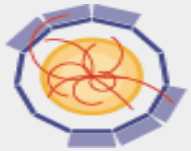
Common Geometry Primitives library
WP3

Gabriele Cosmo (CERN EP/SFT)
for the VecGeom team



AIDA 2020

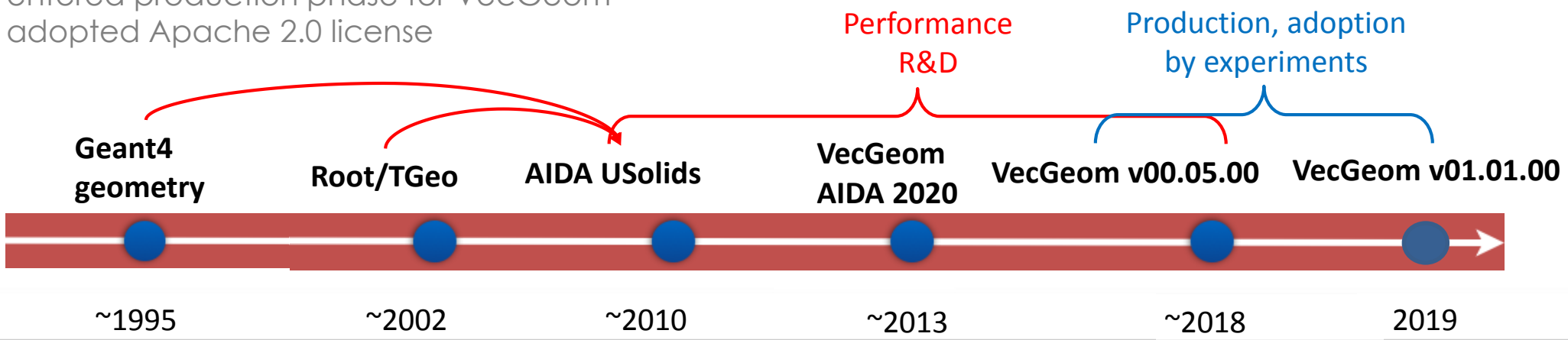
CERN

Outline

- VecGeom evolution
- Status
- Recent developments
  - Optimisation of tessellated shapes
  - Multi-Union structure
  - Added shapes: tetrahedron, elliptical cone, generic polycone
- Current activities & future work/extensions
- Conclusions

VecGeom Evolution

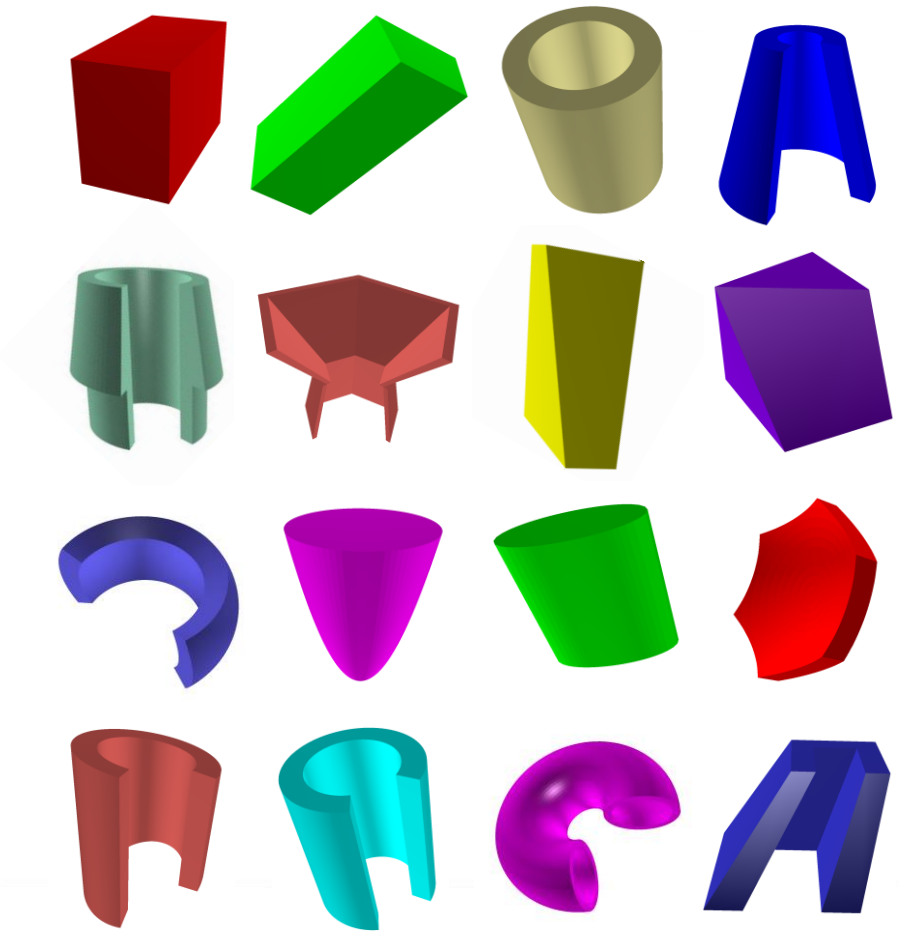
- AIDA project aiming to unify Geant4 and Root geometry algorithms
  - merge code base
  - pick best implementation and increase code quality
  - improve performance and increase long term maintainability
- Extended scope in **VecGeom**
  - encompass parallelism/vectorization
  - multi-architecture/multi-platform support
  - provide advanced navigation features
- Old initial USolids implementation phased out in 2018
  - entered production phase for VecGeom
  - adopted Apache 2.0 license



Shapes implementation status

- Available in VecGeom:
  - Box, Orb, Trapezoid (Trap), Simple Trapezoid (Trd), Sphere (+ sphere section), Tube (+ cylindrical section) , Cone (+ conical section), Generic Trapezoid (Arb8), Polycone, Polyhedron (+generic)
  - Generic Polycone, Elliptical Tube
  - Paraboloid, Parallelepiped (Para), Hyperboloid, Ellipsoid, Torus (+ torus section), Scaled Solid, Boolean (addition, subtraction, intersection), Cut Tube, Simple Extruded Solid (SExtru), Tessellated Solid, Extruded Solid
  - Tetrahedron (Tet), Multi-Union
- Missing:
  - Ellipsoid (+cut), Elliptical Cone
  - Twisted shapes (box, trap, tube)

    complex and infrequent use

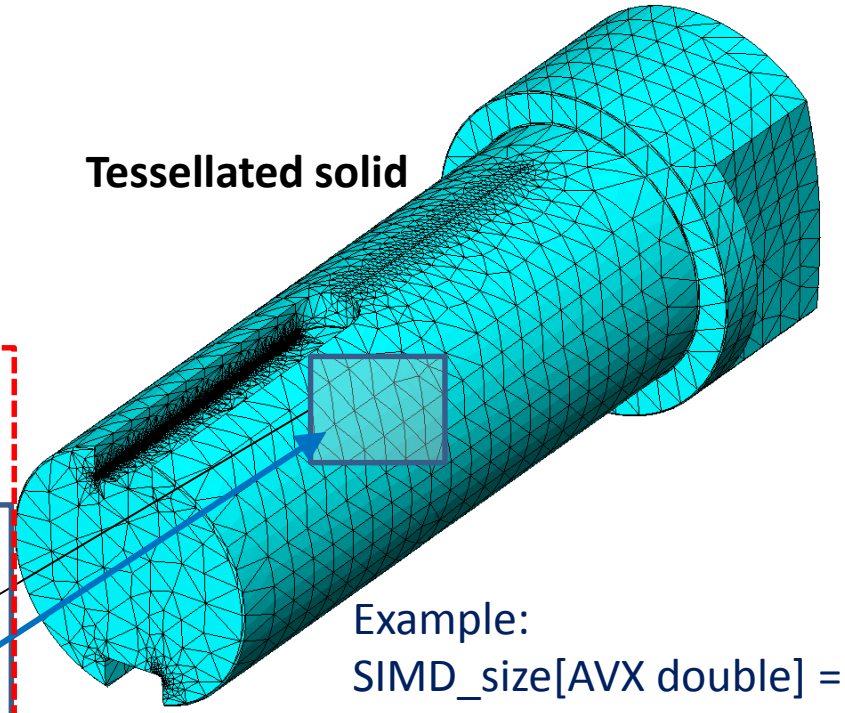


Recent developments

Multi-level vectorization

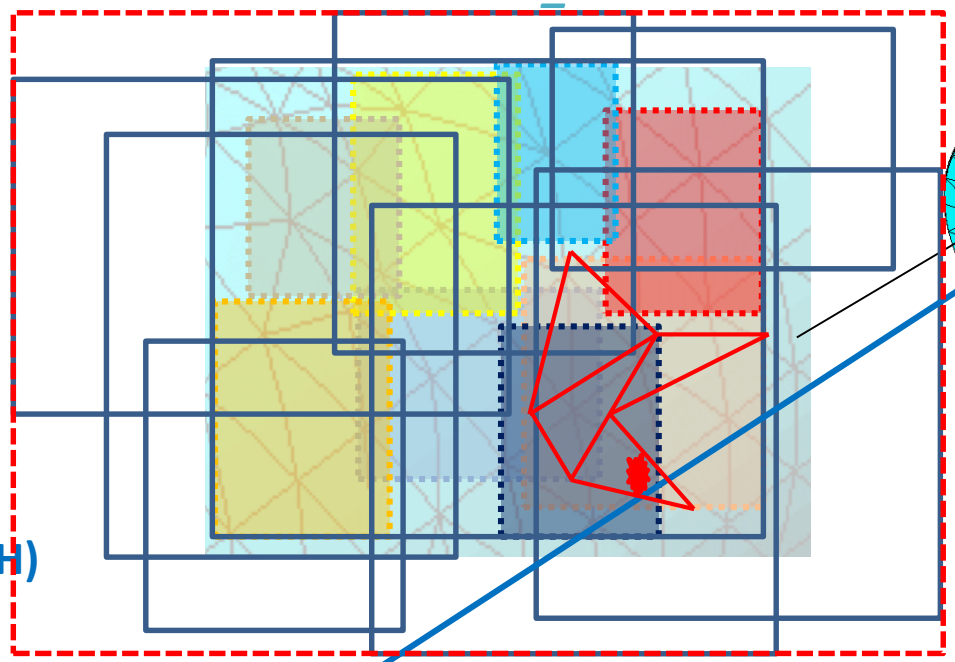
Enhanced distance computing for tessellated solid

Tessellated solid



Example:
SIMD_size[AVX double] = 4
SIMD_size[AVX float] = 8

Group neighbor triangles in clusters, 4 per cluster. Store data in *vecCore::Double_v*



Make groups of bounding boxes of clusters, 8 per group

Continue grouping by 8 and make "super" bounding boxes
-> bounding volume hierarchy (BVH)

Vectorize in double computation of distances to triangles in each cluster

Vectorize in float computation of distances to super-boxes to select only hit candidates

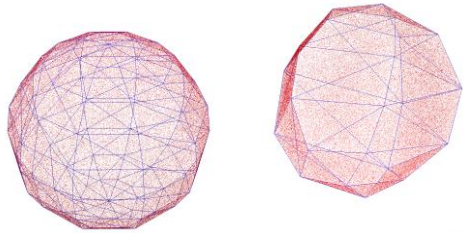
Repeat the same with the content of the boxes being hit, until we get the candidate clusters

e^-

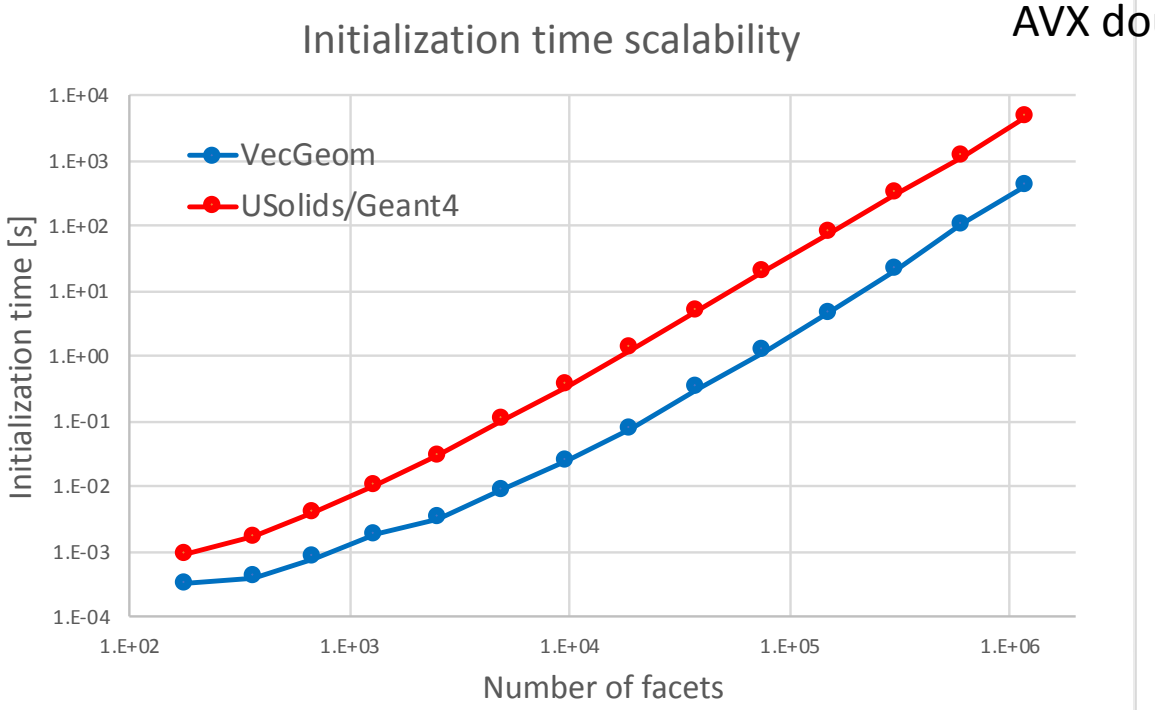
M.Gheata

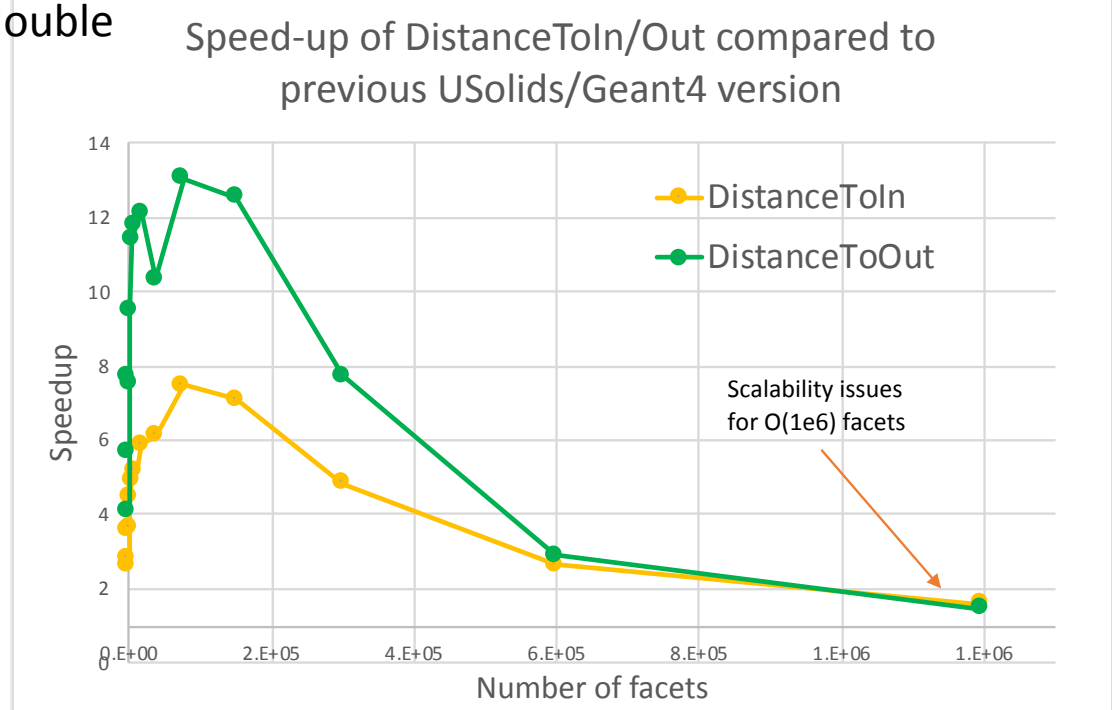
Improved Tessellated Solid performance

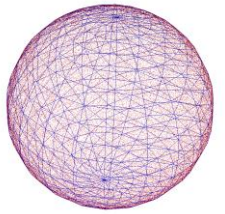
- O(10) speed-up compared to the original USolids/Geant4 implementation in both initialization and run time for up to 100K facets
- Next step: improve of scalability for very large number of facets/components
  - Vectorized approaches are available in industry ray tracers (such as Intel Embree)
  - Trading precision for speed in some calculations (isotropic safe distance)



AVX double





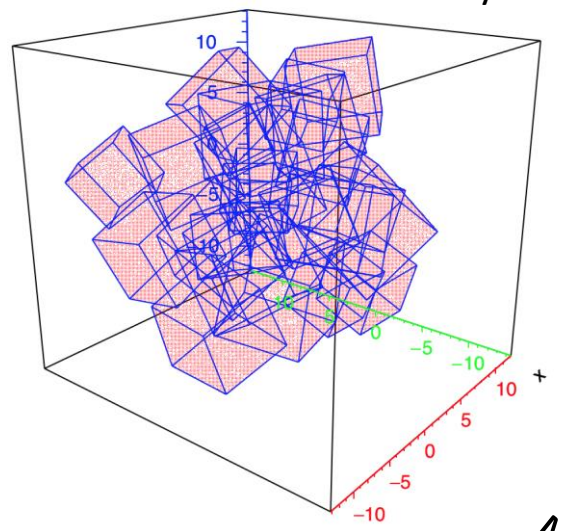


M.Gheata

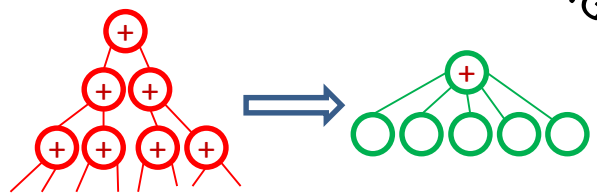
Multi-Union structure in VecGeom

- Boolean unions are represented as binary trees
  - Pathologically slow in simulation - too many individual checks
- Multi-unions representing nodes at same level with same material
  - Optimisation structures to limit selection of candidates
  - Implementation in Geant4 using voxelization helper
- Re-implemented in VecGeom using technique based on Bounding Volume Hierarchies (BVH)
  - Vectorised search of candidates
  - Similar technique adopted as for Tessellated Solid

Artificial model for scalability testing



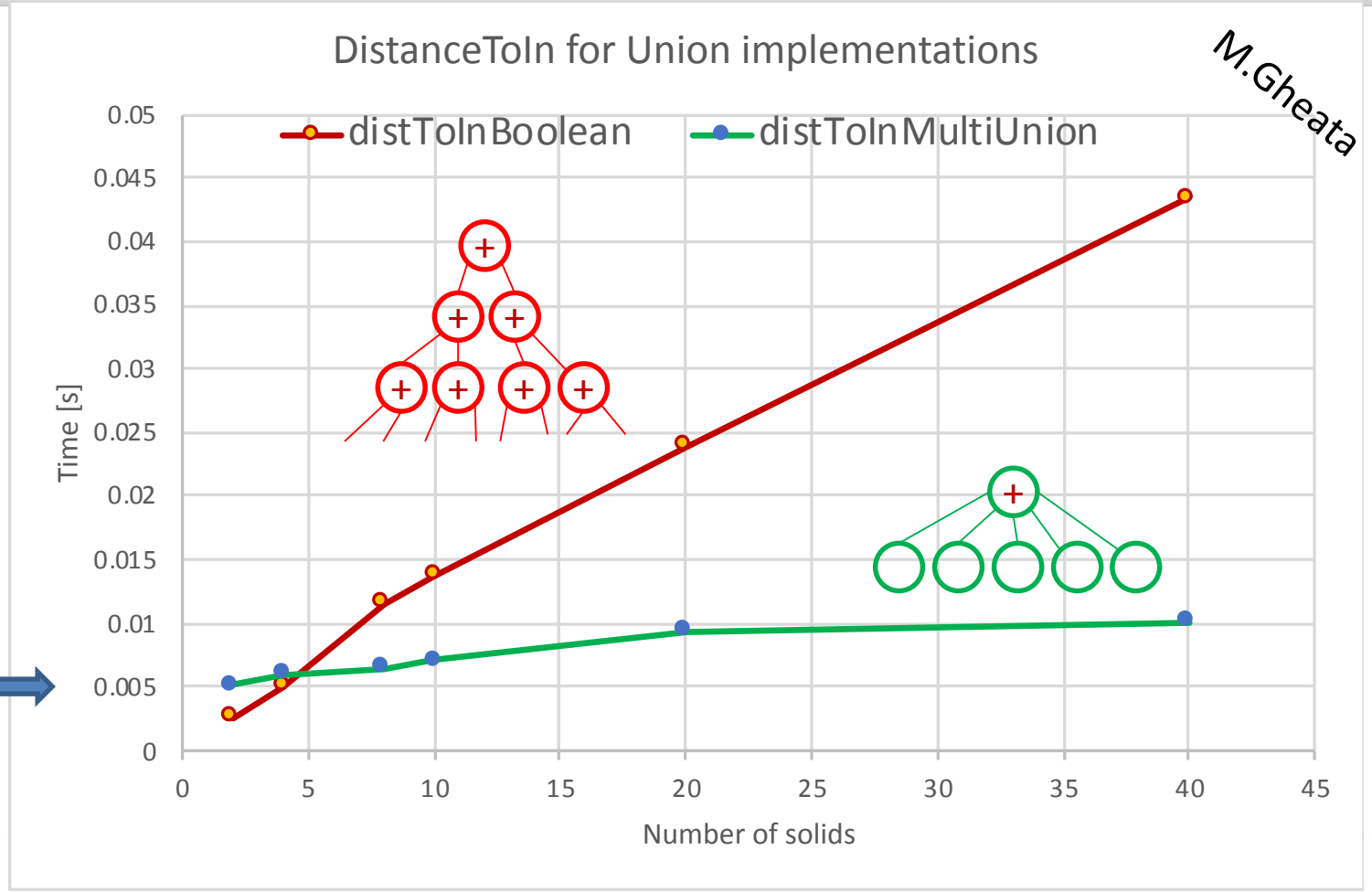
M.Gheata



Boolean union Multi-union

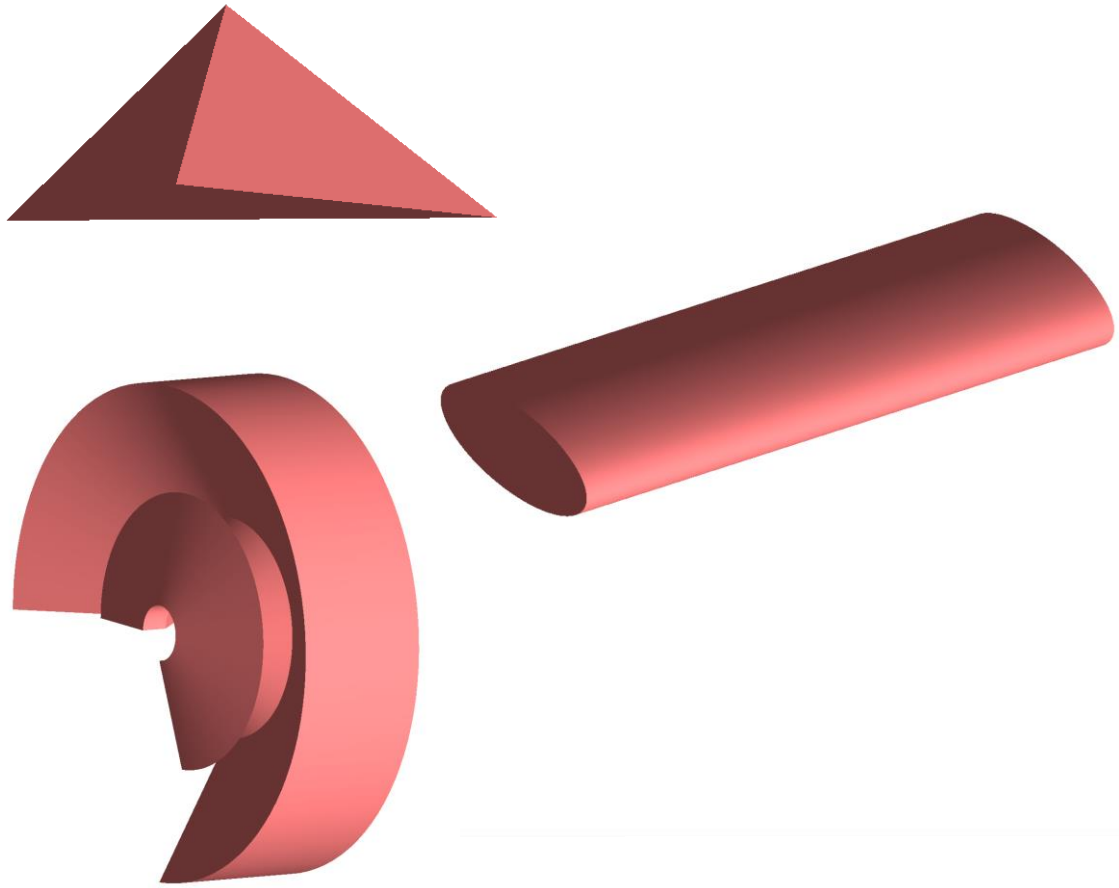
Replacing Boolean union with Multi-Union

- 2x-4x speed-up compared to corresponding implementation in USolids/Geant4 for up to several hundred components
- VecGeom implements automatic conversion of classic Boolean union of volumes to the new multi-union structure
- Much better scaling performance for large number of components



New missing shapes added

- Tetrahedron
  - A solid defined by four points in space
- Elliptical Tube
  - A cone with elliptical cross-section
- Generic Polycone
  - A polycone constructed by specifying points through (r,z) coordinates and optionally non-monotonic order for Z sections



Current VecGeom version

- Version v1.1.0 of VecGeom containing all latest new features and fixes
  - Fixes for corner-case problems in tube and polyhedron
  - First implementation of GDML reader based on Xerces-C
  - Faster safety computation for tessellated solid
  - Removed obsolete USolids module and configuration
- Reference version for latest Geant4 10.5 release

Building Geant4 to use VecGeom shapes

a. Install VecGeom library
   - Scalar mode:

```
cmake -DBACKEND=Scalar -DGEANT4=OFF -DCMAKE_BUILD_TYPE=Release \
   [...other optional VecGeom switches as needed...] \
   -DCMAKE_INSTALL_PREFIX=${VecGeomINSTALLDIR} ${VecGeomSOURCE}
make -j8 install
```

   - Or .. Vector mode:

```
cmake -DBACKEND=Vc -DGEANT4=OFF -DCMAKE_BUILD_TYPE=Release -DVECGEOM_VECTOR=native \
   [...other optional VecGeom switches as needed...] \
   -DCMAKE_INSTALL_PREFIX=${VecGeomINSTALLDIR} ${VecGeomSOURCE}
make -j8 install
```

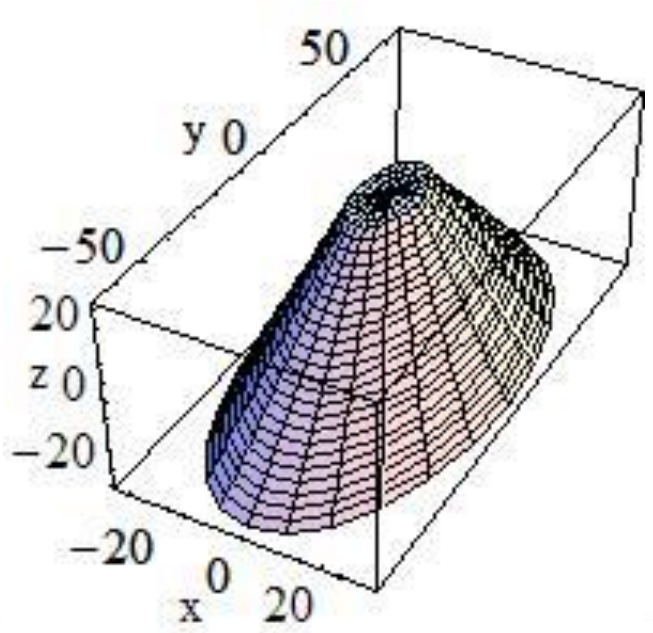
b. Install Geant4
   - `export VecGeom_DIR=${VecGeomINSTALLDIR}/lib/Cmake/VecGeom`
   - Add `-DGEANT4_USE_USOLIDS=ALL` when configuring Geant4 with Cmake, to use all shapes currently being exercised
   - OR, `-DGEANT4_USE_USOLIDS="box;trap"` for configuring to use/replace only specified shapes

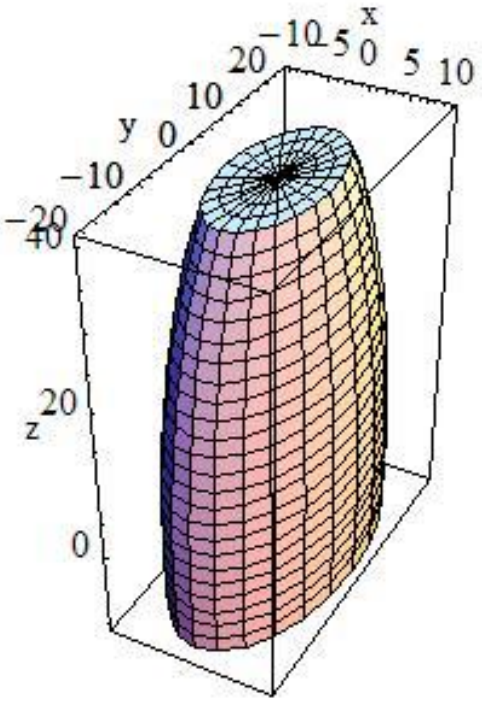
NOTES: Reasonably recent version of the gcc/clang compilers required. Windows VC++ currently not supported

On going activity…

Addition of missing shapes

- G4Ellipsoid
  - A general triaxial ellipsoid with optional cut in Z
- G4EllipticalCone
  - A cone with elliptical cross-section



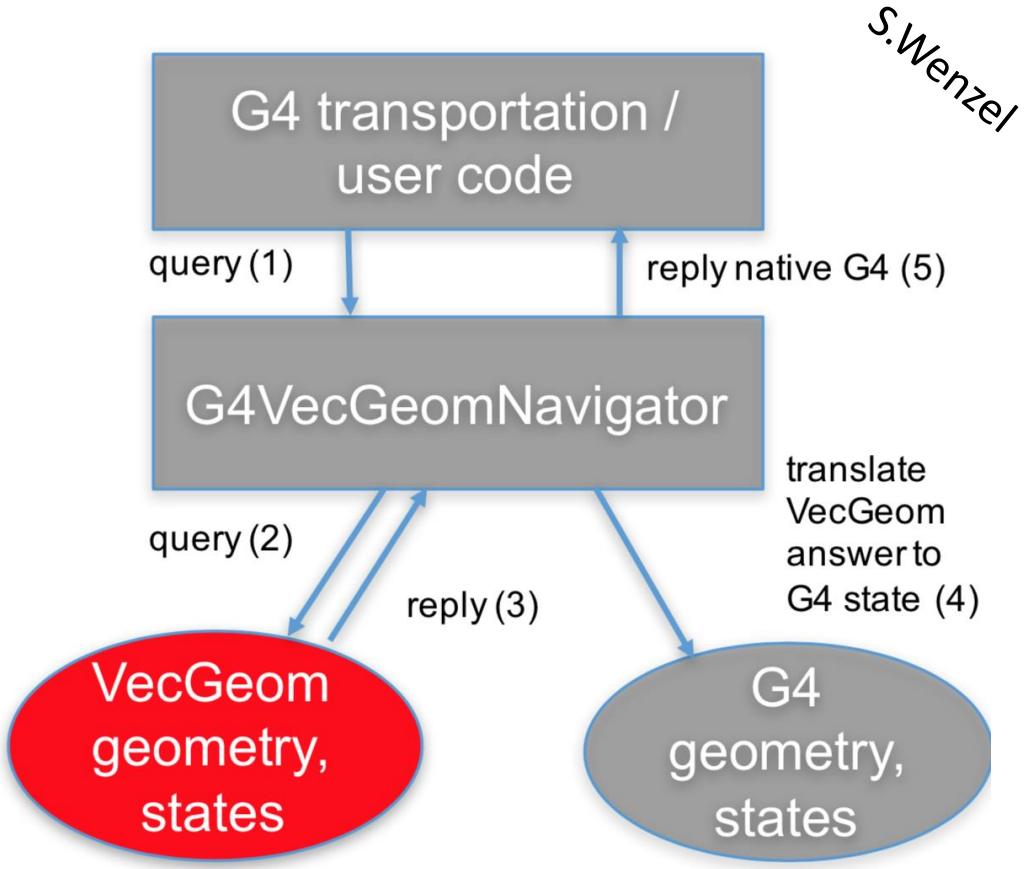


Interface to VecGeom navigation

- VecGeom implements structures which benefits from SIMD (vectorized search/traversal of structure)
  - Increasing size of vector registers on future hardware will automatically make algorithm faster
- Other advantages like strong solid specialization would be made available, especially important for simple solids
- Two options for interfacing navigation:

a) No overhead, full integration but much more complex and potentially user- disturbing
  - Complex changes required in VecGeom and in Geant4; API and type evolution, more abstraction layers, ...

b) User friendly, easy-to-implement, but "some overhead" option:
  - Simultaneous existence of Geant4 and VecGeom geometry with necessary synchronization/translation of states/objects
  - Applicable at first to placed/static geometries only



Student projects & more…

- Enhancements to specialized navigators, neighbor volume detection
- Addition of replicas/divisions
- Use of Intel Embree library for tessellated shapes
- GDML writer & Root I/O persistency
- Generation of polyhedral meshes for shapes
- Addition of overlaps checking
- Improve error logging / diagnostics

Documentation

- Invest some efforts to write a Users Guide
  - VecGeom primitives can be transparently built through either Geant4 or Root modelers
  - Still it is required to document original VecGeom API and features specific to VecGeom navigation and tools
- Set up dedicated web page entry for the project

Summary

- VecGeom now available as production-quality since version v0.5.0
  - Tested by several experiments, adopted by CMS in production
  - Latest version v1.1.0 including most recent features and reference for Geant4 10.5
- Most primitives from the GDML schema now supported
  - Recently added: Generic Polycone, Tetrahedron, Elliptical Cone
  - Added Multi-Union structure and enhanced Tessellated Solid queries
- Added GDML reader and removed old USolids module
- Ongoing work for interfacing Geant4 and Root navigation with VecGeom
- Several student projects defined for completing features of the modeler
- Documentation

Contributors

- *CERN-EP/SFT + AIDA 2020: G.Amadio, J.Apostolakis, G.Cosmo, A.Gheata, M.Gheata, P.Mato, W.Pokorski, E.Tcherniaev*
- *J.Martinez Castro, A.Miranda Aguillar (Mexico), P.Canal, G.Lima (FNAL),* D.Savin (GSoC student), *R.Sehgal (BARC), S.Wenzel (CERN-ALICE)*

- *Repository for VecGeom*
  - *https://gitlab.cern.ch/VecGeom/VecGeom*
- JIRA issue tracking tool
  - *https://its.cern.ch/jira/projects/VECGEOM*
- *AIDA USolids page*
  - *http://cern.ch/aidasoft/USolids*