# VecGeom

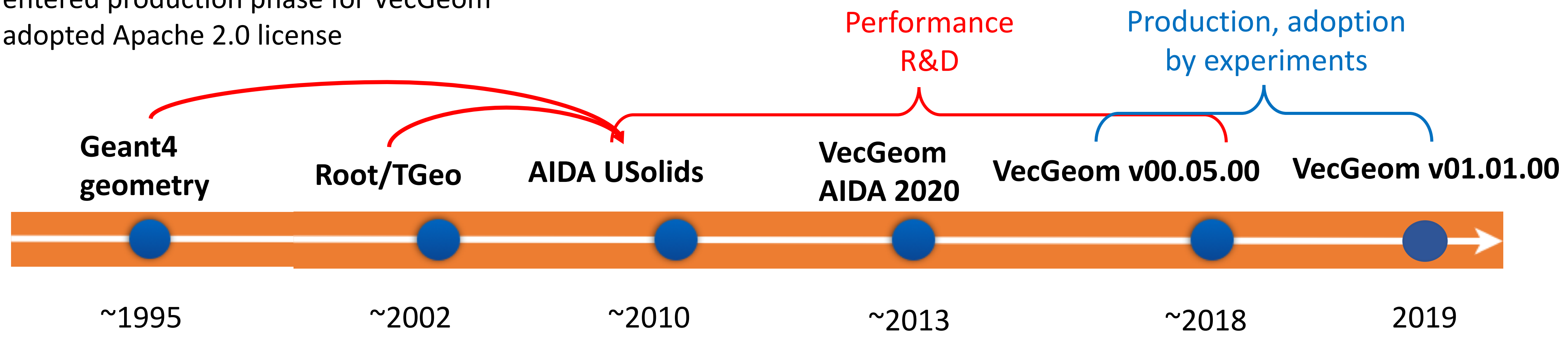
## Vectorised Geometry Library

Gabriele Cosmo (CERN EP/SFT)

*For the VecGeom team*

# VecGeom Evolution

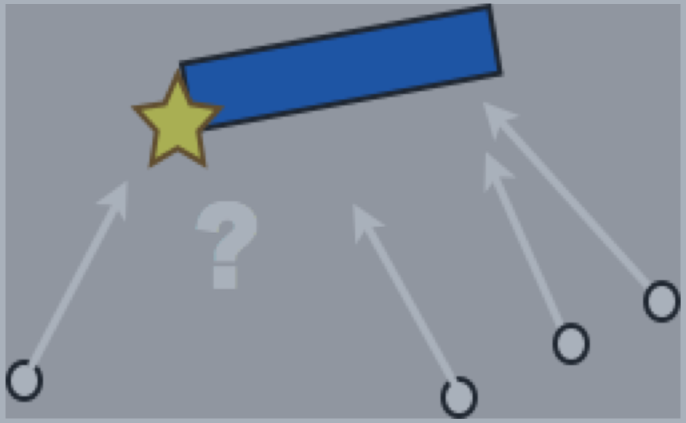
- AIDA project aiming to unify Geant4 and Root geometry algorithms
  - merge code base
  - pick best implementation and increase code quality
  - improve performance and increase long term maintainability
- Extended scope in **VecGeom**
  - encompass parallelism/vectorization
  - multi-architecture/multi-platform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



# VecGeom: targeting vectorisation

***Vector signatures***

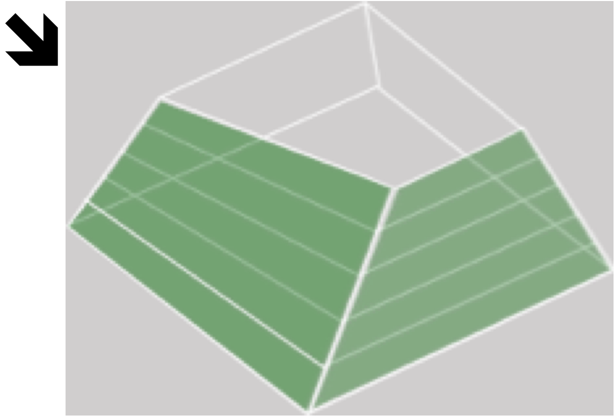
"parallel" collision detection



*Multi-particles queries*

***Internal algorithm vectorization***

internal loop over
lateral planes for
distance calculation

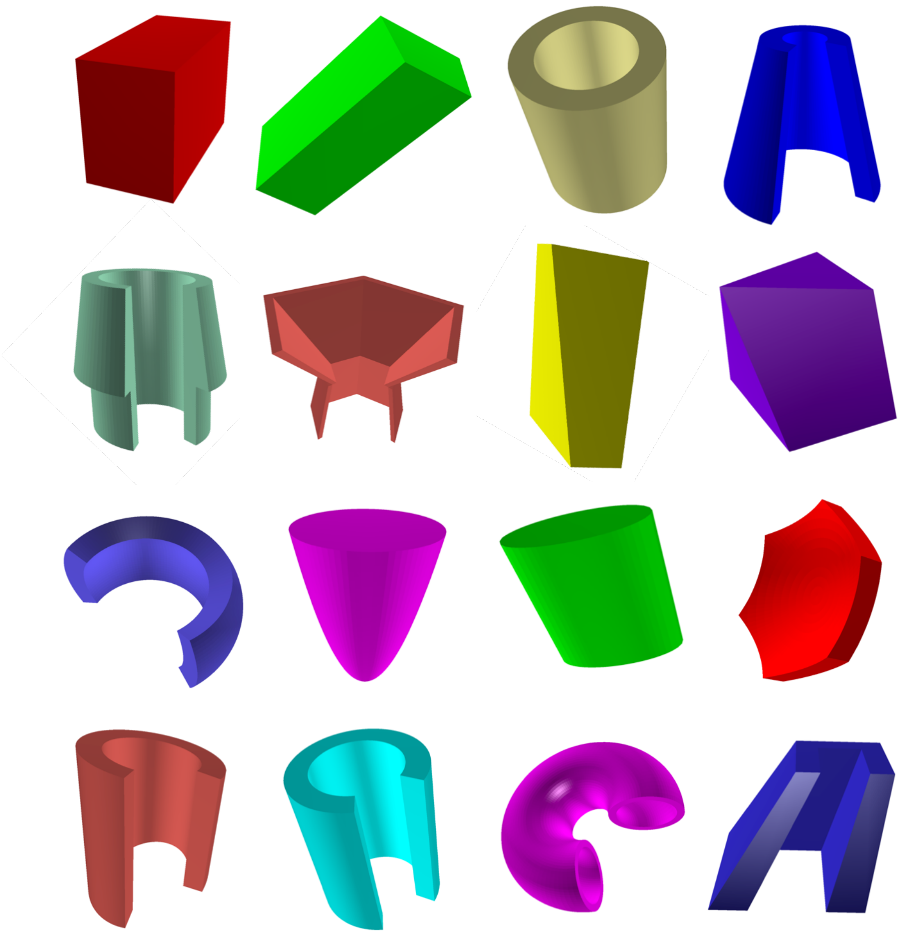


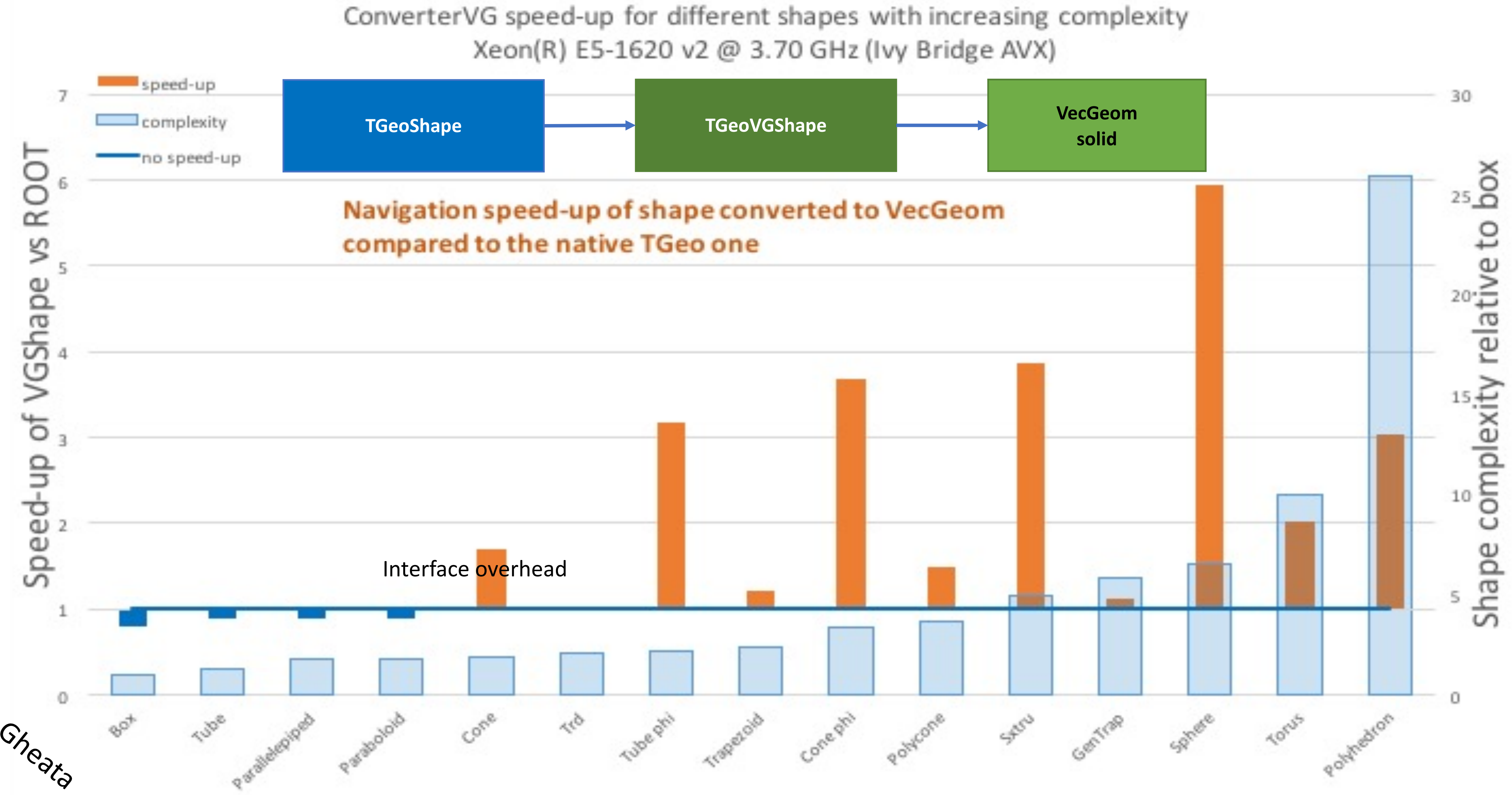
*Beneficial for current simulations*

# 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, Elliptical Cone
  - 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)
  - Twisted shapes (box, trap, tube)

    *complex and infrequent use*





ConverterVG speed-up for different shapes with increasing complexity
Xeon(R) E5-1620 v2 @ 3.70 GHz (Ivy Bridge AVX)

TGeoShape → TGeoVGShape → VecGeom solid

**Navigation speed-up of shape converted to VecGeom compared to the native TGeo one**

Interface overhead

M.Gheata

# Recent developments

# Multi-level vectorization

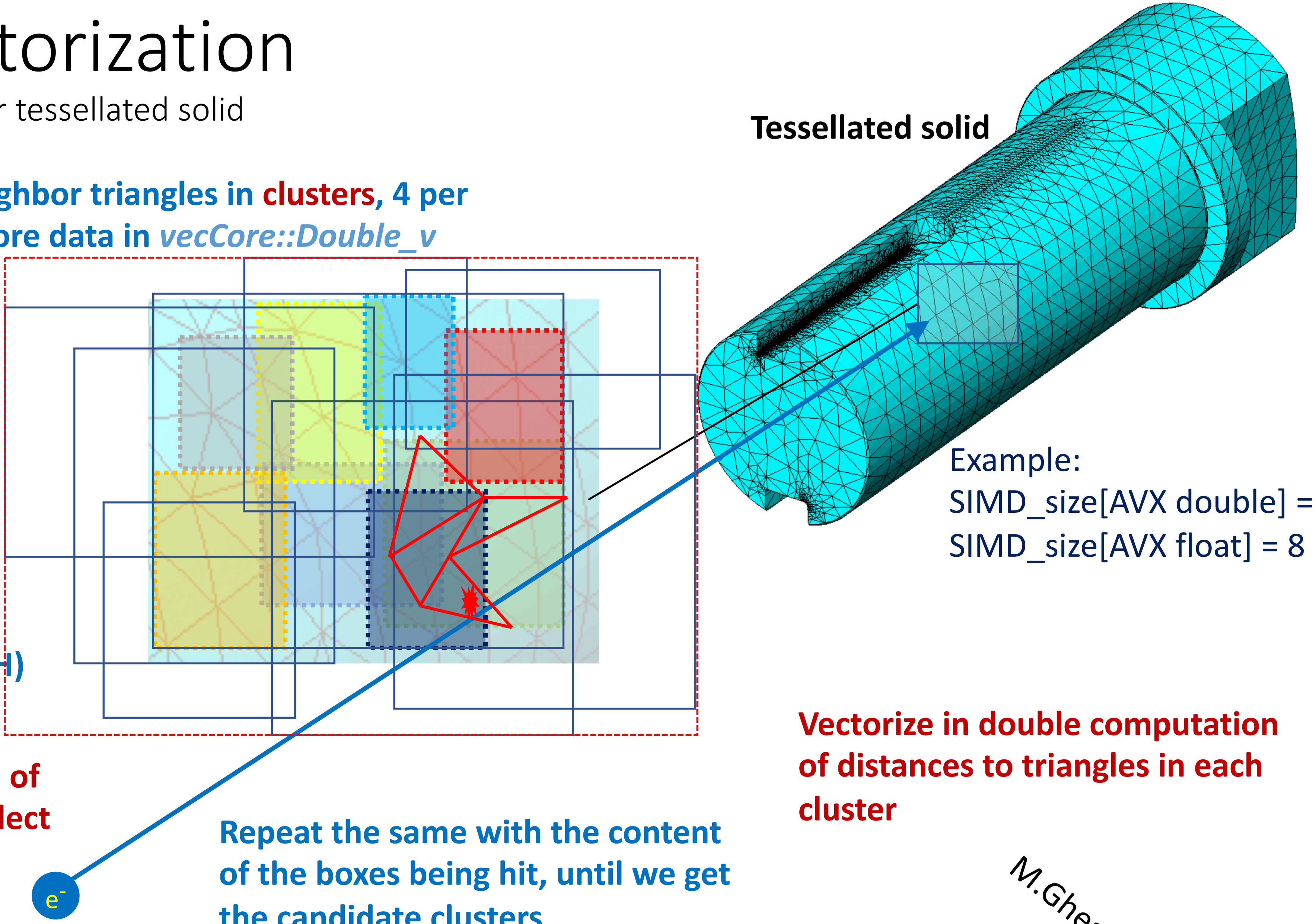
Enhanced distance computing for tessellated solid

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



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

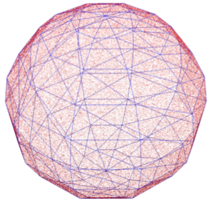
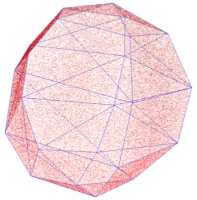
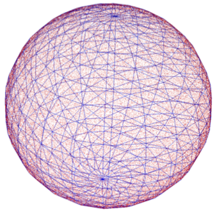
**Vectorize in double computation of distances to triangles in each cluster**

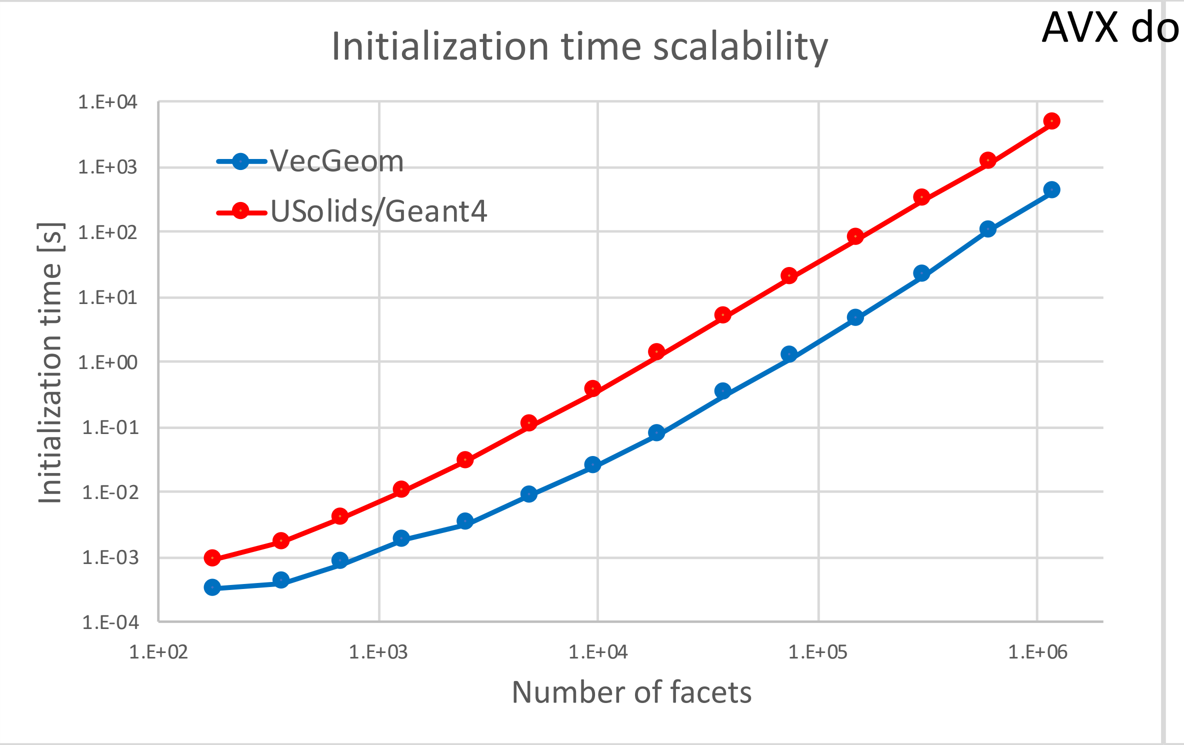
**Repeat the same with the content of the boxes being hit, until we get the candidate clusters**

M.Gheata

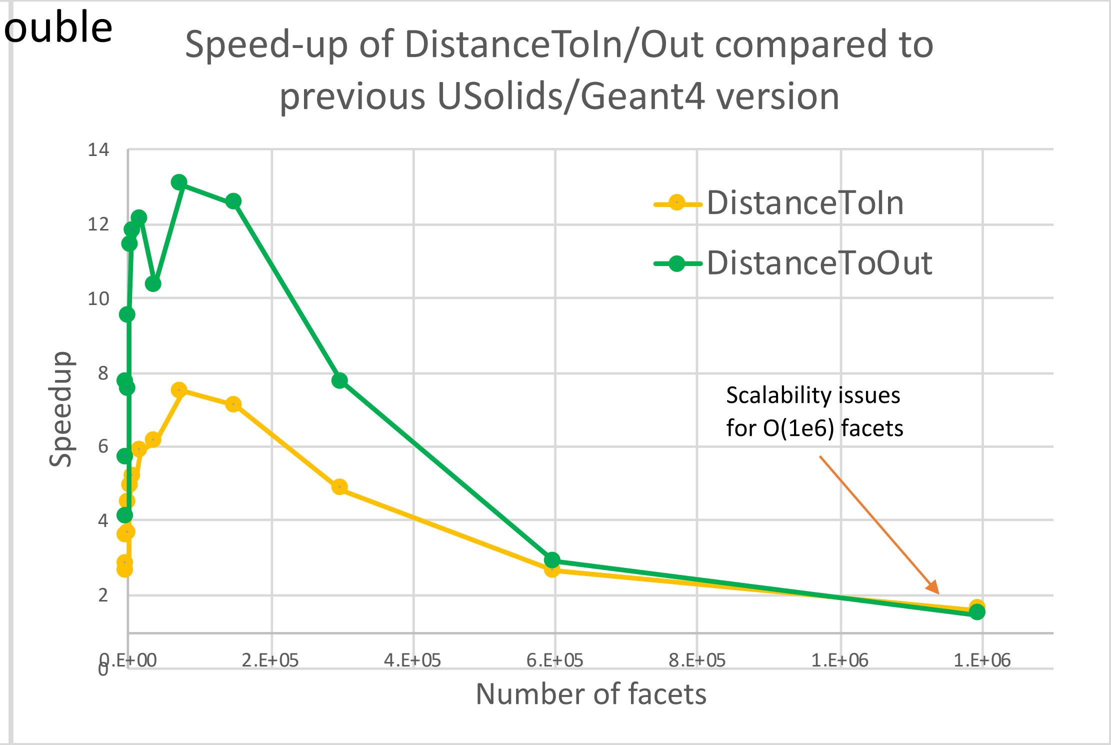
# Effect on Tessellated Solid performance

- 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)


Initialization time scalability

AVX double


Speed-up of DistanceToIn/Out compared to previous USolids/Geant4 version

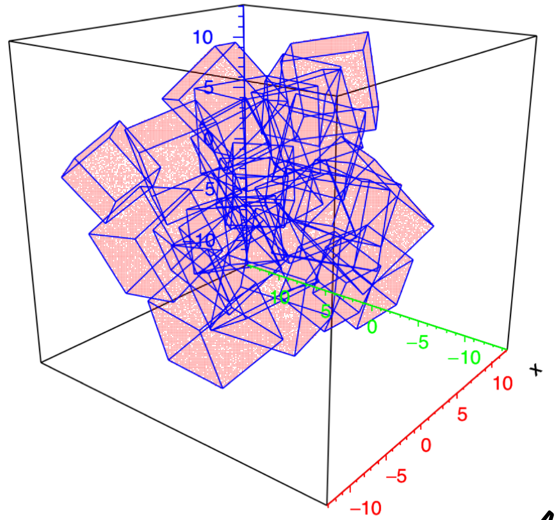
Scalability issues for O(1e6) facets

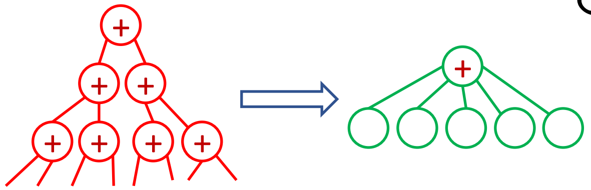
M.Gheata

# Multi-Union structure in VecGeom

- Boolean unions traditionally 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





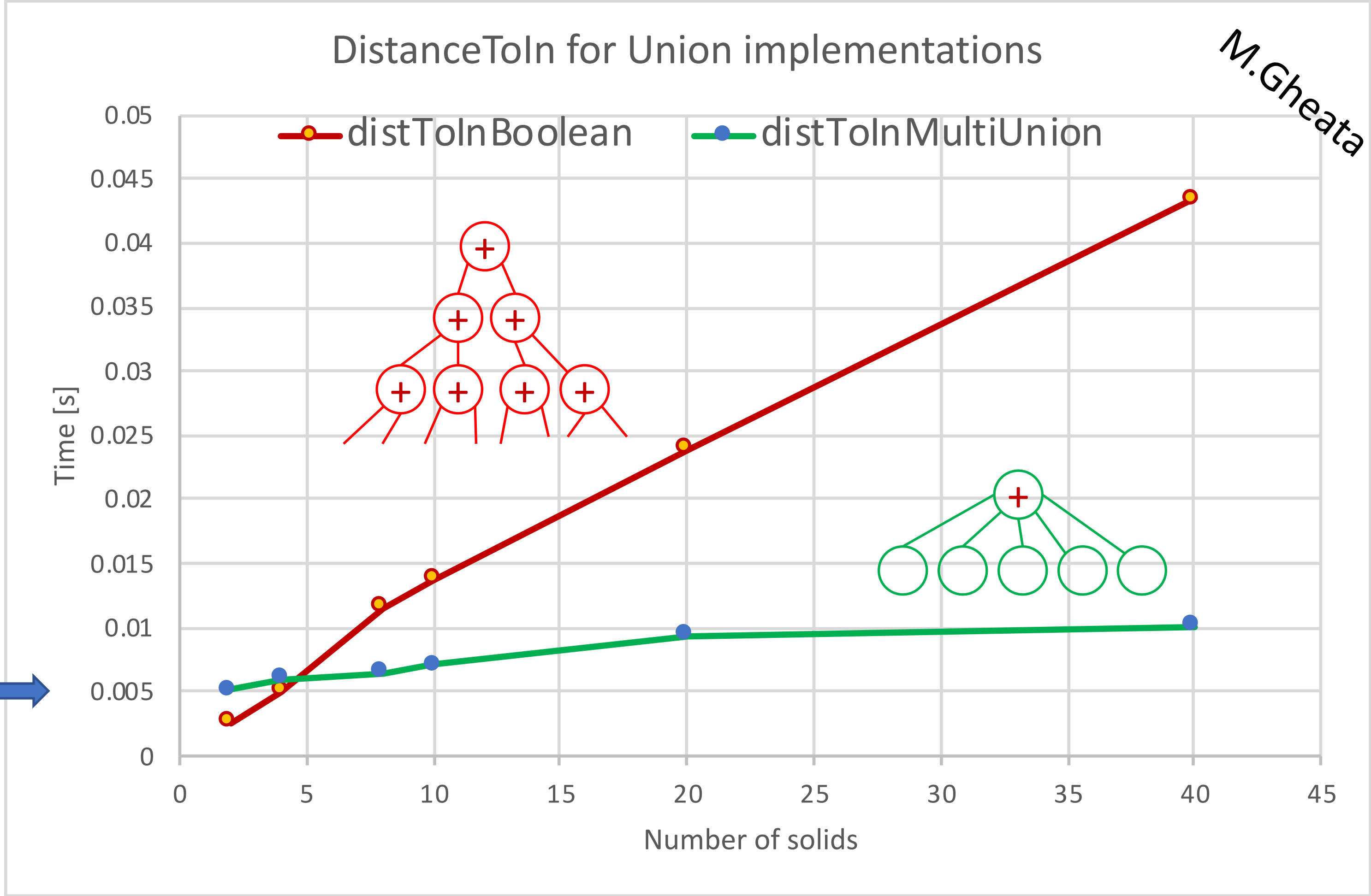
Boolean union

Multi-union

M.Gheata

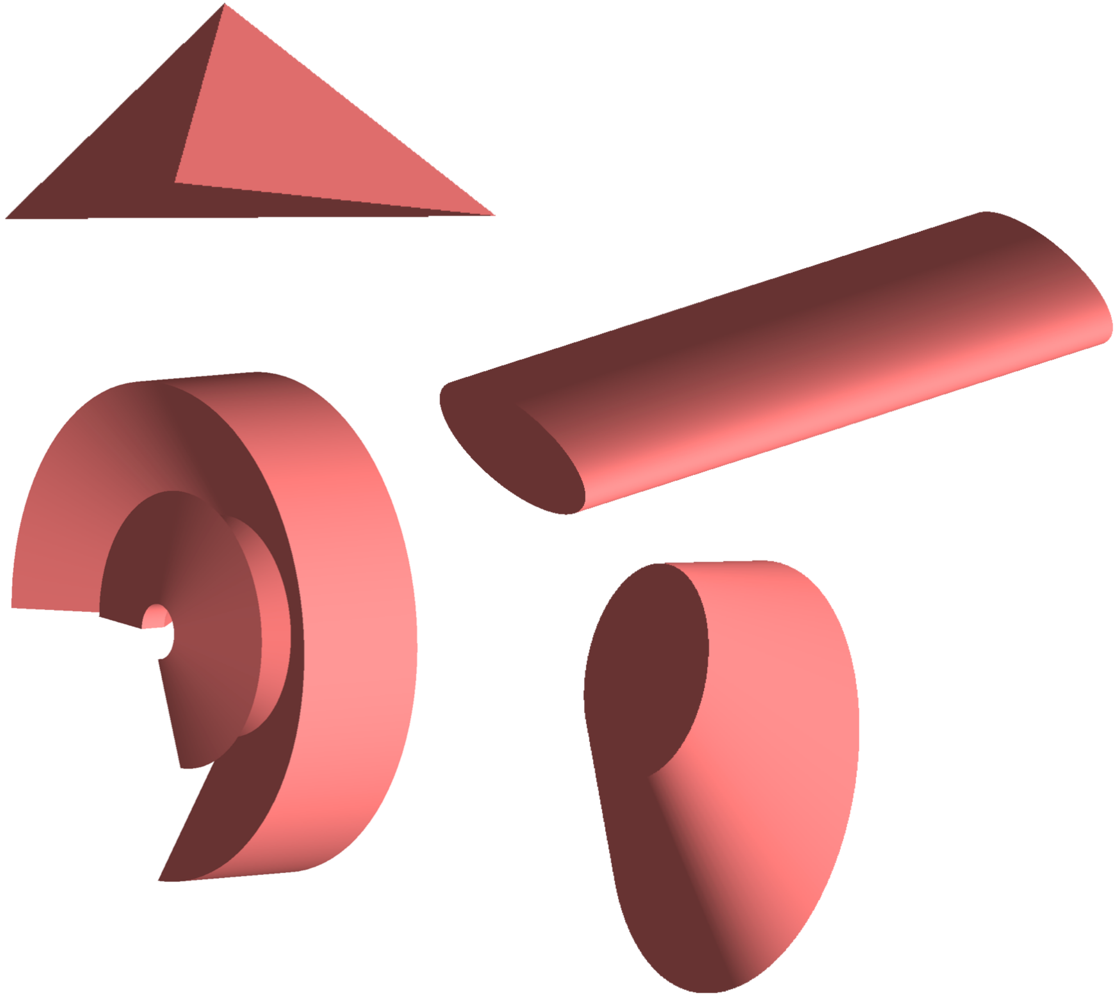
# Replacing Boolean union with Multi-Union

- 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



# Shapes recently added

- Tetrahedron
  - A solid defined by four points in space
- Elliptical Tube
  - A tube with elliptical cross-section
- Elliptical Cone
  - 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

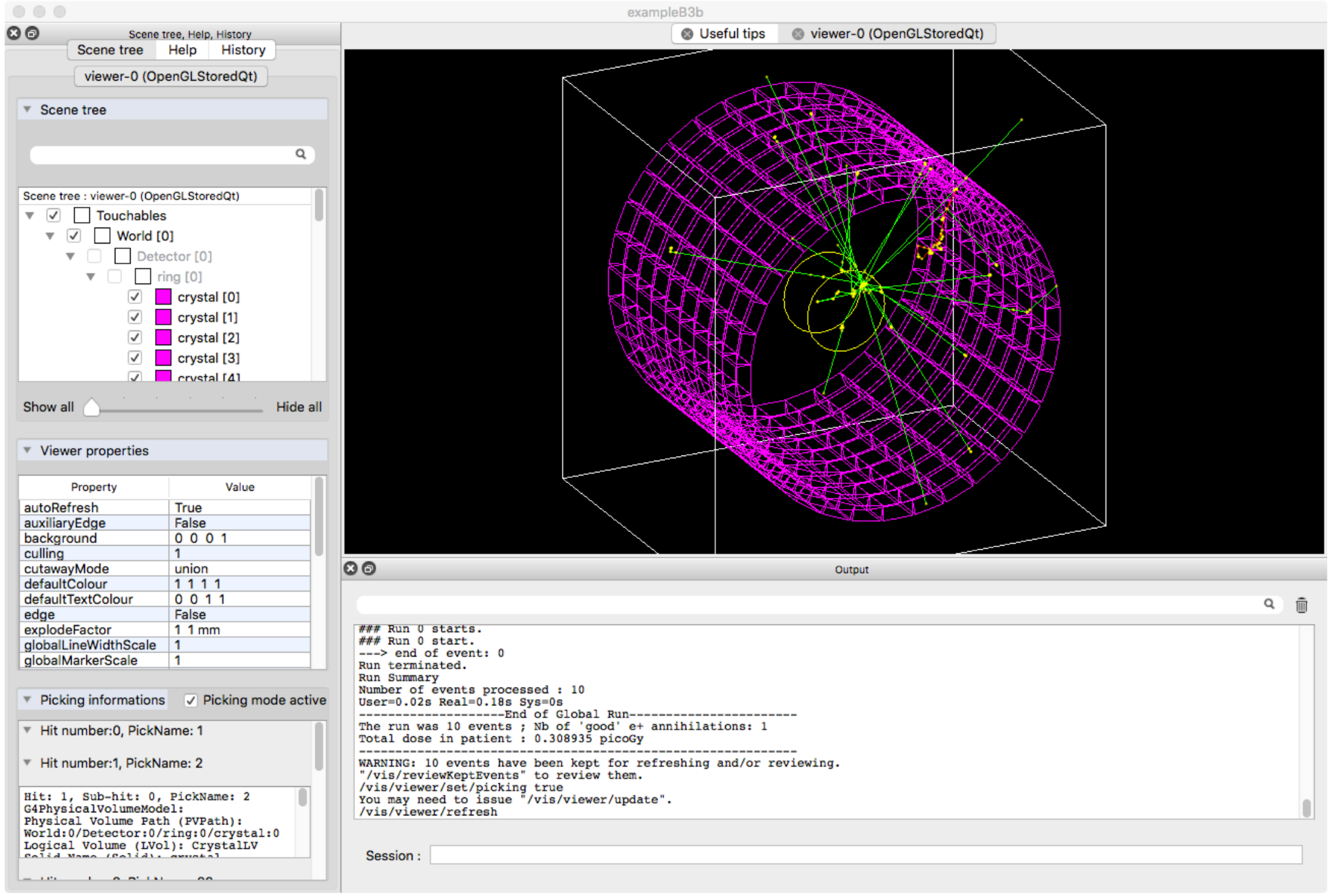


# Current VecGeom version

- Version v1.1.1 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
- Soon will be tagging v1.1.2
  - Fixes in Para, Cone, Tube, Extruded Solid, …
  - Doxygen documentation
- Reference versions for latest Geant4 10.5.p01 release and coming 10.6-Beta

# Using VecGeom in Geant4

- VecGeom can be used in the last Geant4 version 10.5
- VecGeom solids can be used seamlessly since Geant4 10.2 as external library
- Part of nightly builds in Geant4
  - VecGeom shapes from Git master
  - Tested on CentOS/gcc-5.3
- VecGeom in Geant4 works either in batch or interactive mode
  - Including multi-threading (MT)
  - Including replicas/parameterisations



| Property | Value |
|---|---|
| autoRefresh | True |
| auxiliaryEdge | False |
| background | 0 0 0 1 |
| culling | 1 |
| cutawayMode | union |
| defaultColour | 1 1 1 1 |
| defaultTextColour | 0 0 1 1 |
| edge | False |
| explodeFactor | 1 1 mm |
| globalLineWidthScale | 1 |
| globalMarkerScale | 1 |

Hit number:0, PickName: 1

Hit number:1, PickName: 2

```
Hit: 1, Sub-hit: 0, PickName: 2
G4PhysicalVolumeModel:
Physical Volume Path (PVPath):
World:0/Detector:0/ring:0/crystal:0
Logical Volume (LVol): CrystalLV
Solid Name (Solid): crystal
```

```
### Run 0 starts.
### Run 0 start.
---> end of event: 0
Run terminated.
Run Summary
Number of events processed : 10
User=0.02s Real=0.18s Sys=0s
--------------------End of Global Run-----------------------
The run was 10 events ; Nb of 'good' e+ annihilations: 1
Total dose in patient : 0.308935 picoGy
------------------------------------------------------------
WARNING: 10 events have been kept for refreshing and/or reviewing.
"/vis/reviewKeptEvents" to review them.
/vis/viewer/set/picking true
You may need to issue "/vis/viewer/update".
/vis/viewer/refresh
```

# 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

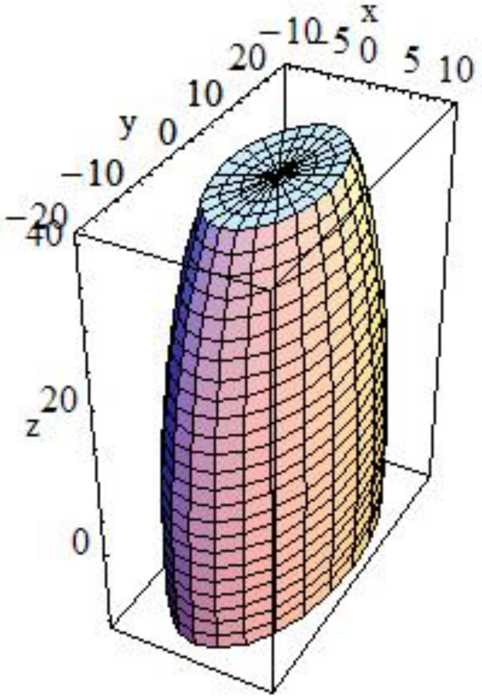
NOTES: Reasonably recent version of the gcc/clang compilers required. Windows VC++ currently not supported

On going activity…

# Addition of missing shapes

- G4Ellipsoid
  - A general triaxial ellipsoid with optional cut in Z

*This completes the set of shapes in the GDML schema (still excluding twisted shapes)*



# Interface to VecGeom navigation

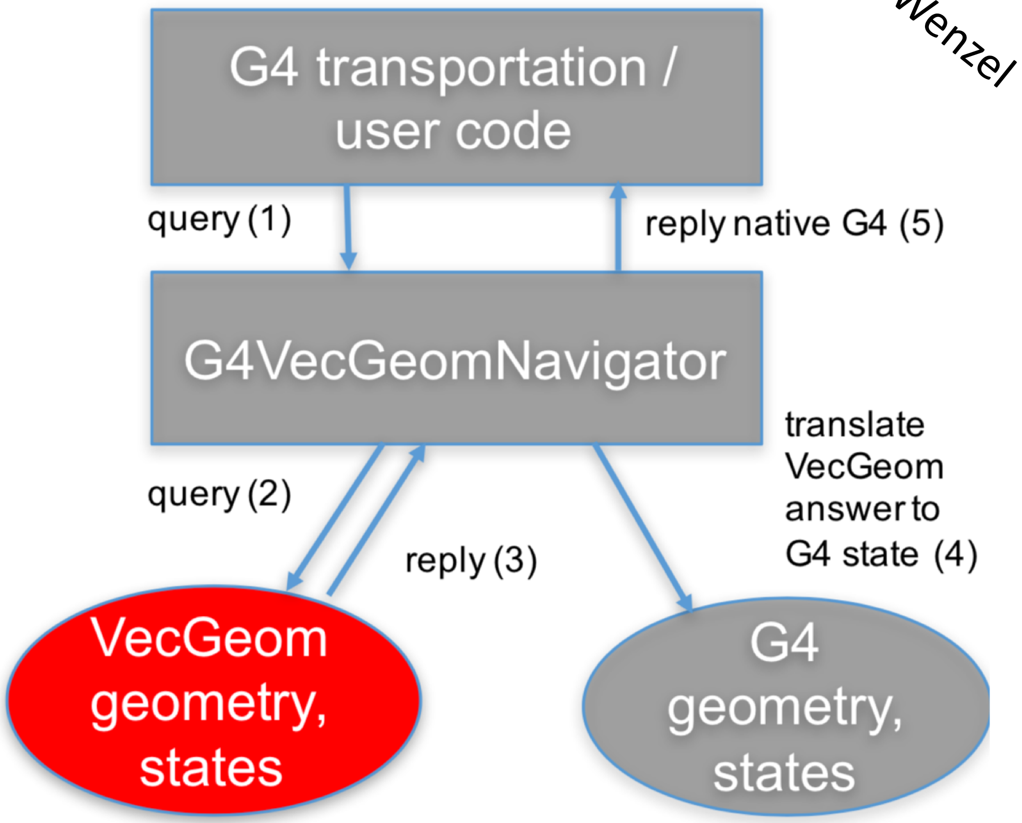
- VecGeom implements structures which benefits from SIMD (vectorized search/traversal of structure)
  - Increasing size of vector registers on future hardware will automatically make algorithm faster
- Other advantages like strong solid specialization would be made available, especially important for simple solids
- Two options for interfacing VecGeom navigation in Geant4:

a) No overhead, full integration but much more complex and potentially user-disturbing

  - Relatively 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
  - Useful as first approach and/or feasibility study



S.Wenzel

# 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
  - Currently working to setup/complete Doxygen reference guide

# Summary

- VecGeom now available as production-quality since version v0.5.0
    - Tested by several experiments, adopted by CMS in production
    - Latest version v1.1.1 including most recent features and reference for Geant4 10.5.p01
- Most primitives from the GDML schema now supported
    - Recently added: Generic Polycone, Tetrahedron, Elliptical Tube, 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://sft.its.cern.ch/jira/projects/VECGEOM*