



Centro de Investigación
en Computación
Instituto Politécnico Nacional



A vectorization approach for multifaceted solids in VecGeom

Mihaela Gheata for the VecGeom team

CHEP 2018

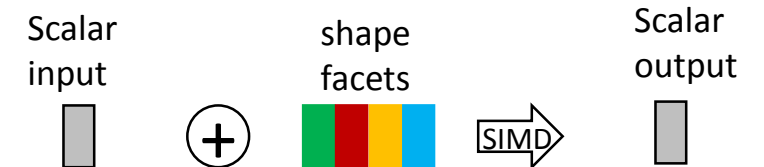
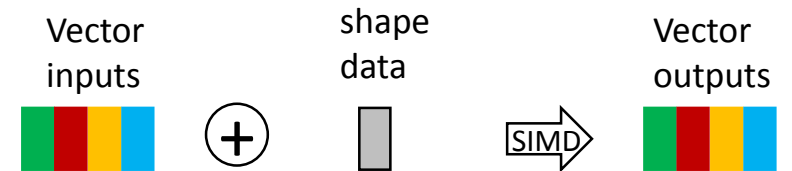
9-13 July, Sofia

Context

- Geometry modeling is a performance-critical component for detector simulation and tracking
 - Locating tracks, computing line/solid intersections, surface normals, ...
- VecGeom development started 5 years ago aiming to optimize this functionality for multi-particle queries
 - Vectorize on multiple inputs provided by GeantV as “baskets” of tracks
- VecGeom started an important R&D on vectorization techniques
 - Resulting in the VecCore vectorization library
<https://github.com/root-project/veccore>
See L.Moneta’s [talk](#)
- Extended USolids effort for unifying/replacing the existing geometry solids algorithms (AIDA project)
 - <http://aida2020.web.cern.ch>

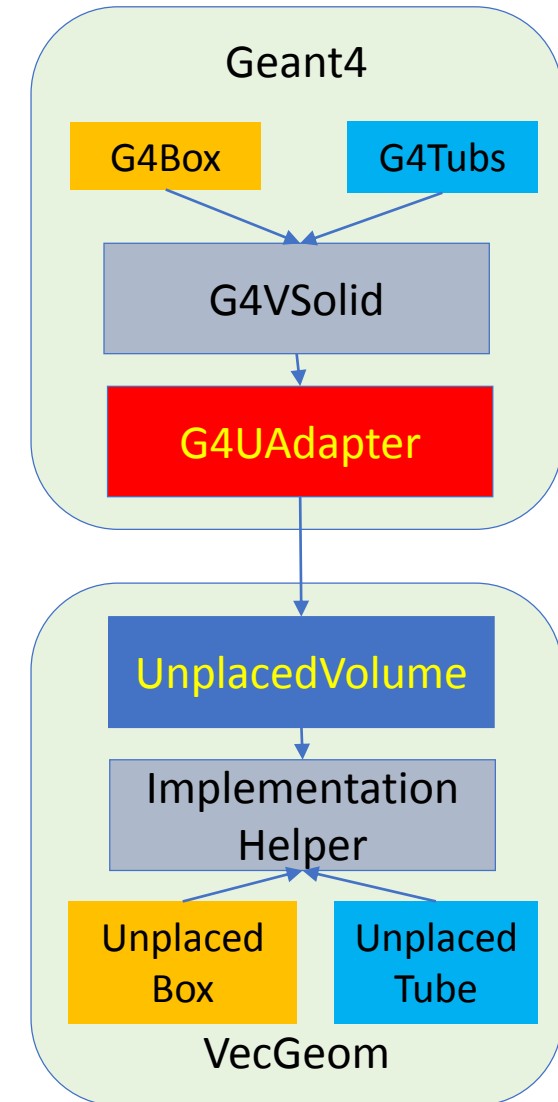
Vectorization modes

- Many particle input -> vectorization on input data
 - Using SIMD operations based on VecCore
 - Close to ideal efficiency for simple solids, not so good for very complex ones
 - Complexity (branching, early returns) creates bottlenecks for vectorization
- Single particle input -> vectorization on internal loops
 - Multiple facets (trapezoids, polyhedra, tessellated, extruded solids), or multiple sections of same type (polycones)
 - **Very important optimization mode for scalar clients** (Geant4, tracking)



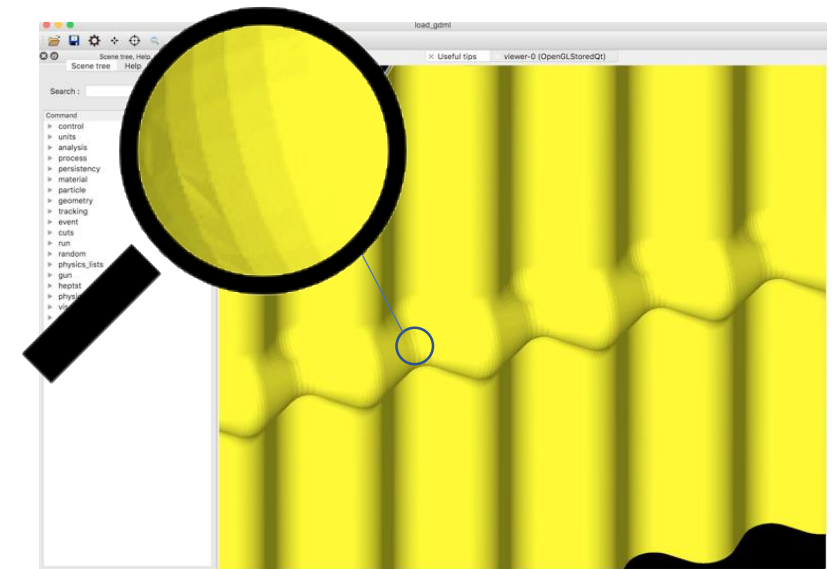
VecGeom & Geant4

- VecGeom solids can be used instead of native Geant4 solids since release 10.2
 - Activated with compilation option – transparent to users
- Used in production by CMS in 2018
 - Overall simulation speedup of ~7-13%
 - Other experiments investigating this option
- Most benefits are due to internal vectorization of complex solids (polycones, polyhedra)
 - **Worth extending vectorization in scalar input mode for more multi-faceted solids**



The obvious candidate: tessellated solid

- Simulation of complex surfaces is needed in certain cases
 - Medical applications, applications sensitive to material budget, space applications for shielding studies
- Surfaces are represented as meshes of connected facets (coming from CAD programs)
 - Vectorization seems natural for the loop over facets
- A better approach is to pre-select the facets to be checked
 - In most cases not all facets have to be checked. Can we select the subset of candidates also in vectorized mode?
 - In Geant4 voxelization is used for this purpose

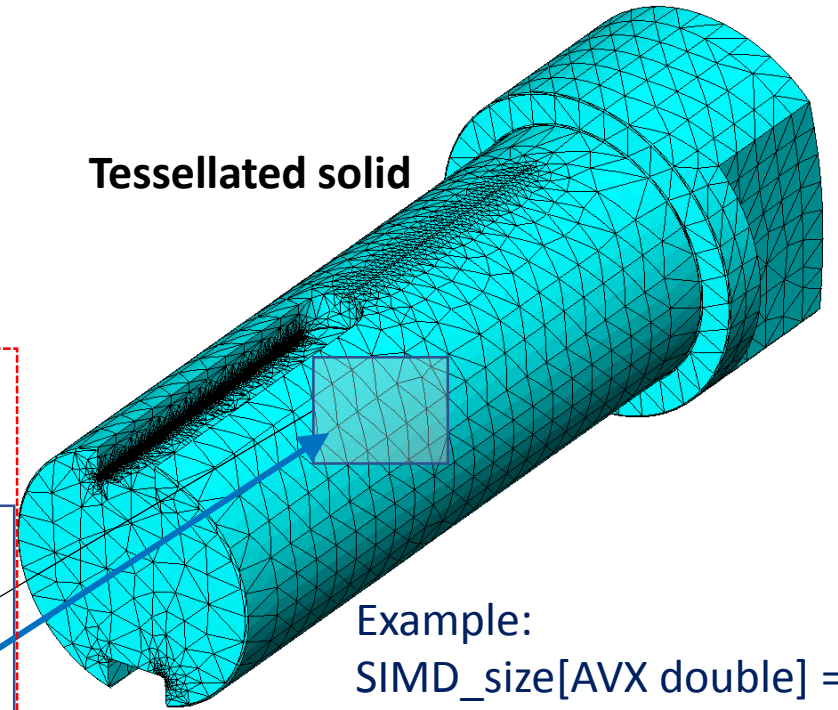


The LHCb RF-foil loaded from GDML as tessellated solid (164k facets) and visualised in Geant4 through VecGeom

Multi-level vectorization

Computing the distance to the tessellated solid

Tessellated solid

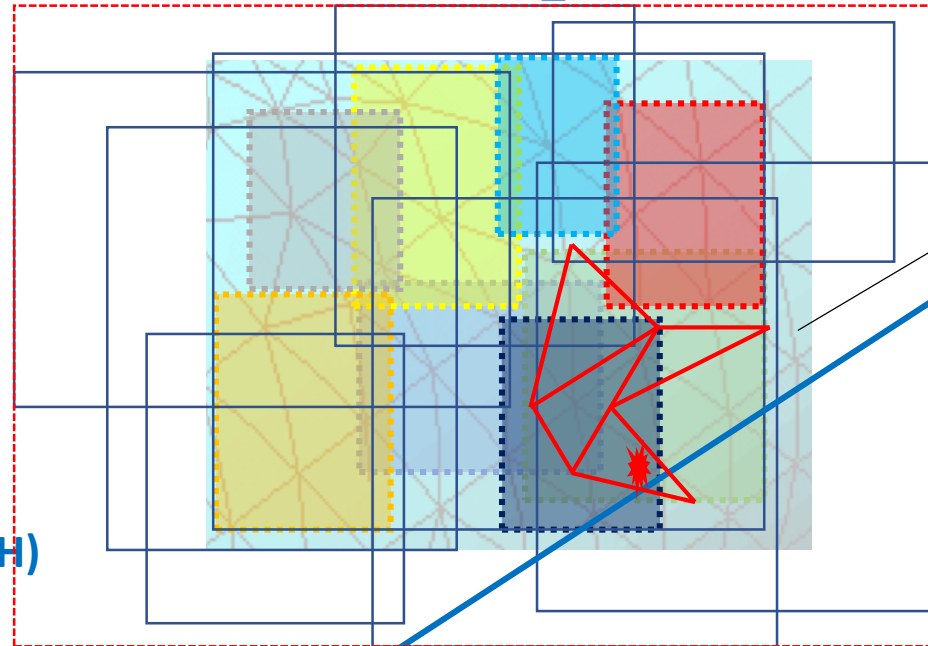


Group neighbour triangles in **clusters**, 4 per cluster. Store data in `vecCore::Double_v`

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



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

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

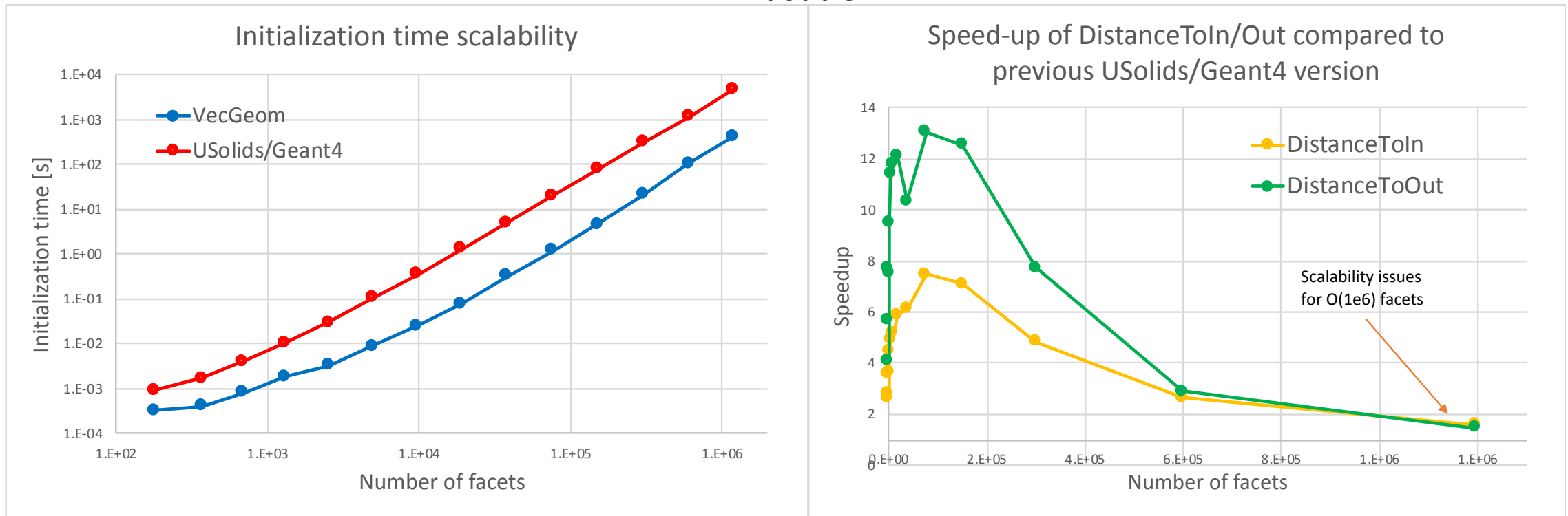
Vectorize in double computation of distances to triangles in each cluster



Tessellated solid performance

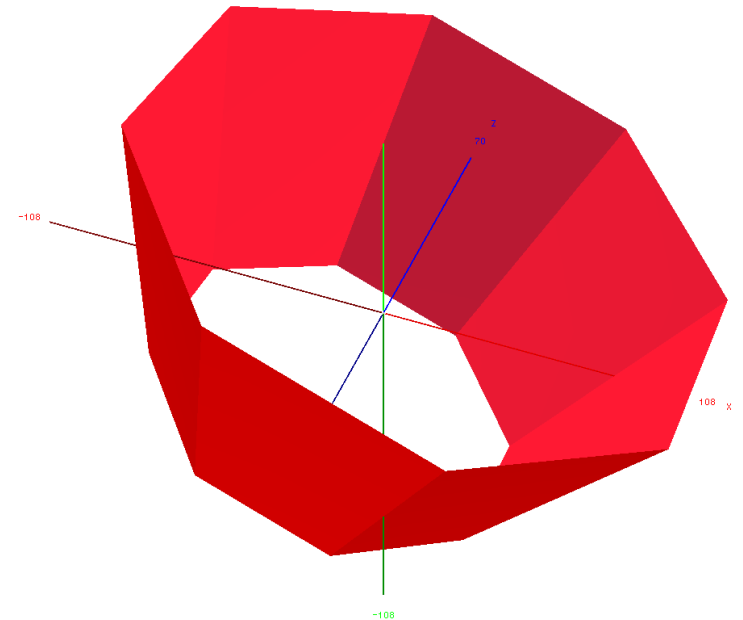
- $O(10)$ speed-up compared to Geant4 in both initialization and run time for up to 100K facets

AVX double



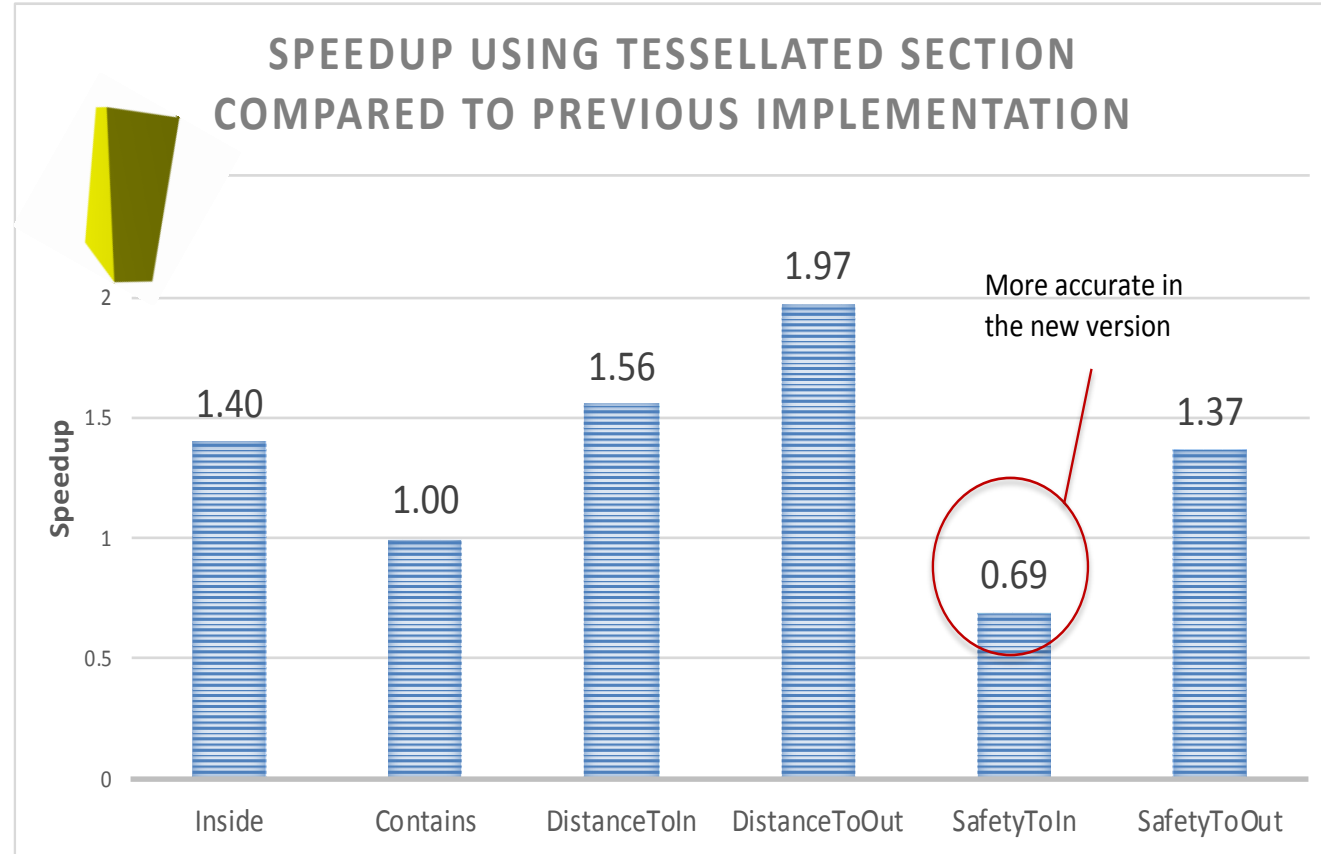
Extending multi-faceted approach to other solids

- Generalized to more multi-faceted solids
 - Trapezoids, polyhedra, extruded solids
- Created a new SIMD helper class
 - Representing a surface made of quadrilateral tiles, organized in clusters of size = vector length
 - Delimited by two Z planes
 - Using explicit vectorization on tiles based on VecCore



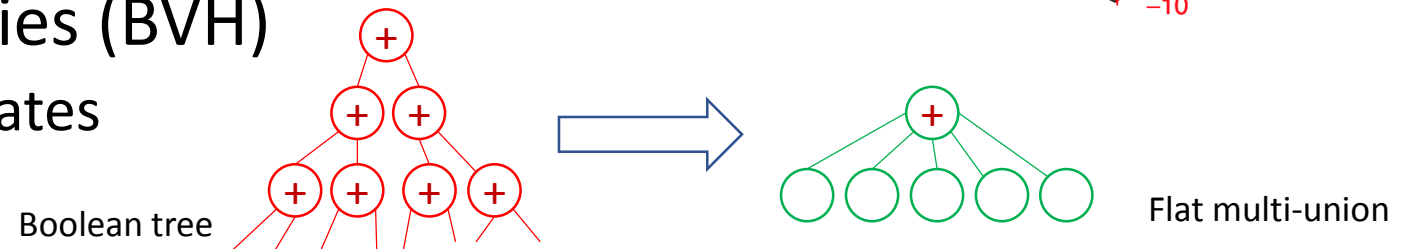
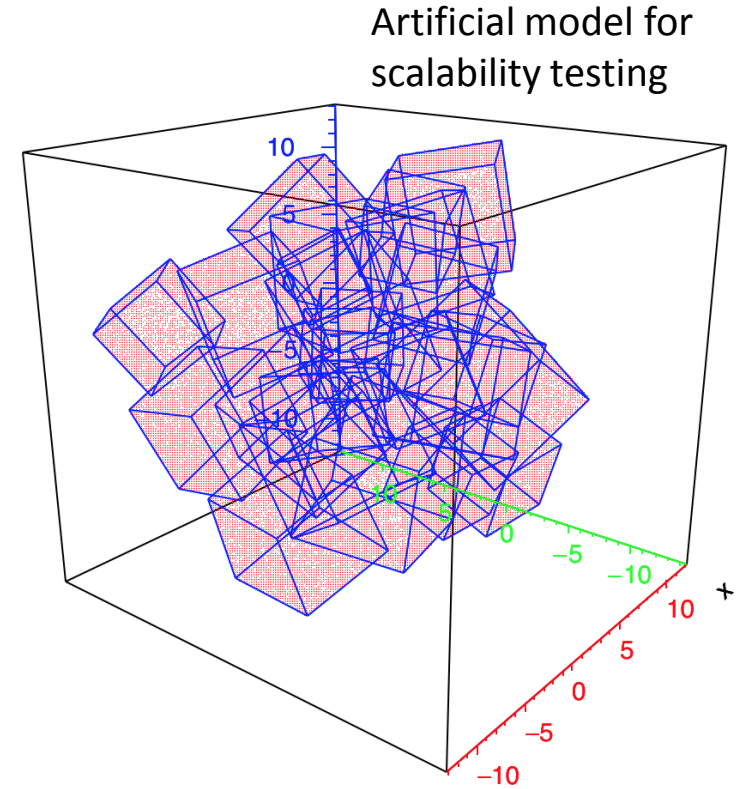
Tessellating other solids

- No benefits measured for simple solids (box, simple trapezoids)
 - Faster without loops
- Other solids already vectorized in the scalar case -> marginal gain
 - Polyhedron, Trapezoid
- Important improvements observed for previously non vectorized cases
 - General planar trapezoids
 - Extruded solids



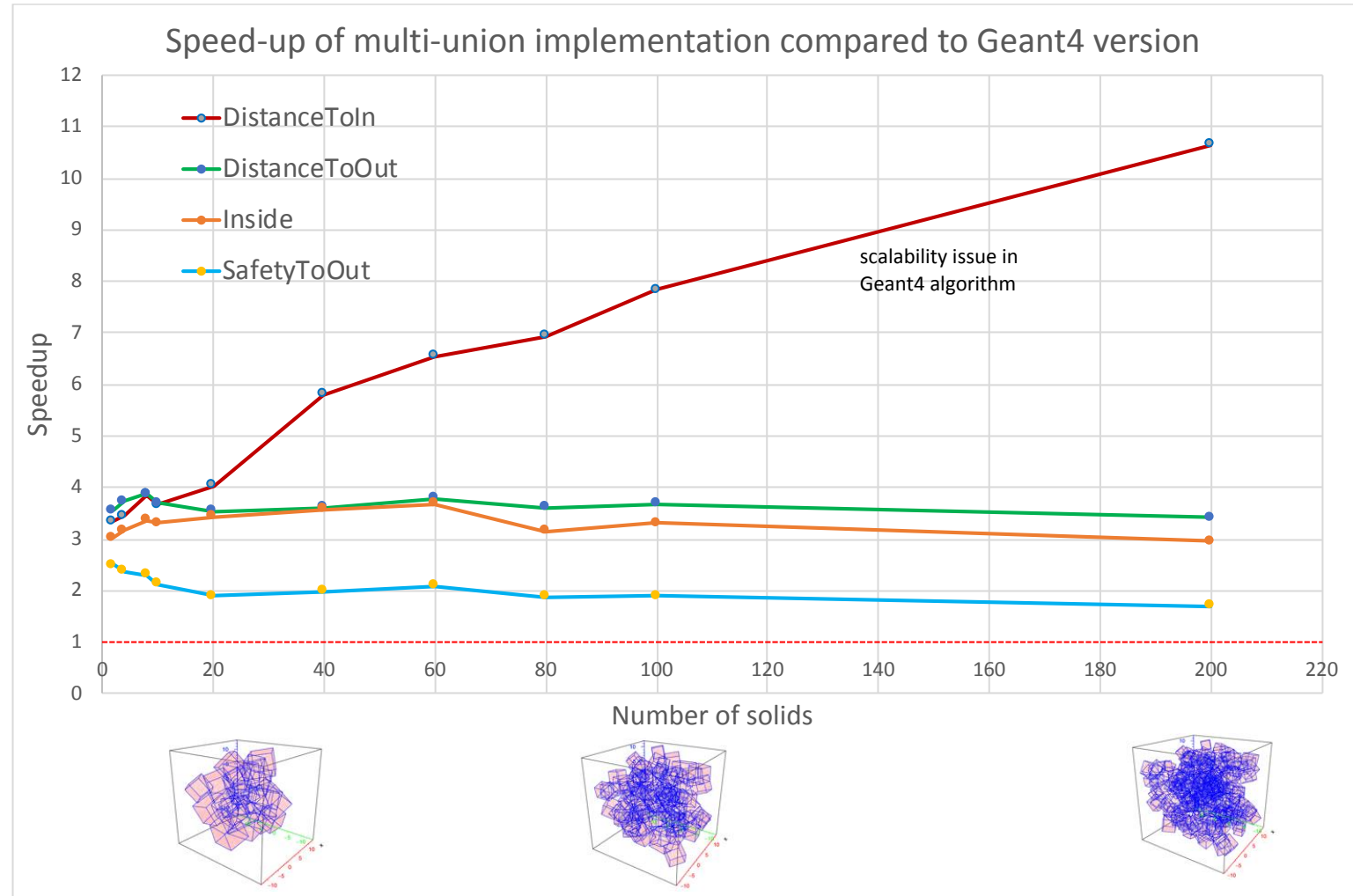
Other optimizations: multi-unions

- Boolean unions are represented as binary trees
 - Pathologically slow in simulation - too many individual checks
- Multi-unions representing nodes at same level
 - Optimization structures to limit candidates
 - Implementation in Geant4 using voxelization helper
- Re-implemented in VecGeom based on Bounding Volume Hierarchies (BVH)
 - Vectorized search of candidates



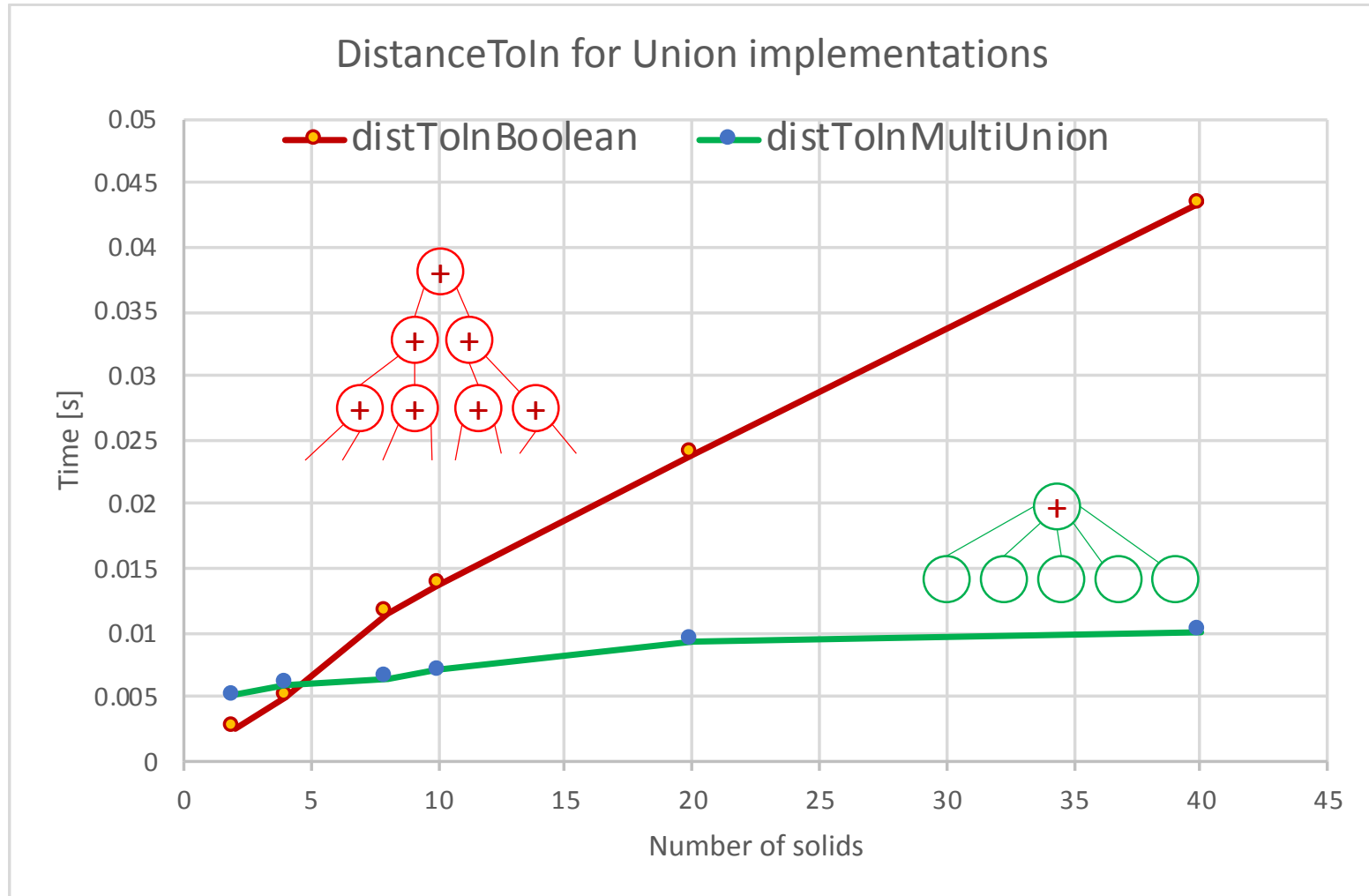
Benefits of using VecGeom multi-union

Large speed-up compared to corresponding implementation in Geant4 for up to several hundred components



Replacing Boolean union with VecGeom multi-union

- Implemented automatic conversion of Boolean volumes to the new multi-union structure
 - Much better performance for large number of components



Outlook

- VecGeom library became production quality
 - Can be used transparently with Geant4 simulations
- Vector optimizations in single particle mode very important for Geant4 simulation
 - Multi-faceted and multi-component solids with internal loops
 - Important performance gains in some cases, available in release 1.0 of VecGeom library
- Next steps: improvement 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)

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), R. Sehgal (BARC), S. Wenzel (CERN-ALICE), D. Savin (GSoC student)*
- *Repository for VecGeom*
 - <https://gitlab.cern.ch/VecGeom/VecGeom>
- *JIRA issue tracking tool*
 - <https://its.cern.ch/jira/projects/VECGEOM>



This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 654168.

