Geometry Parallel Session Summary

J. Apostolakis, G. Cosmo

26 September 2013, Geant4 Collaboration Meeting, Sevilla

The talks

- Unified Solids: Status report - Marek
  - developments in the improved library of Solids
- Optimizing geometry methods for SIMD - Sandro
  - Primary target: vector particle tracking
- “Workspaces”
  - Thread Building Blocks, Task parallelism and Geant4-MT
- Use of precise safety by EM processes & open issues

Current status of the development of the Unified Solids library

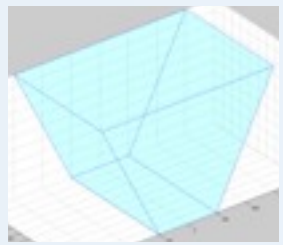
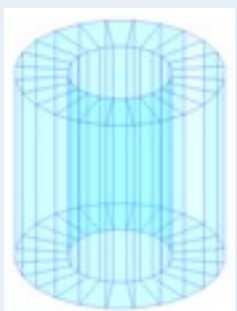
Marek Gayer

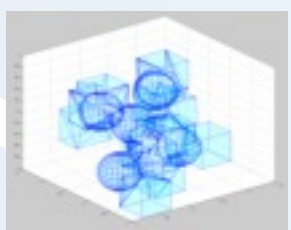
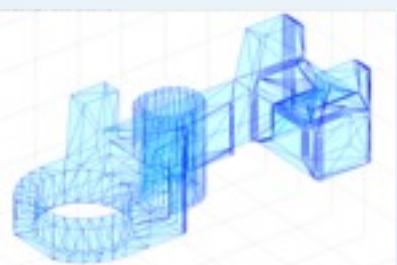
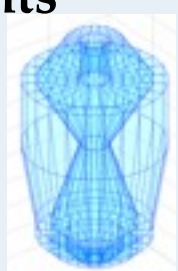
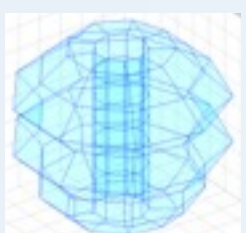
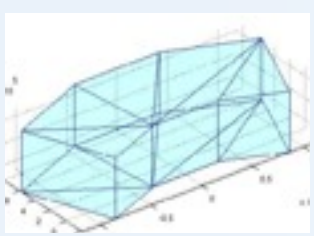
CERN PH/SFT

Solids implemented so far

- Box
- Orb
- Trapezoid
- Sphere (+ sphere section)
- Tube (+ cylindrical section)
- Cone (+ conical section)
- Generic trapezoid
- Tetrahedron
- Arbitrary Trapezoid (ongoing)
- **Multi-Union**
- **Tessellated Solid**
- **Polycone**
- **Polyhedra**
- Extruded solid (ongoing)

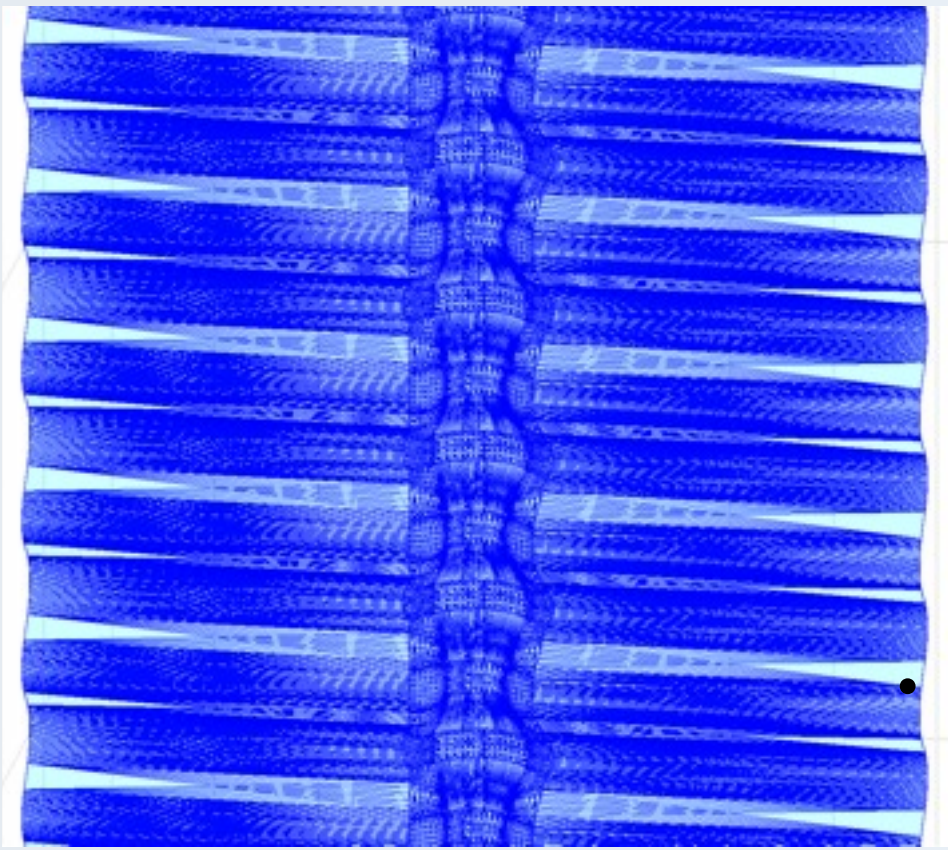
BIG improvements

Fast Tessellated Solid

- Example of LHCb Velo foil



Method	Speedup
Inside	2423x
DistanceToIn	1334x
DistanceToOut	1976x
Information	**Value**
Number of facets	**164.149**
Number of voxels	**158.928**
Memory saved	22%

- **Speedup factor ~10x** additional improvement since the previous Geant4 collaboration meeting

Refactoring and Optimizing Geometry Routines for (SIMD) Vector Particle Processing

-- goals and status report --

Sandro Wenzel / CERN-PH-SFT

(for the "Geant-Vector Prototype" team)

Geant4 collaboration meeting, Sevilla, 24.09.2013

The programming model

In order to use SIMD CPU capabilities, need to emit special assembly instructions ("add" versus "vaddp") to the hardware.

Multiple options exist:

- **"autovectorization:"** Let the compiler figure this out himself (without code changes).
  - Pro: best option for portability and maintenance
  - Cons: This currently never works (but in a few cases)....

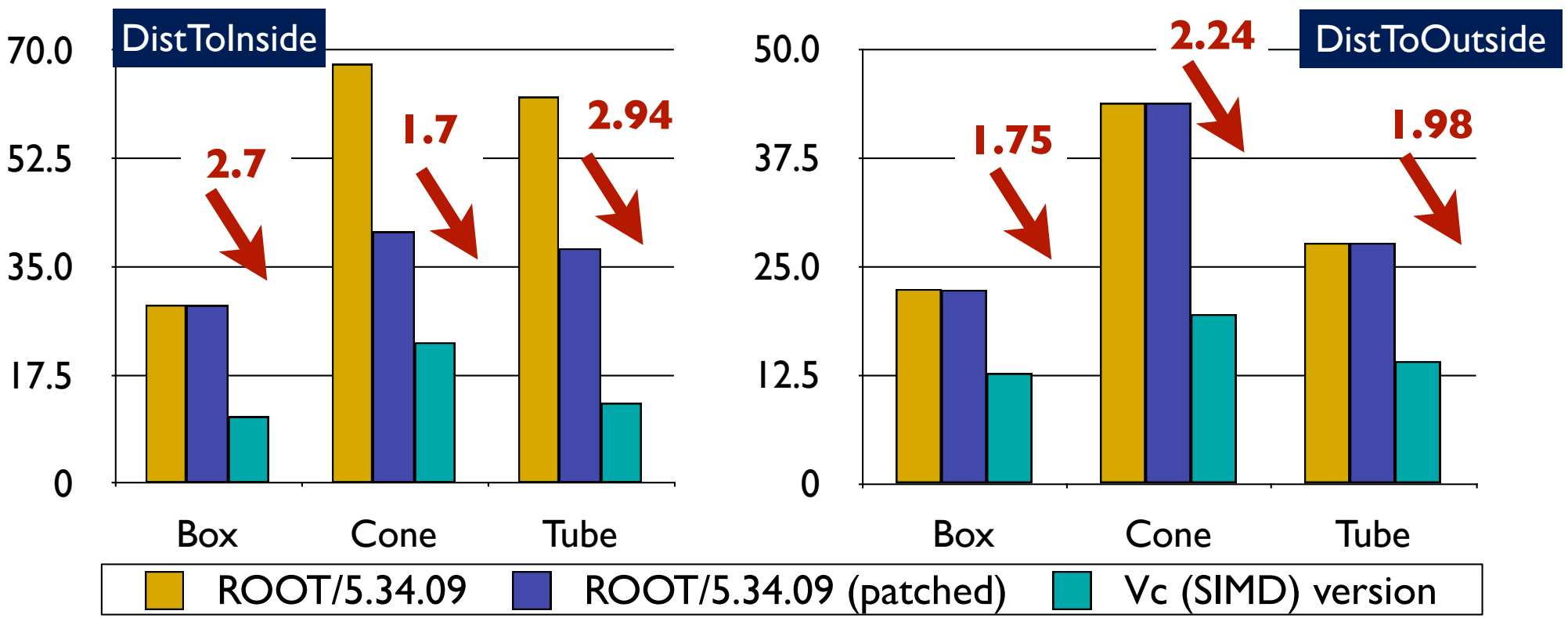
- **explicit vector oriented programming via intrinsics:** Manually instruct the compiler to use vector instructions:
  - at the lowest level: intrinsics
  - at higher level: template based APIs that hide low level details like the **Vc library**
  - Pro: good performance, portability, only little platform dependency (templates!)
  - Cons: requires some code changes, refactoring of code

code.compeng.uni-frankfurt.de/projects/Vc

- **language extensions**, such as Intel Cilk Plus Array notation
  - similiar to point 2, investigated but not covered in this talk

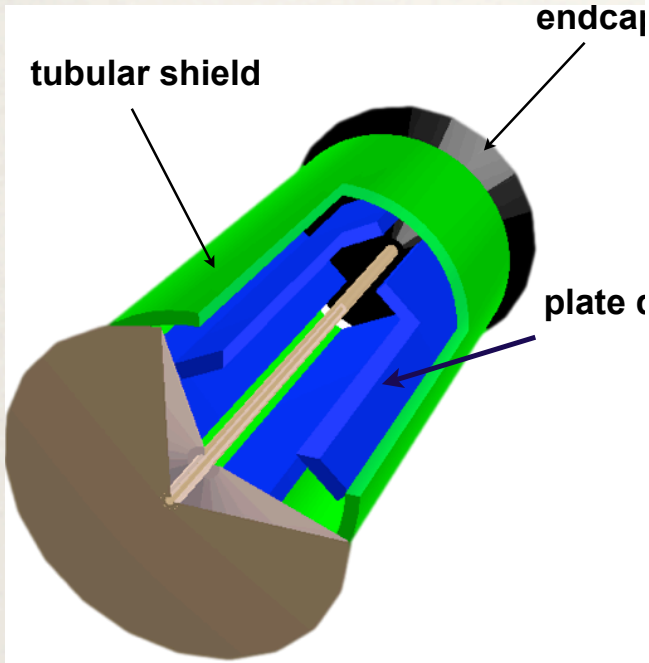
Status of simple shape/algorithm investigations

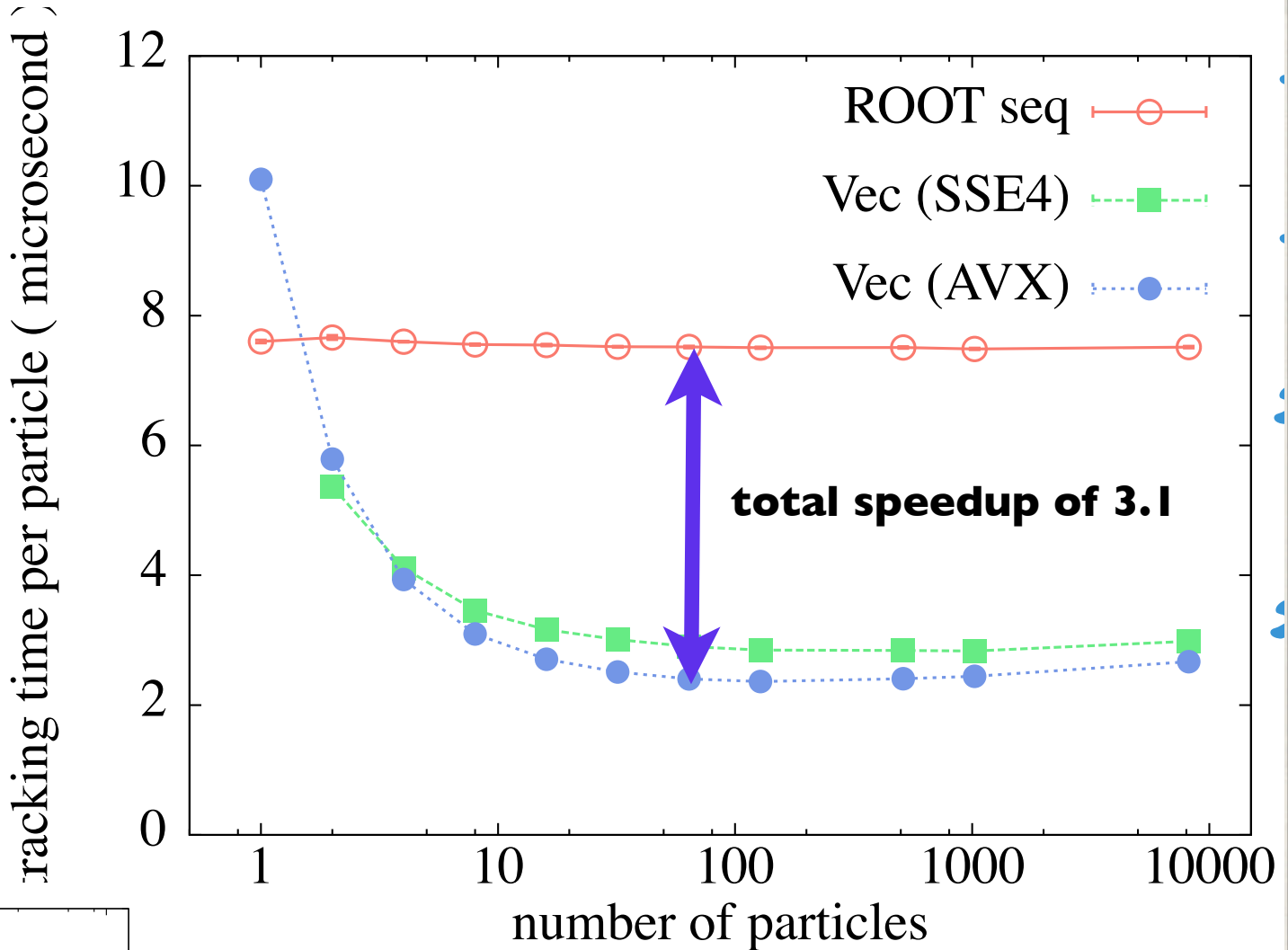
- provided vector interfaces to all shapes and optimized code to simple shapes for functions
  - **" DistToInside", "DistToOutside", "Safety", "IsInside/Contains"**
  - here: using the ROOT shapes (but USolids will come)
  - obtained good experience and results using the Vc programming model
- For simple shapes the **performance gains match our expectations**

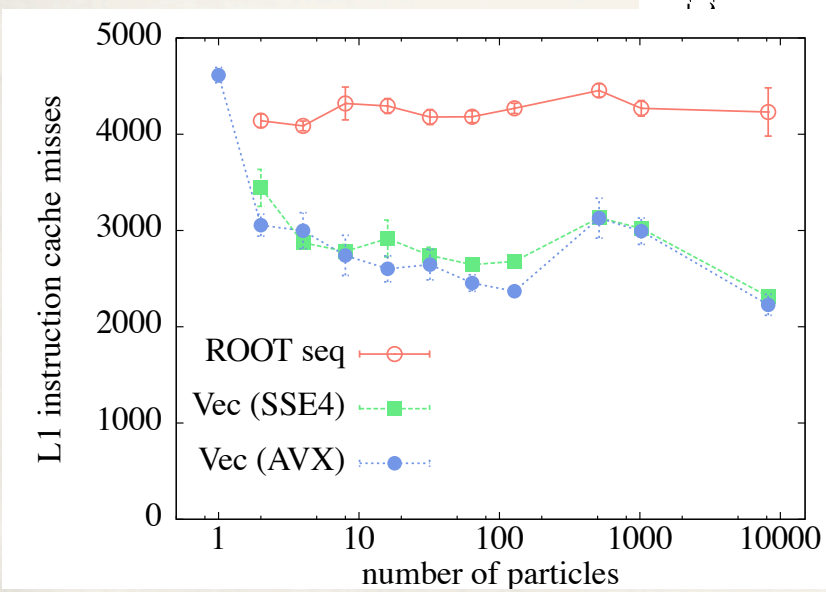


comparison of processing times for 1024 particles (AVX instructions), times in microseconds

Navigation







Toy Geometry with 8 volumes

Promising first result -

Vectorizing Solids: Summary/Next

Summary

- vectorization is not threading and needs to be cared for additionally!
- a vector/basket centric architecture allows to make use of SIMD instruction sets, needs less functions calls, and is more instruction cache friendly
- provided a first refactored vector API in ROOT geometry/ navigation library and showed good performance gains for individual as well as complex algorithms on commodity hardware

Text

- Very good experience with explicit vector oriented programming model (Vc, Intel Cilk Plus Arrays)

Plans: Vectorize harder solids, and USolids library (2014), XeonPhi, full-flow of vectors in Vector Prototype (Geant-V)

Workspaces: Extending G4MT for task parallelism

From threads to tasks

- HEP experiments have chosen Intel Thread Building Blocks™ (TBB) as the foundation library for parallelization for production in 2014/15
- Is Geant4-MT compatible with tasks / TBB ?
  - Yes - Andrea demonstrated first, proof-of-principle prototype (Dec 2012)
- Motivated by extension of MT to
-

Differences between TBB and G4MT

Multi-threading

- Explicit spawning and control of threads
  - G4MTRunManager
- All threads initialized at same time,
  - before any event is simulated.

Task-parallelism

- Control is by the task system
  - e.g. TBB's runtime system
  - It creates and reuses thread
- Lazy initialization
  - initialize context only when needed
- Switching to a new task is fast

Multi Scattering and Geometry

Multiple Scatter & Geometry

- Isotropic safety has important uses in Mult. Scat.
  - Its value is used to constrain lateral displacement
  - It is consulted several times a step
- Moving displacement to AlongStep
  - Creates issue due to Transportation/Navigator - John Ap. to find & fix.
- Issue with revision by SteppingManager
  - 0 safety is changed to 0.5 * kCarTolerance
  - request that this is reverted:
  - Zero safety = probably tracks is at surface.
- Potential to revise GS MSc model
  - which will then require 2 geometry inquiries per step
  - become more similar to Penelope.