Bounding Volume Hierarchy Acceleration in Geant4

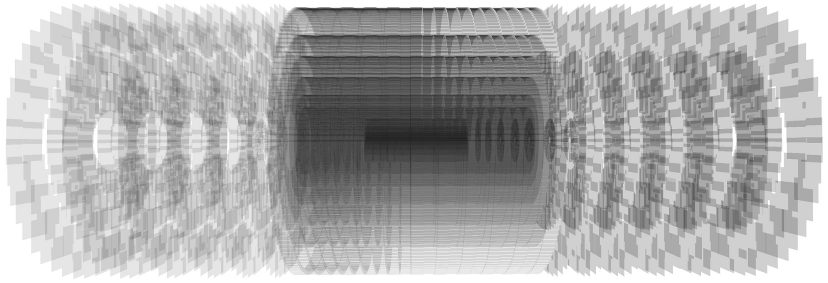
G. Amadio

29 Sep 2022

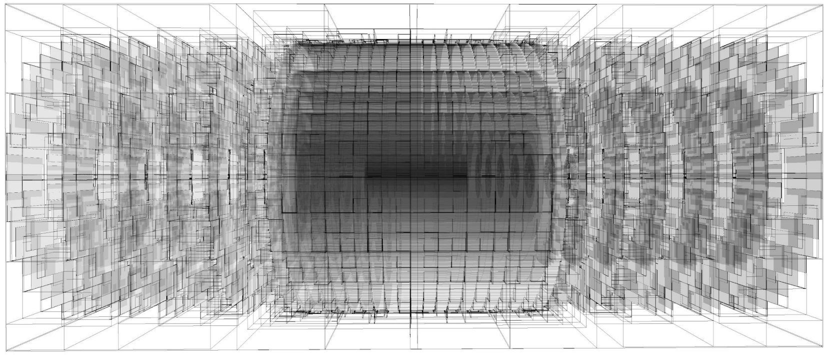
27th Geant4 Collaboration Meeting, Rennes, France

Bounding Volume Hierarchy Acceleration

- First acceleration structure in VecGeom with support for both CPU and GPU
- Optimized for HEP geometry corner cases, using Surface Area Heuristic (SAH) algorithm for better quality BVH (S. Hageböck)
- Ported to Geant4 as potential alternative to G4VoxelNavigator

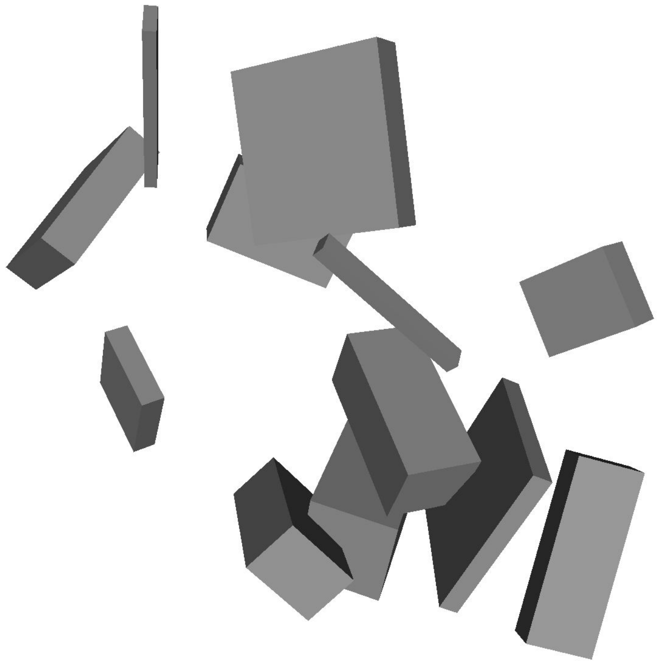


Track ML Geometry



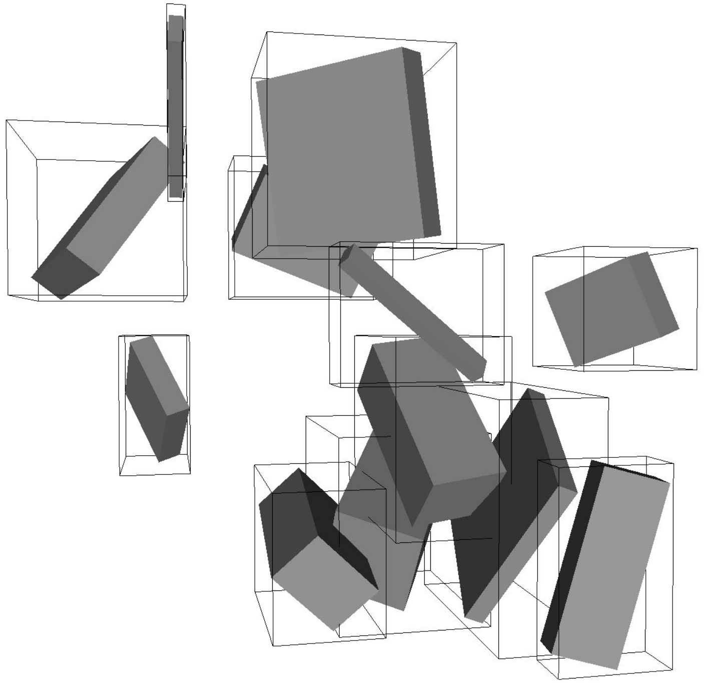
Simple Prototype in OpenGL

- **Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin**
- Compute AABB for each box
- Construct top-down BVH
- Shoot rays and check with which boxes it intersects
- Optionally highlight all intersections along the ray, for debugging/testing



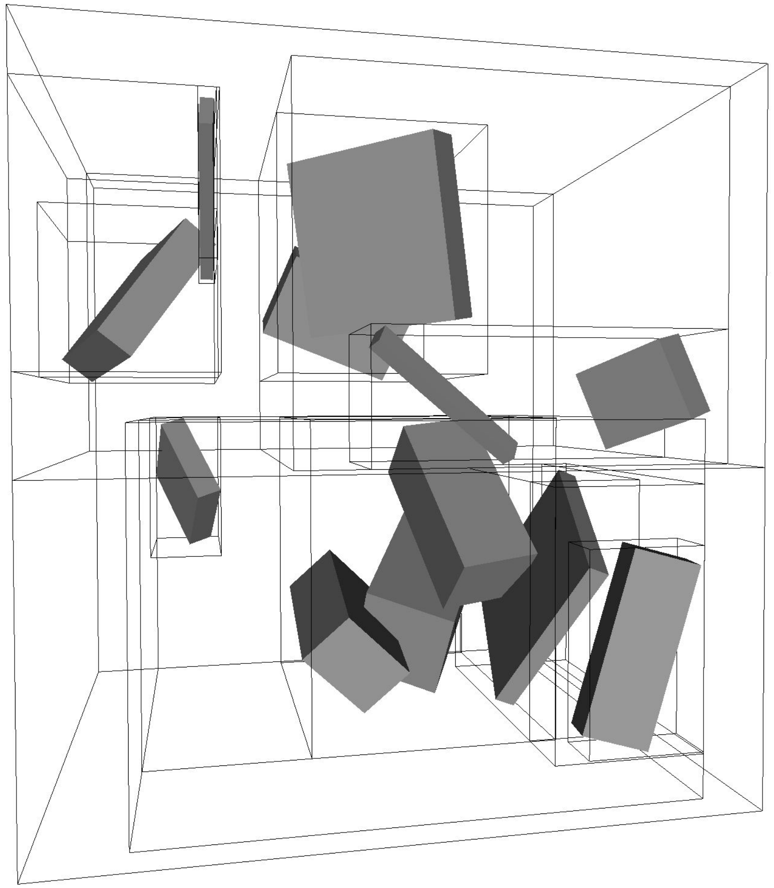
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- **Compute AABB for each box**
- Construct top-down BVH
- Shoot rays and check with which boxes it intersects
- Optionally highlight all intersections along the ray, for debugging/testing



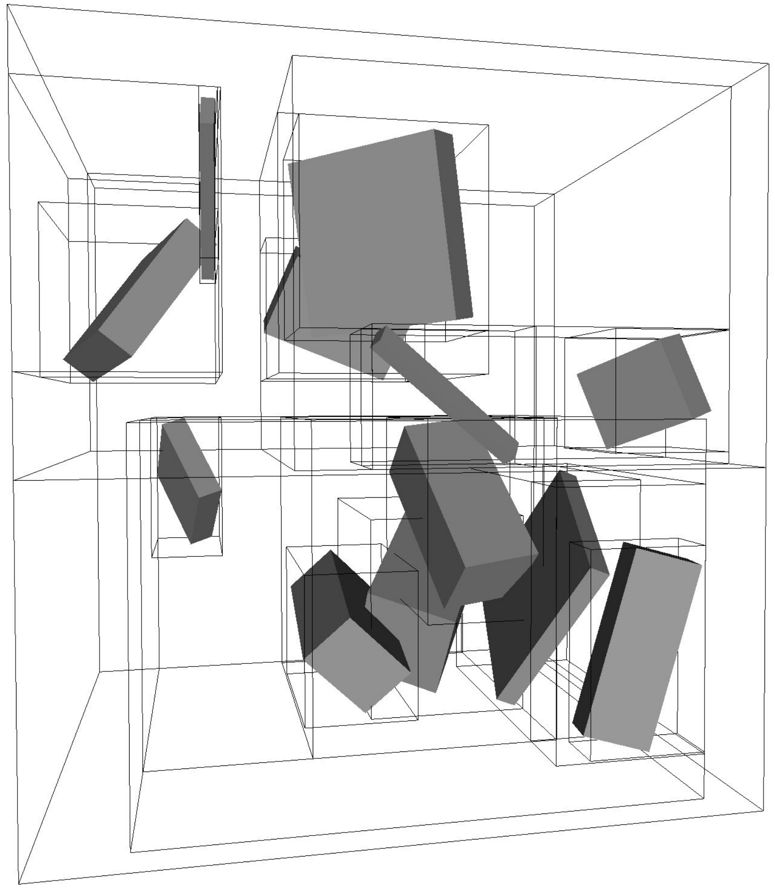
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- Compute AABB for each box
- **Construct top-down BVH**
- Shoot rays and check with which boxes it intersects
- Optionally highlight all intersections along the ray, for debugging/testing



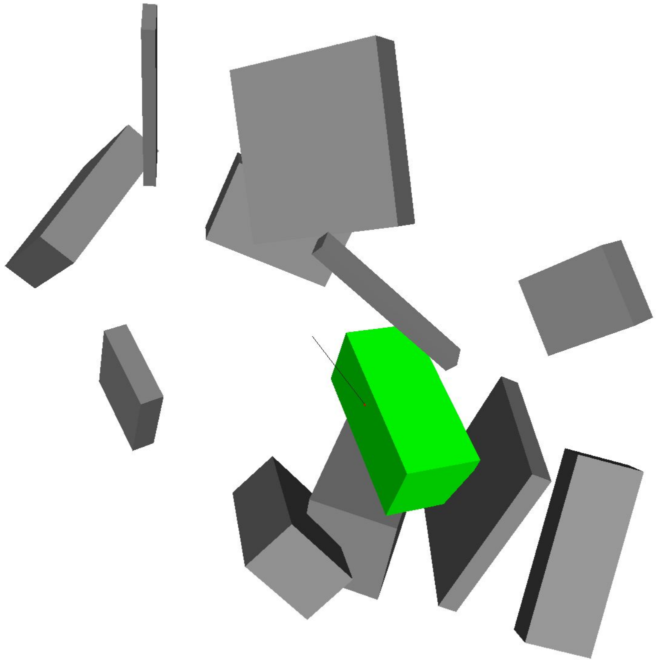
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- Compute AABB for each box
- **Construct top-down BVH**
- Shoot rays and check with which boxes it intersects
- Optionally highlight all intersections along the ray, for debugging/testing



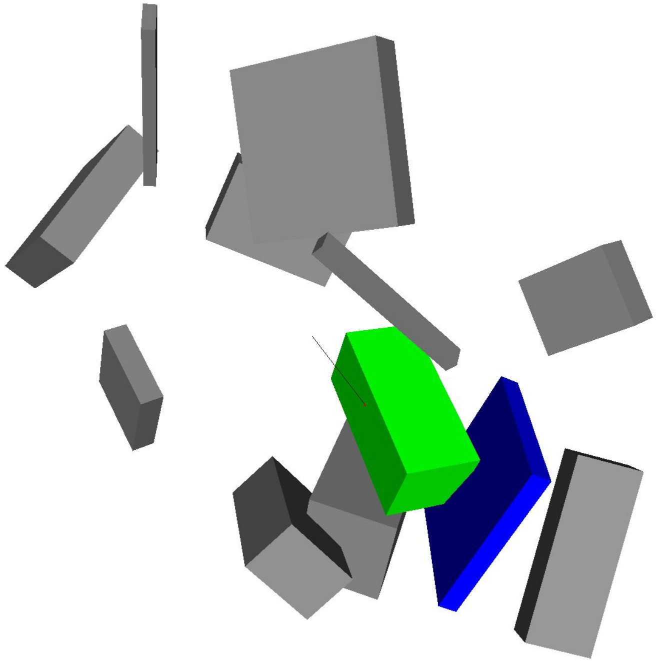
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- Compute AABB for each box
- Construct top-down BVH
- **Shoot rays and check with which boxes it intersects**
- Optionally highlight all intersections along the ray, for debugging/testing



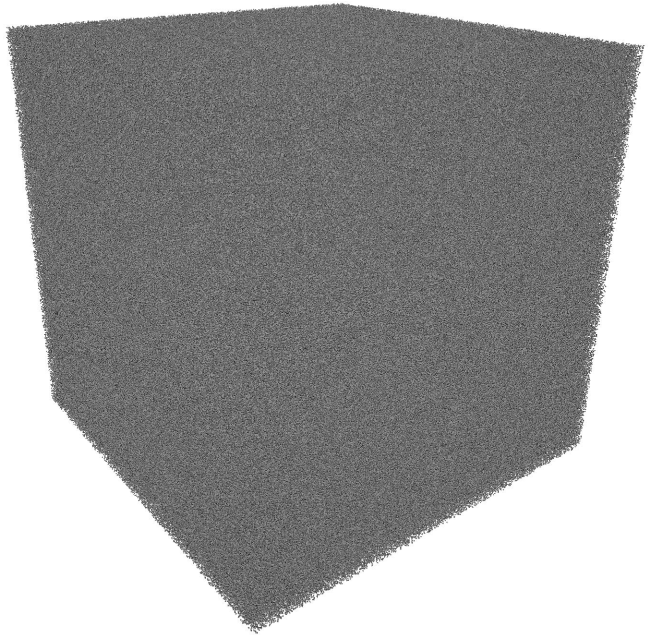
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- Compute AABB for each box
- Construct top-down BVH
- Shoot rays and check with which boxes it intersects
- **Optionally highlight all intersections along the ray, for debugging/testing**



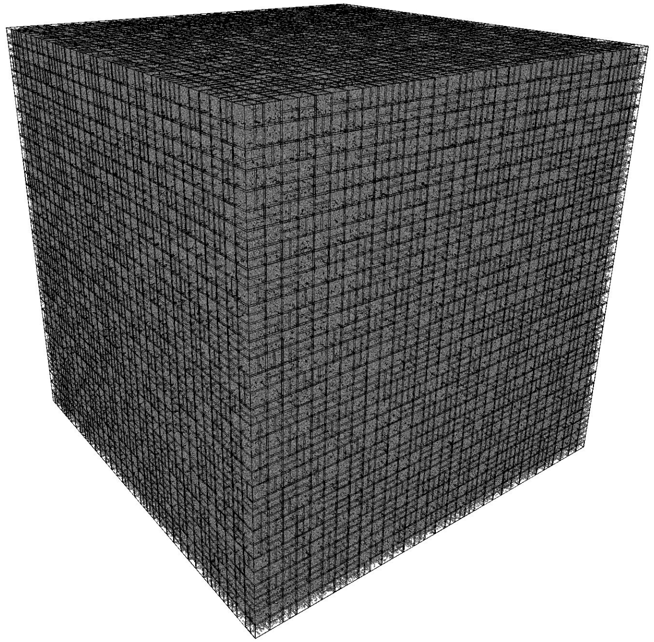
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- Compute AABB for each box
- Construct top-down BVH
- Shoot rays and check with which boxes it intersects
- Optionally highlight all intersections along the ray, for debugging/testing
- Works with many volumes
  - 10^7 shown on the right

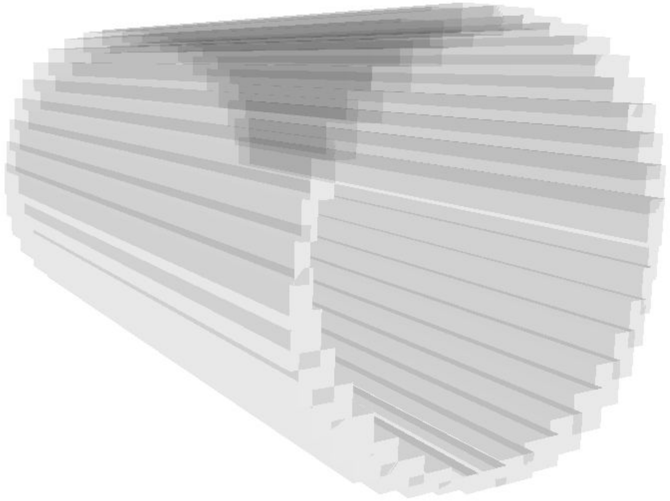


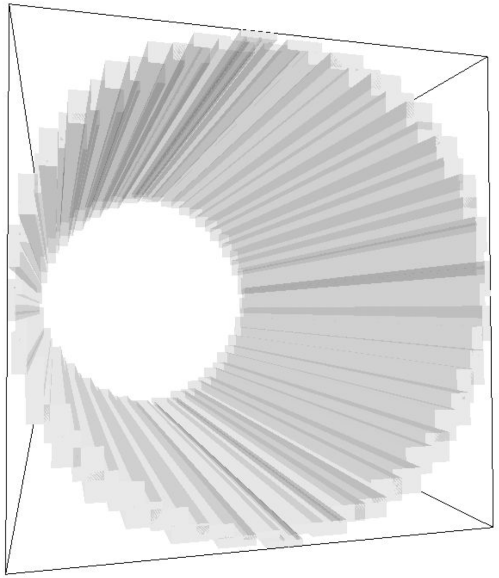
Simple Prototype in OpenGL

- Initialization: boxes of random sizes, positions, and orientations inside a unit cube centered at the origin
- Compute AABB for each box
- Construct top-down BVH
- Shoot rays and check with which boxes it intersects
- Optionally highlight all intersections along the ray, for debugging/testing
- Works with many volumes
    - **10^7 shown on the right**

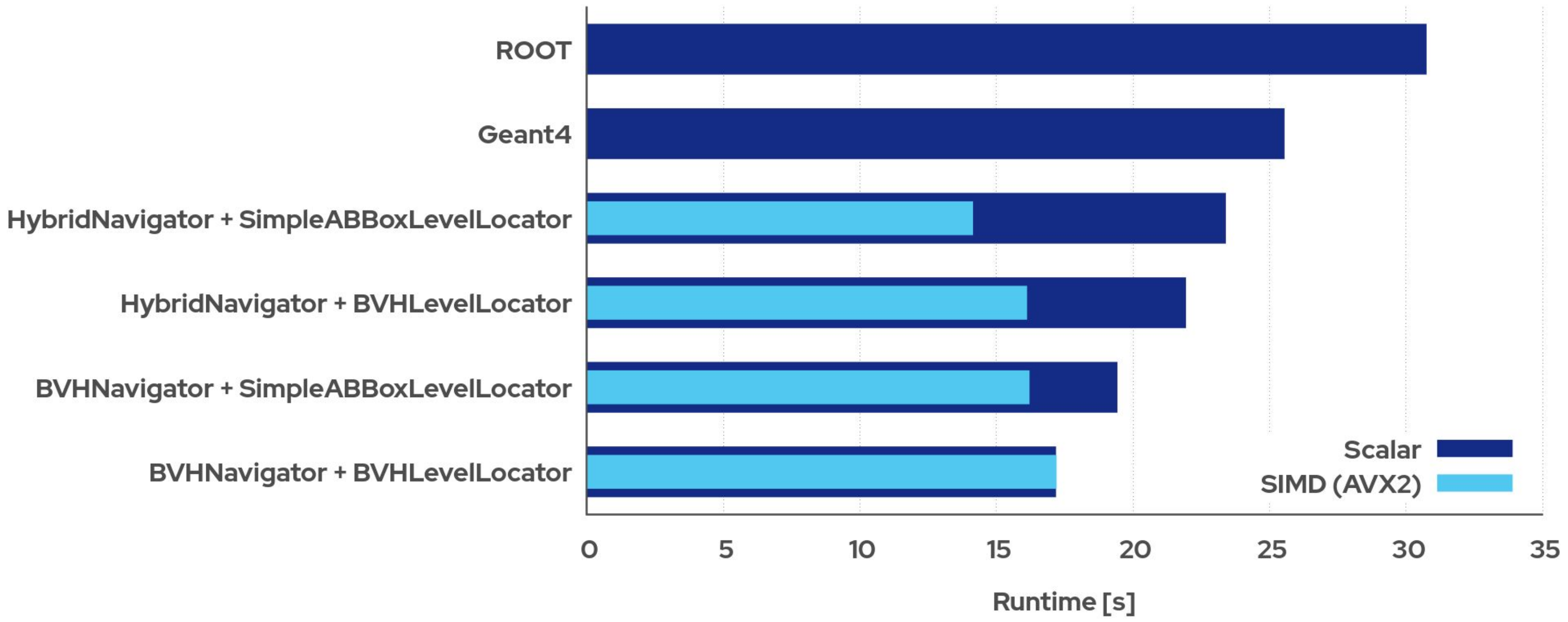


Surface Area Heuristic (SAH) Algorithm


Initial splitting algorithm, no subdivisions.


SAH correctly computes subdivisions.

Performance of BVH Navigator in VecGeom

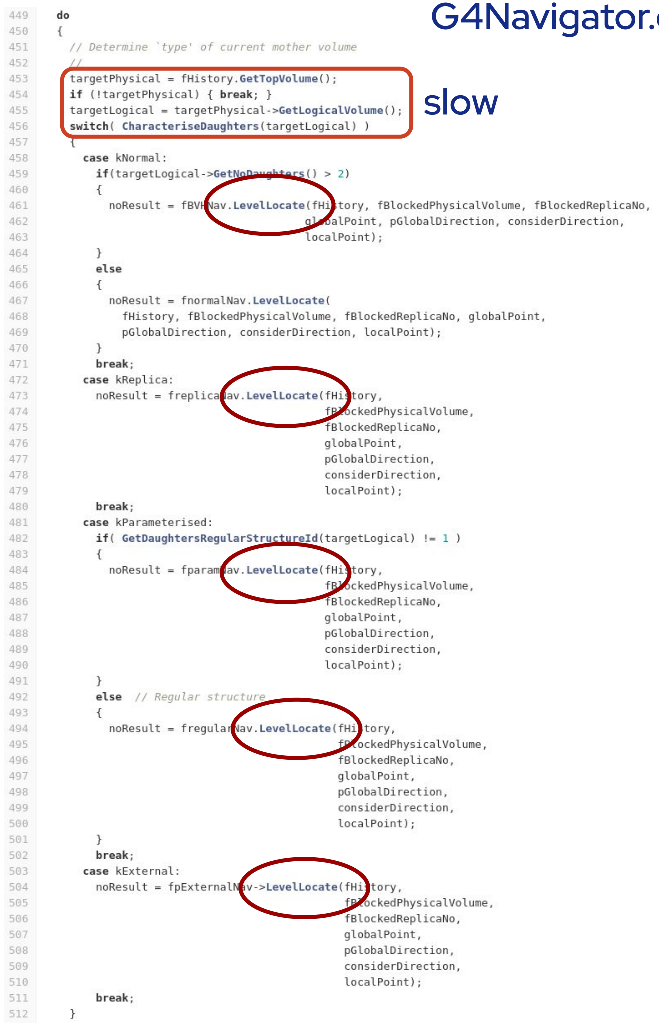


```
test/XRayBenchmarkFromROOTFile test/cms2015.root MUON y 500
```

G4Navigation needs Code Refactoring

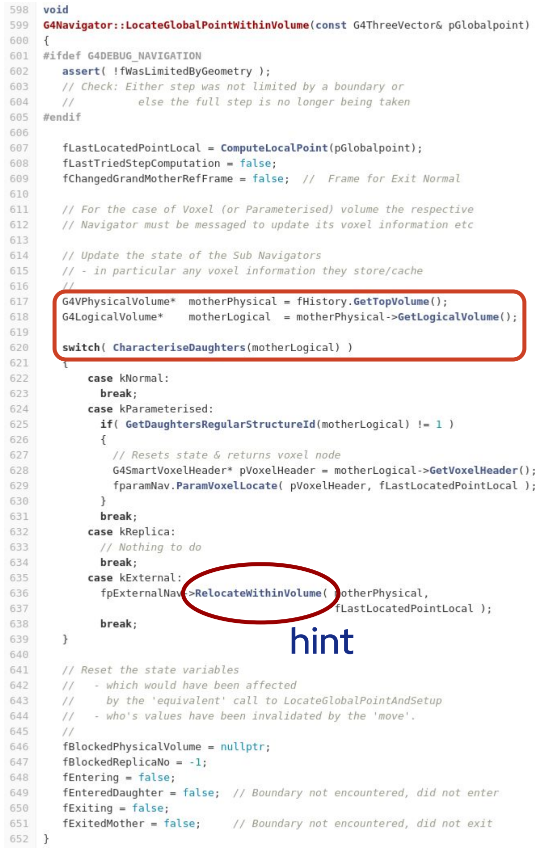
This is screaming for an abstract base class, but due to this sort of switch statement appearing in several places in Geant4 geometry with slight variations, so it's not easy to refactor the code without invasive changes.

G4Navigator.cc



slow

```
do
{
  // Determine `type' of current mother volume
  //
  targetPhysical = fHistory.GetTopVolume();
  if (!targetPhysical) { break; }
  targetLogical = targetPhysical->GetLogicalVolume();
  switch( CharacteriseDaughters(targetLogical) )
  {
    case kNormal:
      if(targetLogical->GetNoDaughters() > 2)
      {
        noResult = fBVHNav.LevelLocate(fHistory, fBlockedPhysicalVolume, fBlockedReplicaNo,
                                       globalPoint, pGlobalDirection, considerDirection,
                                       localPoint);
      }
      else
      {
        noResult = fnormalNav.LevelLocate(
          fHistory, fBlockedPhysicalVolume, fBlockedReplicaNo, globalPoint,
          pGlobalDirection, considerDirection, localPoint);
      }
      break;
    case kReplica:
      noResult = freplicaNav.LevelLocate(fHistory,
                                         fBlockedPhysicalVolume,
                                         fBlockedReplicaNo,
                                         globalPoint,
                                         pGlobalDirection,
                                         considerDirection,
                                         localPoint);
      break;
    case kParameterised:
      if( GetDaughtersRegularStructureId(targetLogical) != 1 )
      {
        noResult = fparamNav.LevelLocate(fHistory,
                                         fBlockedPhysicalVolume,
                                         fBlockedReplicaNo,
                                         globalPoint,
                                         pGlobalDirection,
                                         considerDirection,
                                         localPoint);
      }
      else  // Regular structure
      {
        noResult = fregularNav.LevelLocate(fHistory,
                                           fBlockedPhysicalVolume,
                                           fBlockedReplicaNo,
                                           globalPoint,
                                           pGlobalDirection,
                                           considerDirection,
                                           localPoint);
      }
      break;
    case kExternal:
      noResult = fpExternalNav->LevelLocate(fHistory,
                                            fBlockedPhysicalVolume,
                                            fBlockedReplicaNo,
                                            globalPoint,
                                            pGlobalDirection,
                                            considerDirection,
                                            localPoint);
      break;
  }
```



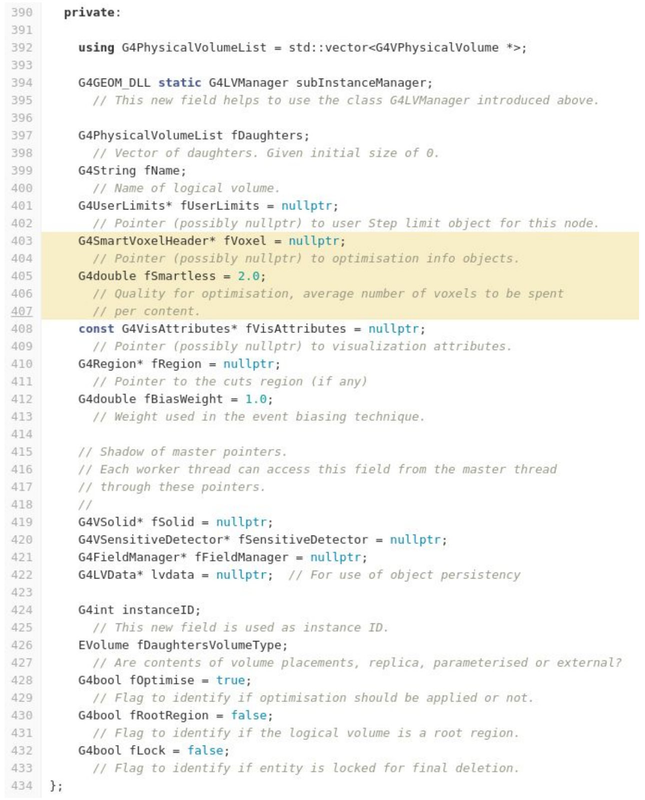
hint

```
void
G4Navigator::LocateGlobalPointWithinVolume(const G4ThreeVector& pGlobalpoint)
{
#ifdef G4DEBUG_NAVIGATION
   assert( !fWasLimitedByGeometry );
   // Check: Either step was not limited by a boundary or
   //        else the full step is no longer being taken
#endif

   fLastLocatedPointLocal = ComputeLocalPoint(pGlobalpoint);
   fLastTriedStepComputation = false;
   fChangedGrandMotherRefFrame = false;  //  Frame for Exit Normal

   // For the case of Voxel (or Parameterised) volume the respective
   // Navigator must be messaged to update its voxel information etc

   // Update the state of the Sub Navigators
   // - in particular any voxel information they store/cache
   //
   G4VPhysicalVolume*  motherPhysical = fHistory.GetTopVolume();
   G4LogicalVolume*    motherLogical  = motherPhysical->GetLogicalVolume();

   switch( CharacteriseDaughters(motherLogical) )
   {
       case kNormal:
         break;
       case kParameterised:
         if( GetDaughtersRegularStructureId(motherLogical) != 1 )
         {
           // Resets state & returns voxel node
           G4SmartVoxelHeader* pVoxelHeader = motherLogical->GetVoxelHeader();
           fparamNav.ParamVoxelLocate( pVoxelHeader, fLastLocatedPointLocal );
         }
         break;
       case kReplica:
         // Nothing to do
         break;
       case kExternal:
         fpExternalNav->RelocateWithinVolume( motherPhysical,
                                              fLastLocatedPointLocal );
         break;
   }

   // Reset the state variables
   //   - which would have been affected
   //     by the 'equivalent' call to LocateGlobalPointAndSetup
   //   - who's values have been invalidated by the 'move'.
   //
   fBlockedPhysicalVolume = nullptr;
   fBlockedReplicaNo = -1;
   fEntering = false;
   fEnteredDaughter = false;  // Boundary not encountered, did not enter
   fExiting = false;
   fExitedMother = false;     // Boundary not encountered, did not exit
}
```

G4LogicalVolume + G4VoxelNavigation

G4LogicalVolume closely linked with **G4VoxelNavigator**. Needs a generalization of per-volume data for acceleration structures to be able to add BVH without extending **G4LogicalVolume** to also have a pointer to a BVH. Currently have a “BVHStore” so that no changes are needed in **G4LogicalVolume**.

G4LogicalVolume.hh



```cpp
  private:

    using G4PhysicalVolumeList = std::vector<G4VPhysicalVolume *>;

    G4GEOM_DLL static G4LVManager subInstanceManager;
      // This new field helps to use the class G4LVManager introduced above.

    G4PhysicalVolumeList fDaughters;
      // Vector of daughters. Given initial size of 0.
    G4String fName;
      // Name of logical volume.
    G4UserLimits* fUserLimits = nullptr;
      // Pointer (possibly nullptr) to user Step limit object for this node.
    G4SmartVoxelHeader* fVoxel = nullptr;
      // Pointer (possibly nullptr) to optimisation info objects.
    G4double fSmartless = 2.0;
      // Quality for optimisation, average number of voxels to be spent
      // per content.
    const G4VisAttributes* fVisAttributes = nullptr;
      // Pointer (possibly nullptr) to visualization attributes.
    G4Region* fRegion = nullptr;
      // Pointer to the cuts region (if any)
    G4double fBiasWeight = 1.0;
      // Weight used in the event biasing technique.

    // Shadow of master pointers.
    // Each worker thread can access this field from the master thread
    // through these pointers.
    //
    G4VSolid* fSolid = nullptr;
    G4VSensitiveDetector* fSensitiveDetector = nullptr;
    G4FieldManager* fFieldManager = nullptr;
    G4LVData* lvdata = nullptr;  // For use of object persistency

    G4int instanceID;
      // This new field is used as instance ID.
    EVolume fDaughtersVolumeType;
      // Are contents of volume placements, replica, parameterised or external?
    G4bool fOptimise = true;
      // Flag to identify if optimisation should be applied or not.
    G4bool fRootRegion = false;
      // Flag to identify if the logical volume is a root region.
    G4bool fLock = false;
      // Flag to identify if entity is locked for final deletion.
};
```

Integration of BVH classes into Geant4

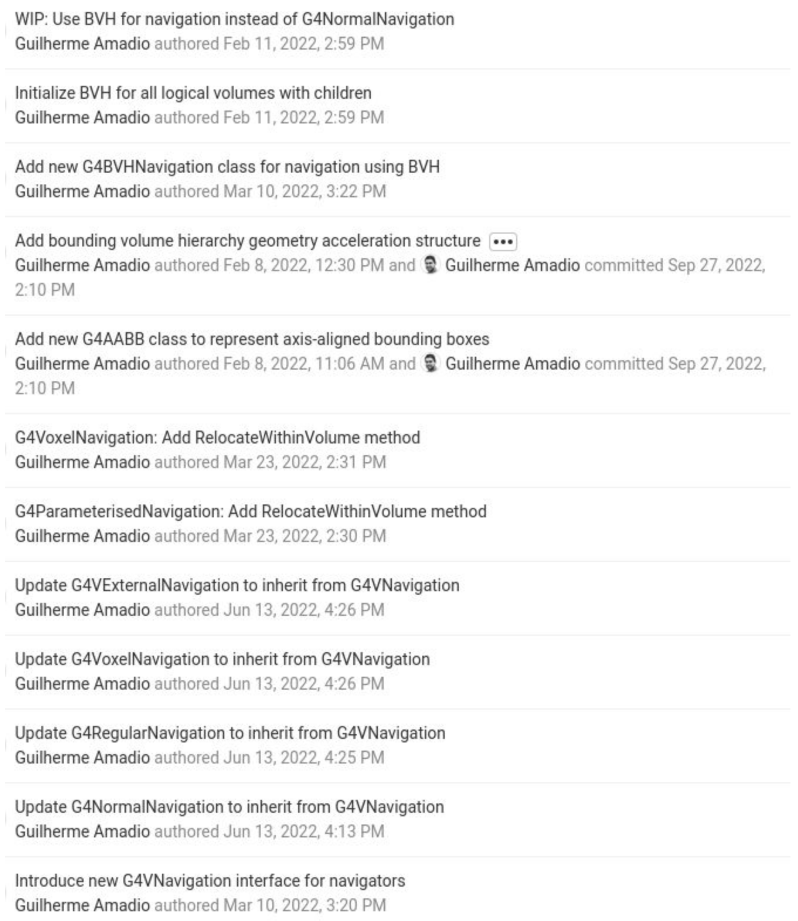
Added a new interface class called **G4VNavigation** with common interface for all navigator types and modified most navigators to inherit from it.

Adapted BVH implementation from VecGeom and added it to Geant4.

Added a few changes to forcibly replace G4VoxelNavigator with BVH for testing purposes.

Needs more work

Mostly fine to merge



WIP: Use BVH for navigation instead of G4NormalNavigation
Guilherme Amadio authored Feb 11, 2022, 2:59 PM

Initialize BVH for all logical volumes with children
Guilherme Amadio authored Feb 11, 2022, 2:59 PM

Add new G4BVHNavigation class for navigation using BVH
Guilherme Amadio authored Mar 10, 2022, 3:22 PM

Add bounding volume hierarchy geometry acceleration structure •••
Guilherme Amadio authored Feb 8, 2022, 12:30 PM and Guilherme Amadio committed Sep 27, 2022, 2:10 PM

Add new G4AABB class to represent axis-aligned bounding boxes
Guilherme Amadio authored Feb 8, 2022, 11:06 AM and Guilherme Amadio committed Sep 27, 2022, 2:10 PM

G4VoxelNavigation: Add RelocateWithinVolume method
Guilherme Amadio authored Mar 23, 2022, 2:31 PM

G4ParameterisedNavigation: Add RelocateWithinVolume method
Guilherme Amadio authored Mar 23, 2022, 2:30 PM

Update G4VExternalNavigation to inherit from G4VNavigation
Guilherme Amadio authored Jun 13, 2022, 4:26 PM

Update G4VoxelNavigation to inherit from G4VNavigation
Guilherme Amadio authored Jun 13, 2022, 4:26 PM

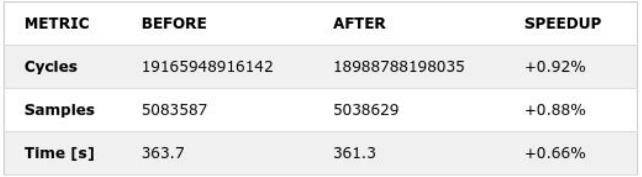
Update G4RegularNavigation to inherit from G4VNavigation
Guilherme Amadio authored Jun 13, 2022, 4:25 PM

Update G4NormalNavigation to inherit from G4VNavigation
Guilherme Amadio authored Jun 13, 2022, 4:13 PM

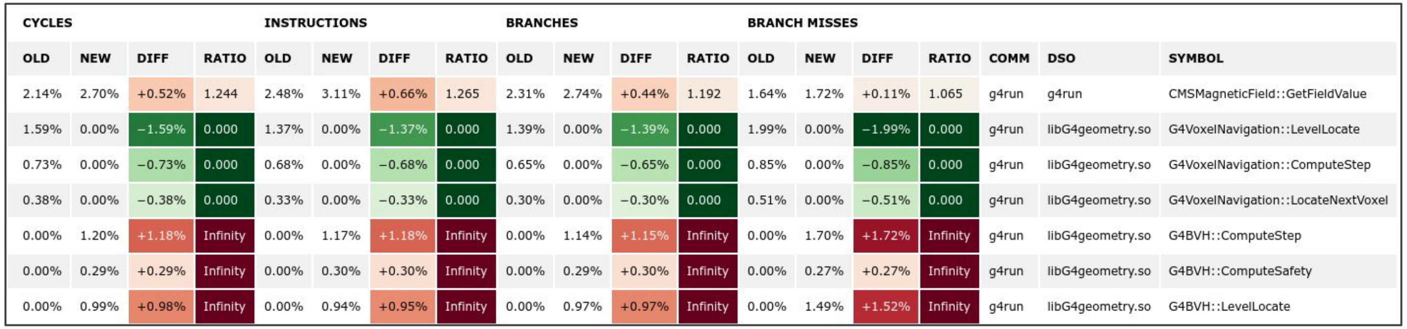
Introduce new G4VNavigation interface for navigators
Guilherme Amadio authored Mar 10, 2022, 3:20 PM

Validation and Performance of BVH in Geant4

- Validated by tracing particles from center of CMS geometry outward in all directions, and checking that steps and safeties are identical to the current navigators (Normal & Voxel)
- Since full integration into Geant4 requires invasive changes in multiple places in Geant4 geometry classes, this task has been given a lower priority
- Performance comparable to G4VoxelNavigator for a CMS simulation (speedup of <1%). Tracing geantinos is 8.4% faster with BVH (init excluded)
- Memory cost: 452MB → 456MB for CMS

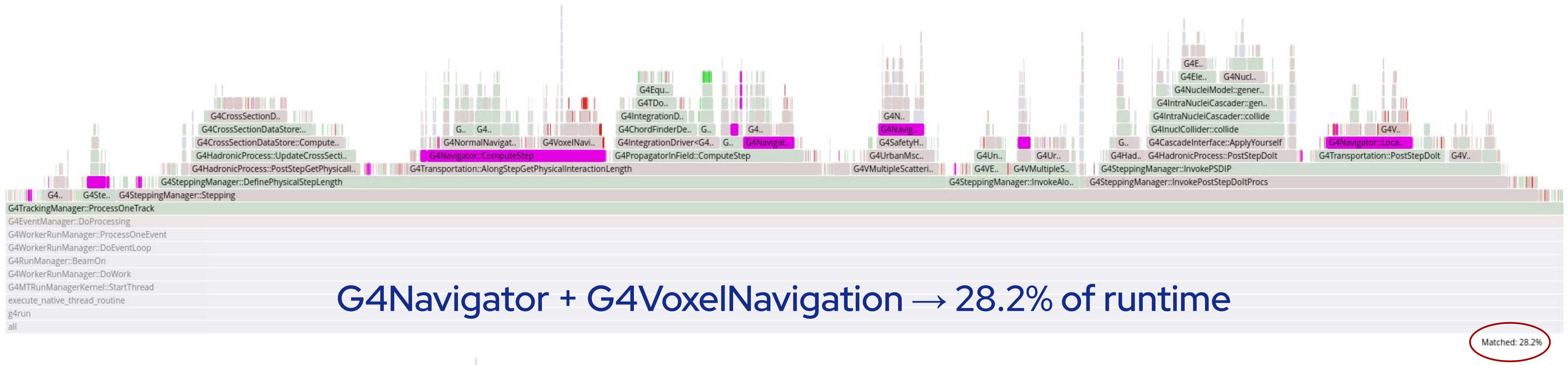


METRIC	BEFORE	AFTER	SPEEDUP
Cycles	19165948916142	18988788198035	+0.92%
Samples	5083587	5038629	+0.88%
Time [s]	363.7	361.3	+0.66%

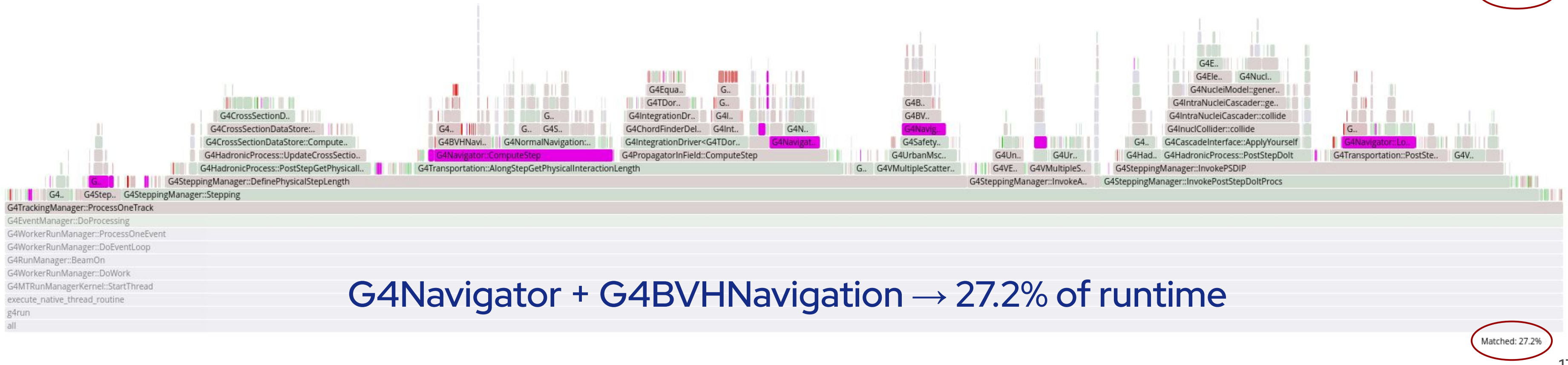


CYCLES				INSTRUCTIONS				BRANCHES				BRANCH MISSES						
OLD	NEW	DIFF	RATIO	OLD	NEW	DIFF	RATIO	OLD	NEW	DIFF	RATIO	OLD	NEW	DIFF	RATIO	COMM	DSO	SYMBOL
2.14%	2.70%	+0.52%	1.244	2.48%	3.11%	+0.66%	1.265	2.31%	2.74%	+0.44%	1.192	1.64%	1.72%	+0.11%	1.065	g4run	g4run	CMSMagneticField::GetFieldValue
1.59%	0.00%	−1.59%	0.000	1.37%	0.00%	−1.37%	0.000	1.39%	0.00%	−1.39%	0.000	1.99%	0.00%	−1.99%	0.000	g4run	libG4geometry.so	G4VoxelNavigation::LevelLocate
0.73%	0.00%	−0.73%	0.000	0.68%	0.00%	−0.68%	0.000	0.65%	0.00%	−0.65%	0.000	0.85%	0.00%	−0.85%	0.000	g4run	libG4geometry.so	G4VoxelNavigation::ComputeStep
0.38%	0.00%	−0.38%	0.000	0.33%	0.00%	−0.33%	0.000	0.30%	0.00%	−0.30%	0.000	0.51%	0.00%	−0.51%	0.000	g4run	libG4geometry.so	G4VoxelNavigation::LocateNextVoxel
0.00%	1.20%	+1.18%	Infinity	0.00%	1.17%	+1.18%	Infinity	0.00%	1.14%	+1.15%	Infinity	0.00%	1.70%	+1.72%	Infinity	g4run	libG4geometry.so	G4BVH::ComputeStep
0.00%	0.29%	+0.29%	Infinity	0.00%	0.30%	+0.30%	Infinity	0.00%	0.29%	+0.30%	Infinity	0.00%	0.27%	+0.27%	Infinity	g4run	libG4geometry.so	G4BVH::ComputeSafety
0.00%	0.99%	+0.98%	Infinity	0.00%	0.94%	+0.95%	Infinity	0.00%	0.97%	+0.97%	Infinity	0.00%	1.49%	+1.52%	Infinity	g4run	libG4geometry.so	G4BVH::LevelLocate

Flamegraphs

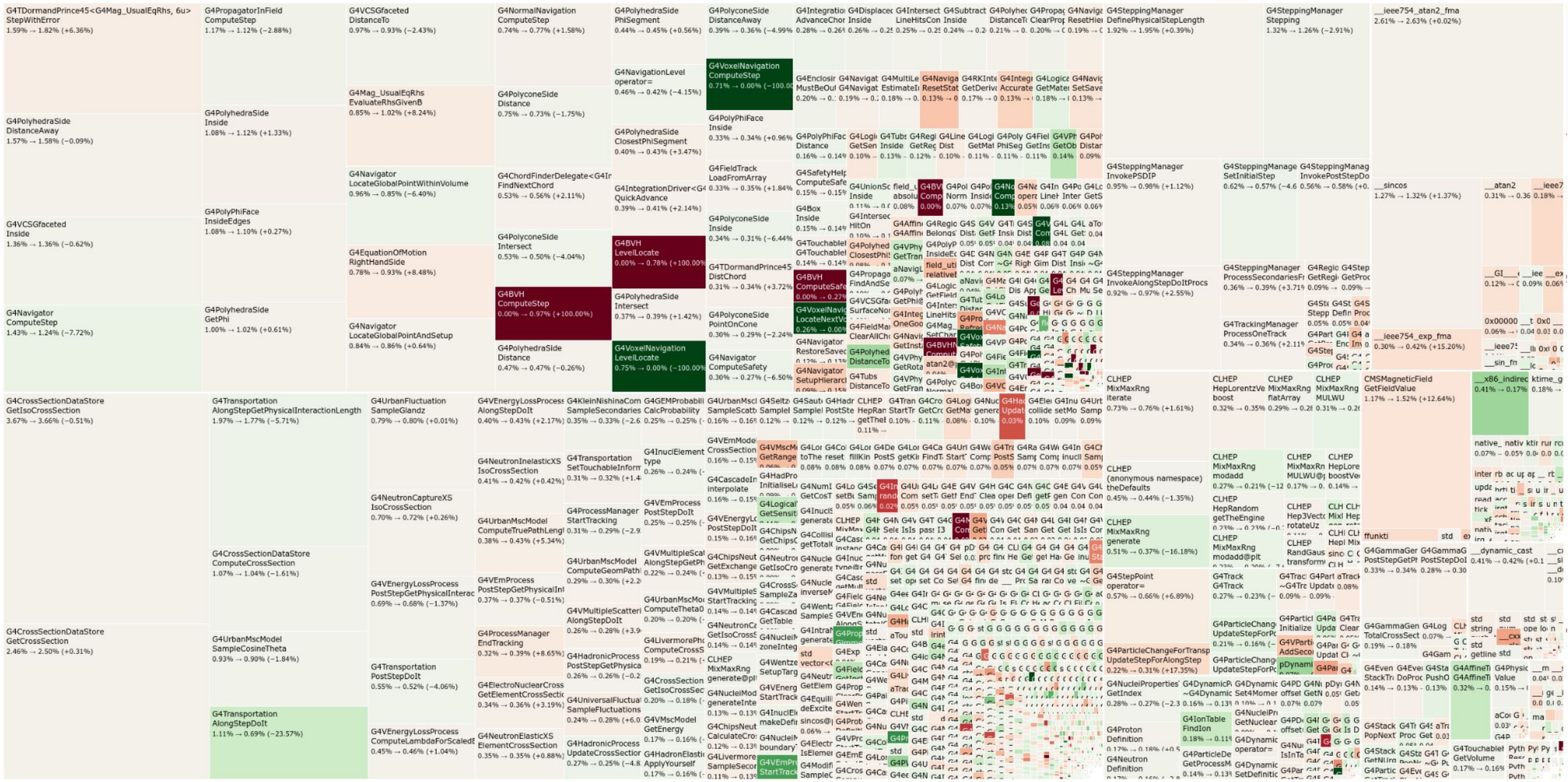


G4Navigator + G4VoxelNavigation → 28.2% of runtime



G4Navigator + G4BVHNavigation → 27.2% of runtime

Treemap

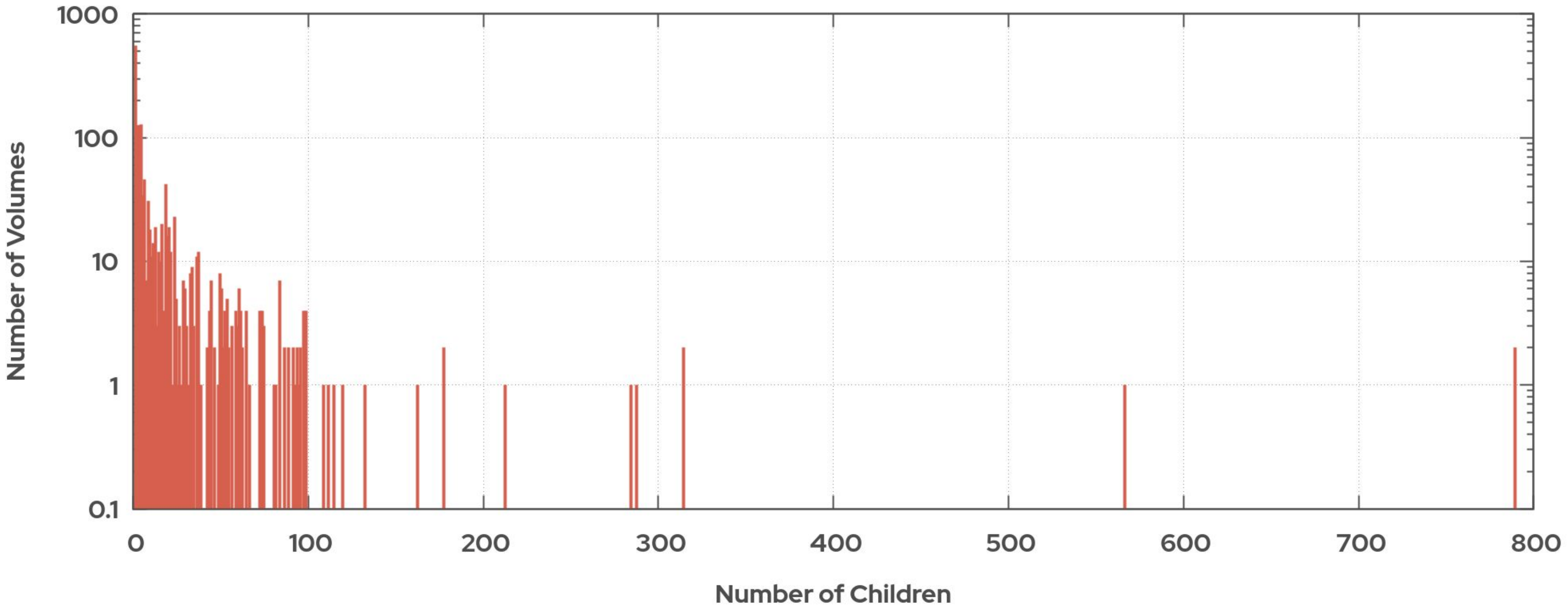

G4TDormandPrince45<G4Mag_UsualEqRhs, 6u> StepWithError 1.59% → 1.82% (+6.36%)
G4PropagatorInField ComputeStep 1.17% → 1.12% (−2.88%)
G4VCSGfaceted DistanceTo 0.97% → 0.93% (−2.43%)
G4NormalNavigation ComputeStep 0.74% → 0.77% (+1.58%)
G4PolyhedraSide PhiSegment 0.44% → 0.45% (+0.56%)
G4PolyconeSide DistanceAway 0.39% → 0.36% (−4.99%)
G4SteppingManager DefinePhysicalStepLength 1.92% → 1.95% (+0.39%)
G4SteppingManager Stepping 1.32% → 1.26% (−2.91%)
__ieee754_atan2_fma 2.61% → 2.63% (+0.02%)
G4PolyhedraSide DistanceAway 1.57% → 1.58% (−0.09%)
G4PolyhedraSide Inside 1.08% → 1.12% (+1.33%)
G4Mag_UsualEqRhs EvaluateRhsGivenB 0.85% → 1.02% (+8.24%)
G4PolyconeSide Distance 0.75% → 0.73% (−1.75%)
G4NavigationLevel operator= 0.46% → 0.42% (−4.15%)
G4VoxelNavigation ComputeStep 0.71% → 0.00% (−100.00%)
G4PolyPhiFace Inside 0.33% → 0.34% (+0.96%)
G4PolyhedraSide ClosestPhiSegment 0.40% → 0.43% (+3.47%)
G4Navigator LocateGlobalPointWithinVolume 0.96% → 0.85% (−6.40%)
G4ChordFinderDelegate<G4Ir FindNextChord 0.53% → 0.56% (+2.11%)
G4IntegrationDriver<G4 QuickAdvance 0.39% → 0.41% (+2.14%)
G4FieldTrack LoadFromArray 0.33% → 0.35% (+1.84%)
G4VCSGfaceted Inside 1.36% → 1.36% (−0.62%)
G4PolyPhiFace InsideEdges 1.08% → 1.10% (+0.27%)
G4EquationOfMotion RightHandSide 0.78% → 0.93% (+8.48%)
G4PolyconeSide Intersect 0.53% → 0.50% (−4.04%)
G4BVH LevelLocate 0.00% → 0.78% (+100.00%)
G4PolyconeSide Inside 0.34% → 0.31% (−6.44%)
G4TDormandPrince45 DistChord 0.31% → 0.34% (+3.72%)
G4SteppingManager InvokePSDIP 0.95% → 0.98% (+1.12%)
G4SteppingManager SetInitialStep 0.62% → 0.57%
G4SteppingManager InvokePostStepDo 0.56% → 0.58%
__sincos 1.27% → 1.32% (+1.37%)
G4Navigator ComputeStep 1.43% → 1.24% (−7.72%)
G4PolyhedraSide GetPhi 1.00% → 1.02% (+0.61%)
G4Navigator LocateGlobalPointAndSetup 0.84% → 0.86% (+0.64%)
G4BVH ComputeStep 0.00% → 0.97% (+100.00%)
G4PolyhedraSide Distance 0.47% → 0.47% (−0.26%)
G4PolyhedraSide Intersect 0.37% → 0.39% (+1.42%)
G4VoxelNavigation LevelLocate 0.75% → 0.00% (−100.00%)
G4PolyconeSide PointOnCone 0.30% → 0.29% (−2.24%)
G4Navigator ComputeSafety 0.30% → 0.27% (−6.50%)
G4SteppingManager InvokeAlongStepDoItProcs 0.92% → 0.97% (+2.55%)
G4SteppingManager ProcessSecondariesFr 0.36% → 0.39% (+3.71%)
G4TrackingManager ProcessOneTrack 0.34% → 0.36% (+2.11%)
__ieee754_exp_fma 0.30% → 0.42% (+15.20%)
G4CrossSectionDataStore GetIsoCrossSection 3.67% → 3.66% (−0.51%)
G4Transportation AlongStepGetPhysicalInteractionLength 1.97% → 1.77% (−5.71%)
G4UrbanFluctuation SampleGlandz 0.79% → 0.80% (+0.01%)
G4VEnergyLossProcess AlongStepDoIt 0.40% → 0.43% (+2.17%)
G4KleinNishinaCom SampleSecondaries 0.35% → 0.33%
G4GEMProbability CalcProbability 0.25% → 0.25%
CLHEP MixMaxRng iterate 0.73% → 0.76% (+1.61%)
CLHEP HepLorentzVe boost 0.32% → 0.35%
CLHEP MixMaxRng flatArray 0.29% → 0.28%
CLHEP MixMaxRng MULWU 0.31% → 0.28%
CMSMagneticField GetFieldValue 1.17% → 1.52% (+12.64%)
G4NeutronInelasticXS IsoCrossSection 0.41% → 0.42% (+0.42%)
G4Transportation SetTouchableInform 0.31% → 0.32%
G4NeutronCaptureXS IsoCrossSection 0.70% → 0.72% (+0.26%)
CLHEP (anonymous namespace) theDefaults 0.45% → 0.44% (−1.35%)
CLHEP MixMaxRng modadd 0.27% → 0.21%
CLHEP HepRandom getTheEngine
G4UrbanMscModel ComputeTruePathLengt 0.38% → 0.43% (+5.34%)
G4ProcessManager StartTracking 0.31% → 0.29%
G4VEmProcess PostStepDoIt 0.25% → 0.25%
CLHEP MixMaxRng generate 0.51% → 0.37% (−16.18%)
G4CrossSectionDataStore ComputeCrossSection 1.07% → 1.04% (−1.61%)
G4VEnergyLossProcess PostStepGetPhysicalInterac 0.69% → 0.68% (−1.37%)
G4VEmProcess PostStepGetPhysicalInt 0.37% → 0.37% (−0.51%)
G4UrbanMscModel ComputeGeomPath 0.29% → 0.30%
G4StepPoint operator= 0.57% → 0.66% (+6.89%)
G4Track G4Track 0.27% → 0.23%
G4GammaGen PostStepGetPh
__dynamic_cast 0.41% → 0.42%
G4CrossSectionDataStore GetCrossSection 2.46% → 2.50% (+0.31%)
G4UrbanMscModel SampleCosineTheta 0.93% → 0.90% (−1.84%)
G4ProcessManager EndTracking 0.32% → 0.39% (+8.65%)
G4VMultipleScatteri AlongStepDoIt 0.26% → 0.28%
G4ParticleChangeForTransp UpdateStepForAlongStep 0.22% → 0.31% (+17.35%)
G4ParticleChang UpdateStepForPc 0.21% → 0.16%
G4Transportation PostStepDoIt 0.55% → 0.52% (−4.06%)
G4ElectroNuclearCross GetElementCrossSecti 0.34% → 0.36% (+3.19%)
G4HadronicProcess PostStepGetPhysica 0.26% → 0.26%
G4NucleiProperties GetIndex 0.28% → 0.27%
G4DynamicParticle ~G4Dynamic 0.16% → 0.13%
G4Transportation AlongStepDoIt 1.11% → 0.69% (−23.57%)
G4VEnergyLossProcess ComputeLambdaForScaledE 0.45% → 0.46% (+1.04%)
G4NeutronElasticXS ElementCrossSection 0.35% → 0.35% (+0.88%)
G4HadronicProcess UpdateCrossSection 0.27% → 0.25%
G4Proton Definition
G4IonTable FindIon 0.18% → 0.11%
G4Neutron Definition
G4ParticleDe GetProcessM 0.14% → 0.13%
G4AffineT Value 0.32%
G4Touchable GetVolume 0.17% → 0.16%
funkti

Summary

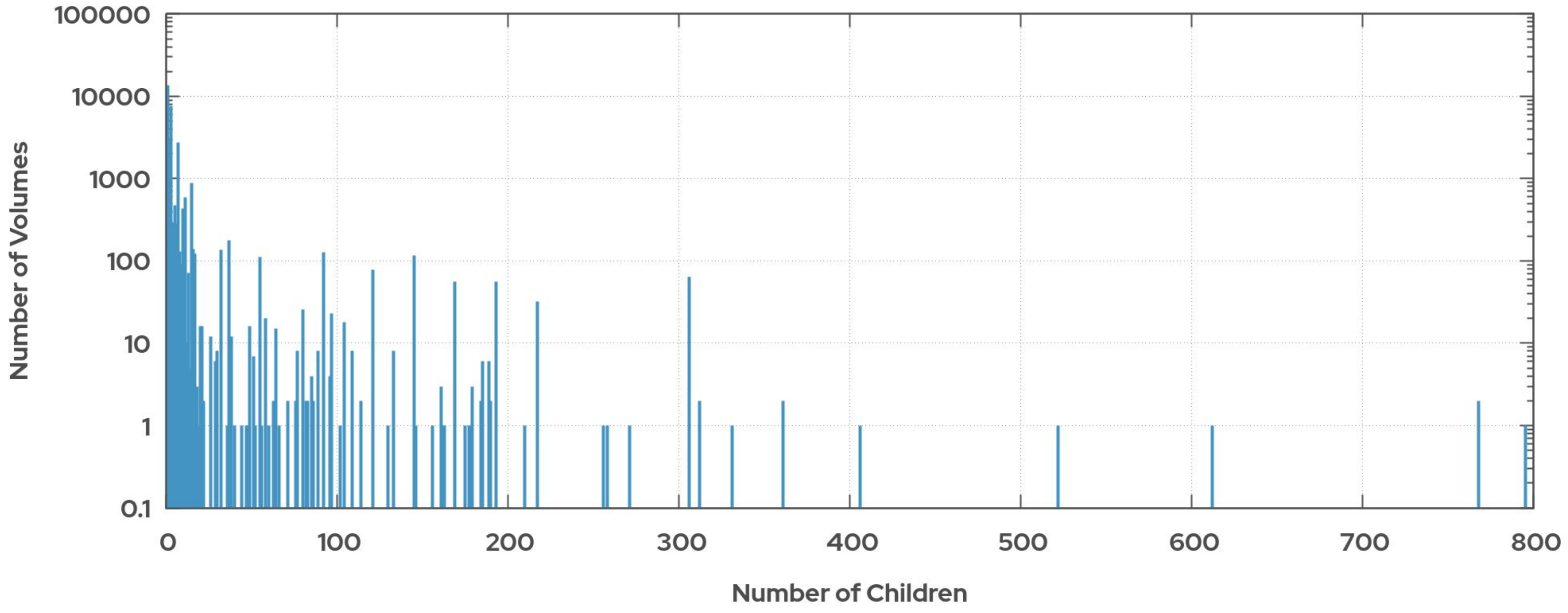
- BVH implementation from VecGeom is working inside Geant4
  - Uses native Geant4 navigation functionality
  - G4BVHNavigator needs further work to support replicas and parameterized volumes
  - Can use BVH for normal volumes and voxel navigator for the rest with current implementation
  - Proper integration needs more code refactoring of core geometry classes
- Performance is comparable to G4VoxelNavigator for a full detector simulation
  - Not worth investing a lot of time for a small gain in performance
  - Performance benefit of BVH more visible with volumes with lots of children
  - Detector geometries have many logical volumes with only a few children, which limits benefit

Backup

Number of Children per Logical Volume (CMS)



Number of Children per Logical Volume (ATLAS)



Note: 4 volumes have between 1k and 10k children in ATLAS, and 4 volumes have more than 10k. Excluded to avoid squishing data to the left.