Motivation

- We have vectorized voxelization working nicely.
- However:
  - **Initialization** of **helper structures does not scale**
  - **Only 2-level tree implemented which has assymptotic O(N) scaling**; real BVH needs further care
  - Voxelization based on AABBoxes
    - generalization clear but never done
  - lack of manpower …
- Could we benefit from externally maintained resources?

First investigation of Intel Embree inside VecGeom

Sandro Wenzel, May 2018

Intro to Embree

https://embree.github.io/

- What is Intel Embree?
  - Embree is an open source library offering “high-performance ray-tracing kernels”
    - maintained by Intel, used by Industry
    - specialized on CPU renderings
    - uses SIMD instructions under the hood for BVH traversal + triangle intersections
- What can it do?
  - for our purposes it can setup a static scene of many triangles/quads and query the distance to the scene in float precision
- Why might it be interesting for us?
  - has fast construction of acceleration structures (BVH trees)
    - bottleneck for very large tessellated solids
  - less development/maintenance on our side
  - nice feature of selecting ISA code path dynamically depending on where it is run (get AVX on AVX machine without recompiling)

Possible Use cases

- Use Embree to model the bounding box (polygonal hull) setup used for fast volume exclusions
  - relevant for navigation, better voxelization?
  - could improve beyond our native vectorized navigators (e.g. HybridNavigator)
- Use Embree to actually model/query volumes consisting of facets
  - speed up single volume distance queries
  - tessellated solid, polygon, trapezoid, etc.

Use case 1: for navigation

- Develop(ed) an EmbreeNavigator which can make an acceleration structure from
  - aligned bounding boxes (like HybridNavigator)
  - arbitrary (tight) bounding boxes
  - todo: arbitrary (tight) facet-hulls
- Embree scene is made from triangles that make up these boxes/hulls
- The navigator queries the scene for intersections thereby constructing a list of actual candidate vecgeom volumes to query later on

Use case 1: navigation – first tests

- Tested on most complex volume found in CMS: MBWheel_1N
  - ~800 daughters consisting of trapezoids, boolean volumes, tubes
- So far done an extremely good job! Our native navigator outperforms Embree here
- But at least 2 possibilities here
  - Embree AVX outperforms Hybrid when VecGeom was compiled with SSE
  - We could easily make a tight facet-hull navigator with Embree with potentially better timing
    - need polygonal **outer** approximations for each volume!!

ISA	HybridNavigator (AABBoxes)	Embree (AABBoxes)	Embree (tight BBBoxes)	Embree (tight facet-hulls)
SSE4.2	**0.0855**	**0.134**	**0.091**	**?**
AVX	**0.061**	**0.115**	**0.074**	**?**

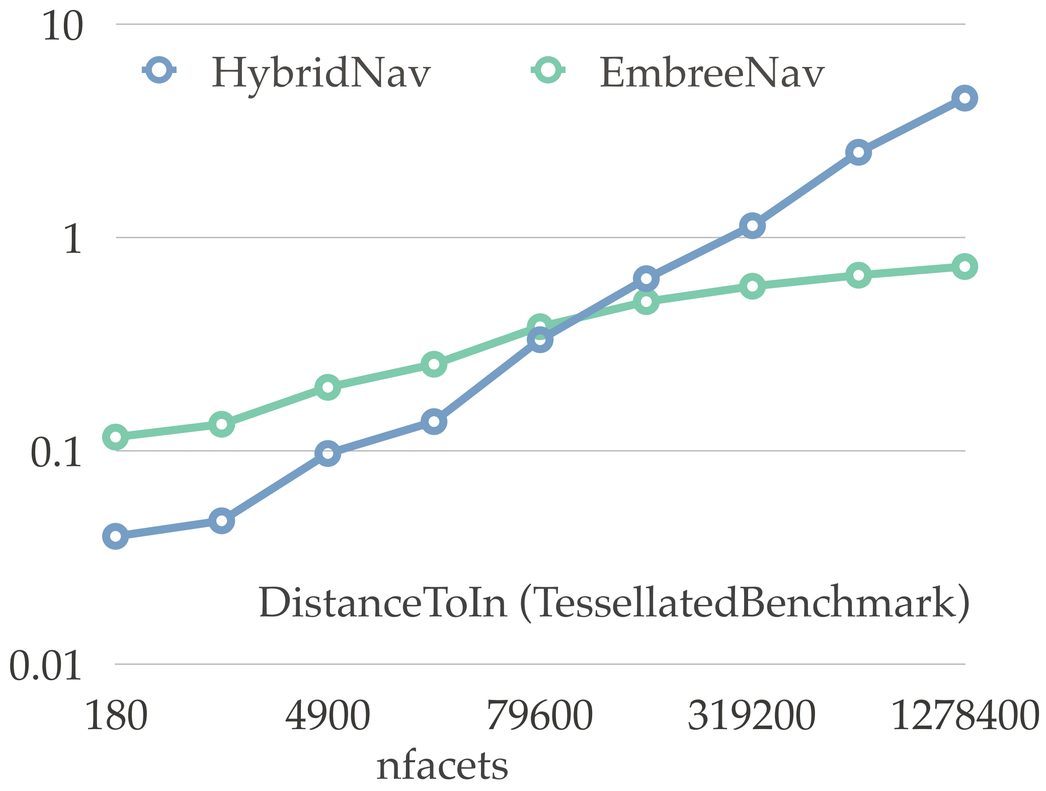
ref for TGeo is 0.239s

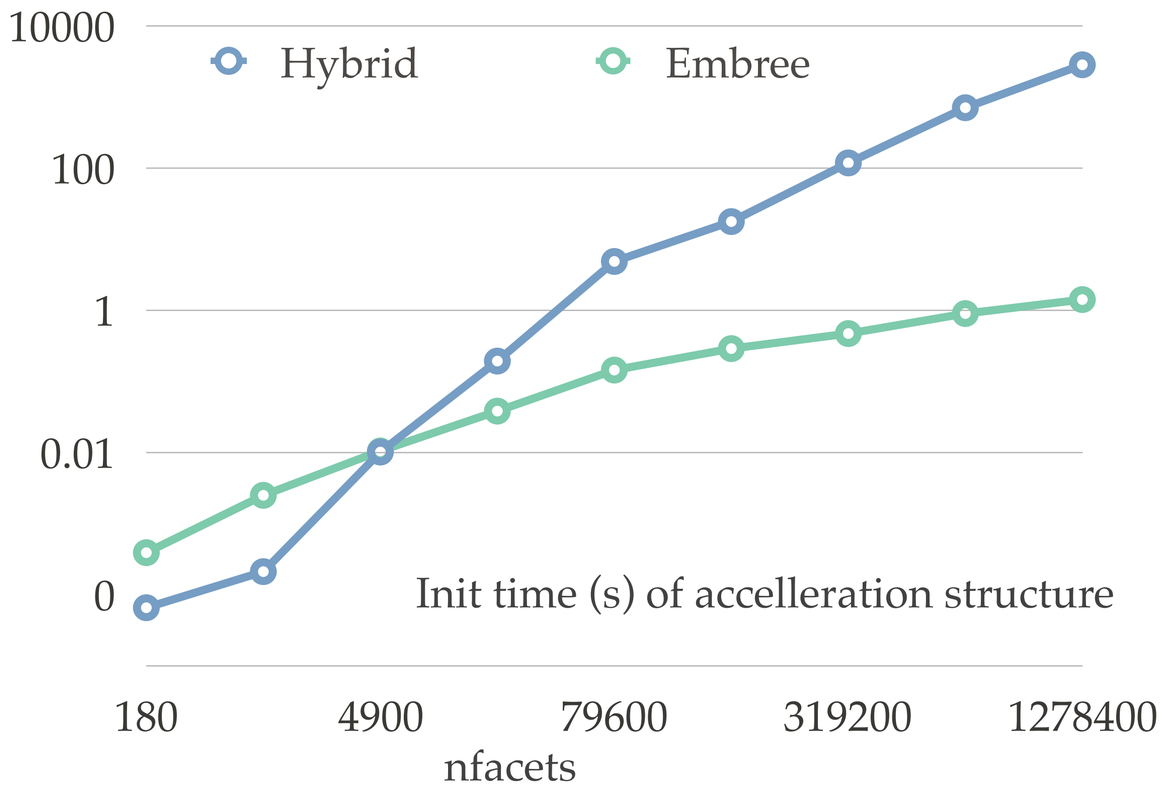
numbers are seconds

Reasons why slower

- We have dedicated algorithms for AABoxes
  - almost never do a division
  - we know we are talking to boxes and exploit symmetries
- Embree always just talks to triangles
  - might be doing too much work here
- On the other hand, this could be exploited because (as said) we can now go away to AABBoxes to much tighter hull-objects

Tessellated - Replacing HybridN with EmbreeN





- Can easily use `EmbreeNavigator` in Tessellated solid
- Better DistanceToIn for nfacets > ~ 80000
- Better Initialization times for nfacets > ~5000

Scaling problem solved

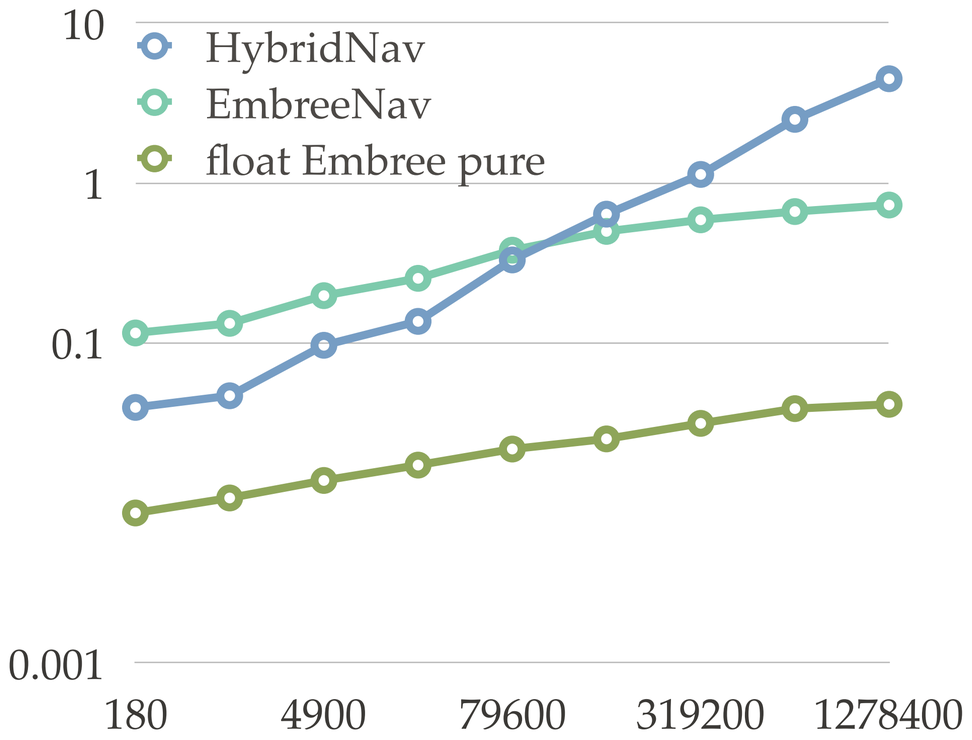
```
./TessellatedBenchmark -nrep 1 -npoints 100000 -ngrid xx
```

Use case 2: facet-type volumes

- What if we model directly the tessellated solid as Embree scene?

Use case 2: facet-type volumes

- What if we model directly the tessellated solid as Embree scene?
- **Wow!!**
  - a lot faster (order of magnitude) compared to previous numbers
  - but keep in mind difference in precision
- Couldn't we use Embree to get a quick/dirty intersection + floating-point distance to the facets and then do a refinement step in double precision?
  - should be ok in general
  - what about boundaries???




```
./TessellatedBenchmark -nrep 1 -npoints 100000 -ngrid xx
```

My conclusions

- We've done a nice job so far with our vectorized navigators
- Embree can push this further in some cases:
  - Superior scaling in large facet limit (as expected)
  - Use it as a crude approximator for all kinds of facet-like shapes
  - Use it for candidate selection using tight facet hulls

- A couple of additional benefits
  - automatic ISA selection
- Action items:
  - provide (configurable) facet hulls for each volume
  - investigate stability when directly using for shapes

Things to consider

- Embree is a large library and takes some time to compile
- Part of the better timing comes from **multithreading inside Embree** (in particular during voxel building??)
- My implementation of EmbreeNavigator not yet thread-safe
- How would one ship it??
  - could be both internal to VecGeom + external

P.S.

- One thing I noticed is that Safety for tessellated solid is MUCH slower than distance
- I guess we should work on this; Crude safety rather than precise one, etc.
- Embree does not have “Safety” function … but it offers us access to the BVH