SPONSORED BY THE

Federal Ministry
of Education
and Research

GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Update on the detray Geometry

ACTS Parallelization Meeting

Joana Niermann
Andreas Salzburger

15.10.2021

CERN and University of Göttingen

detray Geometry

General Considerations

- Geometry without polymorphism
- Flat container structure
- Indexing instead of pointers

Components

- Navigation **Volumes** with boundaries that keep surfaces
- **Surface** class, that can either be used as module surface or portal
- Various **Mask** classes that provide surface boundaries
- Surface **Transforms** (including their inverse)
- **Grid** as volume and surface finders

=> Kept in dedicated container structures, linking between objects by direct indexing

Source: https://github.com/acts-project/detray

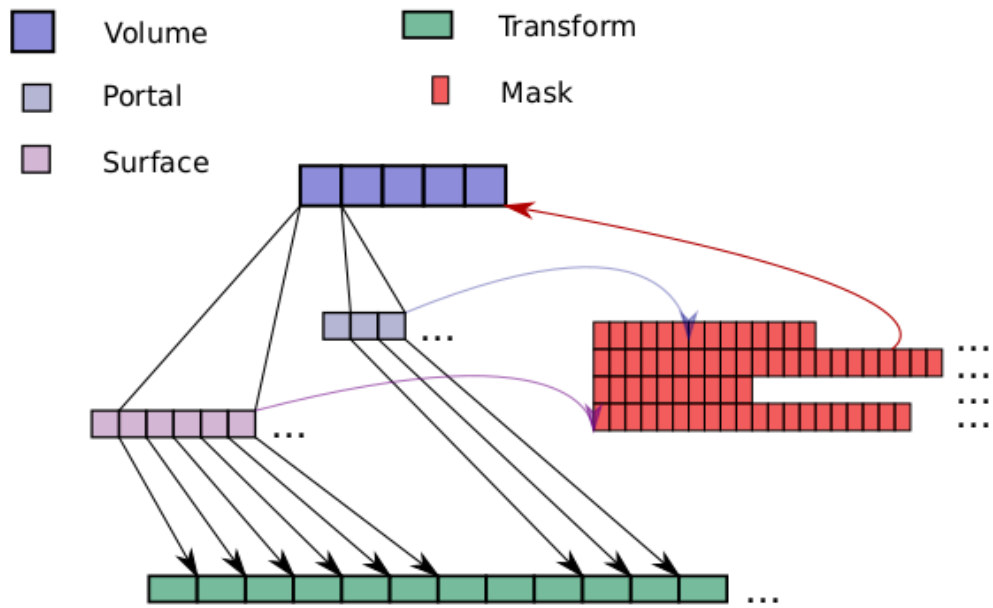
Geometry Implementation

Structure

- Detector class as interface (keeps transforms, masks, geometry and grid)
- Geometry provides the indexing data
- Navigator moves through geometry by its links and feeds the data containers to intersection kernel
- Rudimentary propagator and line-stepper classes steer navigation status and target calls
- csv reader to load different geometries (tml, odd, itk)

Linking

- The geometry class keep a vector of volumes, sorted by volume index
- The volumes keeps index ranges into surface and portal containers
- The surface class keeps index into the transform and mask containers
- The masks keeps links towards the next volume and surface finder in case they belong to a portal



=> Every type needs its own container: Variadic unrolling of mask container

Source: https://github.com/acts-project/detray

Recent Development

Modularization

- Detector class as interface between navigator geometry, data and grid
- Abstract container interfaces (transform store, mask store)
- Different geometry implementations possible
- Transparent switch between algebra implementations

Memory pooling

- Transforms and mask are handled in global container structures
- No vectors of vectors left in the geometry
- Container types interchangeable (by templating)

Indexed Geometry

- Standard implementation
- Can distinguish portal and module surface types (different linking behaviour)
- Portal mask batching

Unified index Geometry (WIP)

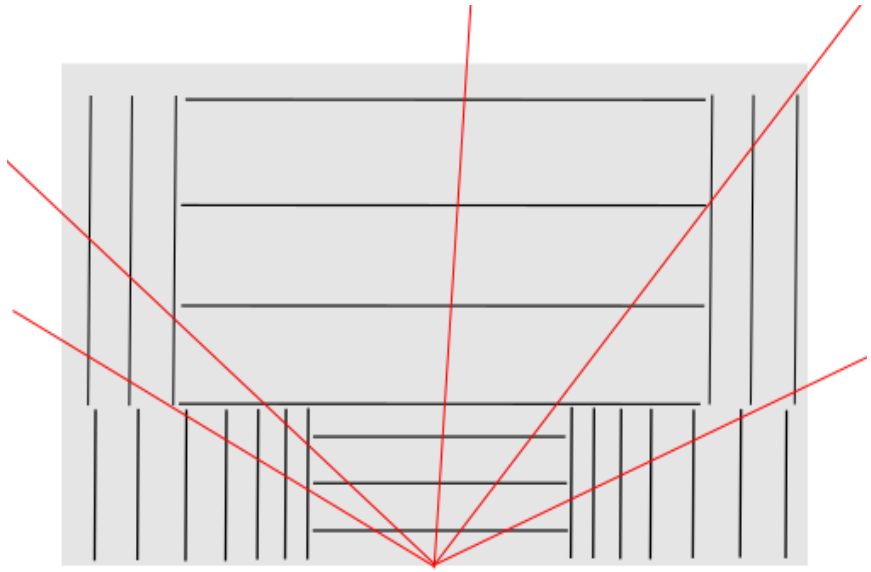
- Simplified model with lower number of containers
- No difference between portal and module surface type (surface links back to mother volume)
- No batching whatsoever

Source: https://github.com/acts-project/detray

Validation Effort

Ray scan

- Shoot straight line ray through detector
- Record every crossed portal and volume index
- Match intersection distance for pairs of portals
- Form a chain of volumes



⇒ Seems to work with TML

⇒ Some portals don't match for ODD

Navigation validation (WIP)

- Shoot ray, but this time follow with navigator
- So far: stuck on invalid links in TML

Geometry linking validation (WIP)

- Treat geometry as graph
- Check adjacency list against ray scan

Validation Effort - Outlook

Toy Geometry

- Implement a very simple, small geometry apart from csv reading
- All links are set manually and checked for consistency
- Uses raw volume, surface, transform and mask containers, outside of the 'big classes'

The toy geometry follows the TML pixel detector. It contains:

- A beampipe ($r = 27\,\mathrm{mm}$)
- An inner layer ($r_{\mathrm{min}} = 27\,\mathrm{mm}$, $r_{\mathrm{max}} = 38\,\mathrm{mm}$) with 224 pixel module surfaces
- A gap volume ($r_{\mathrm{min}} = 38\,\mathrm{mm}$, $r_{\mathrm{max}} = 64\,\mathrm{mm}$)
- An outer layer ($r_{\mathrm{min}} = 64\,\mathrm{mm}$, $r_{\mathrm{max}} = 80\,\mathrm{mm}$) with 448 pixel module surfaces
- TODO: Add grid

Toy Navigator (WIP)

- To handle this geometry, a navigator is needed
- No distinction between portals and surfaces, just follow volume links
- Less bookkeeping between portal and surface kernel structures
- First candidate to port to device?