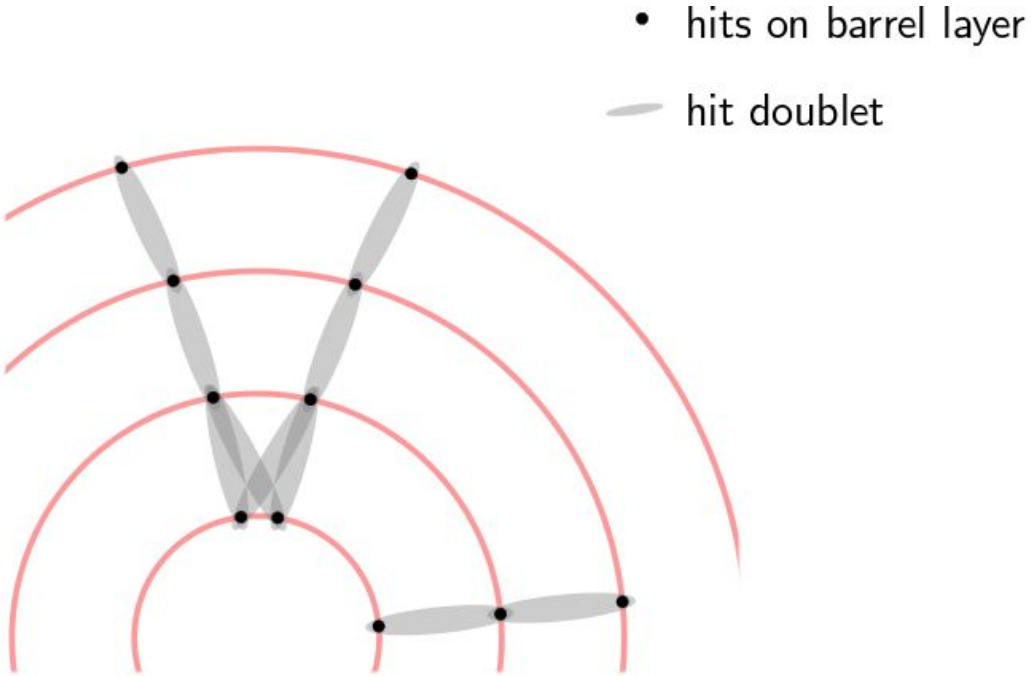
Cellular Automaton in ACTS

Valentin Volkl, ACTS developers meeting 09.07.2017

Cellular Automaton in ACTS

- Original CMS Implementation:
  - https://github.com/cms-sw/cmssw/blob/60f3ed2dbf1d980a7bcd5e26fd9528bde7386cb1/RecoPixelVertexing/PixelTriplets/plugins/CellularAutomaton.cc
  - F. Pantaleo et al.
- Decoupled already in FCCSW:
  - https://github.com/HEP-FCC/FCCSW/pull/191
- Main Parts:
  - Input: Hit spacepoints, organised in layers
  - Doublet creation, filter candidates using geometric cuts
  - Turn doublets into CA-cells, connecting them according to alignment and curvature
  - Evolve to find 2-, 3-, 4-, N-tuples

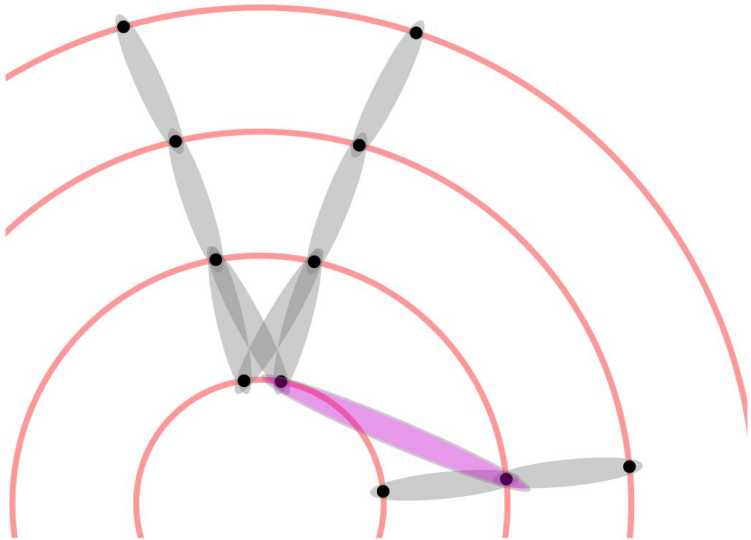
Doublet Creation



Doublets implementation

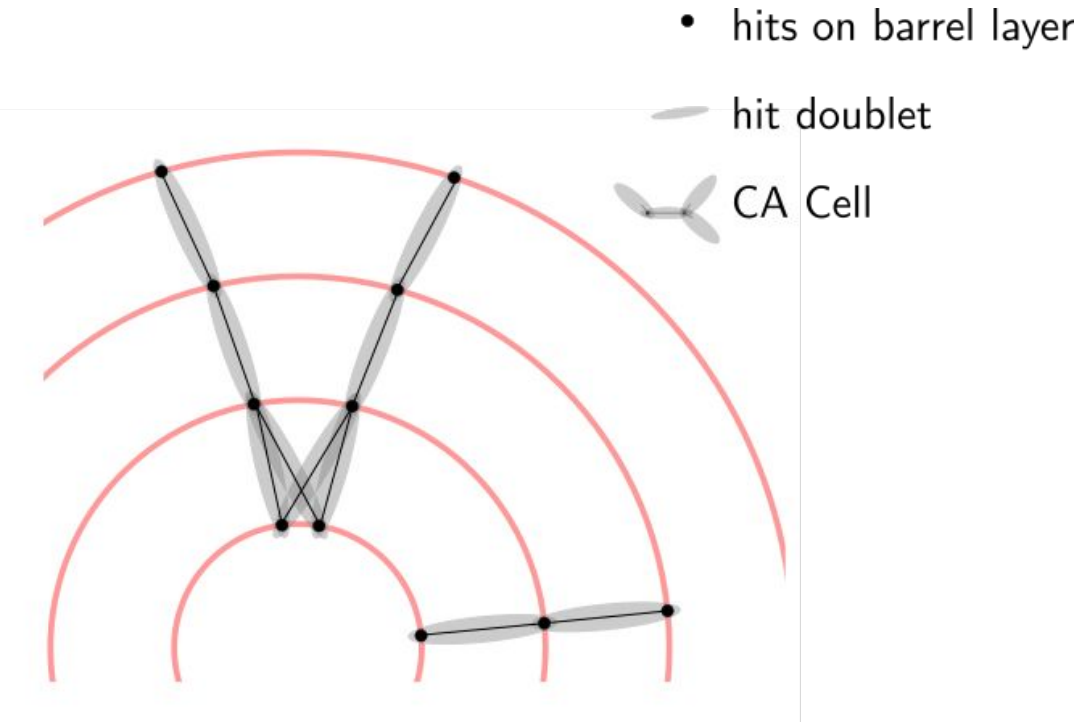
- Container of all hits on inner / outer layer along with indices which hits form a doublet
- Reuses Acts::Seeding::SpacePoint

Filter track candidates on Doublet creation



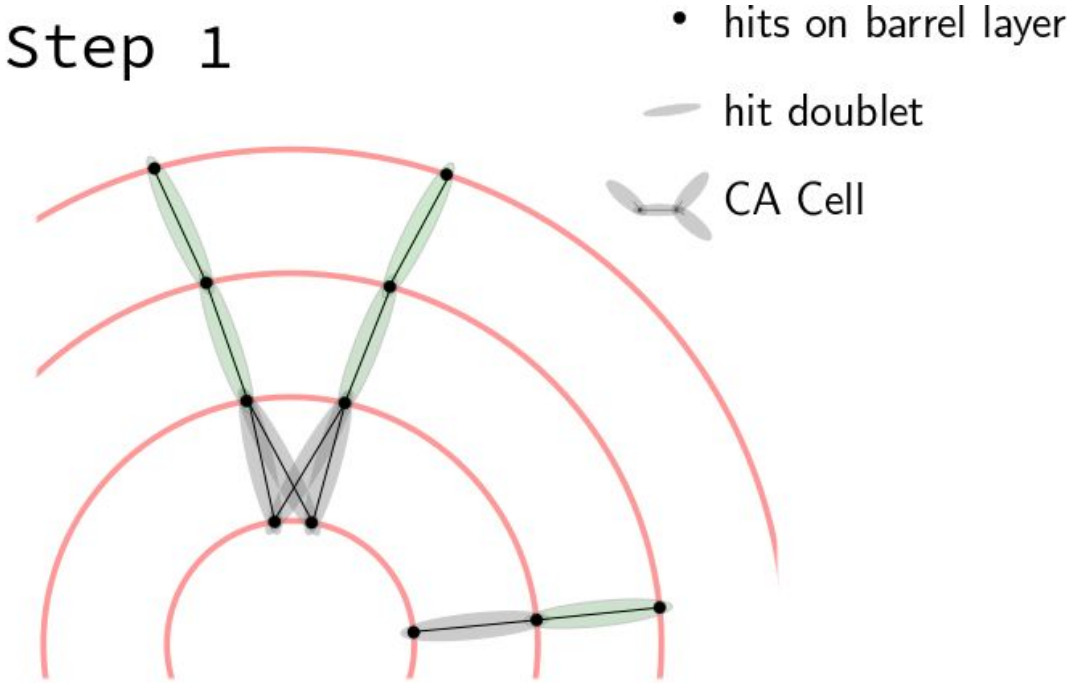
- Many track candidates can already be filtered from two hit information alone
- Felice's KDTree implementation would be a very flexible means of doing this
- For now, I reused the phi-/theta cut methods already present in Acts::Seeding

Connecting the Cells



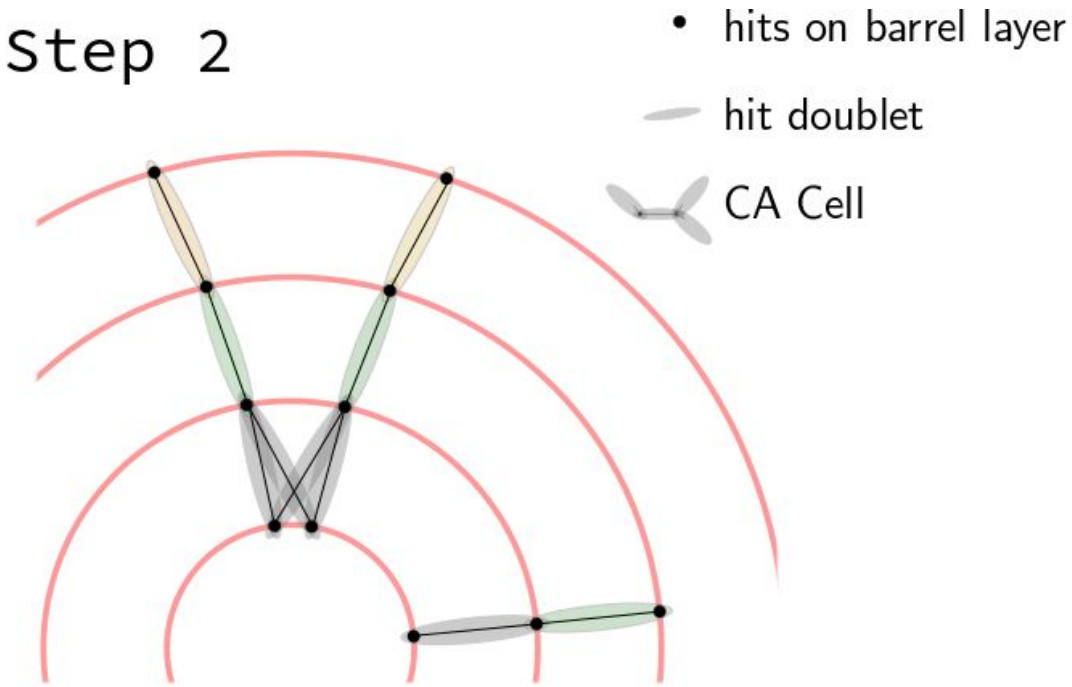
- The building blocks of the CA are the Cells
- For each hit doublet, a cell is created, consisting of
  - Identifiers of the doublet and hits making up the cell
  - Pointers to Cells connected on the inside and outside
  - The state of the Cell
- Has methods for the local evolution and n-tuple discovery
- Additional filtering possible here

Evolution



- Finally, the CellularAutomaton class steers the evolution
- At each iteration, the state of cells that have inner neighbors with the same state is incremented

Evolution



- Finally, the CellularAutomaton class steers the evolution
- At each iteration, the state of cells that have inner neighbors with the same state is incremented
- This successively finds the longest connected tracklets and returns them as a vector of pointers to the cells involved

ACTS implementation

The PR so far includes work done on:

- Removing all references to CMS specific types (no problem the CA was already fairly decoupled)
- Adding a test (reproducing the same behavior as the BarrelSeedFinder)
- Documentation and Code formatting

What should be discussed:

- Licensing /Attribution to original authors (for now via doxygen @author tag) / getting maintenance
- More sophisticated doublet creation using the KDTree
- Doublet / CACell restructuring

Source Code organisation

```
Core/include/ACTS/Seeding
     - CACell.hpp
     - SpacePoint.hpp
     - KDDoublets.hpp
     - TrackSeed.hpp
     - BarrelSeedFinder.hpp
     - CellularAutomaton.hpp
     - CellularAutomaton.ipp
     - detail
     - - cyclic_range.hpp
     - - geometry.hpp
Tests/Seeding
- CMakeLists.txt
- SeedingToolsTests.cpp
- SeedingTestsCommon.hpp
- SeedingTests.cpp
- CellularAutomatonTests.cpp
```