Track Seeding in EICrecon

Joe Osborn

January 19, 2023

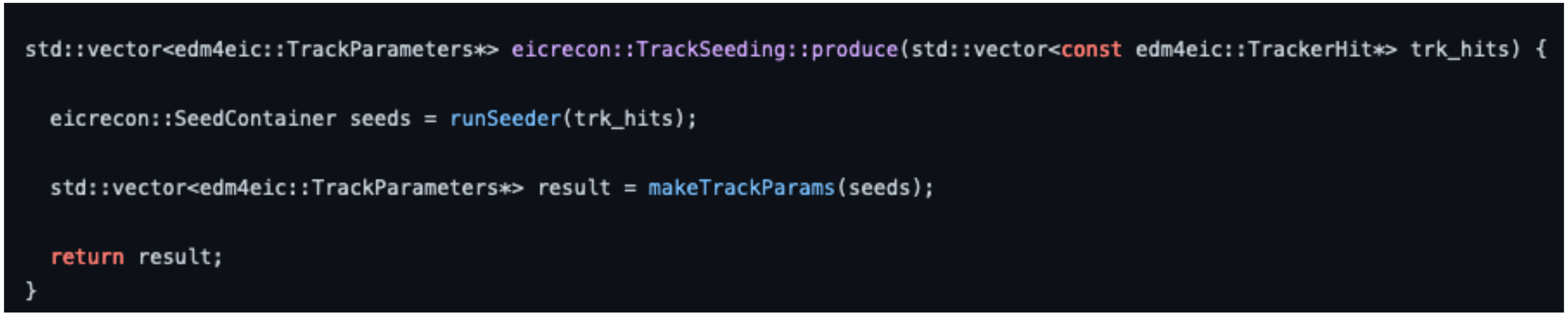
Overview

- Acts::OrthogonalSeedFinder is now implemented in EICrecon, but not tuned
- Need to tune the ~20 parameters to test efficiency, duplication rate, and seed quality performance
- Parameters can be tested from the command line, no compilation of EICrecon necessary

Seeder Implementation

- The seeding algorithm itself is pretty simple
  - Configuration class which contains all of the parameters
  - Factory class which calls the algorithm
    - Take hits, convert to space points, ship to Acts, then circle + line analytic fit to get estimate of track parameters
  - Put results in PODIO output file in the form of a collection of track parameters called CentralTrackSeedResults

Track Seeding Algorithm



```
std::vector<edm4eic::TrackParameters*> eicrecon::TrackSeeding::produce(std::vector<const edm4eic::TrackerHit*> trk_hits) {

  eicrecon::SeedContainer seeds = runSeeder(trk_hits);

  std::vector<edm4eic::TrackParameters*> result = makeTrackParams(seeds);

  return result;
}
```

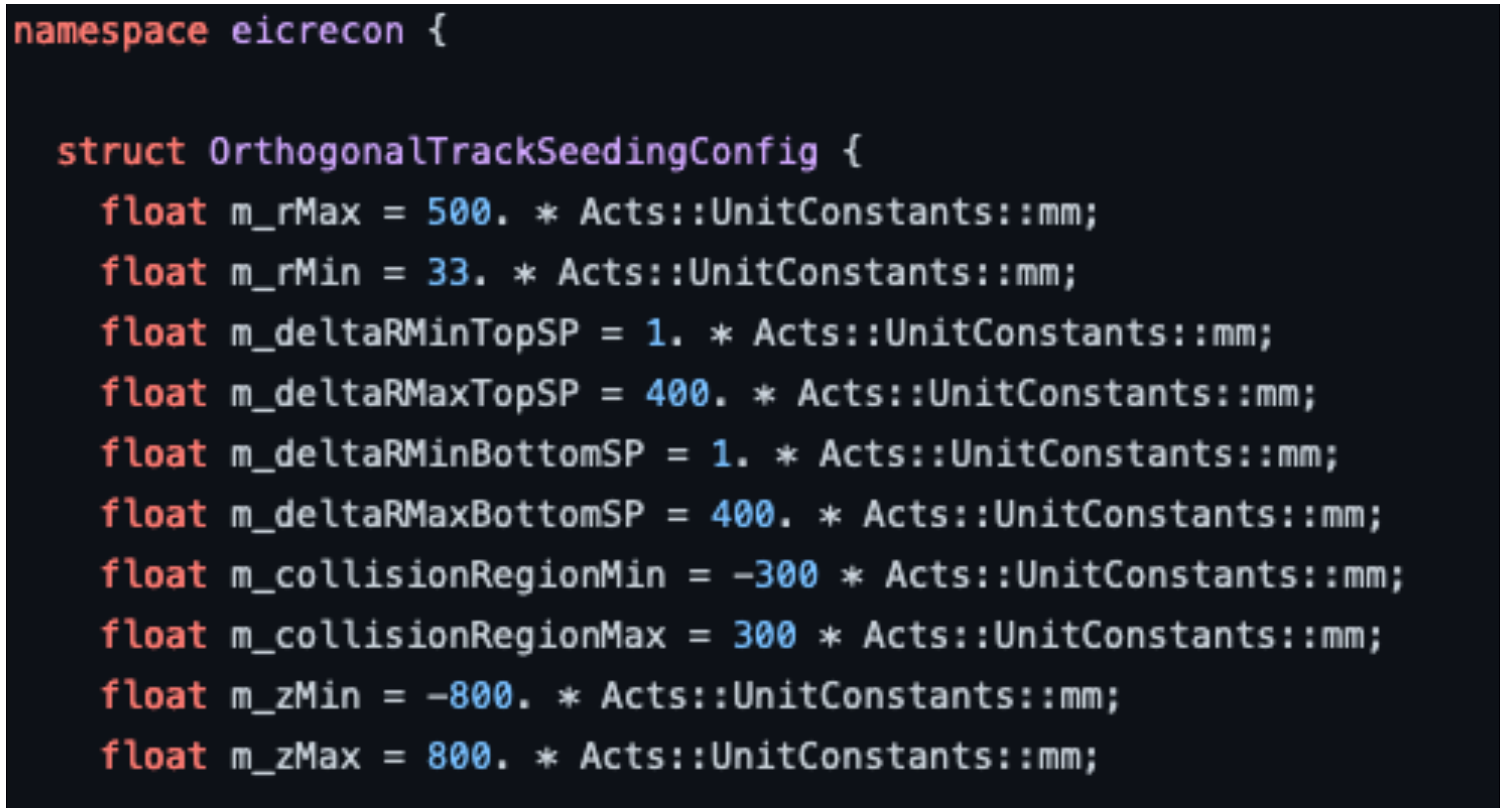
- runSeeder converts edm4eic::TrackerHits into Acts::SpacePoints which can be interpreted by Acts and calls Acts seeder - returns a vector of seeds

- makeTrackParams takes those seeds and does the analytic circle+line fit

Track Seeding Configuration

- Configuration parameters are stored in OrthogonalTrackSeedingConfig struct within eicrecon
    - These are a collection of the primary (and minimum) configuration parameters the Acts::OrthogonalSeedFinder takes
        - All parameters can be found in SeedFinderOrthogonalConfig.hpp within Acts repository, with some (limited) description of what they are

Track Seeding Configuration

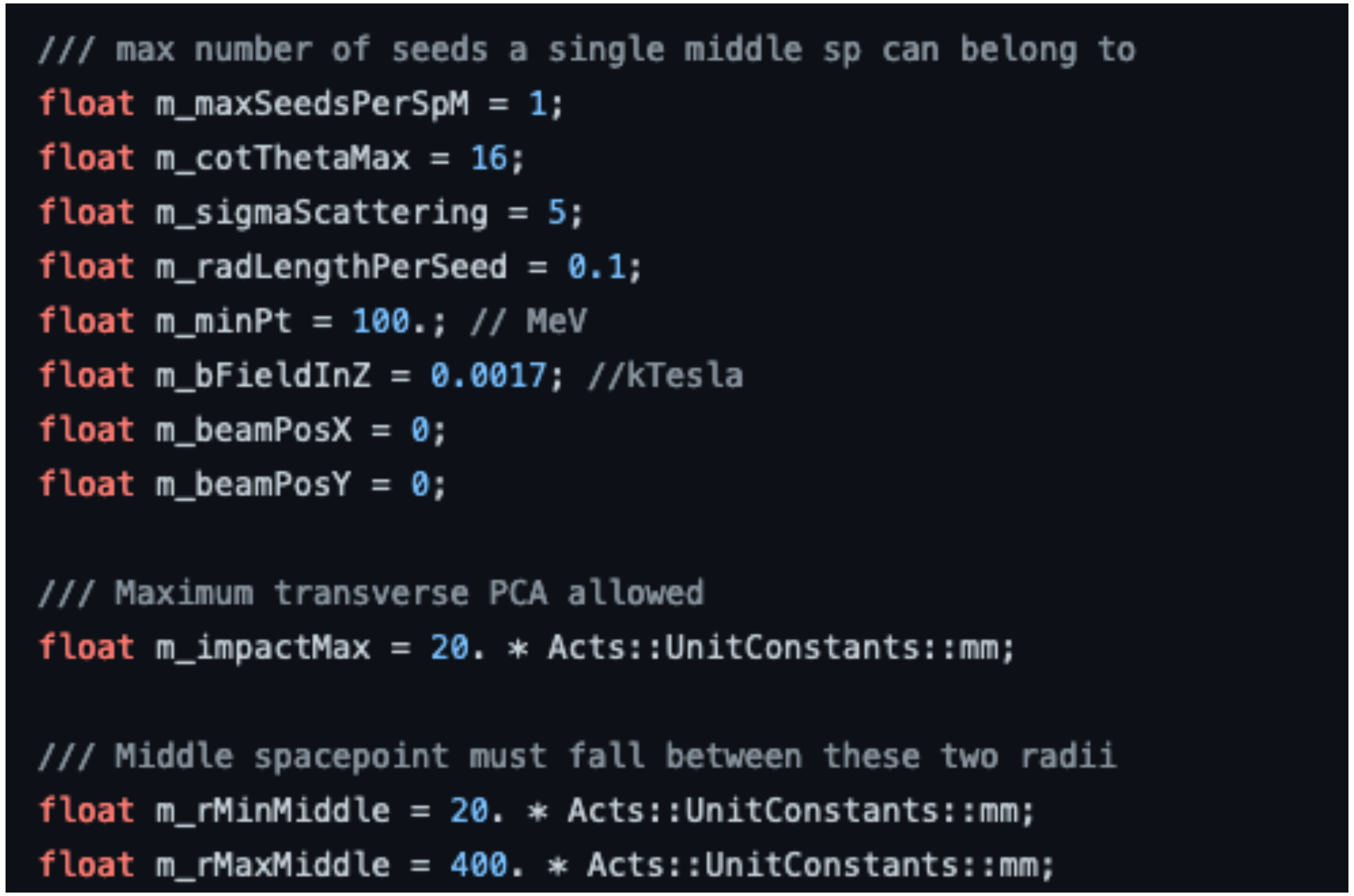


```
namespace eicrecon {

  struct OrthogonalTrackSeedingConfig {
    float m_rMax = 500. * Acts::UnitConstants::mm;
    float m_rMin = 33. * Acts::UnitConstants::mm;
    float m_deltaRMinTopSP = 1. * Acts::UnitConstants::mm;
    float m_deltaRMaxTopSP = 400. * Acts::UnitConstants::mm;
    float m_deltaRMinBottomSP = 1. * Acts::UnitConstants::mm;
    float m_deltaRMaxBottomSP = 400. * Acts::UnitConstants::mm;
    float m_collisionRegionMin = -300 * Acts::UnitConstants::mm;
    float m_collisionRegionMax = 300 * Acts::UnitConstants::mm;
    float m_zMin = -800. * Acts::UnitConstants::mm;
    float m_zMax = 800. * Acts::UnitConstants::mm;
```

- First set of parameters can be thought of as defining the volume within to search for seeds
  - Collision region, r and z min/max, min/max distance between hits
- As discussed last week, should set these to include outermost silicon layers

Track Seeding Configuration



```
/// max number of seeds a single middle sp can belong to
float m_maxSeedsPerSpM = 1;
float m_cotThetaMax = 16;
float m_sigmaScattering = 5;
float m_radLengthPerSeed = 0.1;
float m_minPt = 100.; // MeV
float m_bFieldInZ = 0.0017; //kTesla
float m_beamPosX = 0;
float m_beamPosY = 0;

/// Maximum transverse PCA allowed
float m_impactMax = 20. * Acts::UnitConstants::mm;

/// Middle spacepoint must fall between these two radii
float m_rMinMiddle = 20. * Acts::UnitConstants::mm;
float m_rMaxMiddle = 400. * Acts::UnitConstants::mm;
```

- These are parameters about the seeds themselves
- Once the volume is determined above, these will need to be tuned to provide adequate performance

How to run at CL

- The seeder is now included in our default eicrecon version within the eic-shell container
  - Follow these instructions to get the container going
  - Follow these instructions to make yourself a single particle simulation sample (producing a file named e.g. pions.edm4hep.root)
  - Run eicrecon on it with $ eicrecon pions.edm4hep.root
- That's it! The output PODIO file will have CentralTrackSeedResults, which are the track parameters associated to the found seeds
- The above will run with the default parameters set in OrthogonalTrackSeedingConfig. To change a parameter, simply add the parameter to the eicrecon call :
  $ eicrecon -Ptracking:CentralTrackSeedingResults:impactMax=5 pions.edm4hep.root
  where you just exchange impactMax with whatever parameter you want to configure

How to run with compilation

- If you want to tune many parameters at a time and don't want to have a long CLI command, you can compile eicrecon and just point to your local build as follows

- Instructions for how to build EICrecon and have your environment point to the local build

  - Then you can just go change whatever parameters you like in src/algorithms/tracking/OrthogonalTrackSeedingConfig.h, rebuild following the instructions, and run eicrecon like usual with eicrecon some_edm4hep_file.root

- The instructions are basically 5 CLI commands. Not too tricky and I was able to do it successfully this morning. Happy to help debug issues if people want to try this and run into problems