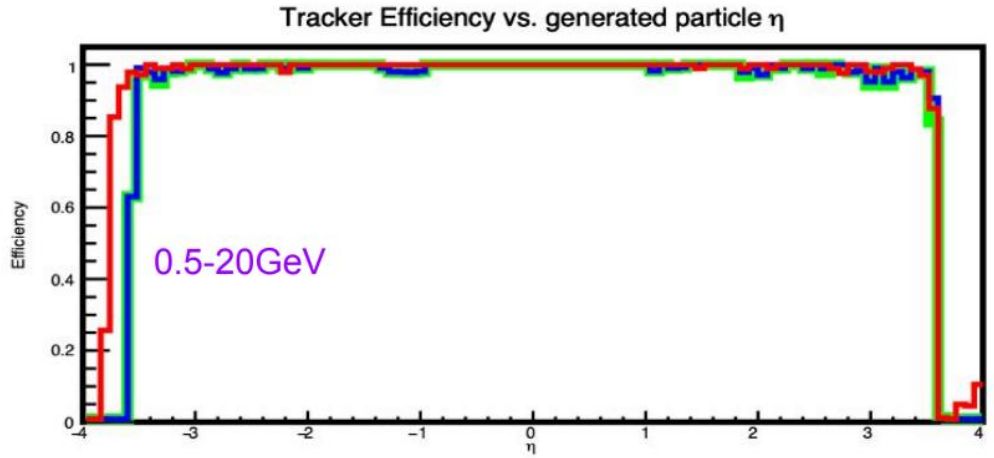
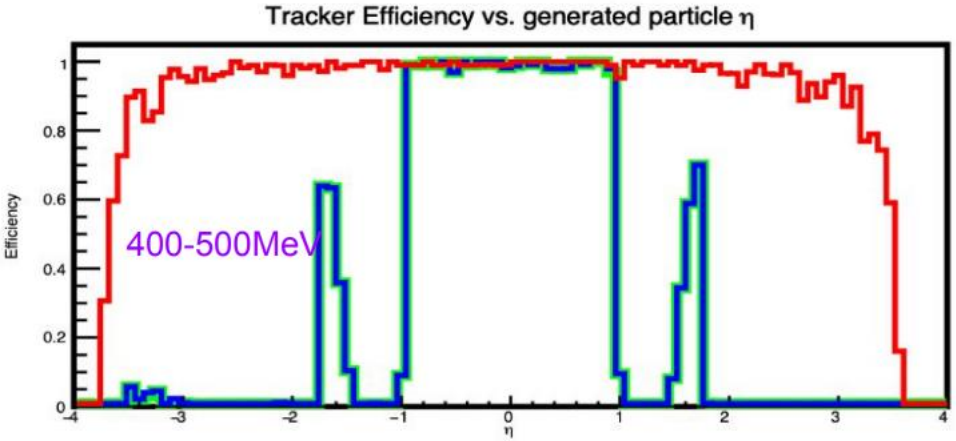
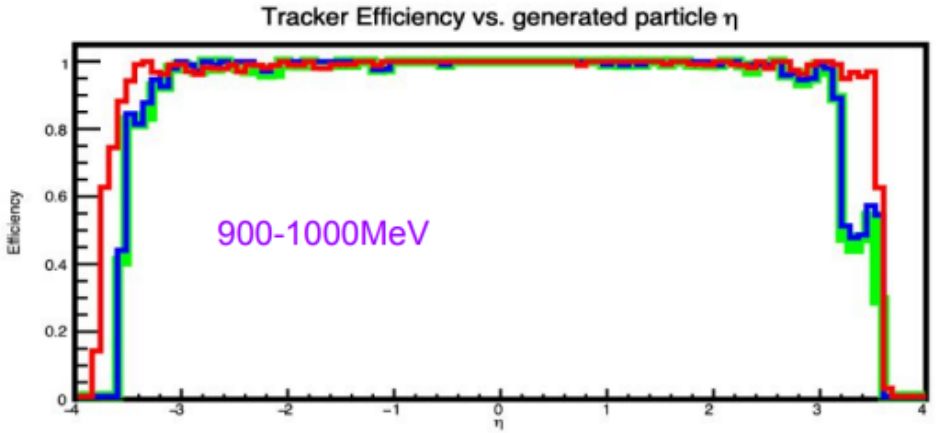
Seed-finding inefficiencies at low momentum

Jeetendra Gupta and Barak Schmookler

Seed-finding inefficiencies – single-particle (negative muon) simulation







Seed Level
Track Level

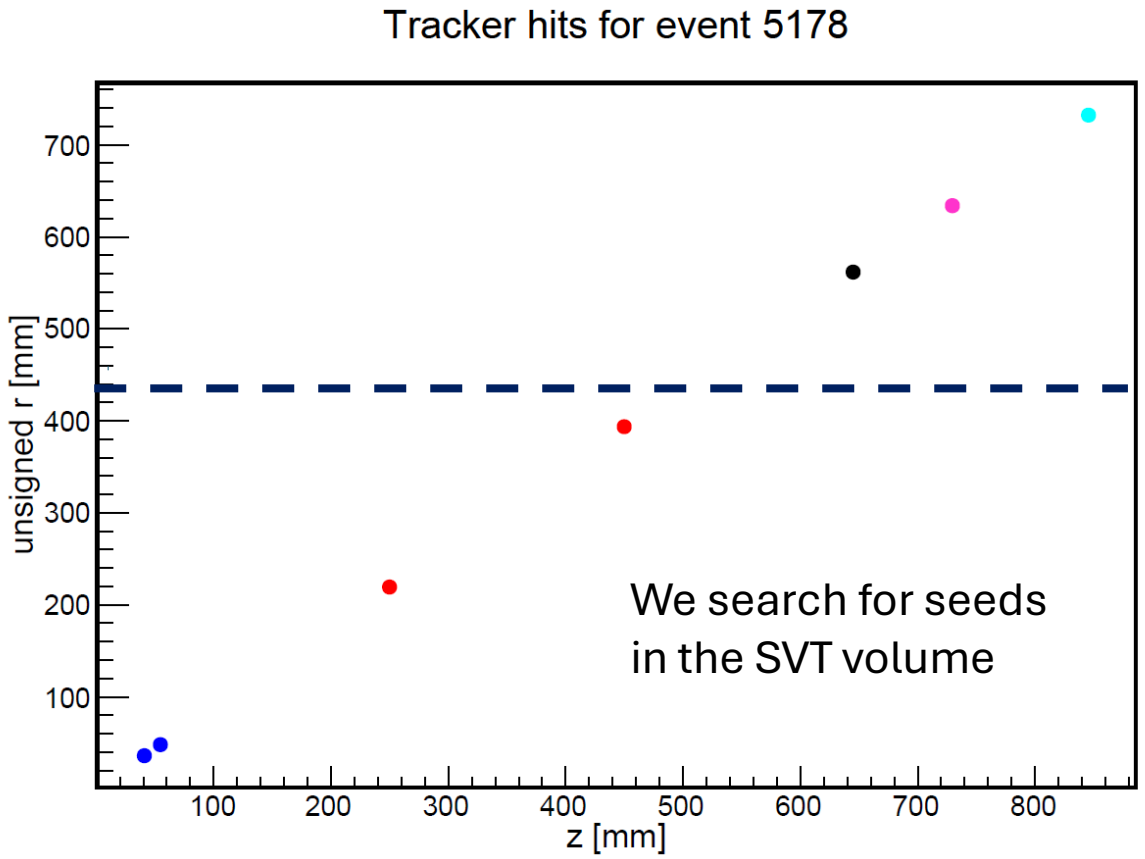
Truth-seeded tracking

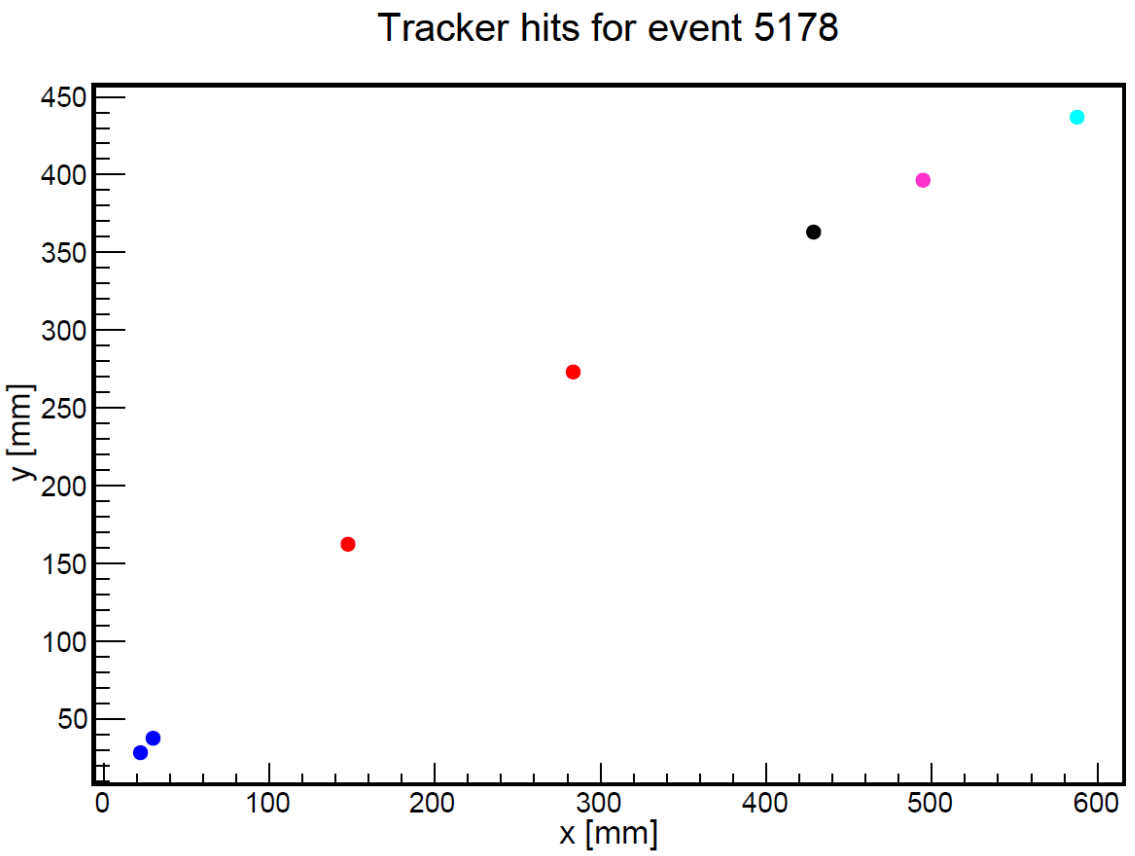
See Oct. 10th presentation

Si Endcap
Si Vertex
Si Barrel
Fwd MPGD
Bwd MPGD
Barrel MPGD
Out Barrel MPGD
TOF Endcap
TOF Barrel

Example single-particle event where seeds are formed

Generated particle: Negative muon
P_{tot} = 1 GeV/c; eta = +0.974; P_t = 661 MeV/c

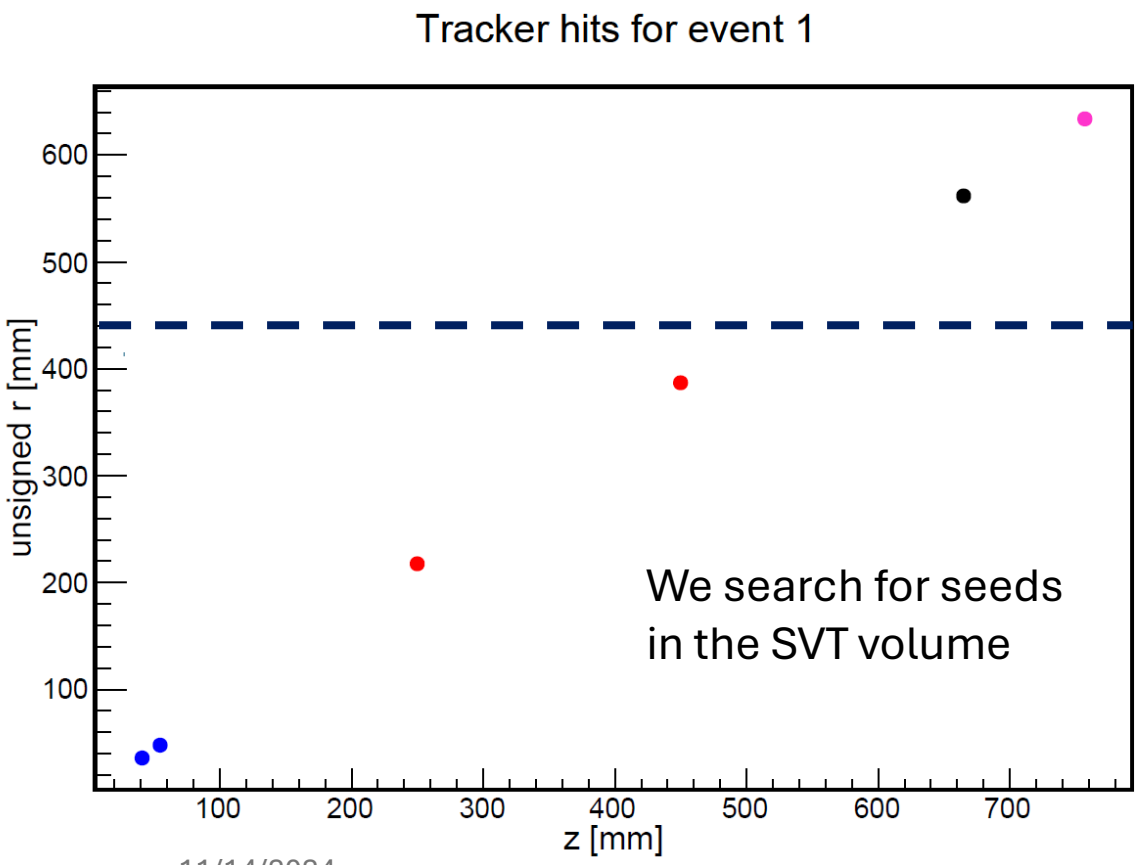


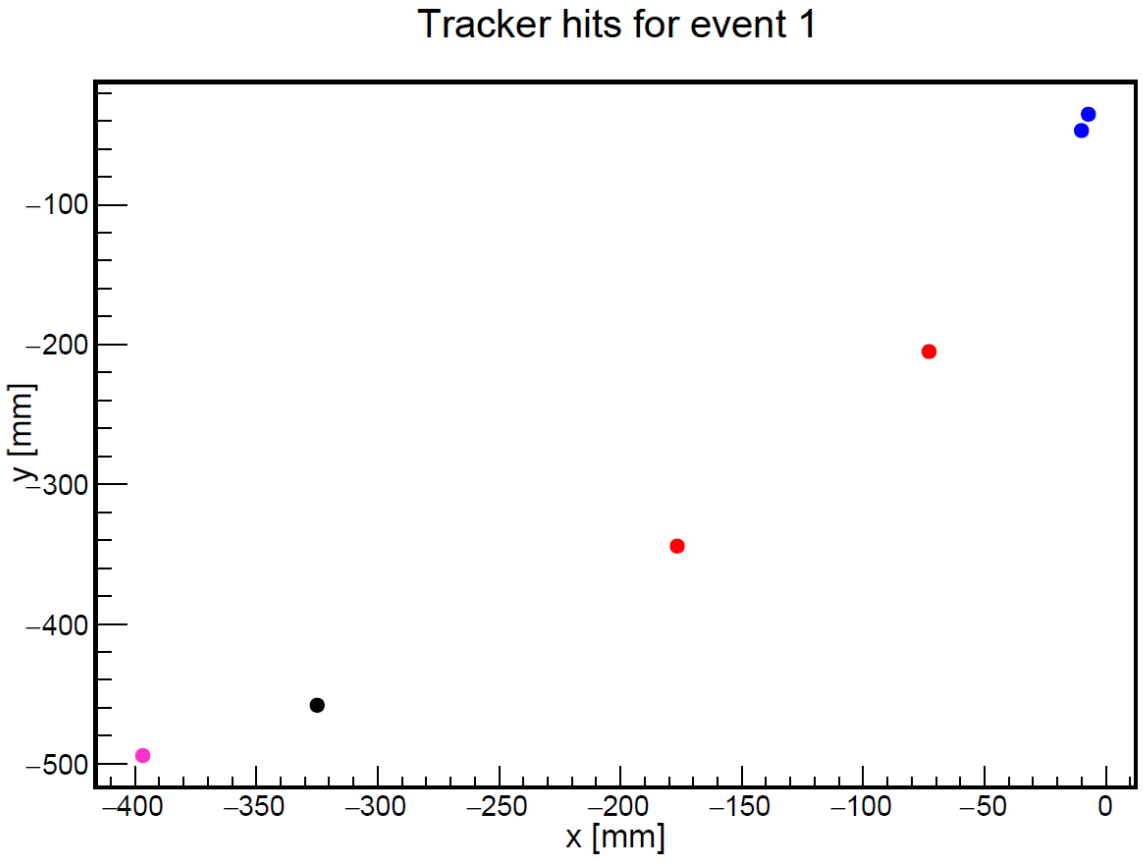


Si Endcap
Si Vertex
Si Barrel
Fwd MPGD
Bwd MPGD
Barrel MPGD
Out Barrel MPGD
TOF Endcap
TOF Barrel

Example single-particle event where no seeds are formed

Generated particle: Negative muon
P_{tot} = 0.5 GeV/c; eta = +0.975; P_t = 330 MeV/c
Generated particle's end point: (r,z) = (1227.9650 , 1528.9735) mm





Criteria to form a seed

- The Acts documentation mentions three criteria that need to be satisfied for a given triplet of space points to form a seed:
  1. Difference between middle-top and middle-bottom angles in r-z plane

$$(\cot\theta_b - \cot\theta_t)^2 < \sigma^2_{p_T^{estimated}} + \sigma^2_{f}$$

  2. Bend radius must be larger than a configurable minimum value

$$\frac{A^2+1}{B^2} > (2R^{min})^2 = \left(\frac{2\cdot p_T^{min}}{300\cdot B_z}\right)^2$$

  3. PoCA of circle in (x,y) plane to (x,y) = (0,0) must be smaller than a configurable value

$$d_0 \leq |(A - B\cdot r_M)\cdot r_M|$$

- The event on the last slide satisfies all these requirements.

Criteria to form a seed

- The Acts documentation also discusses the formation of a grid and grouping of the space points:

The SPs in each detector layer are projected on a rectangular grid of configurable granularity. The search for seed starts by selecting SP in the middle detector layer. Then matching SPs are searched in the inner and outer layers. Grouping of the SPs in the aforementioned grid allows to limit the search to neighbouring grid cells thus improving significantly algorithm performance (see Fig. 32). The number of neighboring bins used in the SP search can be defined separately for the bottom and top layer SPs in the z and ϕ directions.

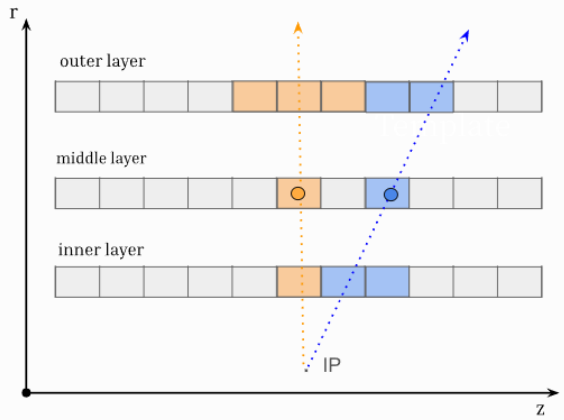


Fig. 32 *Representation of the search for triplet combinations in the (r, z) plane. The bins used in the search are represented in different colours.*

Space point formation in EICRecon

The *eicrecon::SpacePoint* class is used in the seed finder:

```
std::vector<const eicrecon::SpacePoint*> spacePoints = getSpacePoints(trk_hits);

std::function<std::tuple<Acts::Vector3, Acts::Vector2, std::optional<Acts::ActsScalar>>(
    const eicrecon::SpacePoint *sp)>
    create_coordinates = [](const eicrecon::SpacePoint *sp) {
      Acts::Vector3 position(sp->x(), sp->y(), sp->z());
      Acts::Vector2 variance(sp->varianceR(), sp->varianceZ());
      return std::make_tuple(position, variance, sp->t());
    };


eicrecon::SeedContainer seeds = finder.createSeeds(m_seedFinderOptions, spacePoints, create_coordinates);
```

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackSeeding.cc

Space point formation in EICRecon

The eicrecon::SpacePoint class derives from the edm4eic::TrackerHit class. The positions and uncertainties are calculated as

```
float x() const { return getPosition()[0]; }
float y() const { return getPosition()[1]; }
float z() const { return getPosition()[2]; }
float r() const { return std::hypot(x(), y()); }
float varianceR() const
{
  return (std::pow(x(), 2) * getPositionError().xx +
          std::pow(y(), 2) * getPositionError().yy) /
    (std::pow(x(), 2) + std::pow(y(), 2));
}
float varianceZ() const { return getPositionError().zz; }
```

Issue: as tracker hit uncertainties as saved in local coordinates, this leads to incorrect uncertainties for barrel layers

```
SiBarrelVertexRecHits = (vector<edm4eic::TrackerHitData>*)0x556999fa7e90
SiBarrelVertexRecHits.cellID = 17678317541925442079, 17870283364372959519
SiBarrelVertexRecHits.position.x = -10.070501, -7.219409
SiBarrelVertexRecHits.position.y = -46.937229, -35.269253
SiBarrelVertexRecHits.position.z = 54.599998, 40.959999
SiBarrelVertexRecHits.positionError.xx = 0.000033, 0.000033
SiBarrelVertexRecHits.positionError.yy = 0.000033, 0.000033
SiBarrelVertexRecHits.positionError.zz = 0.000000, 0.000000
SiBarrelVertexRecHits.time = 15.620000, -1.313000
SiBarrelVertexRecHits.timeError = 10.000000, 10.000000
SiBarrelVertexRecHits.edep = 0.000018, 0.000013
SiBarrelVertexRecHits.edepError = 0.000000, 0.000000
```

```
SiEndcapTrackerRecHits = (vector<edm4eic::TrackerHitData>*)0x55699a0b8550
SiEndcapTrackerRecHits.cellID = 2271217463926485326, 1121958896408232269
SiEndcapTrackerRecHits.position.x = -176.598190, -72.853355
SiEndcapTrackerRecHits.position.y = -344.277039, -205.074936
SiEndcapTrackerRecHits.position.z = 449.864990, 249.865005
SiEndcapTrackerRecHits.positionError.xx = 0.000033, 0.000033
SiEndcapTrackerRecHits.positionError.yy = 0.000033, 0.000033
SiEndcapTrackerRecHits.positionError.zz = 0.000000, 0.000000
SiEndcapTrackerRecHits.time = 17.409000, -0.367000
SiEndcapTrackerRecHits.timeError = 10.000000, 10.000000
SiEndcapTrackerRecHits.edep = 0.000032, 0.000020
SiEndcapTrackerRecHits.edepError = 0.000000, 0.000000
```

Variance = 20 um /sqrt(12) x 20 um /sqrt(12) = 3.3 x 10^{-5} mm

Proposed fix and question

- Sort the eicrecon::SpacePoint collection based on whether a given SpacePoint is bound to a barrel or end-cap layer:

  hit.getCellID()&0xFF

- Do we only need to pass in r and z variances to the seed finder?