Integration of ACTS with CEPCSW

YiZhou Zhang, Xiaocong Ai, Tao Lin, WeiDong Li
zhangyz@ihep.ac.cn

23rd July 2024

Outline

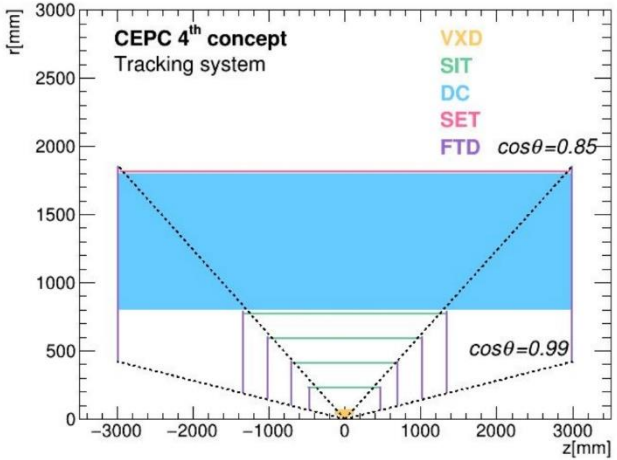
1 Introduction

Circular Electron Positron Collider (CEPC)

The CEPC is a future experiment mainly designed to precisely measure the Higgs boson's properties and search for new physics beyond the Standard Model.

- At 250 GeV: Higgs bosons are produced (4×10^6)
- At 160 GeV: W bosons are produced ($> 10^8$)
- At 90 GeV: Z bosons are produced ($> 4 \times 10^{12}$)

The conceptual design report (CDR) has been completed in Oct. 2018.



Schematic view of CEPC detector

Overview of This Contribution

The Technical Design Report (TDR) of CEPC is now being written.
https://arxiv.org/abs/2312.14363

We plan to apply ACTS' reconstruction tool in the reference detector of TDR, and to compare its performance with our origin reconstruction algorithm.

This Contribution will introduce the integration of ACTS with CEPC software (CEPCSW) environment.

Code working in progress:
https://code.ihep.ac.cn/zhangyz/cepcsw-acts/-/tree/master/Reconstruction/InDetActsTracking

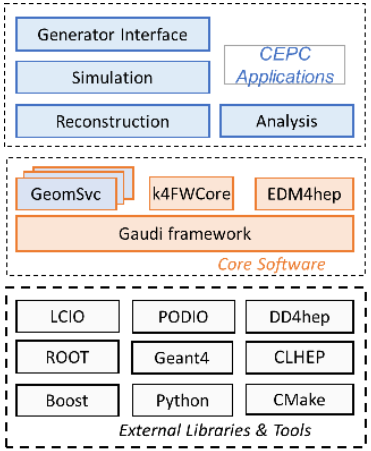
1 Introduction

CEPC software (CEPCSW) environment

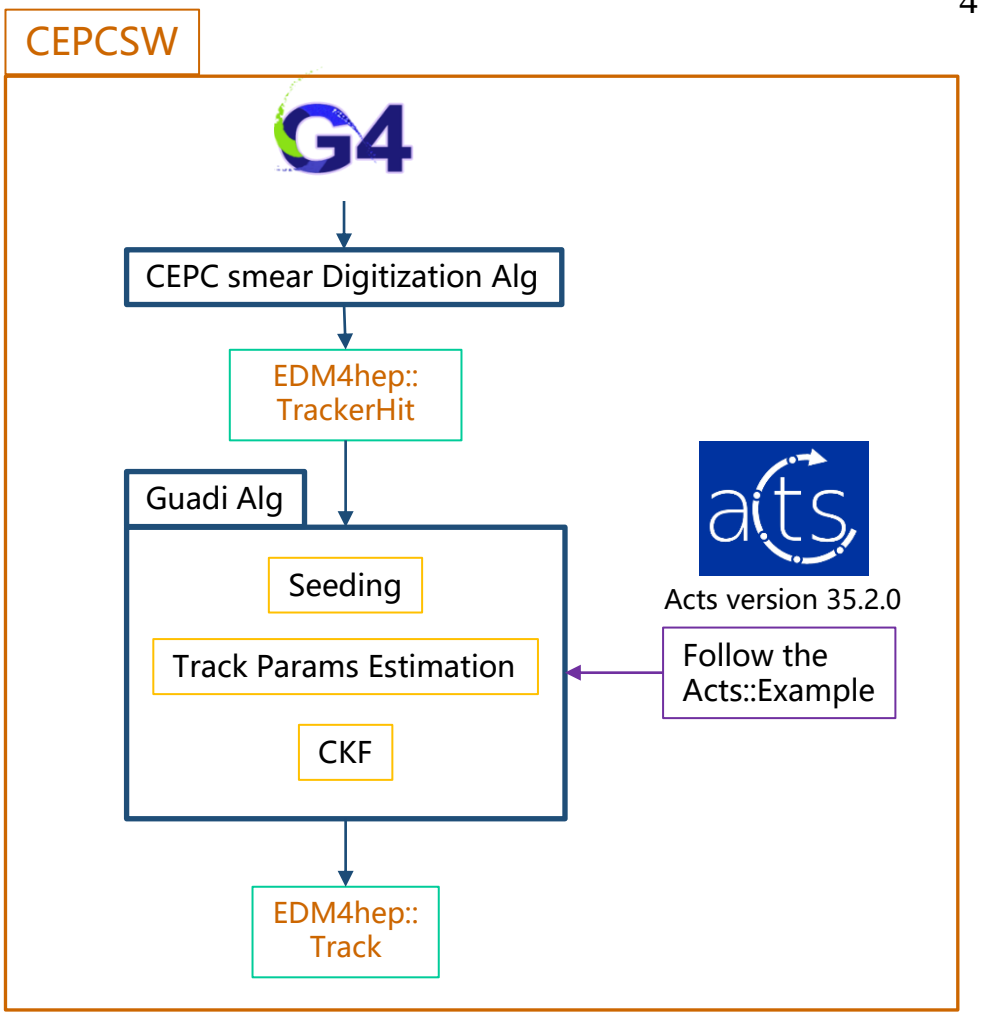
Applications: simulation, reconstruction and analysis

Core software:

- framework: Gaudi
- detector description tool: DD4hep
- event data model: EDM4hep
- event data manager: k4FWCore
- Other CEPC-specific components

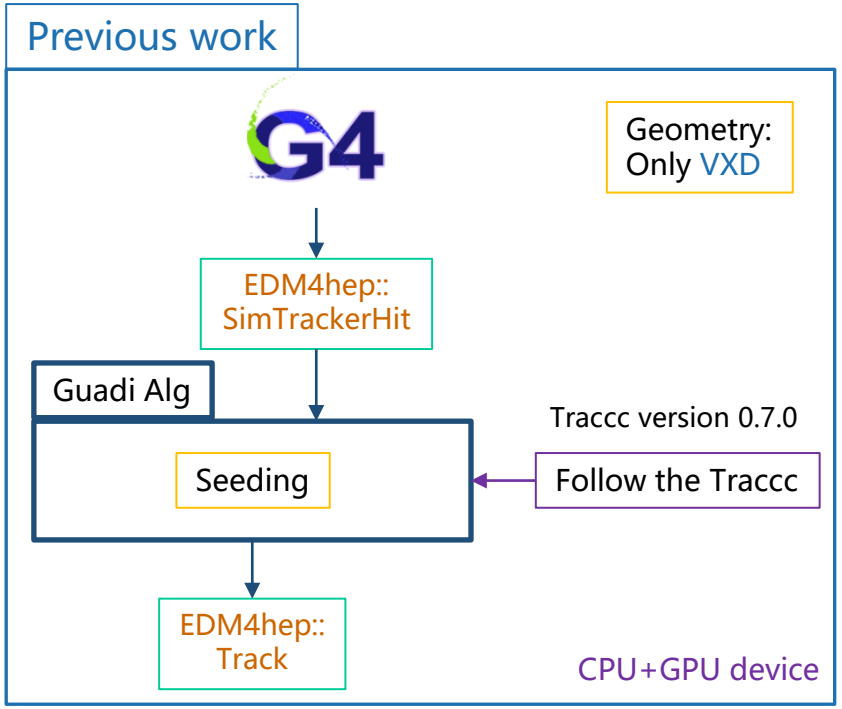


CEPCSW structure



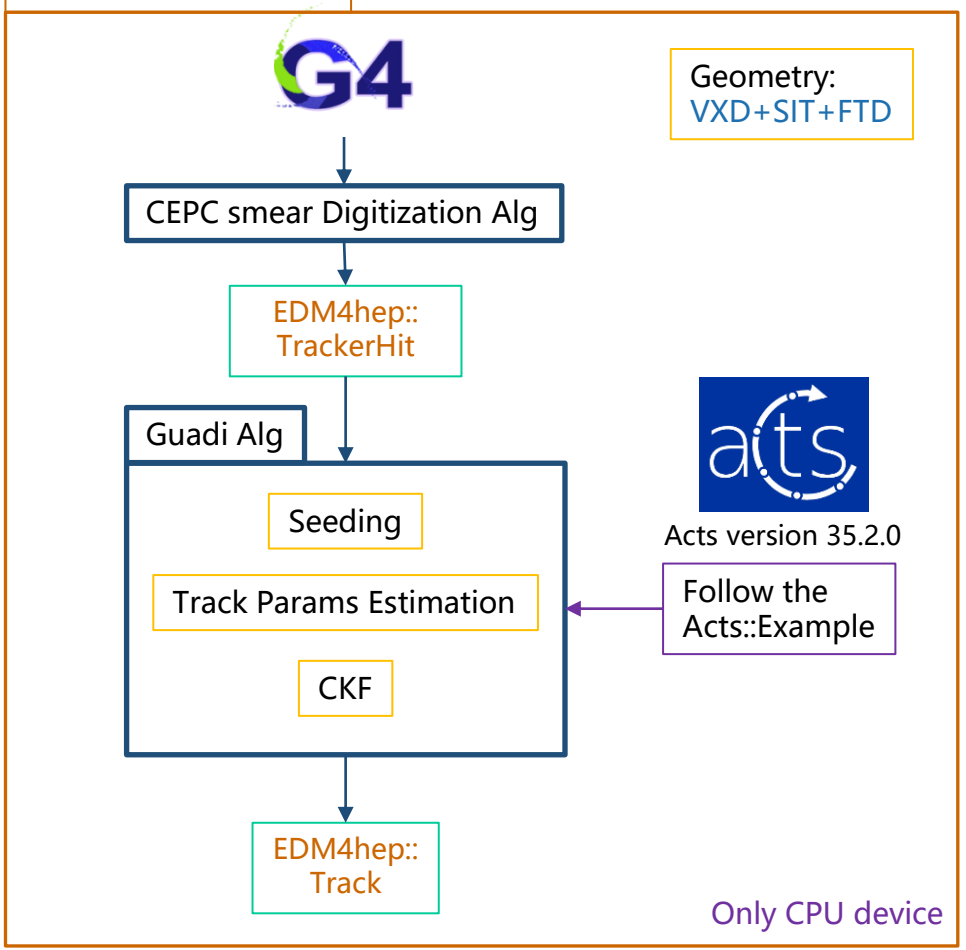
1 Introduction

Previous work



Previous talk:
https://indico.cern.ch/event/1388561/

This contribution



Outline

2 CEPC Geometry in ACTS

Geometry Conversion

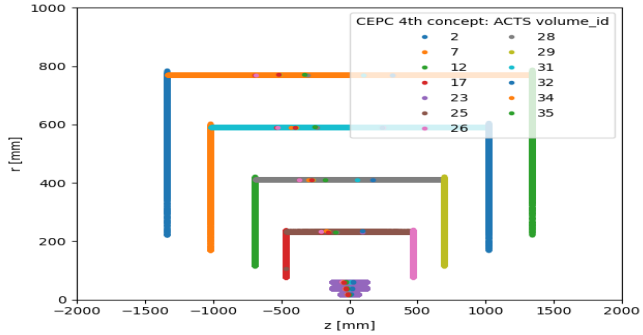
CEPCSW describes the geometry using DD4hep.

1. Convert the CEPC geometry file to tgeo format
2. Write the config file to specify the volumes (VXD, SIT, and FTD) that needs to be generated.
3. Follow the Acts' material mapping and validation tutorials, using the acts python library to generate material mapping json file.

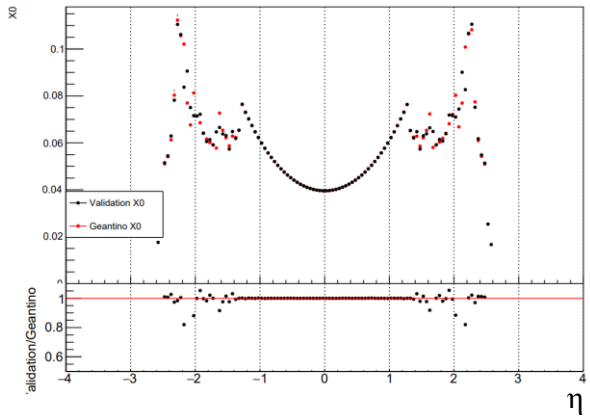
Following the ActsExamples::buildTGeoDetector, we wrote a function that can read the TGeo root file & TGeo config file & material map file, and get *Acts::TrackingGeometry and a vector of *Acts::TGeoDetectorElement.

https://code.ihep.ac.cn/zhangyz/cepcsw-acts/-/blob/master/Reconstruction/InDetActsTracking/src/utils/TGeoDetector.hpp

We now only consider the VXD + SIT, and will add FTD in the future.



FATRAS generates hits in z-r plane
VXD + SIT + FTD (layer 0-3)



Validation of Material Mapping

2 CEPC Geometry in ACTS

Gid Conversion

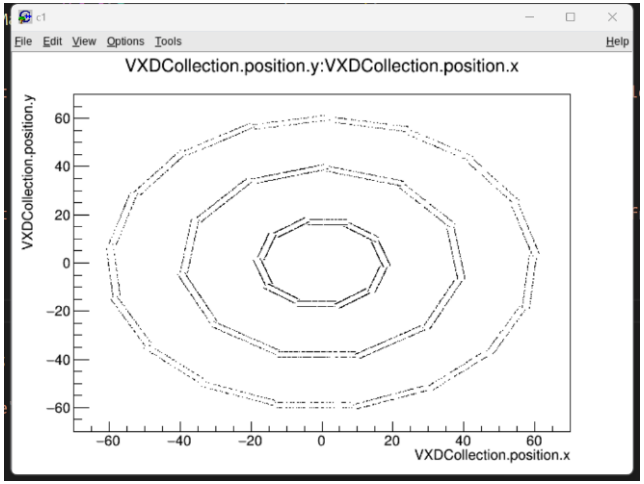
CEPCSW & ACTS use different geometry id format.
To get the correct module, the cell id of EDM4hep::TrackerHit need to be converted to Acts::GeometryIdentifier.

VXD CEPCSW cellid:

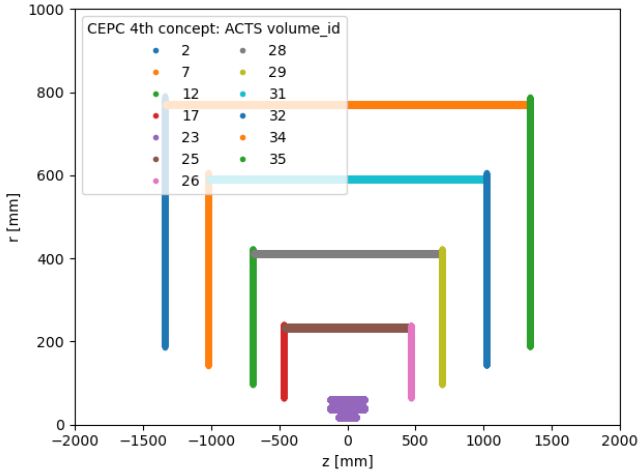
Layer: {0,1,2,3,4,5} # Indicate 6 layers from inside to outside
Module: { L0: 0-9, L1: 0-9,
L2: 0-10, L3: 0-10,
L4: 0-16, L5: 0-16} # Indicate ladders in the φ direction
Sensor: 0
Barrelside: 1 for z > 0 else -1 # one ladders has 2 sensors separated by z

VXD ACTS gid:

volume: {23}
Boundary: 0
Layer: {2, 4, 6} # adjacent layers are too close, so being treated as the same layers
Approach: 0
Sensitive: {L2: 1-40, L4: 1-44, L6:1-68}
The sensitive counts from z>0 to z<0, then counts in φ direction (the order is same to CEPC), and then counts from inner to outer layers.



Generated hits of CEPC VXD by Geant4



ACTS volume ids of VXD + SIT + FTD

2 CEPC Geometry in ACTS

Gid Conversion

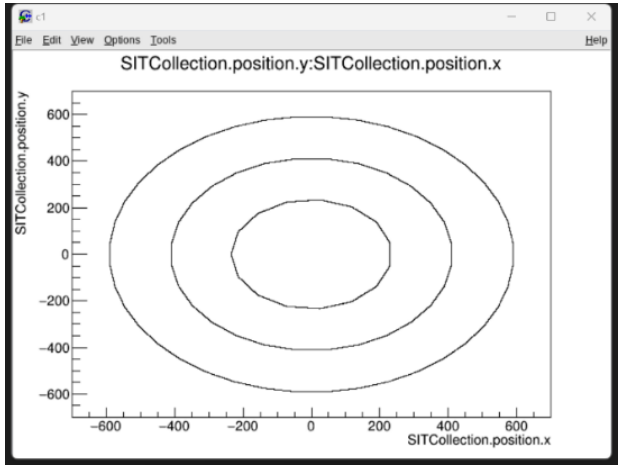
In CEPCSW, the outermost layer of SIT is considered in drift chamber.
We now only consider the inner 3 layers.

SIT CEPCSW cellid:

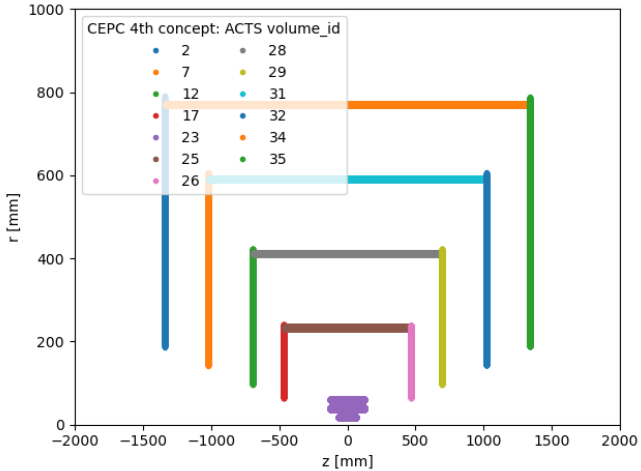
Layer: {0,1,2} # Indicate 3 layers from inside to outside
Module: {L0: 0-14, L1: 0-27, L2: 0-39} # Indicate ladders in the φ direction
Sensor: {L0: 0-9, L1: 0-14, L2:0-21} } # Indicate sensors in the z direction
Barrelside: 0

SIT ACTS gid:

volume: {25, 28, 31} # Indicate 3 layers from inside to outside
Boundary: 0
Layer: 2
Approach: 0
Sensitive: {vol25: 1-150, vol28: 1-420, vol31:1-880}
The sensitive counts in z direction (range z from large to small), then counts in φ direction (the order is same to CEPC), and then counts from inner to outer layers.



Generated hits of CEPC SIT by Geant4



ACTS volume ids of VXD + SIT + FTD

2 CEPC Geometry in ACTS

```
// uint64_t VXD_acts_volume_id = 23;
// std::vector<uint64_t> VXD_NumOfSensors_per_layer{20, 22, 34};
uint64_t barrel_sign = (m_barrelside == 1) ? 1 : 2;
uint64_t acts_volume = VXD_acts_volume_id;
uint64_t acts_boundary = 0;
uint64_t acts_layer = 2 * (m_layer >> 1) + 2;
uint64_t acts_approach = 0;
uint64_t acts_sensitive = VXD_NumOfSensors_per_layer[m_layer >> 1] * (m_layer & 1)
                        + 2 * m_module + barrel_sign;

// set acts geometry identifier
Acts::GeometryIdentifier moduleGeoId;
moduleGeoId.setVolume(acts_volume);
moduleGeoId.setBoundary(acts_boundary);
moduleGeoId.setLayer(acts_layer);
moduleGeoId.setApproach(acts_approach);
moduleGeoId.setSensitive(acts_sensitive);
```

Converter for VXD gid

```
// std::vector<uint64_t> SIT_acts_volume_ids{25, 28, 31};
// std::vector<uint64_t> SIT_NumOfSensors_per_module{10, 15, 22};
uint64_t acts_volume = SIT_acts_volume_ids[m_layer];
uint64_t acts_boundary = 0;
uint64_t acts_layer = 2;
uint64_t acts_approach = 0;
uint64_t acts_sensitive = SIT_NumOfSensors_per_module[m_layer] * m_module
                        + m_sensor + 1;

// set acts geometry identifier
Acts::GeometryIdentifier moduleGeoId;
moduleGeoId.setVolume(acts_volume);
moduleGeoId.setBoundary(acts_boundary);
moduleGeoId.setLayer(acts_layer);
moduleGeoId.setApproach(acts_approach);
moduleGeoId.setSensitive(acts_sensitive);
```

Converter for SIT gid

Validation of Gid Conversion

1. We get the global & local position of EDM4hep::TrackerHit.
2. Give the local position & converted Gid to Acts::Surface, if the gid conversion is correct, Acts::Surface can get the correct global position.

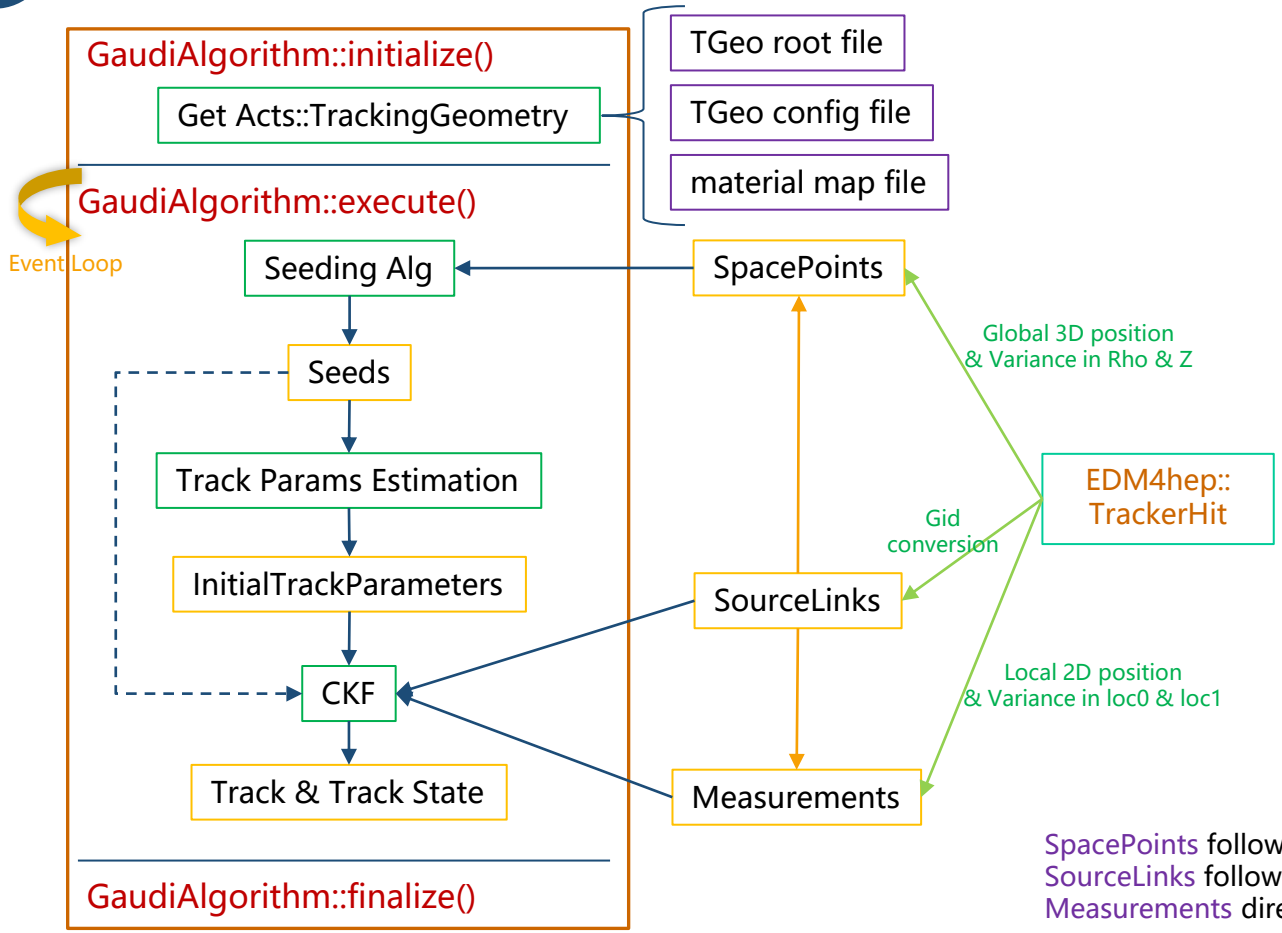
The conversion has been validated.

```
auto simcellid = simhit.getCellID();
Acts::GeometryIdentifier moduleGeoId = getVTXGid(simcellid);
const Acts::Vector2 local_position{loc0, loc1};
const auto& surface = trackingGeometry->findSurface(moduleGeoId);
auto global_postion = surface->localToGlobal(geoContext, local_position, globalFakeMom);
info() << "VXD global position(x,y,z): " << simhit.getPosition()[0] << ", "
                                          << simhit.getPosition()[1] << ", "
                                          << simhit.getPosition()[2]
debug() << "converted acts position(x,y,z): " << global_postion[0] << ", "
                                               << global_postion[1] << ", "
                                               << global_postion[2] << endmsg;
```

Code to check Gid Conversion

Outline

3 Integration of ACTS with CEPCSW



Gaudi Algorithm View

Initialize: read the CEPC geometry, get the Acts::TrackingGeometry.

Execute: read the EDM4hep:: TrackerHit of current event, generate SpacePoints & measurements, and store the connection between ModuleGid & meas idx into SourceLinks.

Follow the Acts::Example, we write the Seeding Alg & Track Params Estimation & CKF using Acts tools.
And finally get the track & track state stored in Acts::TrackContainer.

SpacePoints follows ActsExamples::SimSpacePoint
SourceLinks follows ActsExamples::IndexSourceLink
Measurements directly use Acts::BoundVariantMeasurement

3 Integration of ACTS with CEPCSW

Get local position from EDM4hep

EDM4hep::TrackerHit do not directly provide the local position.

We set the segmentations in the cellid to get the local position.

The grid size of both VTX & SIT is set to 25um.

```
auto cellid = hit.getCellID();
double acts_loc0 = sit_decoder->get(cellid, "y") * grid_size;
double acts_loc1 = sit_decoder->get(cellid, "z") * grid_size;
```

```
// create and store the measurement
const std::array<Acts::BoundIndices, 2> indices{Acts::BoundIndices::eBoundLoc0, Acts::BoundIndices::eBoundLoc1};
Acts::ActsVector<2> par{acts_loc0, acts_loc1};
Acts::ActsSquareMatrix<2> cov = Acts::ActsSquareMatrix<2>::Zero();
measurements.emplace_back(Acts::Measurement<Acts::BoundIndices, 2>(std::move(sl), indices, par, cov));
```

Get and store the local position in measurement

```
<readout name="VXDCollection">
  <!-- <id>system:5,side:-2,layer:9,module:8,sensor:8,barrelside:-2</id> -->
  <segmentation type="CartesianGridXY" grid_size_x="25*um" grid_size_y="25*um"/>
  <id>system:5,side:-2,layer:9,module:8,sensor:8,barrelside:-2,x:-11,y:-14</id>
</readout>
```

VXD

Module size (x-y × z direction)

Layer 0, 1:

11mm*62.5mm

880 * 5000 (25um/bin)

Layer 2, 3, 4, 5:

22mm*125mm

880 * 5000 (25um/bin)

```
<readout name="SITCollection">
  <!-- <id>system:5,side:-2,layer:9,module:8,sensor:8,barrelside:-2</id> -->
  <segmentation type="CartesianGridYZ" grid_size_y="25*um" grid_size_z="25*um"/>
  <id>system:5,side:-2,layer:9,module:8,sensor:8,barrelside:-2,y:-13,z:-13</id>
</readout>
```

SIT

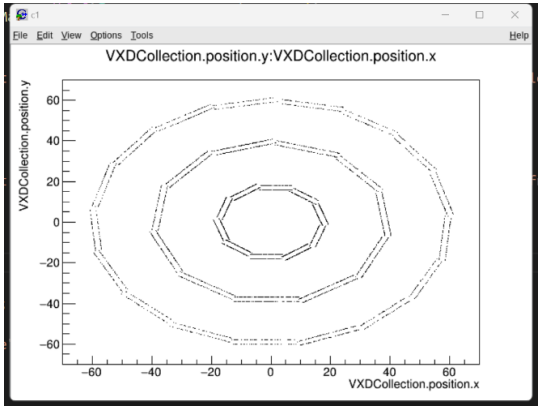
Module size (x-y × z direction)

Layer 0, 1, 2:

97.55mm*91.85mm

3902 * 3674 (25um/bin)

3 Integration of ACTS with CEPCSW

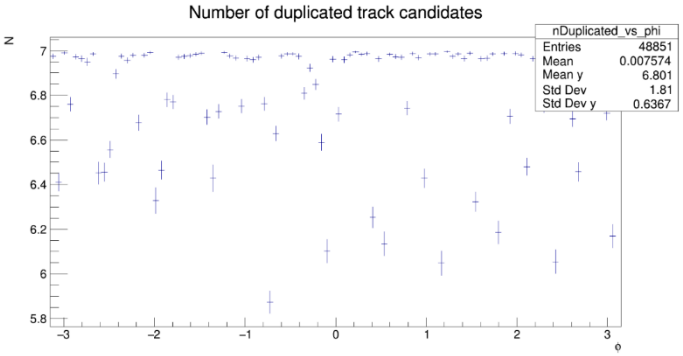


Generated hits of CEPC VXD by Geant4

Seeding for CEPC

Due to the three-layer, two-sided construction of the CEPC VTX, the algorithm that makes up three sp's SEEDs will result in a large duplication rate.

Assuming that each instance has hits in each layer of the VTX, this would result in 2^3-1=7 duplicate seeds.



Duplicated track candidates with Φ

```
InDetActsFullChain    INFO found seed #0: x:-0.149146 y: -15.975 z: -15.1685
InDetActsFullChain    INFO found seed #0: x:-0.333171 y: -36.975 z: -35.1067
InDetActsFullChain    INFO found seed #0: x:-0.498747 y: -57.975 z: -55.0443
InDetActsFullChain    INFO found seed #1: x:-0.149146 y: -15.975 z: -15.1685
InDetActsFullChain    INFO found seed #1: x:-0.333171 y: -36.975 z: -35.1067
InDetActsFullChain    INFO found seed #1: x:-0.513727 y: -60.025 z: -56.9906
InDetActsFullChain    INFO found seed #2: x:-0.167633 y: -18.025 z: -17.1149
InDetActsFullChain    INFO found seed #2: x:-0.333171 y: -36.975 z: -35.1067
InDetActsFullChain    INFO found seed #2: x:-0.498747 y: -57.975 z: -55.0443
InDetActsFullChain    INFO found seed #3: x:-0.167633 y: -18.025 z: -17.1149
InDetActsFullChain    INFO found seed #3: x:-0.333171 y: -36.975 z: -35.1067
InDetActsFullChain    INFO found seed #3: x:-0.513727 y: -60.025 z: -56.9906
InDetActsFullChain    INFO found seed #4: x:-0.149146 y: -15.975 z: -15.1685
InDetActsFullChain    INFO found seed #4: x:-0.349943 y: -39.025 z: -37.053
InDetActsFullChain    INFO found seed #4: x:-0.498747 y: -57.975 z: -55.0443
InDetActsFullChain    INFO found seed #5: x:-0.149146 y: -15.975 z: -15.1685
InDetActsFullChain    INFO found seed #5: x:-0.349943 y: -39.025 z: -37.053
InDetActsFullChain    INFO found seed #5: x:-0.513727 y: -60.025 z: -56.9906
InDetActsFullChain    INFO found seed #6: x:-0.167633 y: -18.025 z: -17.1149
InDetActsFullChain    INFO found seed #6: x:-0.349943 y: -39.025 z: -37.053
InDetActsFullChain    INFO found seed #6: x:-0.498747 y: -57.975 z: -55.0443
InDetActsFullChain    INFO found seed #7: x:-0.167633 y: -18.025 z: -17.1149
InDetActsFullChain    INFO found seed #7: x:-0.349943 y: -39.025 z: -37.053
InDetActsFullChain    INFO found seed #7: x:-0.513727 y: -60.025 z: -56.9906
InDetActsFullChain    INFO Created 8 track seeds from 6 space points
InDetActsFullChain    INFO ---------- track estimation -------------
InDetActsFullChain    INFO Created 8 track Parameters from 6 space points
InDetActsFullChain    INFO creating CKF Navigator Config ...
InDetActsFullChain    INFO creating CKF finder ...
InDetActsFullChain    INFO Finalized track finding with 8 track candidates.
```

Created 8 tracks from 6 space points

3 Integration of ACTS with CEPCSW

CKF for CEPC

When iterating the found seeds, CKF finds all the measurements of the current track, and records all the possible triplets in an unordered_map discoveredSeeds.

CKF will skip the seed who is already in the discoveredSeeds, which avoid the duplication happened in seeding algorithm.

Here is an example, CKF correctly found the 6 meas in VTX and 3 meas in SIT.

```
auto addTrack = [&](const TrackProxy& track)
{
    ++m_nFoundTracks;

    // flag seeds which are covered by the track
    visitSeedIdentifiers(track, [&](const SeedIdentifier& seedIdentifier)
    {
        if (auto it = discoveredSeeds.find(seedIdentifier); it != discoveredSeeds.end())
        { it->second = true; }
    });

    if (m_trackSelector.has_value() && !m_trackSelector->isValidTrack(track)) { return; }

    ++m_nSelectedTracks;
    auto destProxy = tracks.makeTrack();
    // make sure we copy track states!
    destProxy.copyFrom(track, true);
};
```

ActsExamples::visitSeedIdentifiers records all the possible triplets in discoveredSeeds

```
INFO get track: theta 2.33024, phi -1.58069, loc0 0.00310861, loc1 -0.0134383, qOverP -9.89904e-06, absoluteMomentum 101020, nMeasurements 9
INFO ckf track state position(x,y,z): -0.14987, -15.975, -15.175
INFO ckf track state position(x,y,z): -0.17487, -18.025, -17.125
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -0.337941, -36.975, -35.1
INFO ckf track state position(x,y,z): -0.337941, -39.025, -37.05
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -0.486744, -57.975, -55.05
INFO ckf track state position(x,y,z): -0.511744, -60.025, -57
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -1.17643, -231.229, -219.525
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -0.575, -410.085, -389.35
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): 1.375, -590.085, -560.225
INFO Application Manager Stopped successfully
```

VTX

SIT

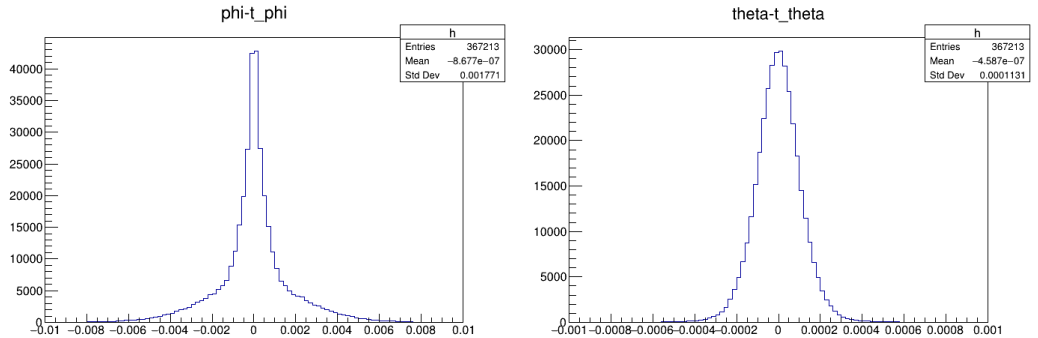
The momentum is too large (Gev)

3 Integration of ACTS with CEPCSW

Track Params for CEPC

The distribution of Δφ & Δθ looks good:
Δφ & Δθ : rec track angle – MC particle angle

Issue:
The momentum of MC particle is set to 30 Gev, but the rec track momentum is too high.



Distribution of Δφ & Δθ

```
INFO get track: theta 2.33024, phi -1.58069, loc0 0.00310861, loc1 -0.0134383, qOverP -9.89904e-06, absoluteMomentum 101020, nMeasurements 9
INFO ckf track state position(x,y,z): -0.14987, -15.975, -15.175
INFO ckf track state position(x,y,z): -0.17487, -18.025, -17.125
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -0.337941, -36.975, -35.1
INFO ckf track state position(x,y,z): -0.337941, -39.025, -37.05
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -0.486744, -57.975, -55.05
INFO ckf track state position(x,y,z): -0.511744, -60.025, -57
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -1.17643, -231.229, -219.525
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): -0.575, -410.085, -389.35
INFO cur track state has no uncalibrated source link
INFO cur track state has no uncalibrated source link
INFO ckf track state position(x,y,z): 1.375, -590.085, -560.225
INFO Application Manager Stopped successfully
```

VTX

SIT

The momentum is too large (Gev)

Thank You

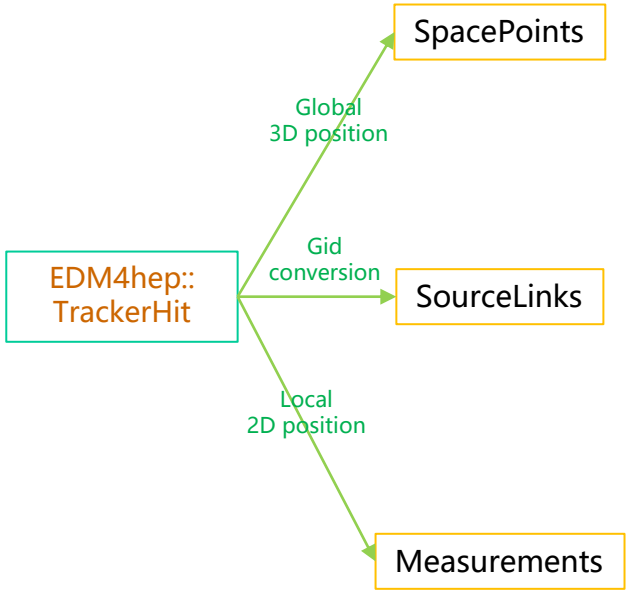
YiZhou Zhang, Xiaocong Ai, Tao Lin, WeiDong Li
zhangyz@ihep.ac.cn

23rd July 2024

Backup

3 Integration of ACTS with CEPCSW

Follow ActsExamples::SimSpacePoint




```
/// @brief Function to build the generic tracking geometry from a TGeo object.
///
/// @param TGeo_ROOTFilePath is the TGeo ROOT file path
/// @param TGeoConfig_jFilePath is the TGeo configuration file path
/// @param MaterialMap_jFilePath is the material map file path
/// @param logger is the logger object
/// @return a shared pointer to the tracking geometry
std::shared_ptr<const Acts::TrackingGeometry> buildTGeoDetector(
    const std::string& TGeo_ROOTFilePath,
    const std::string& TGeoConfig_jFilePath,
    const std::string& MaterialMap_jFilePath,
    const Acts::Logger& logger)
{
```

```
const Acts::Surface* surface = trackingGeometry->findSurface(moduleGeoId);
```

Seeding of ACTS in CEPCSW

输入：CEPCSW的G4模拟并数字化后产生的edm4hep::TrackerHit

```
// Input collections
DataHandle<edm4hep::TrackerHitCollection> _inVTXColHdl{"VXDTrackerHits", Gaudi::DataHandle::Reader, this};
```

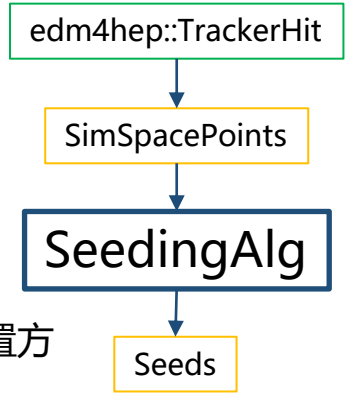
① 将edm4hep::TrackerHit转化为适用于acts seeding的space point数据格式(这里取名SimSpacePoint)。

ACTS Seeding需要使用的space points成员：

position：x, y, z, t(optional)

variance：Rho, Z, T(optional)

对于由两层strip组成的space points，还需要知道两层strip的位置方向信息。




```
class SimSpacePoint {

private:

    edm4hep::TrackerHit m_trackerHit;
    // Global position
    Scalar m_x;
    Scalar m_y;
    Scalar m_z;
    std::optional<Scalar> m_t;
    Scalar m_rho;
    // Variance in rho/z of the global coordinates
    Scalar m_varianceRho;
    Scalar m_varianceZ;
    std::optional<Scalar> m_varianceT;
    // SourceLinks of the corresponding measurements. A
    // (two) sourceLink(s).
    // boost::container::static_vector<Acts::SourceLink,

    // half of the length of the top strip
    float m_topHalfStripLength = 0;
    // half of the length of the bottom strip
    float m_bottomHalfStripLength = 0;
    // direction of the top strip
    Acts::Vector3 m_topStripDirection = {0, 0, 0};
    // direction of the bottom strip
    Acts::Vector3 m_bottomStripDirection = {0, 0, 0};
    // distance between the center of the two strips
    Acts::Vector3 m_stripCenterDistance = {0, 0, 0};
    // position of the center of the bottom strip
    Acts::Vector3 m_topStripCenterPosition = {0, 0, 0};
    bool m_validDoubleMeasurementDetails = false;

}; // SimSpacePoint
```

SimSpacePoint

② 在Gaudi算法initialize时，调整SeedingConfig的各项参数，并初始化Acts::SeedFinder。

③ 在Gaudi算法execute时，生成Grid并将sp Grouping到Grid上。然后运行Acts::SeedFinder::createSeedsForGroup

```
for (const auto [bottom, middle, top] : spacePointsGrouping)
{
    m_seedFinder.createSeedsForGroup(
        seed_cfg.seedFinderOptions, state, spacePointsGrouping.grid(),
        std::back_inserter(seeds), bottom, middle, top, rMiddleSPRange);
}
```

```
// configure the acts tools
m_cfg.seedFinderOptions.bFieldInZ = 3_T;
m_cfg.seedFinderConfig.deltaRMin = 8_mm;
m_cfg.seedFinderConfig.deltaRMax = 25_mm;
m_cfg.seedFinderConfig.rMax = 60_mm;
m_cfg.seedFinderConfig.rMin = 8_mm;
m_cfg.seedFinderConfig.impactMax = 4_mm;
m_cfg.seedFinderConfig.useVariableMiddleSPRange = false;
m_cfg.seedFinderConfig.rMinMiddle = 18_mm; // range for middle spacepoint
m_cfg.seedFinderConfig.rMaxMiddle = 36_mm; // range for middle spacepoint
```

Seeding Config