中国科学技术大学
University of Science and Technology of China

Intergration of ACTS with OSCAR

Hang Zhou

University of Science and Technology of China

the STCF tracking system

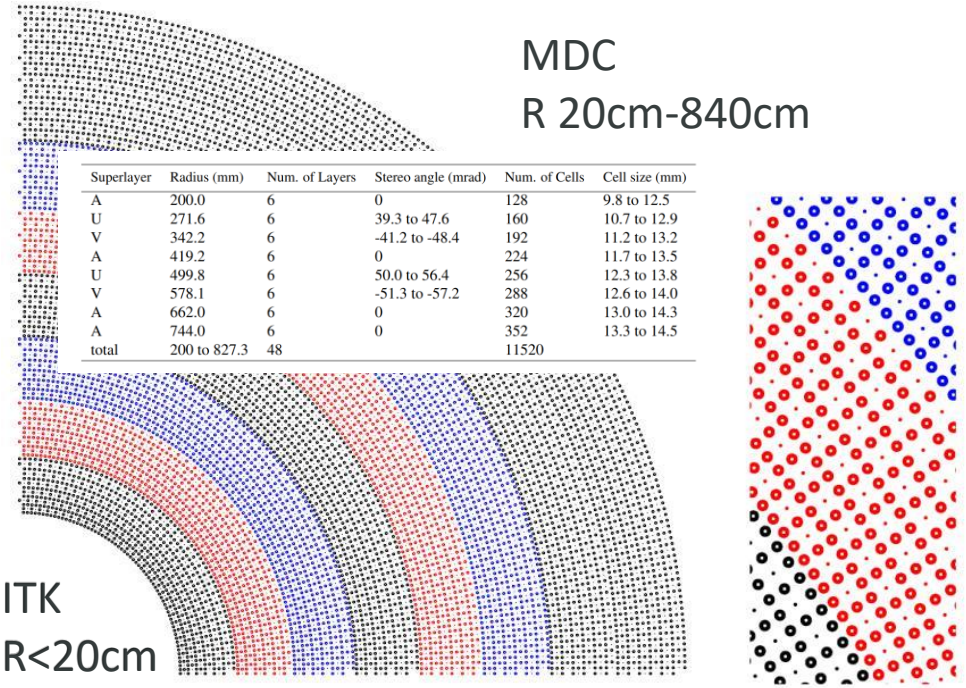
- Consists of **Inner Tracker(ITK)** and Main **Drift Chamber(MDC)**
- ITK: 3 layers

  Two options:

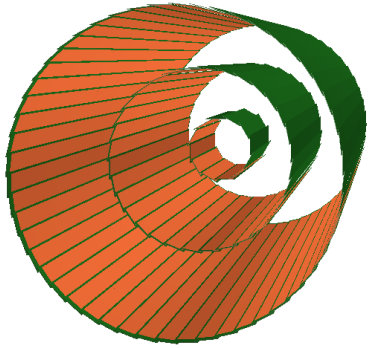
  Monolithic Active Pixel Sensor(MAPS)

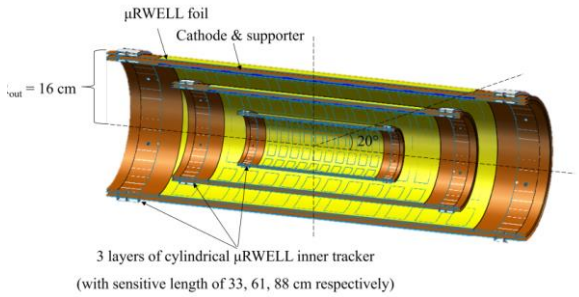
  Micro-Pattern Gaseous Detector(MPGD)
- MDC: 48 layers, 4 stereo super-layers, 4 axial super-layers

- detecting charged particle with a momentum range of 50MeV-3.5GeV



Superlayer	Radius (mm)	Num. of Layers	Stereo angle (mrad)	Num. of Cells	Cell size (mm)
A	200.0	6	0	128	9.8 to 12.5
U	271.6	6	39.3 to 47.6	160	10.7 to 12.9
V	342.2	6	-41.2 to -48.4	192	11.2 to 13.2
A	419.2	6	0	224	11.7 to 13.5
U	499.8	6	50.0 to 56.4	256	12.3 to 13.8
V	578.1	6	-51.3 to -57.2	288	12.6 to 14.0
A	662.0	6	0	320	13.0 to 14.3
A	744.0	6	0	352	13.3 to 14.5
total	200 to 827.3	48		11520	

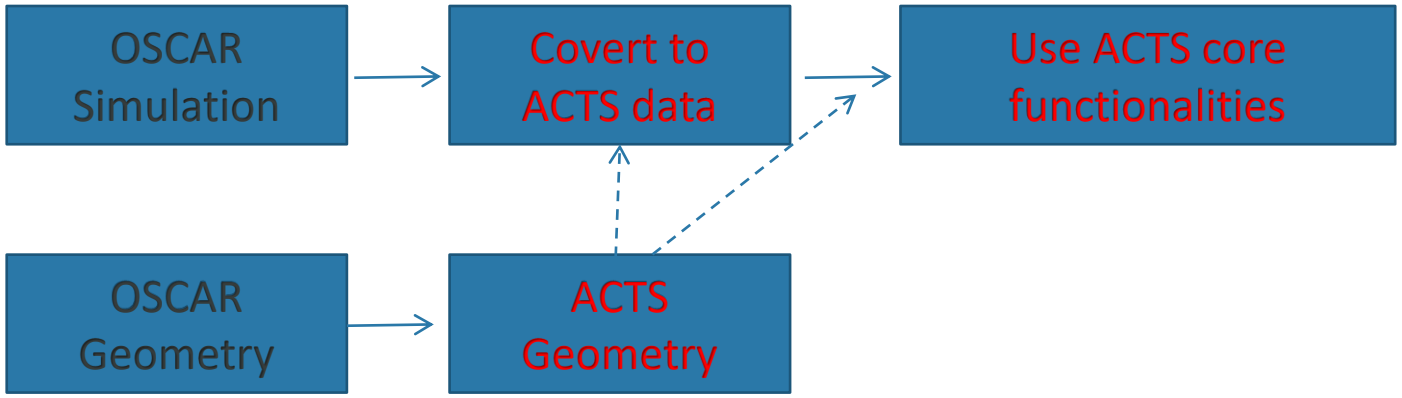




Intergration of ACTS with OSCAR

- Offline Software of Super Tau-Charm Facility(OSCAR),
  Underlying Framework: SNiPER, similar to Gaudi,Lightweighted

Use ACTS as an external library



What needs to be done:
Transform STCF geometry to ACTS geomtry
Create ACTS-style measurements
Invoke the ACTS module for track reconstruction

Software Version ACTS:v32 ROOT:6.22 Geant4:11.2

Geometry transformation

Construct ACTS geometry through Geant4 volume
cylindrical MPGD ITK

```
G4VPhysicalVolume *g4PhysVol = parser.GetWorldVolume();
auto g4LogicalVolume = g4PhysVol->GetLogicalVolume();
```

SensitiveVolume

ITK WorkGas-Cylinder surface
MDC wire-Line surface

PassiveVolume-Cylinder Layer

BeamPipe
ITK Support/electronic
MDC gas

```
Acts::ProtoVolume senGas1;
senGas1.name = "ITKL0";
senGas1.extent.set(Acts::binR, 60, 70);
senGas1.extent.set(Acts::binZ, -200., 200);
senGas1.internal = Acts::ProtoVolume::InternalStructure{
    Acts::Surface::SurfaceType::Cylinder};
```

Set range

```
if (daughter->GetName().find("Drift_Gas_") != std::string::npos)
{
  sensitiveVol.push_back(daughter->GetName());
  convertedSurfaceTypeMap.insert(std::make_pair(
      daughter->GetName(), Acts::Surface::SurfaceType::Other));
}
```

Set volume name to
be converted

Set the range of geometry
to be converted

↓

Set the name of geometry
to be converted

↓

Find Geant4 volumes
by name and range

↓

Read volume parameters
to construct ACTS geometry

Geometry transformation-issues

Acts::Geant4DetectorSurfaceFactory:

```
Acts::Geant4DetectorSurfaceFactory{}.construct(
    g4SurfaceCache, g4ToWorld, *cfg.g4World, cfg.g4SurfaceOptions);
```

Calculate translation using the read parameters,but seems not right

FrameTransform?
ObjectTransform?

Gean4 has two types transform

Acts::Geant4PhysicalVolumeConverter: G4Box,G4Trd->PlaneSurface G4Tubs->CylinderSurface

There is no function to convert volume to Acts::LineSurface

Calculate material properties, but it's not correct

Incorrectly treating mass fraction as mole fraction,

The conversion from Geant4 to ACTS units may be incorrect

The boundary calculation during the surface conversion has led to significant geometric overlaps;

The boundary calculation for LineSurface is incorrect

Calculating envelope parameters requires at least three distinct points, but a line has only two

Geometry transformation-issues

MDC geometry: 48 CylinderLayer, each layer consists of one passive CylinderSuface(gas) and N LineSurface(sensitive wire)

However, during Kalman fitting navigation, sometimes the passivelayer cannot be tracked, which prevents accurate calculation of material effects

why?

I force the material to be assigned to the CylinderLayer itself, which will enforce the calculation of material effects

I have made some modifications, and they seem to be effective for my application.

Convert measurement

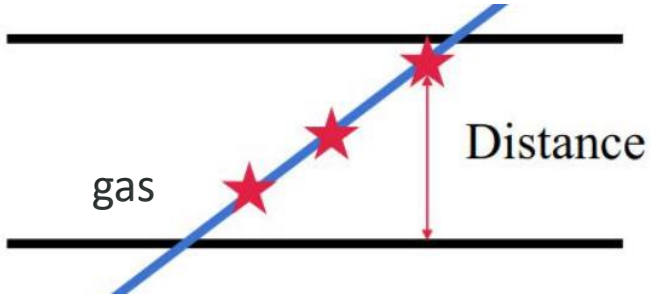
smear simulation hit, resoluton: ITK 100um x 400um MDC 120um

```
Acts::ActsVector<2> par{
  surfaceRadius[layID] * Acts::VectorHelpers::phi(pos) + gRandom->Gaus(0, 0.1),
  pos.z() + gRandom->Gaus(0, 0.4)};
Acts::ActsSquareMatrix<2> cov = Acts::ActsSquareMatrix<2>::Identity();
cov(0, 0) = 0.1 * 0.1;
cov(1, 1) = 0.4 * 0.4;
```

problems:

what's the proper value of covariance matrix?

It seems that the measurement point's R is fixed (tied to the sensitive surface).



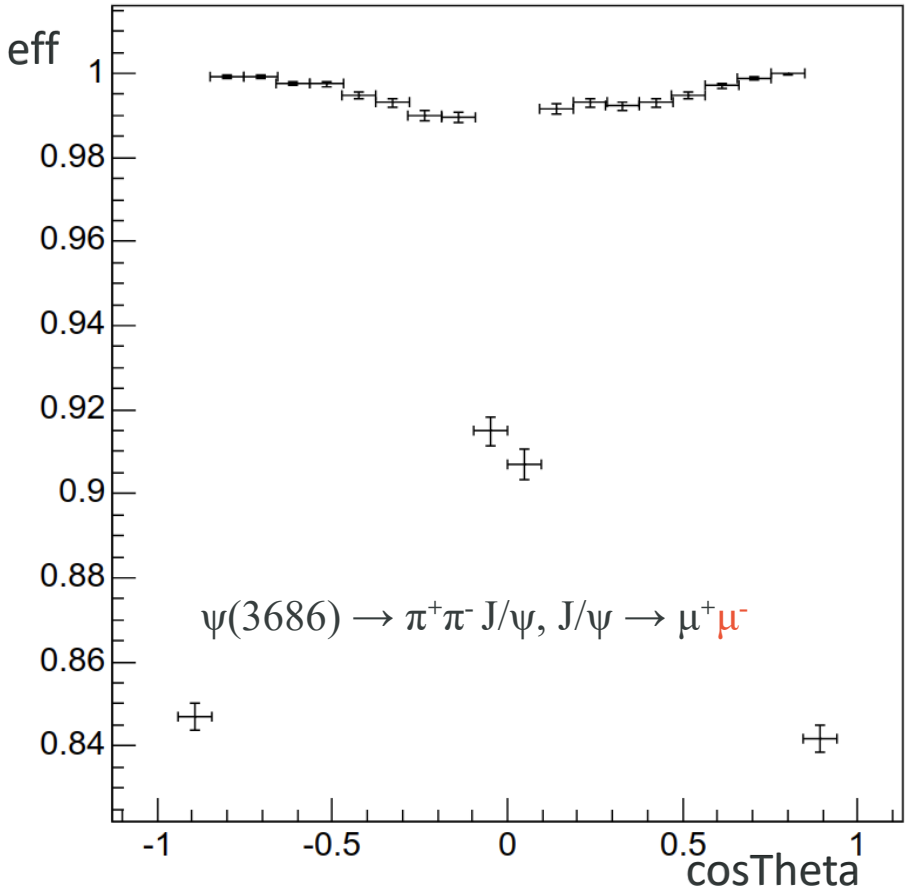
The position reconstructed by ITK may be a point within the working gas range and might not have a fixed R

Track reconstruction

Seeding → Parameters estimation → CKF

Referring to the examples provided by ACTS, I have implemented track reconstruction using ACTS in OSCAR

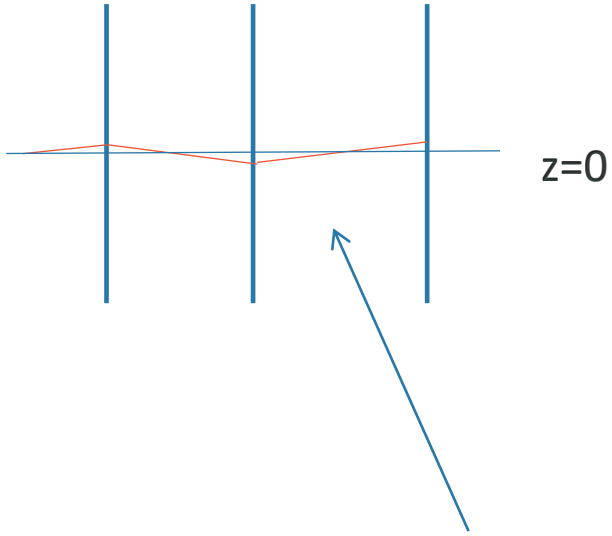
It seems there are some problems with seeding



Seeding-issues

STCF has only three layers ITK -> only one chance to find the correct seed
The space resolution of MPGDs is not good enough compare to silicon detectors -> several hundreds um

```
float cotThetaAvg2 = cotThetaB * cotThetaT;
if constexpr (detailedMeasurement ==
              Acts::DetectorMeasurementInfo::eDetailed)
{
  // use arithmetic average
  float averageCotTheta = 0.5 * (cotThetaB + cotThetaT);
  cotThetaAvg2 = averageCotTheta * averageCotTheta;
}
else if (cotThetaAvg2 <= 0)
{
```



when theta~90 degree, and due to measurement errors,
there is a significant deviation in the measurement position

Seeding-issues


```
float p2scatterSigma = iHelixDiameter2 * sigmaSquaredPtDependent;
if (!std::isinf(m_config.maxPtScattering))
{
  // if pT > maxPtScattering, calculate allowed scattering angle using
  // maxPtScattering instead of pt.
  // To avoid 0-divison the pT check is skipped in case of B2==0, and
  // p2scatterSigma is calculated directly from maxPtScattering
  if (B2 == 0 || options.pTPerHelixRadius * std::sqrt(S2 / B2) >
                 2. * m_config.maxPtScattering)
  {
    float pTscatterSigma =
        (m_config.highland / m_config.maxPtScattering) *
        m_config.sigmaScattering;
    p2scatterSigma = pTscatterSigma * pTscatterSigma * iSinTheta2;
  }
}



// if deltaTheta larger than allowed scattering for calculated pT, skip
if (deltaCotTheta2 > (error2 + p2scatterSigma))
{
```

error estimation

```
float error2 =
    lt.Er + ErB +
    2 * (cotThetaAvg2 * varianceRM + varianceZM) * iDeltaRB * lt.iDeltaR;
```

error2 is the error caused by measurement uncertainty

I believe this error is underestimated
Since STCF ITK does not provide very precise measurements
As a result, many seeds do not pass the filtering criteria

Multiply by a factor?

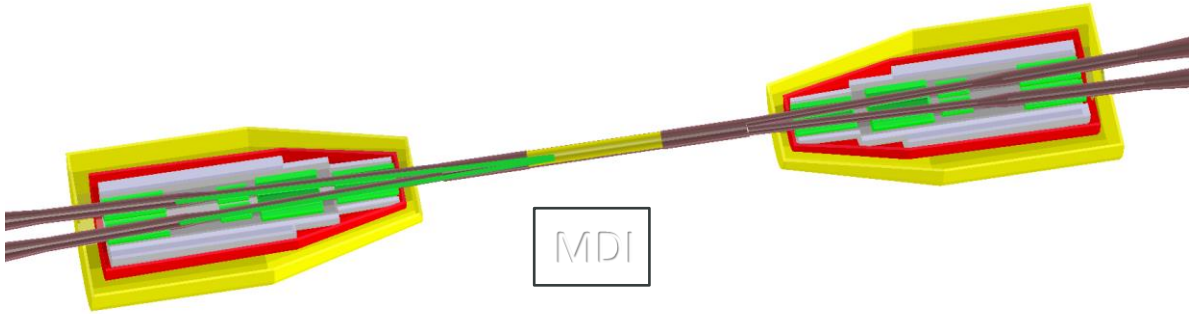
```
cov(0, 0) = 0.1 * 0.1;
cov(1, 1) = 0.4 * 0.4;
```

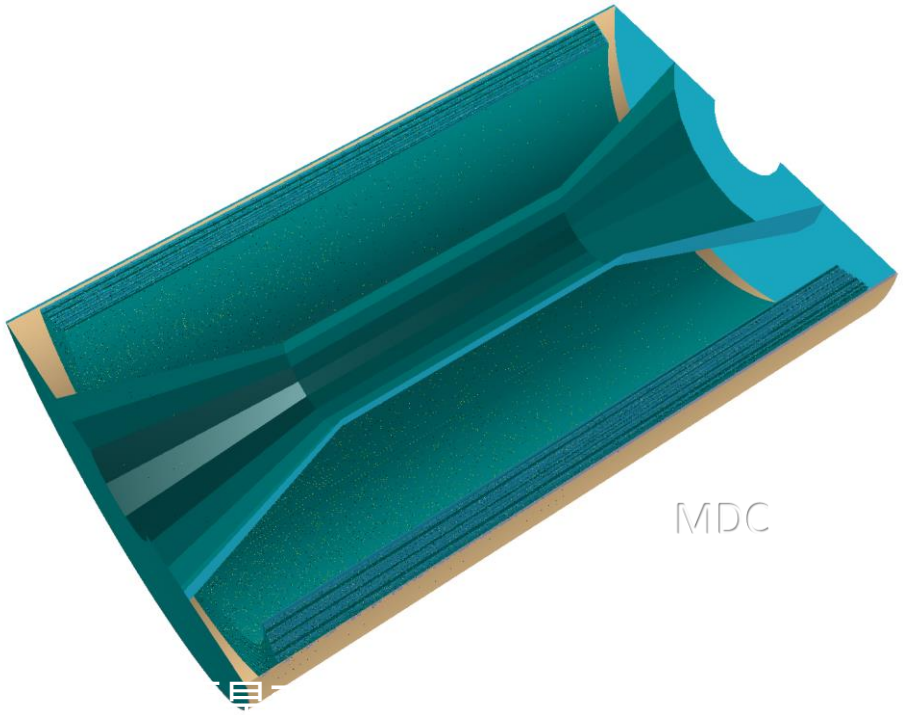
Or set a larger covariance?

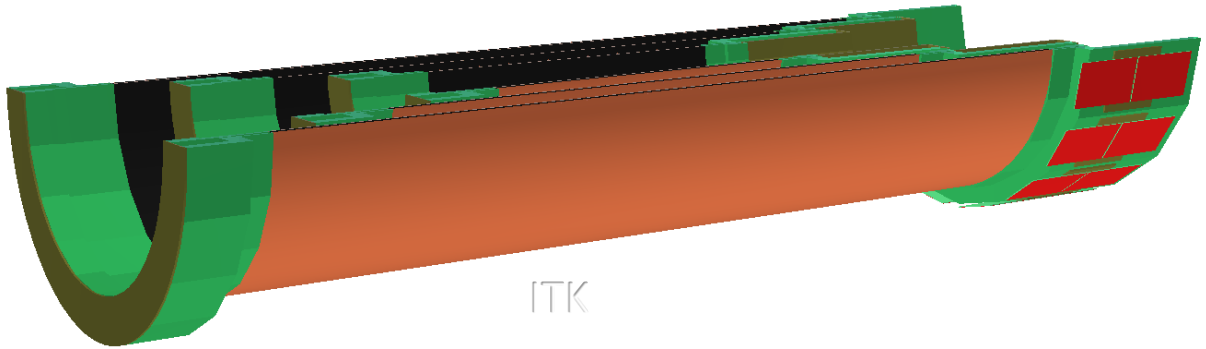
Any suggestions are welcome

Thank you very much

BACK UP


MDI


MDC


ITK