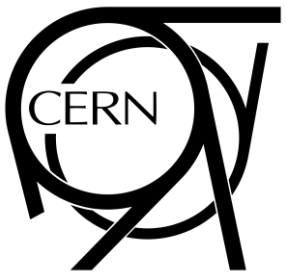


JOHANNES GUTENBERG UNIVERSITÄT MAINZ

CERN Summer Student 2022

Project:

Integration of ACTS into Gaudi

Doğa Elitez

Summer Student in EP-ADP-OS

M.Sc. in JGU Mainz

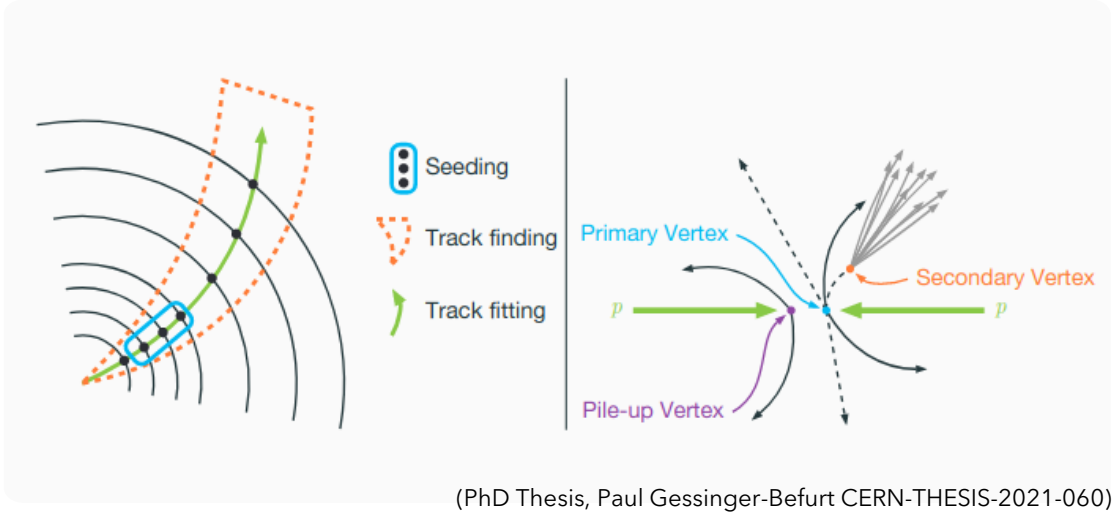
Supervisor:

Paul Gessinger-Befurt

Motivation

- Event reconstruction in high energy physics experiments starts with reconstruction of the trajectory of the charged particles.
- Various software and toolkits are used to describe the tracks in the detectors.
- High computational performance needed for future projects
- Increase of data with HL-LHC .

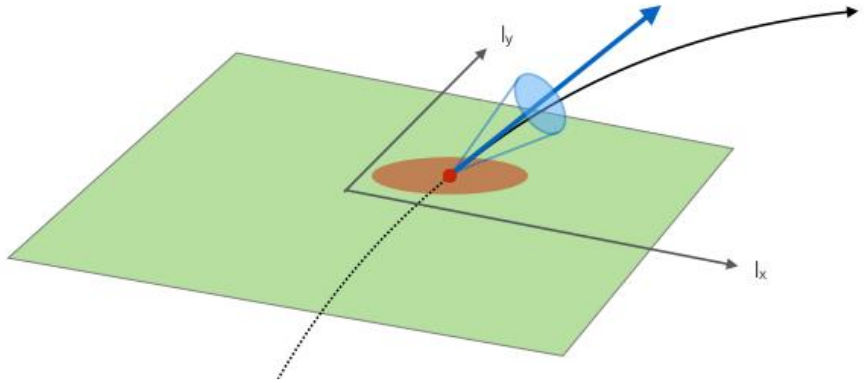
Theory: Track reconstruction in a nutshell



(PhD Thesis, Paul Gessinger-Befurt CERN-THESIS-2021-060)

Track Reconstruction

- "Finding sets of measurements coming from one charged particle and building the associated trajectory through the detector"
- By using tracks, one can create higher level reconstruction objects.
- Trajectory of a charged particle in a magnetic field is defined by track parameters:



$$\vec{q} = (l_1, l_2, \phi, \theta, q/p)$$

(Lectures of Andreas Salzburger on "Track and Vertex Reconstruction")

Track Propagation

- Detectors consist of many different components arranged in various structures.
- Track reconstruction uses the expression of these tracks at different surfaces in the detector.
- Different pattern recognition algorithms are used to recognize the tracks, then they are fitted through i.e. statistical methods and filters.

Toolkits: ACTS and Gaudi

ACTS: A Common Tracking Software

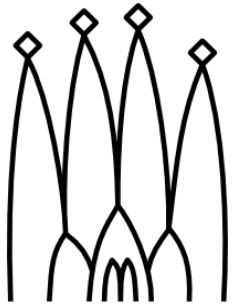
- Experiment-independent toolkit for track reconstruction applications
- Individual and configurable components are provided for the assembly of the experiment
- Main components:
  - Tracking geometry for the navigation and particle propagation
  - Magnetic field description
  - Seed finding algorithms
  - Track find and fitting chains

ACTS integration to key4hep
requires (among others) Gaudi integration

(ACTS Documentation: https://actsreadthedocs.io/en/latest/
Gaudi Documentation: https://gaudi-framework.readthedocs.io/en/latest/)

Gaudi

- A framework software package used to build data processing applications, developed mainly for ATLAS and LHCb experiments
- It is based on C++ with some key elements:
  - Boolean functions called "StatusCode" for initialization, execution and finalization
  - Algorithms and services are configured through python scripts
  - 'Gaudi Properties' allows easy configuration



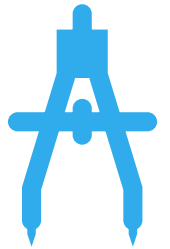
Fun fact:
Gaudi means "big party" in southern german

What have I done?: Development of a service for integrating ACTS into Gaudi

```
33  StatusCode GeoSvc::initialize() {
34    if (m_xmlFileNames.size() > 0) {
35      m_log << MSG::INFO << "Loading xml files from:  '" << m_xmlFileNames << "'" << endmsg;
36    } else {
```

```
98  /// Create a geometry OBJ file
99  StatusCode GeoSvc::createGeoObj() {
100
101
102   Acts::ObjVisualization3D m_obj;
103
104   if (!m_trackingGeo) {
105     return StatusCode::FAILURE;
106   }
107   m_trackingGeo->visitSurfaces([&](const Acts::Surface* surface) {
108     if (surface == nullptr) {
109       info() << "no surface??? " << endmsg;
110       return;
111     }
112     Acts::GeometryView3D::drawSurface(m_obj, *surface, m_trackingGeoCtx);
113   });
114   m_obj.write(m_outputFileName);
115   m_log << MSG::INFO << m_outputFileName << " SUCCESSFULLY written." << endmsg;
116
117   return StatusCode::SUCCESS;
118 }
119
```

```
75
76    return StatusCode::SUCCESS;
77  }
```



Development of a service for the tracking geometry (GeoSvc)

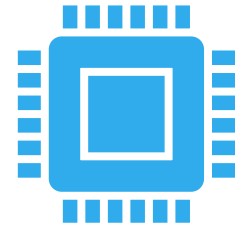
Build detector geometry via DD4Hep from .XML files

Convert tracking geometry using ACTS

Visualize the geometry in a .OBJ file

What have I done?: Development of a service for integrating ACTS into Gaudi

```python
from Configurables import PropagatorAlg
from Configurables import GeoSvc

sys.path.append('/home/delitez/ACTS/spack/k4actstracking')
import actsUnits


algList = []

b = PropagatorAlg("PropagatorAlg")
#
b.mode = 0
b.sterileLogger = False
b.debugOutput = False
b.energyLoss = True
b.multipleScattering = True
b.recordMaterialInteractions = True
b.ntests = 100
b.d0Sigma = 15 * actsUnits.um
b.z0Sigma = 55 * actsUnits.mm
b.phiSigma = 0.001
b.thetaSigma = 0.001
b.covarianceTransport = False
b.qpSigma = 0.0001 / 1 * actsUnits.GeV
b.tSigma = 1 * actsUnits.ns
b.ptLoopers = 500 * actsUnits.MeV
b.maxStepSize = 3 * actsUnits.m
b.sensitiveIDopt = 0

algList.append(b)


a = GeoSvc("GeoSvc")
a.detectors = ["/home/delitez/ACTS/acts/thirdparty/OpenDataDetector/xml/OpenDataDetector.xml"]
a.debugGeometry = True
a.outputFileName = "MyObjFile"


from Configurables import ApplicationMgr

from Configurables import THistSvc
THistSvc().Output = ["rec DATAFILE='propagatorAlgOutput.root' TYP='ROOT' OPT='RECREATE'"]
THistSvc().OutputLevel = DEBUG
THistSvc().PrintAll = True
THistSvc().AutoSave = True
THistSvc().AutoFlush = True

ApplicationMgr(TopAlg=algList, EvtSel="NONE", EvtMax=500, ExtSvc=[a], OutputLevel=DEBUG)
```



Development of a propagation testing algorithm (PropagatorAlg)

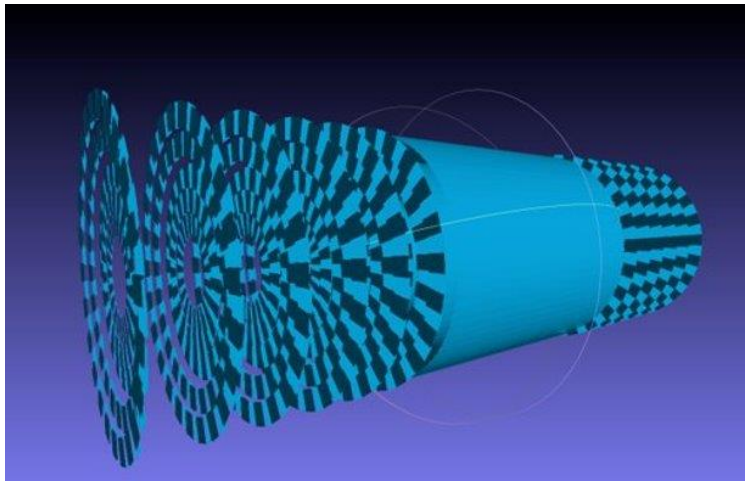
Inherited from the Gaudi Algorithm class

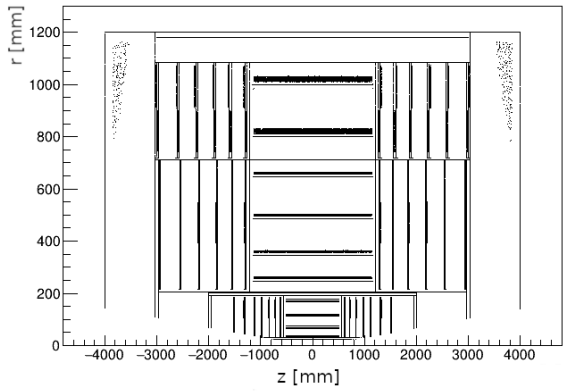
Use GeoSvc to extract the tracking geometry

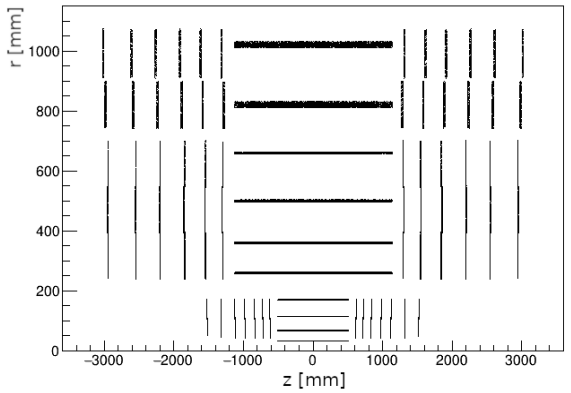
For each particle, extract the track parameters, describe their propagation along a magnetic field

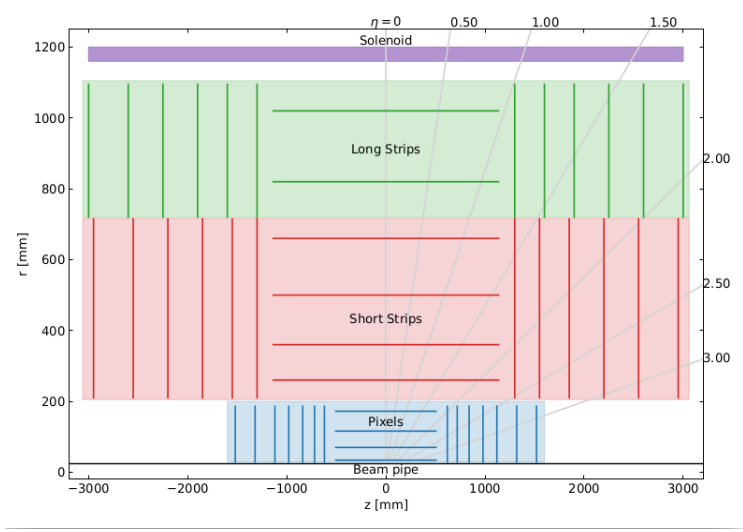
Conclusion: What is next?
Creating a complete example

Build and install	Development of GeoSvc	Development of PropagatorAlg	Simulating events	Track finding algorithm	Management of I/O
• Installing libraries • Building dependencies	• Acts tracking geometry	• Propagation algorithm based on Acts example	• Integration of the particle gun example	• Integration of FATRAS	• .ROOT, .JSON output files • Edm4hep integration









Thank you for your attention!