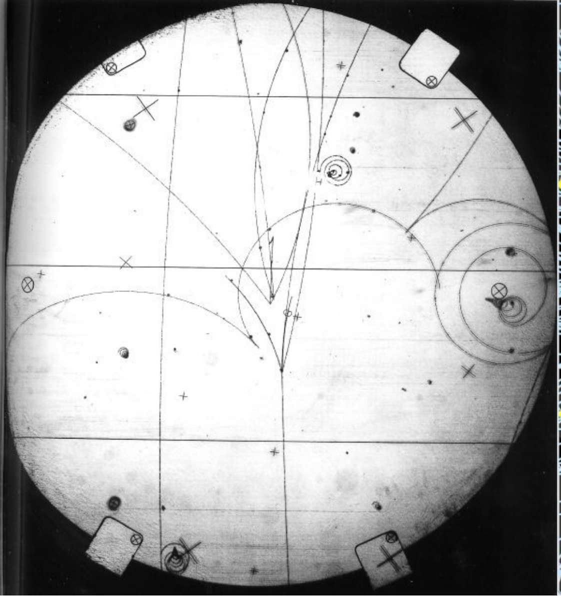
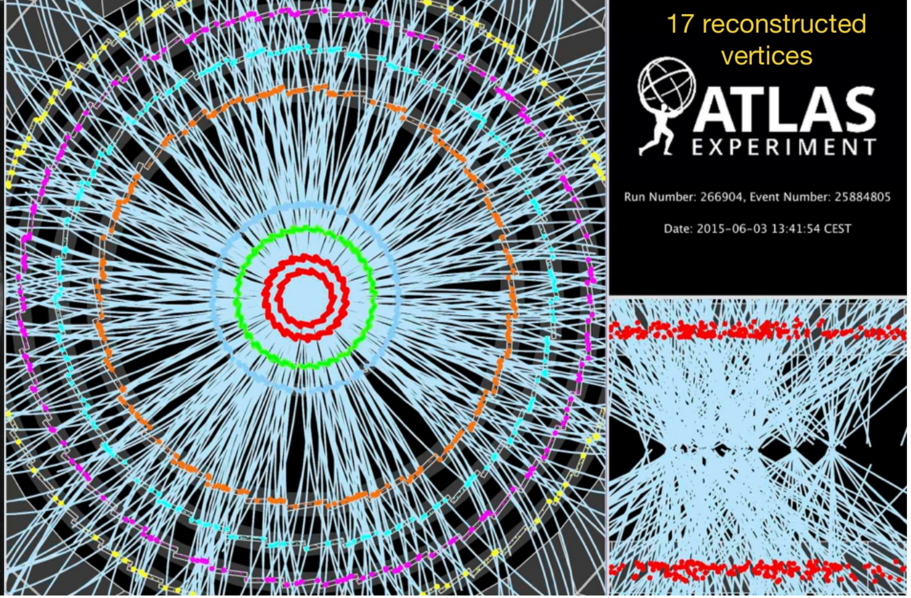




Track reconstruction and beyond

Xiaocong Ai
Dec 6, 2021

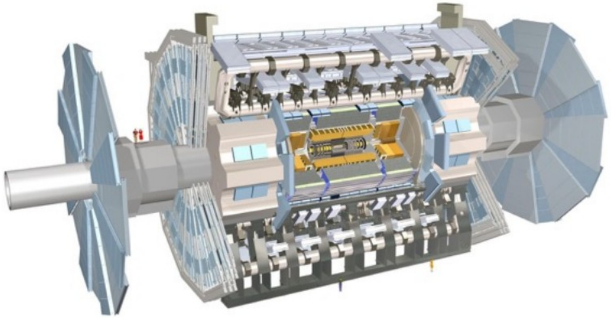
HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

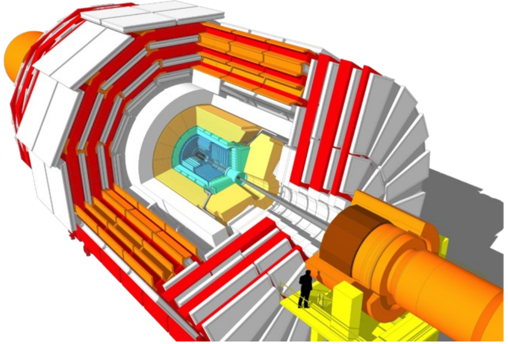
DESY.

Outline

- Event reconstruction at HEP experiments
- Tracking is pivotal
- Tracking strategies
- The tracking challenges
- How to achieve accurate, efficient and fast tracking for various detectors
- Track-based detector alignment
- Summary

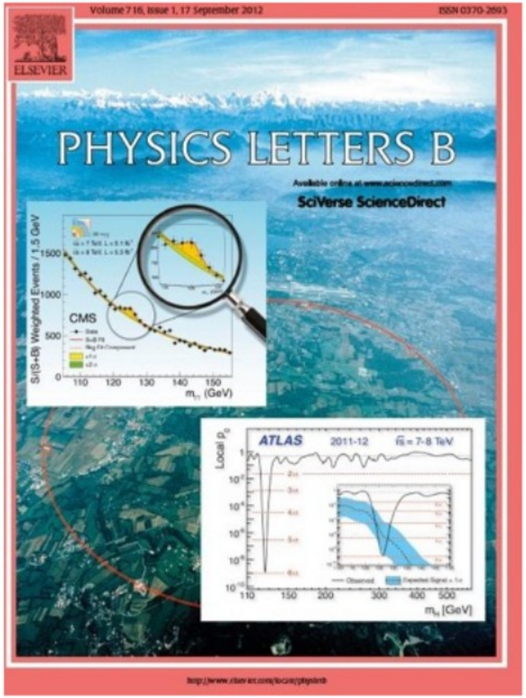
The long long story...



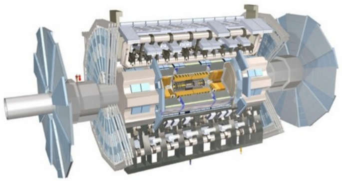


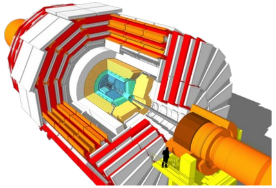
?

The exciting papers



Event reconstruction at HEP experiments





data

Trigger

Reconstruction

Event generation — The interesting physics process

Simulation — Particles interact with detectors:

- Ionization, Bremsstrahlung, Cherenkov, Nuclear interactions ...

Digitization — Energy loss transferred to detector signals:

- hit position, time, shower ...

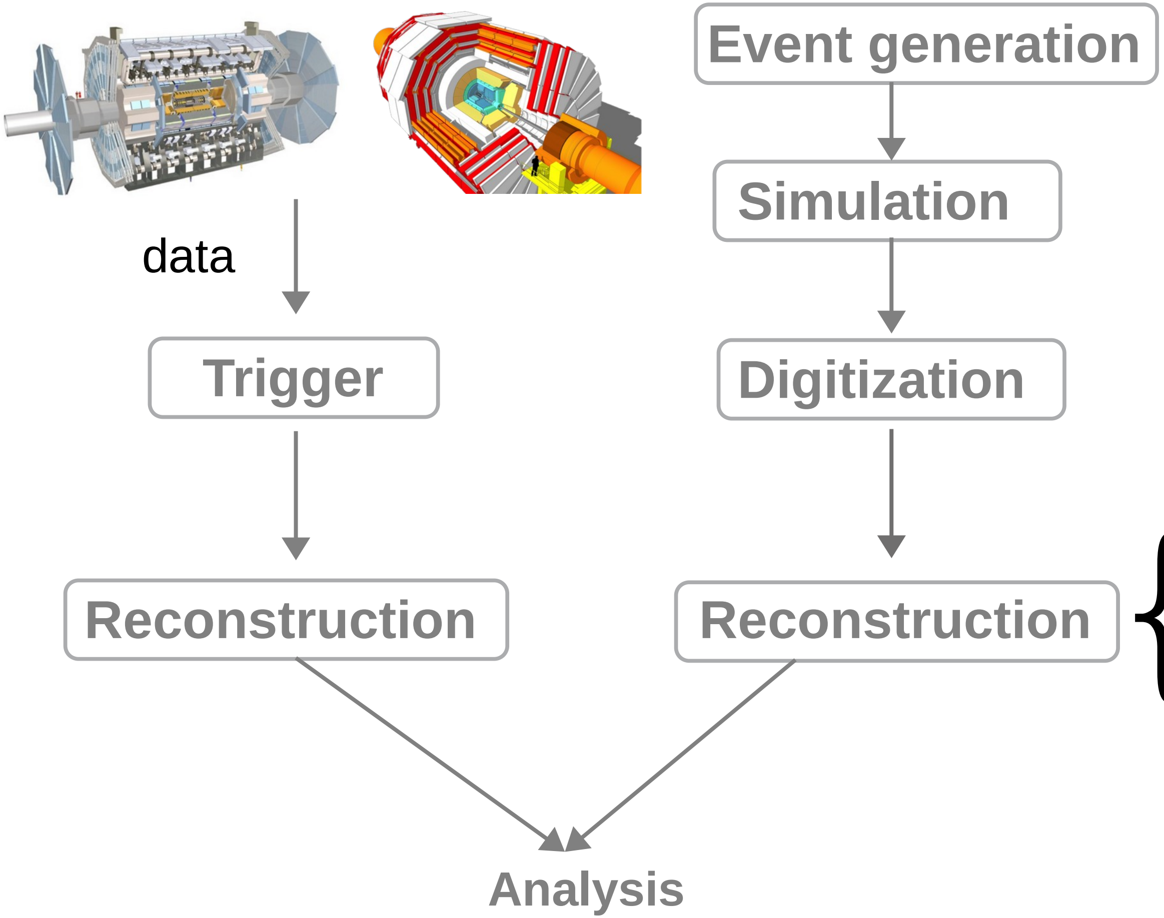
Reconstruction —

Track, Vertex, Calo cluster

High-level physics object
(e, μ, γ, jet, Missing energy)

Analysis

Event reconstruction at HEP experiments



The interesting physics process

Particles interact with detectors:

- Ionization, Bremsstrahlung, Cherenkov, Nuclear interactions ...

Energy loss transferred to detector signals:

- hit position, time, shower ...

Track, Vertex, Calo cluster

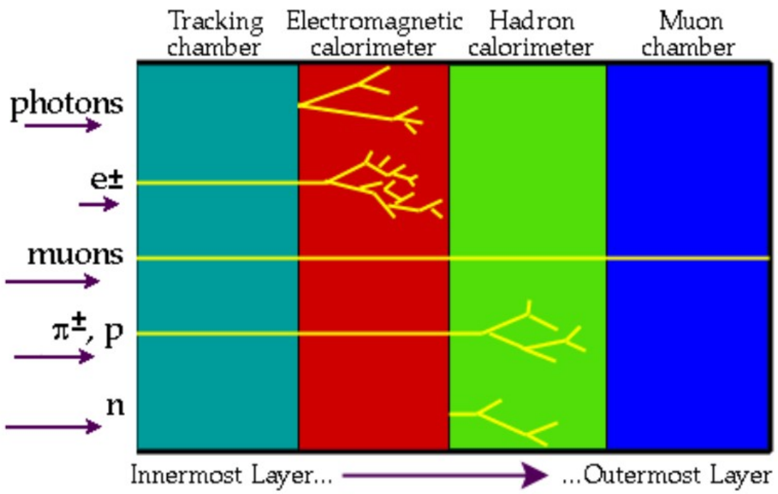
High-level physics object
(e, μ, γ, jet, Missing energy)

My talk will focus on this

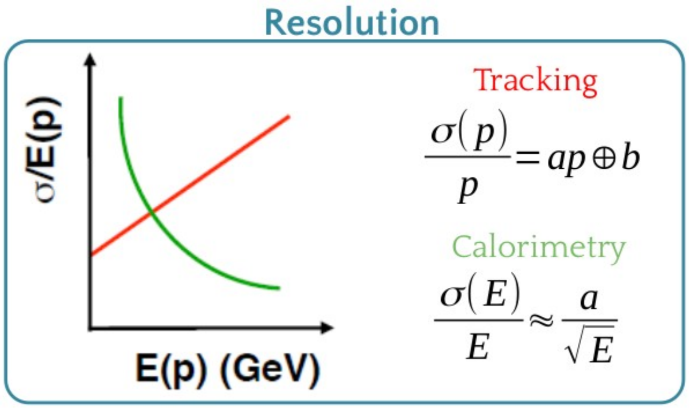
Tracking is pivotal

What is tracking

- Reconstruct **charged particle** (e, μ, charged hadrons) trajectory from **tracker** signals
- Estimate charged particles properties
  - Momentum via curvature in B field
  - Charge
  - Origin and direction
  - Velocity (dE/dx)



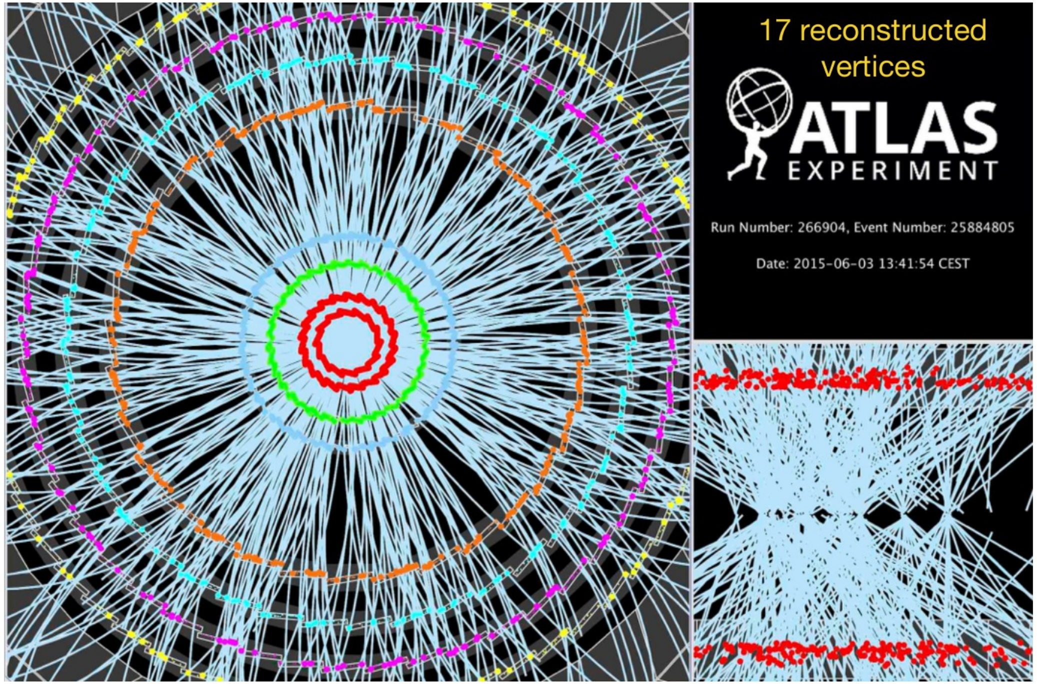
Tracker, calorimeter and muon chamber are complementary to each other!



Sketch from J.-B. Sauvan

Tracking is about vertex reconstruction

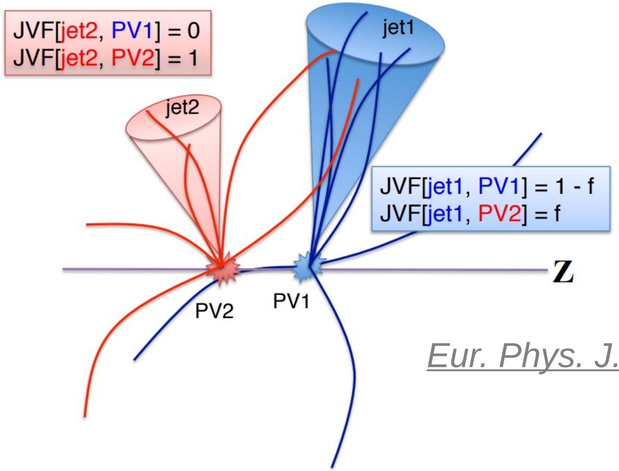
- Primary vertex reconstruction use estimated track parameters of charged particles as inputs
  - Vertex finding
    - Associate tracks to vertices
  - Vertex fitting
    - Estimate vertex position



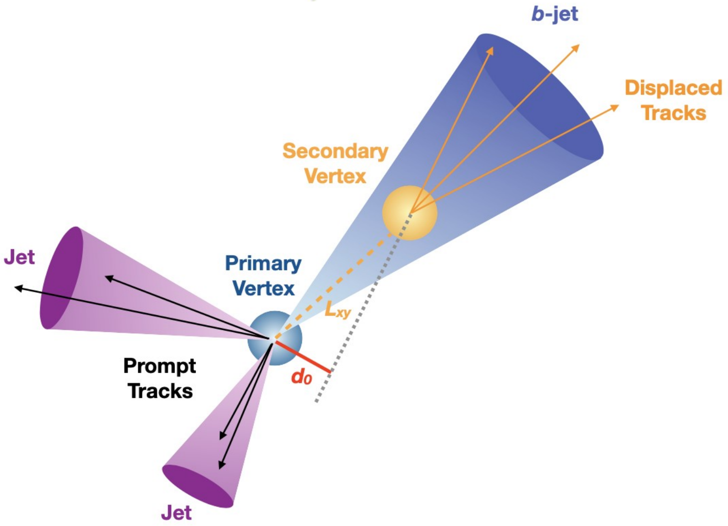
Tens of additional proton–proton collisions accompanying the hard-scatter interaction, i.e. pile-up (μ)

Tracks/vertices are not just about charged particles

- Jets and missing energy reconstruction
  - Better pT resolution for low pT tracks and angular resolution provided by tracker
  - Tracks/vertices are crucial for pile-up mitigation (needs precise jet-vertex association)
- Jet flavor-tagging (b, c or light-flavor jet)
  - Impact parameters, secondary vertices and length of flight
- And track-based alignment of detectors!



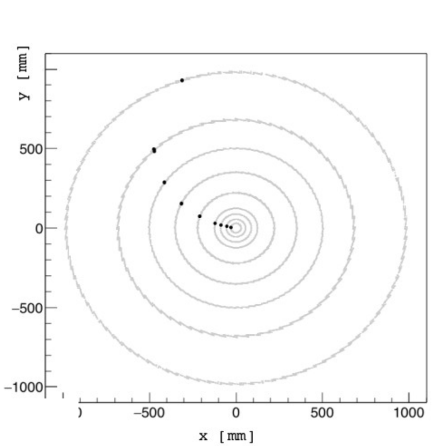
Eur. Phys. J. C 76, 581

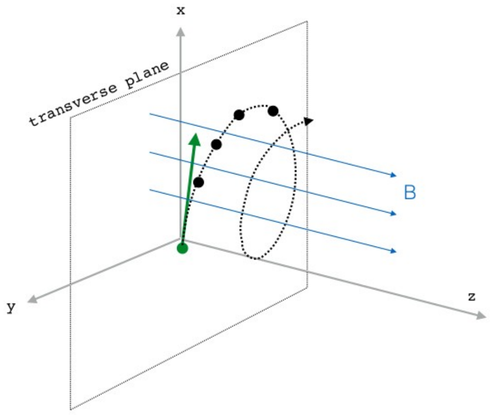


Tracking strategies

Track parameterization

- Helix trajectory of charged particle in homogeneous solonoid magnetic field
- Described by five (or six) parameters
  - e.g. L = (loc0, loc1, phi, theta, q/p, t)

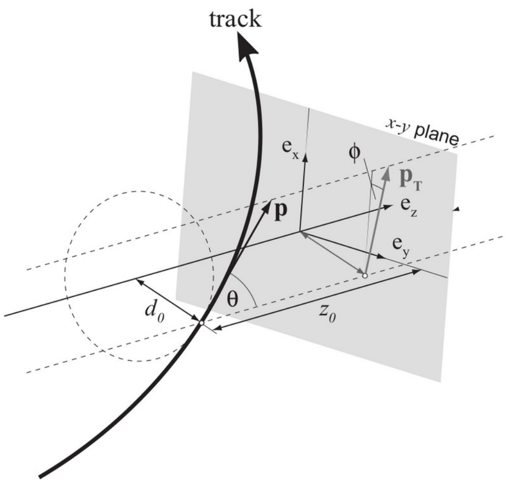




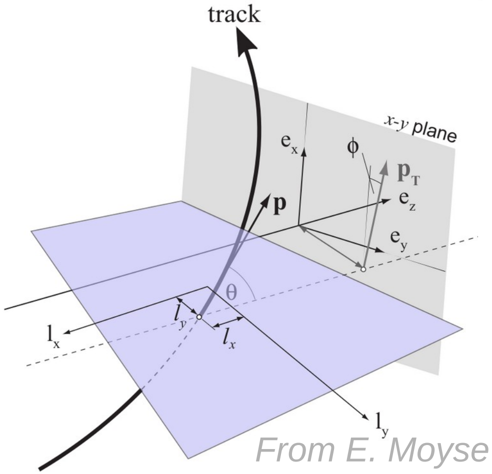
arxiv:1904.06778

$$\frac{d^2\mathbf{r}}{ds^2} = \frac{q}{p}\left[\frac{d\mathbf{r}}{ds} \times \mathbf{B}(\mathbf{r})\right]$$

Solved numerically using Runge-Kutta-Nyström method



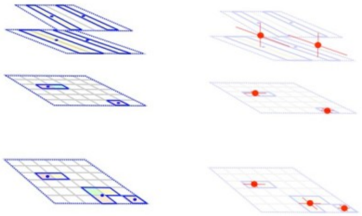
Track parameter represented at the perigee w.r.t. beam line



From E. Moyse

Track parameter represented at detector local surface

How to find & fit tracks ?



Input preparation

↓

Track finding

↓

Track fitting

↓

Vertex and physics object reconstruction

- Raw data converted to cluster/drift circle
- Formation of 3D space point

How to find & fit tracks ?

Input preparation

- Raw data converted to cluster/drift circle
- Formation of 3D space point

Needs calibration and alignment database

↓

Track finding

↓

Track fitting

↓

Vertex and physics object reconstruction

How to find & fit tracks ?

Input preparation

↓

Track finding

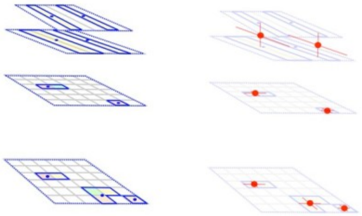
↓

Track fitting

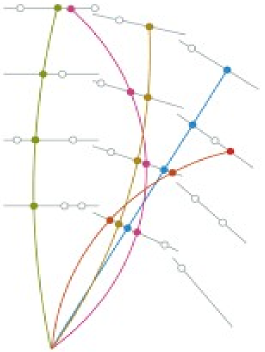
↓

Vertex and physics object reconstruction

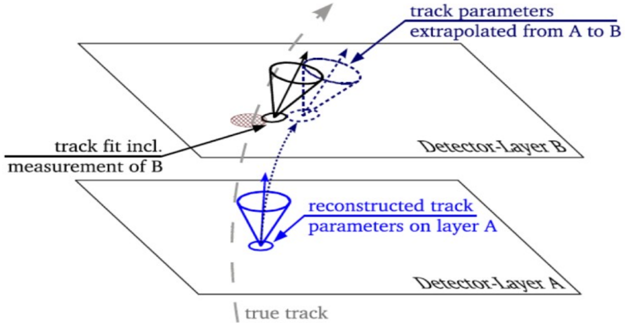
- Raw data converted to cluster/drift circle
- Formation of 3D space point



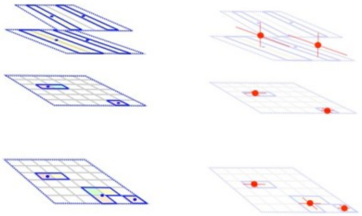
- Identify measurements to individual tracks
  - Global method : Hough transform, Graph Neural Networks
  - Local method: Cellular Automaton, Combinatorial Kalman Filter (CKF)



- Estimate the track parameters
  - Least-square fitter
  - Kalman filter



How to find & fit tracks ?

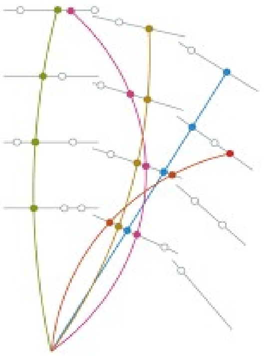


Input preparation

- Raw data converted to cluster/drift circle
- Formation of 3D space point

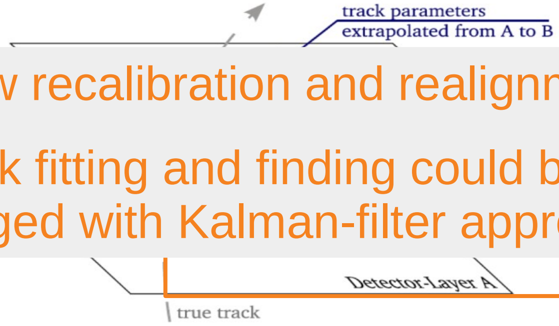
Track finding

- Identify measurements to individual tracks
  - Global method : Hough transform, Graph Neural Networks
  - Local method: Cellular Automaton, Combinatorial Kalman Filter (CKF)



Track fitting

- Estimate the track parameters
  - Least-square fitter
  - Kalman filter



- Allow recalibration and realignment
- Track fitting and finding could be merged with Kalman-filter approach

Vertex and physics object reconstruction

The Least-square fitter for track fitting

- Simultaneously taking into account all measurements
- Cons:
  - Computationally expensive (large size matrix operation)
  - Consideration of material effects is non-trivial
  - Extensions for non-Gaussian noise and non-linear models are difficult

$$\mathcal{P}_i(m_i, \boldsymbol{\lambda}) = \frac{1}{\sqrt{2\pi}} \exp\left[\frac{1}{2}\left(\frac{m_i - h_i(\boldsymbol{\lambda})}{\sigma_i}\right)^2\right]$$

$$\chi^2 = \sum_i \left(\frac{m_i - h_i(\boldsymbol{\lambda})}{\sigma_i}\right)^2 = -2\ln\mathcal{L} + \text{const.}$$

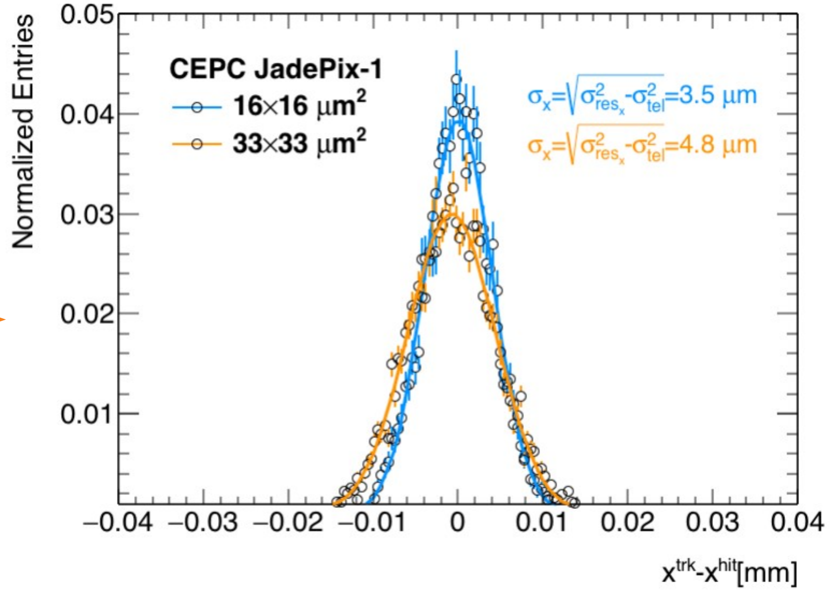
The Least-square fitter for track fitting

$$\mathcal{P}_i(m_i, \boldsymbol{\lambda}) = \frac{1}{\sqrt{2\pi}} \exp\left[\frac{1}{2}\left(\frac{m_i - h_i(\boldsymbol{\lambda})}{\sigma_i}\right)^2\right]$$

$$\chi^2 = \sum_i \left(\frac{m_i - h_i(\boldsymbol{\lambda})}{\sigma_i}\right)^2 = -2\ln\mathcal{L} + \text{const.}$$

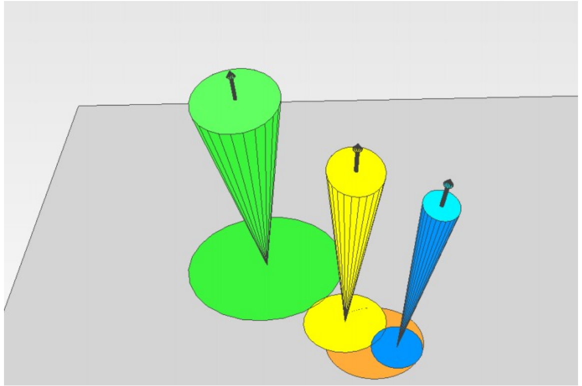
- Simultaneously taking into account all measurements
- Cons:
  - Computationally expensive (large size matrix operation)
  - Consideration of material effects is non-trivial
  - Extensions for non-Gaussian noise and non-linear models are difficult

- Superseded by the Kalman filter, but still used if at least material effects can be considered, e.g. the General Broken Lines (GBL) χ^2 fitter
  - Used for e.g. **beam test tracking** for detector characterization

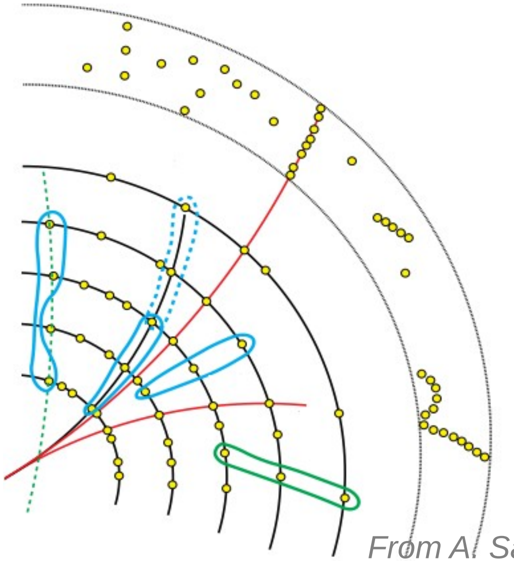


The Kalman Filter-based track fitting and finding

Comput Softw Big Sci **5**, 20 (2021)



- Progressively consider measurements
  - Extrapolate from k-1 to k iteratively: prediction + filtering
  - Backward smoothing when forward filtering is done
- Pros:
  - Straightforward handling of material effects
  - Allow track finding alongside fitting using CKF
  - Extension-friendly
    - For non-gaussian noise, e.g. bremsstrahlung energy loss
      → Gaussian Sum Filter (GSF)
    - For non-linear measurement model
      → Second-order KF!



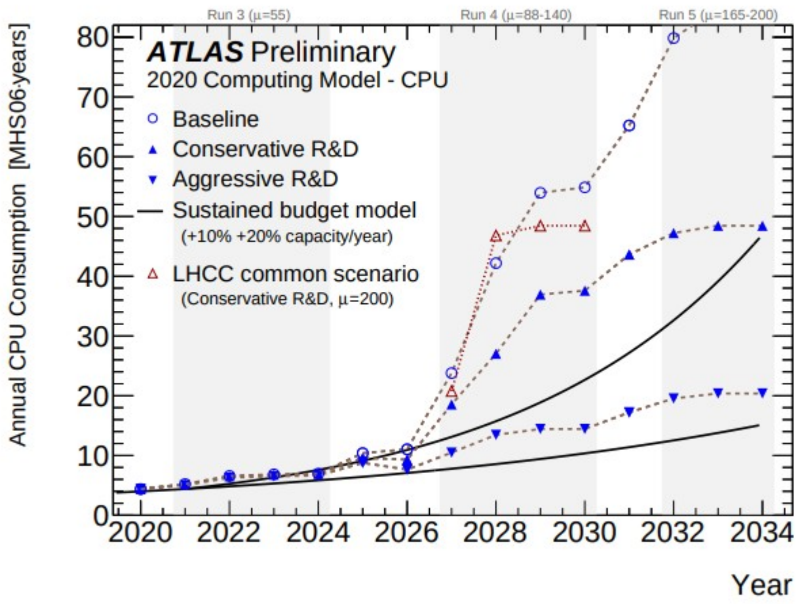
From A. Salzburger

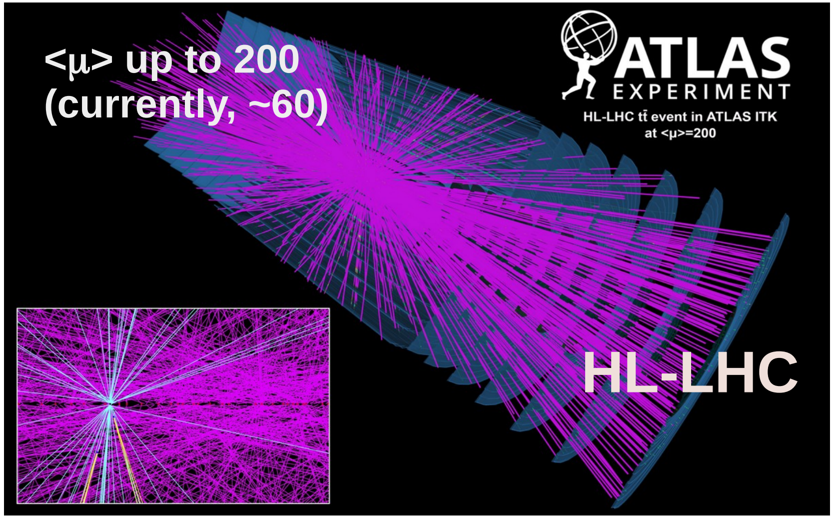
The tracking challenges

Much more dense environment

- Much increased combinatorics with high pileup at future hadron colliders, e.g.
  - ~7k particles/event with <μ> = 200 at High Luminosity LHC (HL-LHC)
  - <μ> = 1000 at FCC-hh
- Much increased CPU needs

More sensitive to rare phyics, and far more combinatorics!



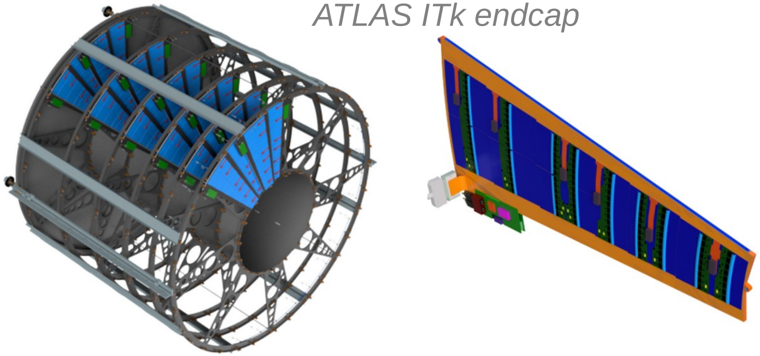


Estimated CPU resources needed for event processing at ATLAS

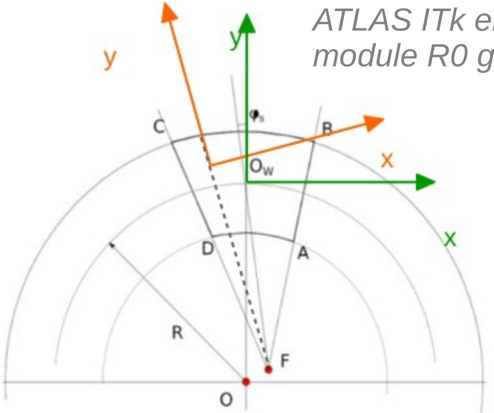
More complicated detector geometry

- For example, new built-in **radial strips** for the ATLAS Phase II Inner Tracker (ITk) Strip endcap
- And then we got lost in the complexity with various coordinate transformations
  - The ATLAS Software (Athena) release was long broken for ITk before 2018
  - The reconstruction software (Eutelescope) for the ITk module test beam was not usable

ATLAS ITk endcap



ATLAS ITk endcap module R0 geometry



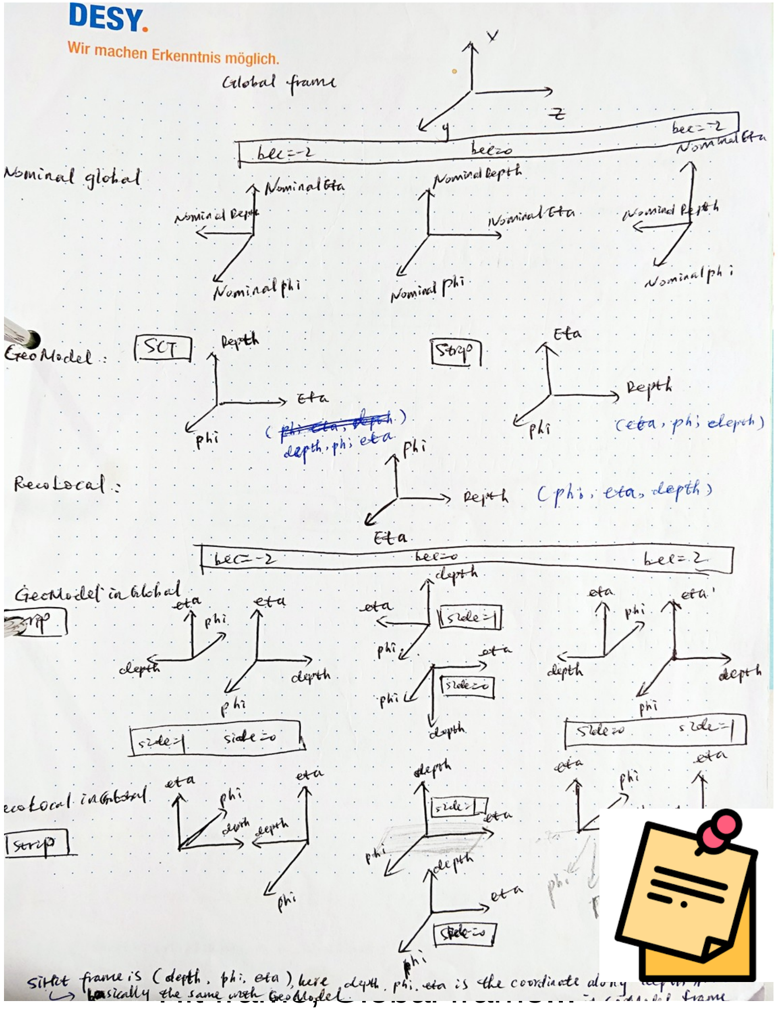
Measurement frame varies with the strip orientation& different to local sensor frame

And we also have the Geant4 Hit frame, Global frame...

More complicated detector geometry

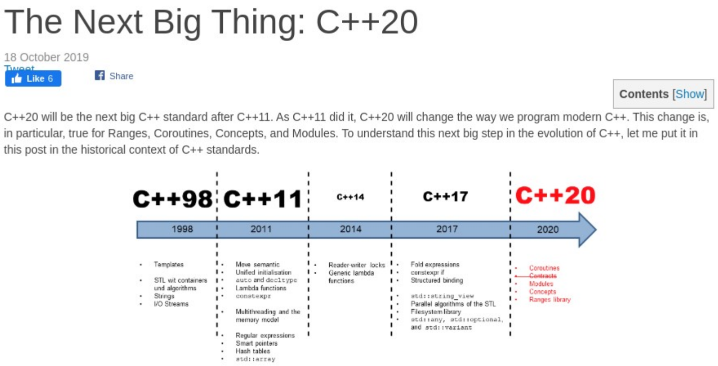
- For example, new built-in **radial strips** for the ATLAS Phase II Inner Tracker (ITk) Strip endcap
- And then we got lost in the complexity with various coordinate transformations
  - The ATLAS Software (Athena) release was long broken for ITk before 2018
  - The reconstruction software (Eutelescope) for the ITk module testbeam was not usable

Lots of efforts to put things in order after sorting out the transformations!



Outdated software framework

- Tracking software used at experiments are often developed before first commission data, i.e. ~tens of years old
  - Old design
  - Poor portability
  - Maintenance is a painstaking!
- Optimization of current old software is never easy
- New technology and architecture is there!


The Next Big Thing: C++20

18 October 2019

Tweet Like 6 Share

Contents [Show]

C++20 will be the next big C++ standard after C++11. As C++11 did it, C++20 will change the way we program modern C++. This change is, in particular, true for Ranges, Coroutines, Concepts, and Modules. To understand this next big step in the evolution of C++, let me put it in this post in the historical context of C++ standards.

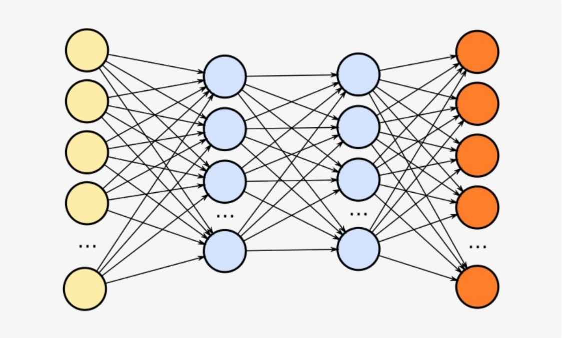
How to achieve accurate, efficient and fast tracking for various detectors?

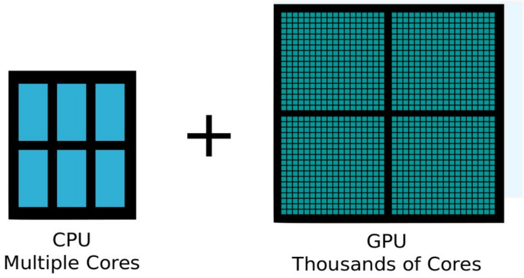
I. Develop a new high performant common tracking toolkit

II. Parallelization, acceleration and ML

The A Common Tracking Software project

- To prepare a modern open-source **experiment-independent tracking toolkit** for **current** and **future** detectors based on LHC tracking experience
  - Targeting at HL-LHC, but also for Belle-II, FASER, sPHENIX, ALICIE, EIC, CEPC…
- To provide an open-source R&D platform to explore **new techniques, parallelization and acceleration**





About acts

Experiment-independent toolkit for (charged) particle track reconstruction in (high energy) physics experiments implemented in modern C++

acts.readthedocs.io

simulation reconstruction particle-track-reconstruction physics-experiment

Readme

MPL-2.0 License

Code of conduct

Cite this repository

Releases 89

v15.0.0 Latest

19 days ago

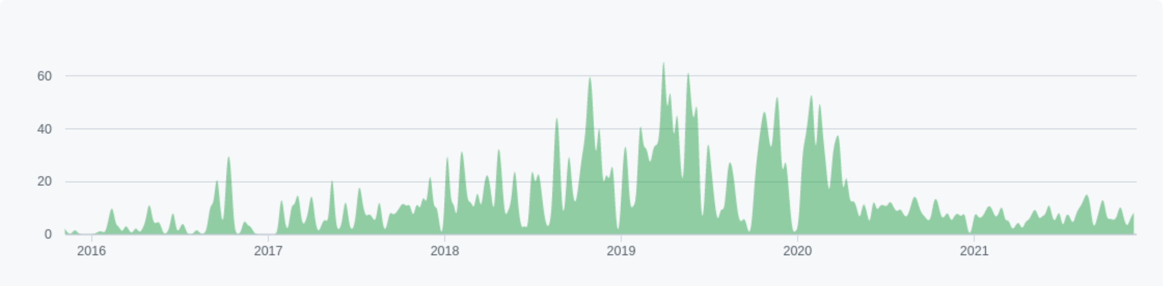
The ACTS developers team

- 10~15 active developers on Core project
  - ATLAS heavy, but increasing external contribution

Nov 8, 2015 – Dec 4, 2021

Contributions: Commits

Contributions to main, excluding merge commits and bot accounts



Contributors 40

+ 29 contributors

Joined the core development in early 2019

ACTS is one of the four projects in IRIS-HEP ($25M from National Science Foundation)

supported by

EP-R&D Programme on Technologies for Future Experiments

iris hep

Bundesministerium für Bildung und Forschung

CERN

UNIVERSITÉ DE GENÈVE

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

TUM

cnrs

DESY.

BERKELEY LAB

cooperations

HSF HEP Software Foundation

intel

CERN openlab

ACTS fosters collaboration

- World-wide users from particle and nuclear physics, collider and non-collider experiments
  - **>10 experiments**
    - ATLAS, Belle-II, ALICE, sPHENIX, FASER, EIC, CEPC, LUXE, PANDA, Muon Collider ...
  - **>15 institutes**
    - CERN, LBNL, ORNL, UC Berkeley, Stanford University, DESY, Universite at Bonn…
  - **~45 forks** of the acts repository
- Regular/irregular discussion between developers and experiment users
  - ATLAS, FASER, sPHENIX, ALICE, EIC...

ACTS design

https://acts.readthedocs.io/en/latest/

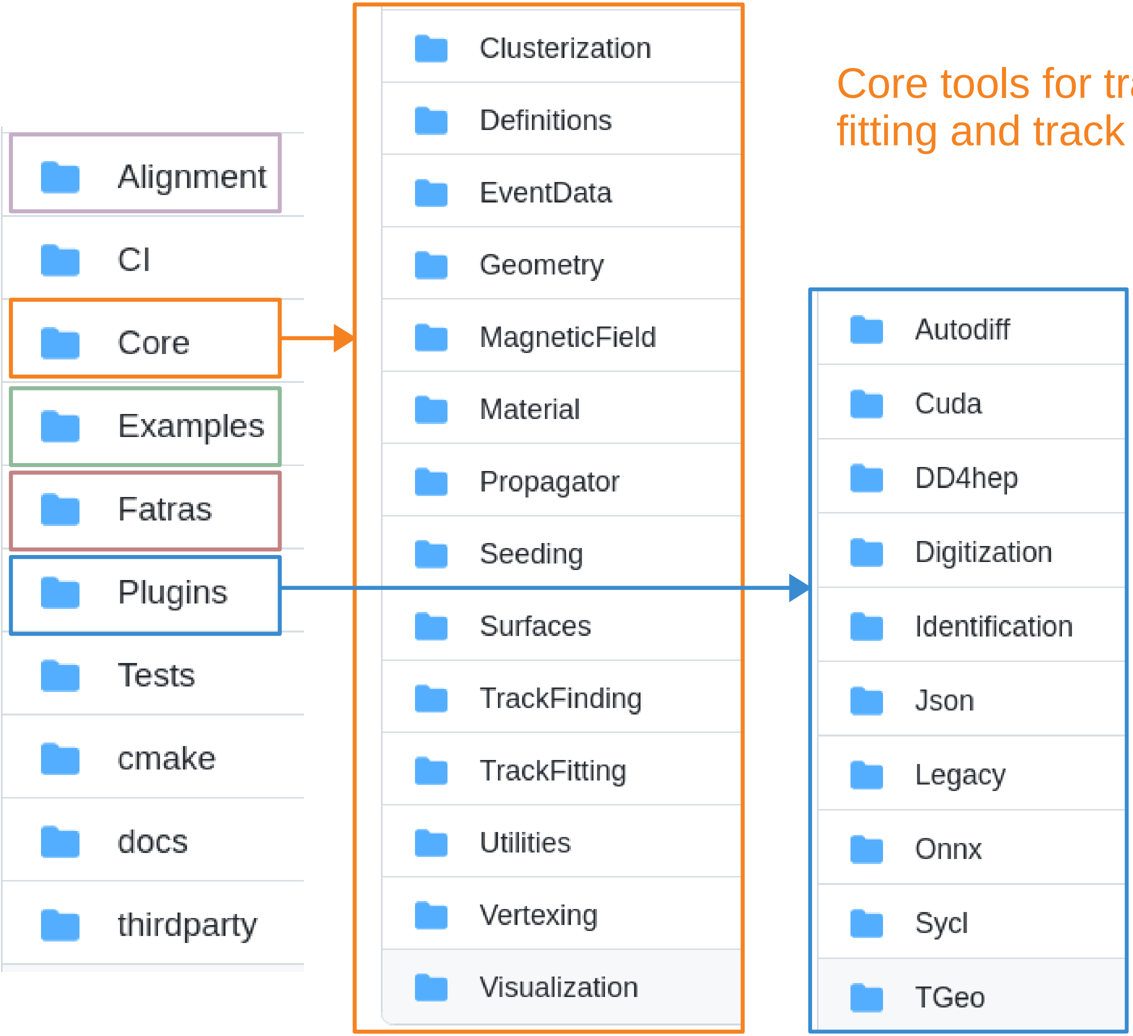
- **Modern C++ 17** concepts
- Highly-templated design
  - **Detector and magnetic field agnostic**
- Strict **thread-safety** to facilitate concurrency
- Supports for **contextual** condition data
  - Concurrent event execution with different Geometry/Calibration/Magnetic field in flight
- Minimal dependency (Eigen)
- Highly configurable for **usability**
- And **well-documented** !

The tracking tools in ACTS

An alignment prototype

A light-weight test framework with application examples

A fast simulation engine



Core tools for track propagation, track fitting and track finding, vertexing...

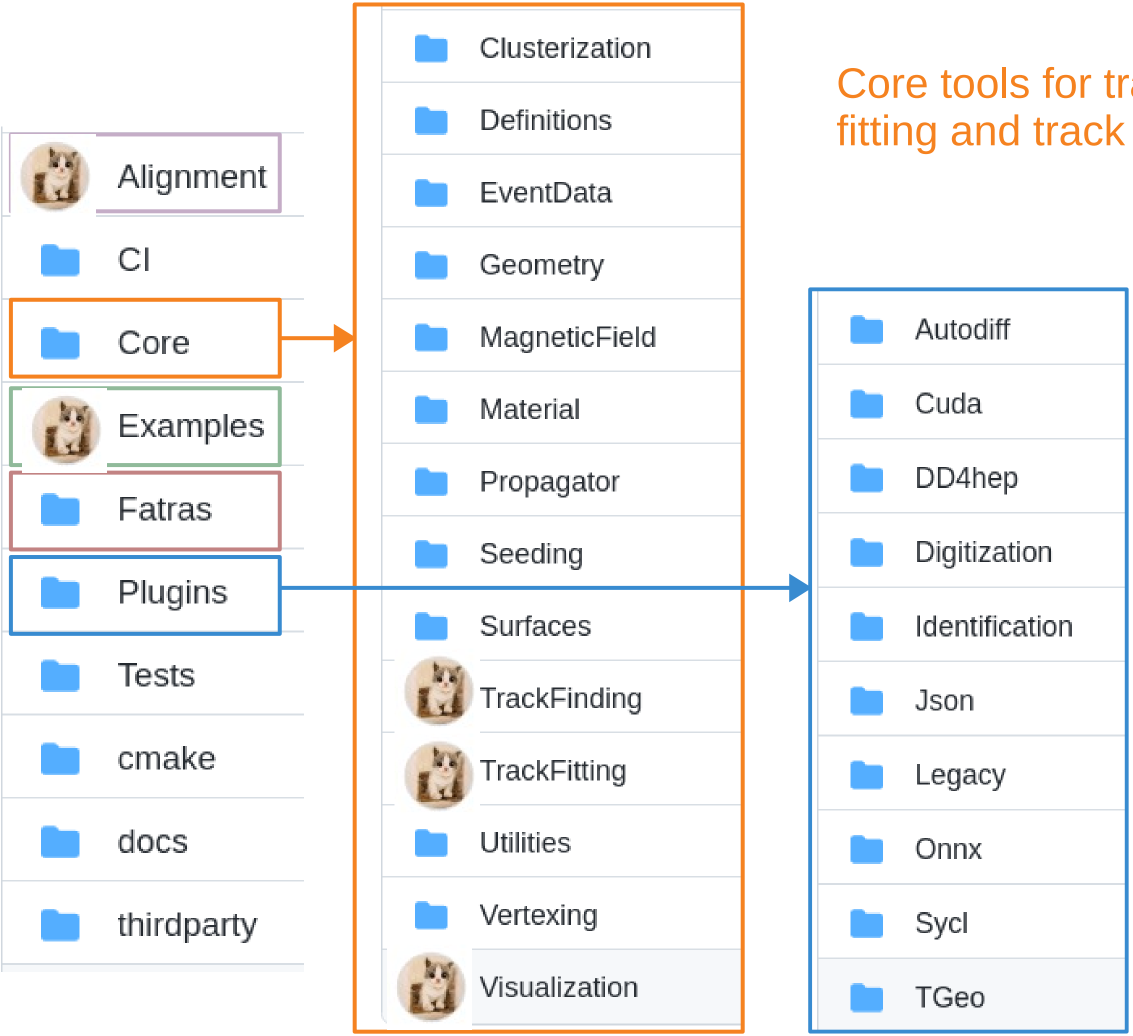
Plugins to support R&D on new techniques!

The tracking tools in ACTS

An alignment prototype

A light-weight test framework with application examples

A fast simulation engine



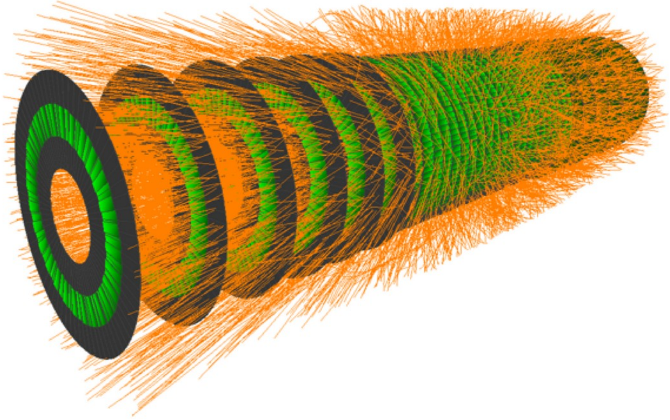
Core tools for track propagation, track fitting and track finding, vertexing...

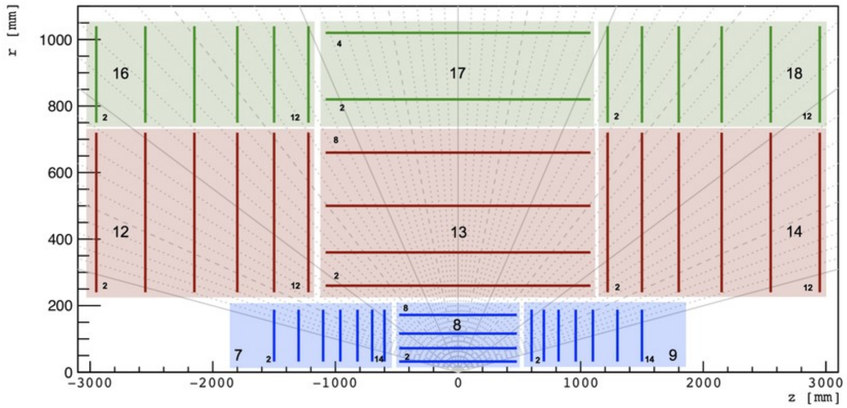
Plugins to support R&D on new techniques!

I'll be happy to help there!

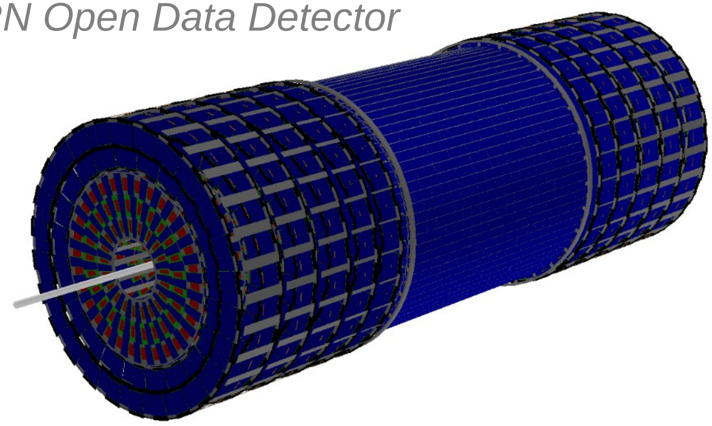
The detectors used for development

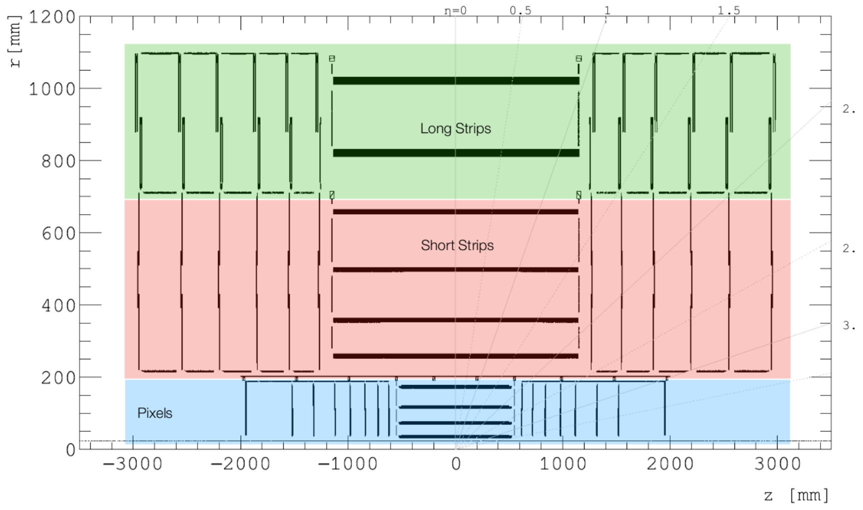
The TrackML detector





The CERN Open Data Detector

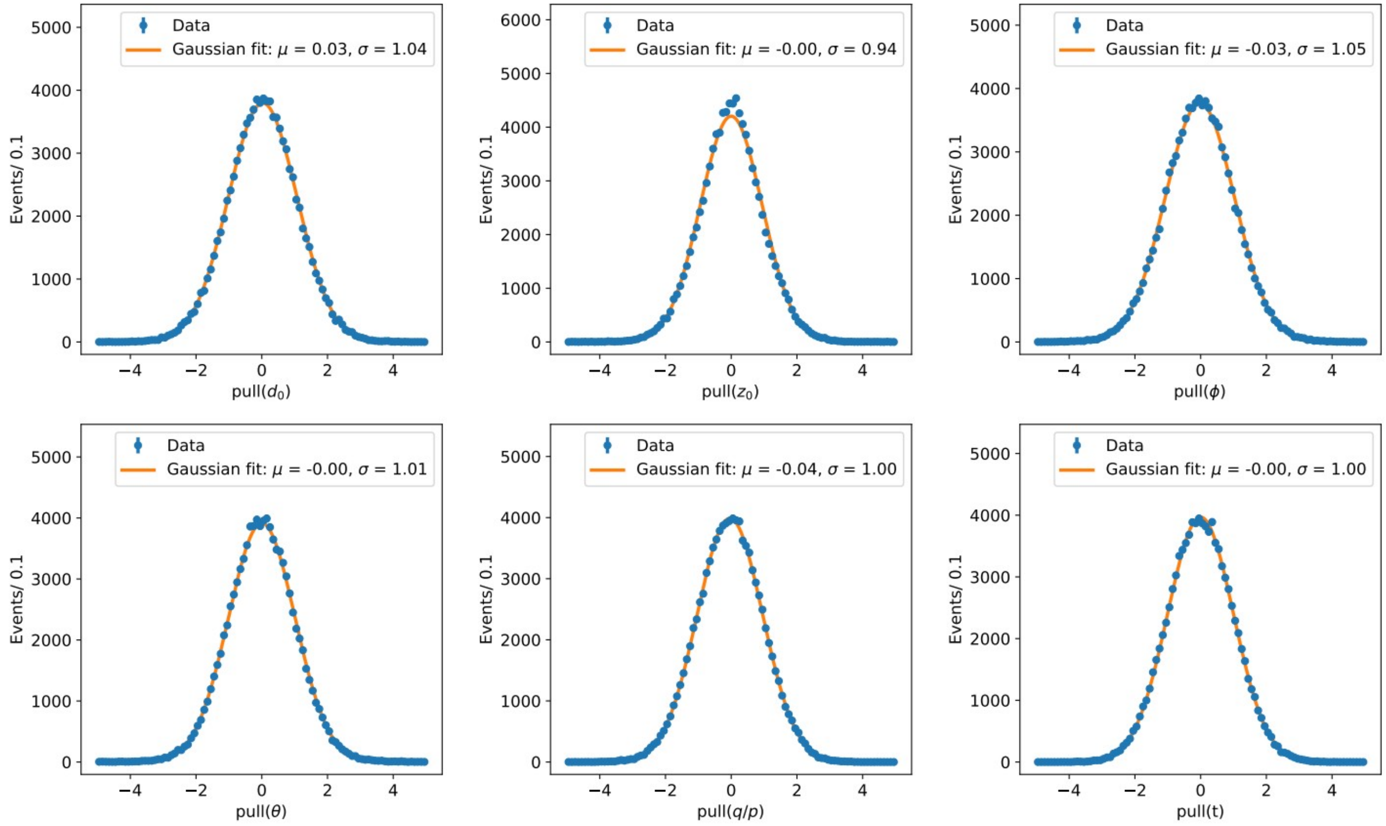




ACTS tracking performance: track parameters estimation

Pull of track parameters represented at the perigee

arXiv:2106.13593v1

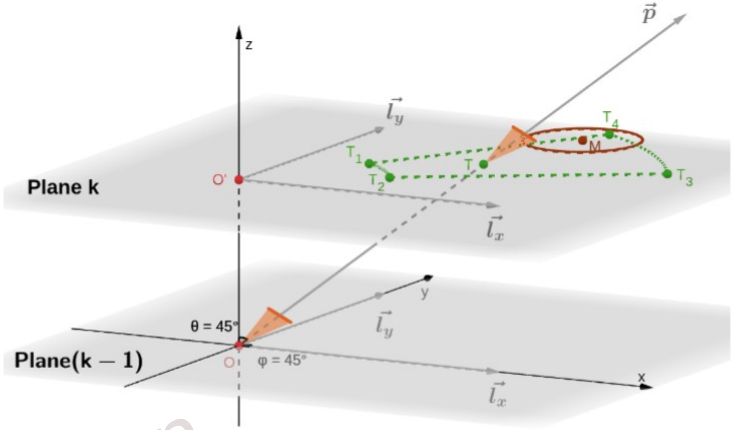


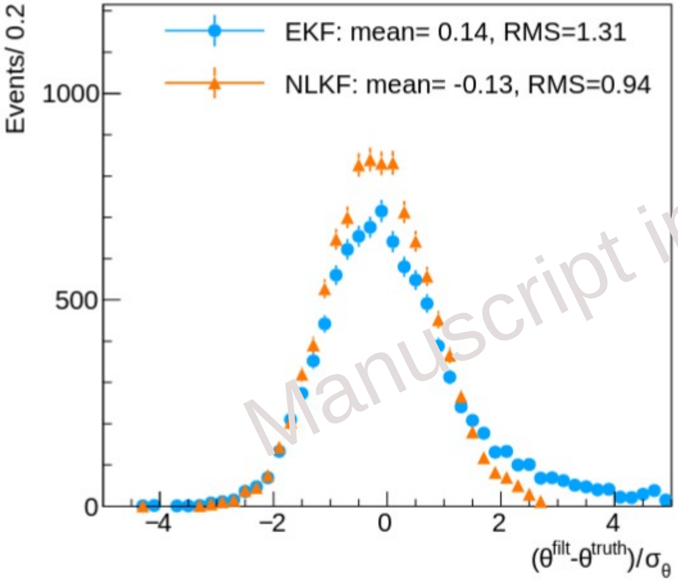
$$v_{\mathrm{pull}} = \frac{v_{\mathrm{fit}} - v_{\mathrm{truth}}}{\sigma_{\mathrm{v}}},$$

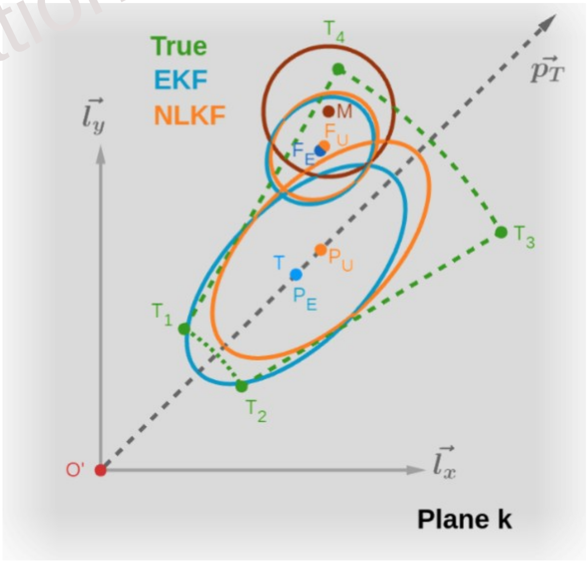
Single muon 400 MeV < pT < 100 GeV, |eta|<2.5
TrackML detector, Bz= 2T

Improve tracking precision with second-order KF

- The (extended) KF used in HEP is optimal for linear system
- Tracking precision is degraded by significant non-linear effects with large incident angle



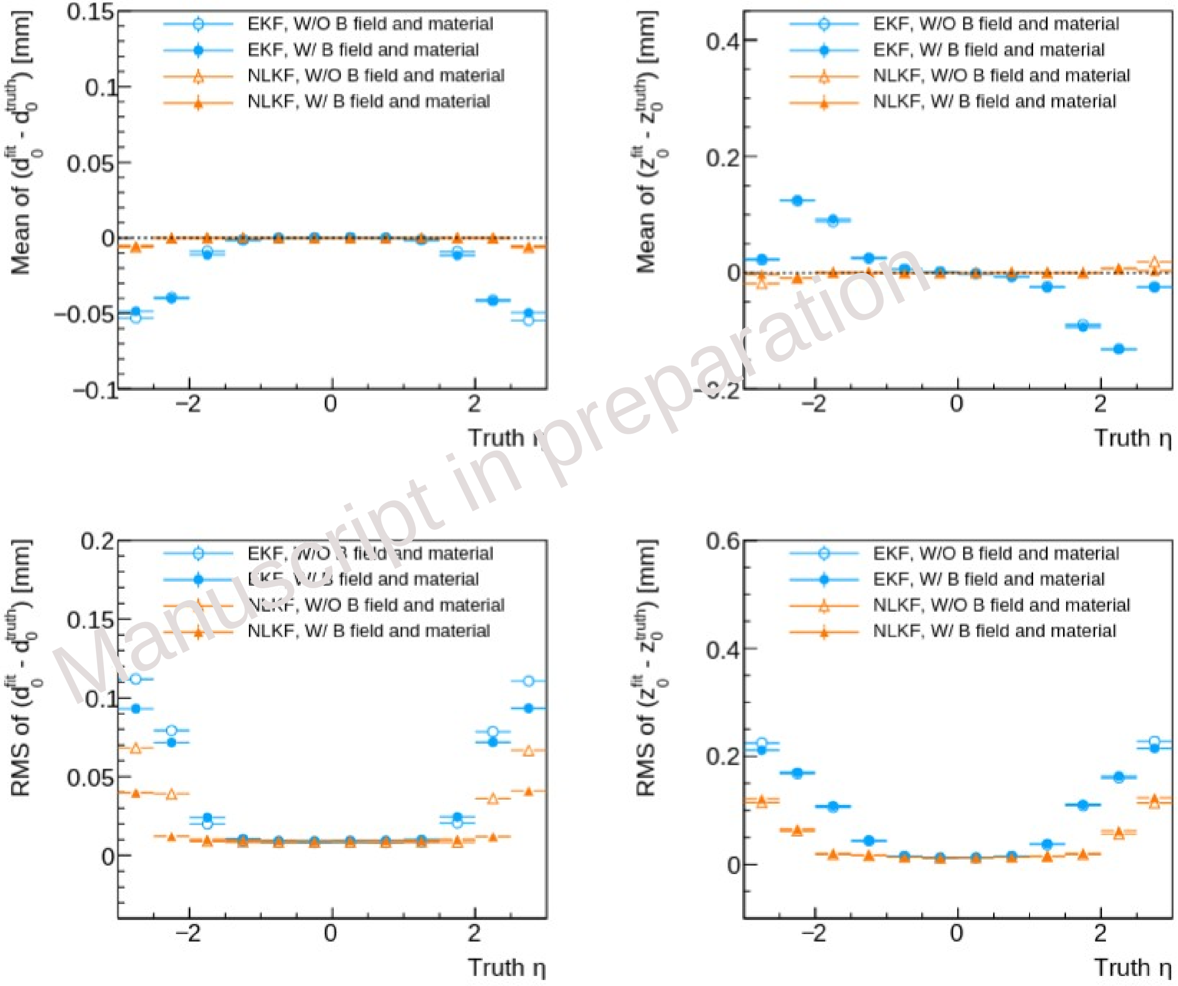




Improve tracking precision with second-order KF

- **Application of second-order correction for KF in HEP for the first time!**
  - Corrects the bias and improves resolution of track parameters significantly!
- The implementation will be ready for deployment in ACTS soon!

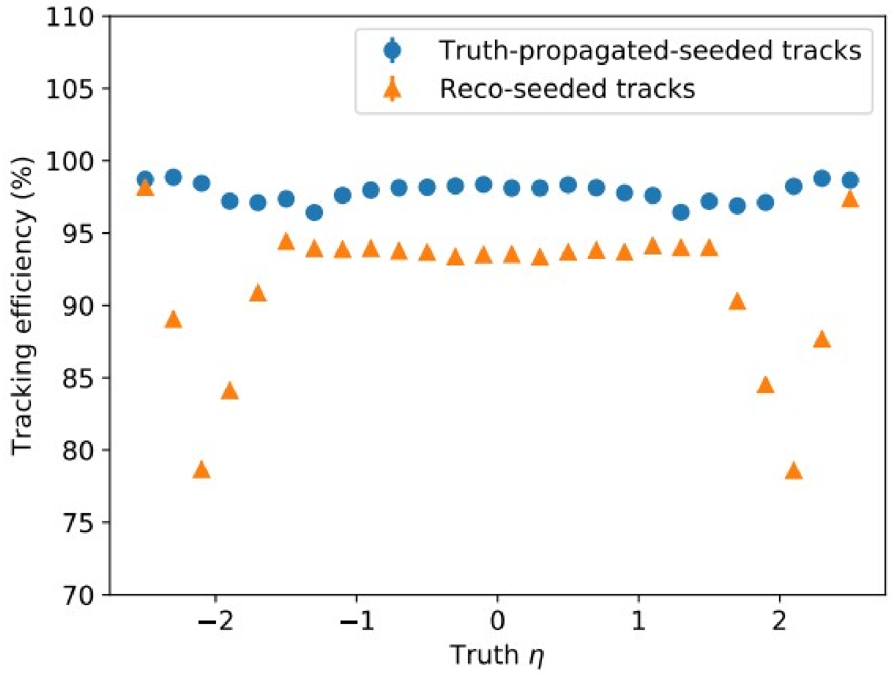
Single muon (20<pT<100 GeV)
Open Data Detector, solenoidal
Bz= 2T (ATLAS-like)



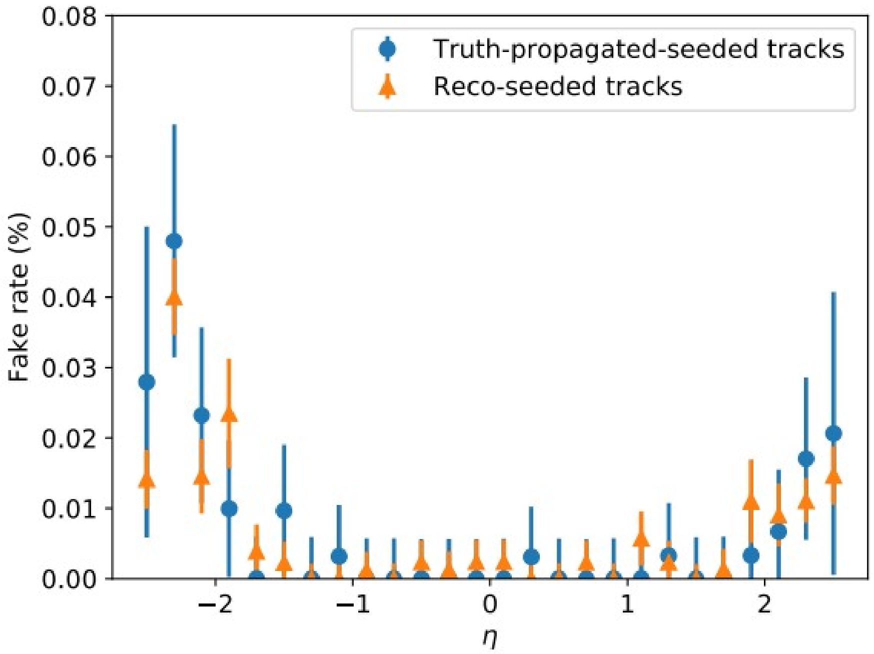
ACTS tracking performance: efficiency and fake rate

- >99% tracking efficiency and <0.01% fake rate for $t\bar{t}$ at <μ>=200 (~3000 charged tracks/event)

Tracking efficiency



Fake rate



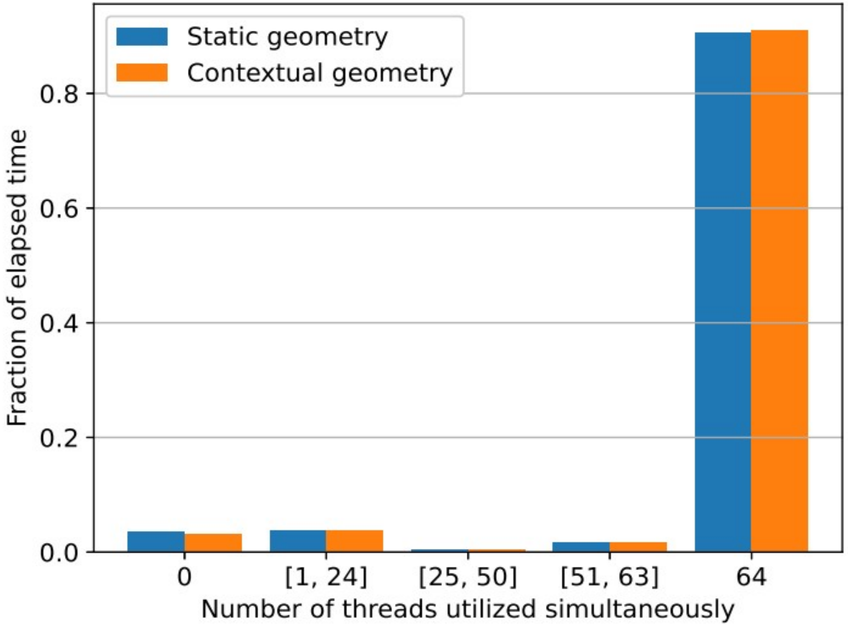
sqrt(s) = 14 TeV, $t\bar{t}$, μ =200
TrackML detector, Bz= 2T

arXiv:2106.13593v1

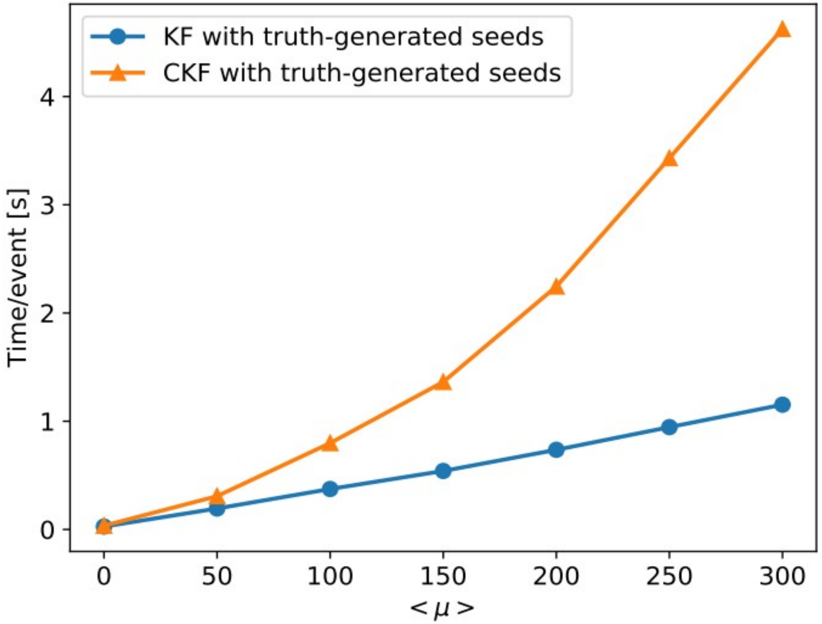
ACTS CPU performance

- Efficient CPU utilization even with contextual geometry
- **Pure track fitting time ~ 0.2 ms/ track for ~15 detector layers with a single thread**

CPU utilization



(C)KF time

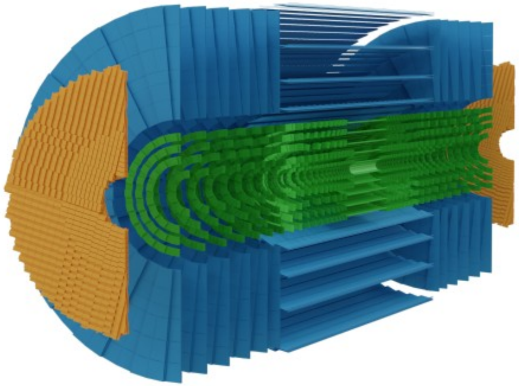


sqrt(s) = 14 TeV, $t\bar{t}$, μ =200
TrackML detector, Bz= 2T

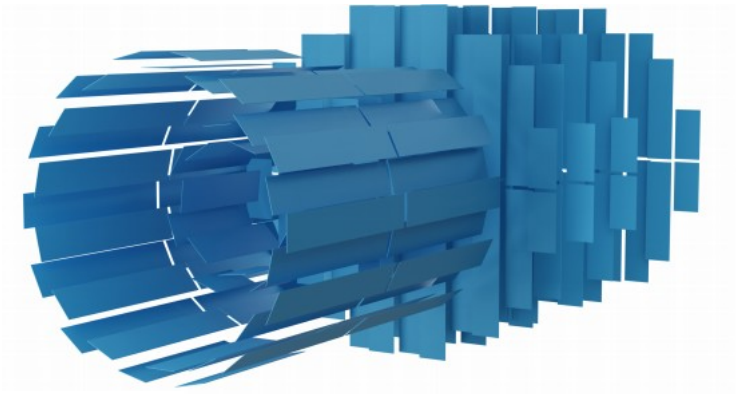
arXiv:2106.13593v1

ACTS application examples

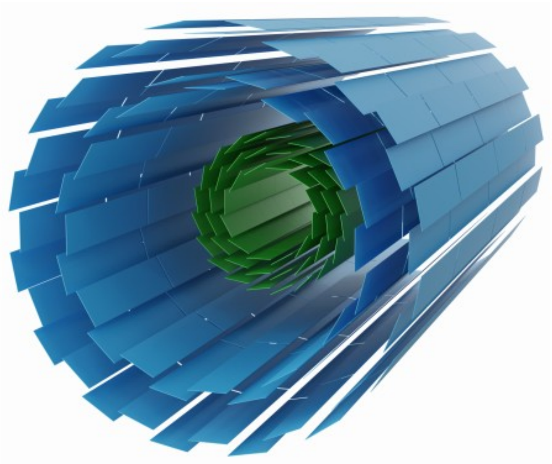
ATLAS ITk (HGTD)

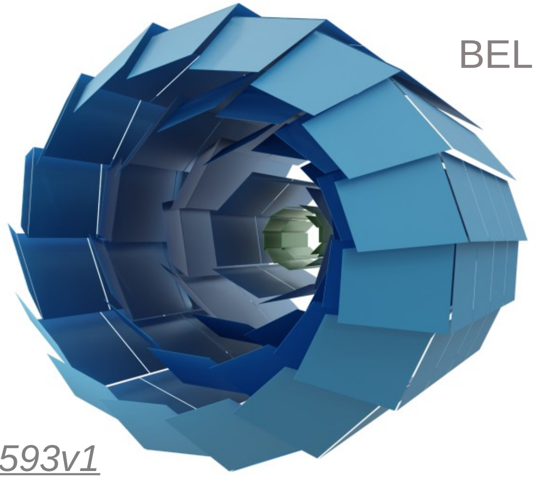


PANDA silicon



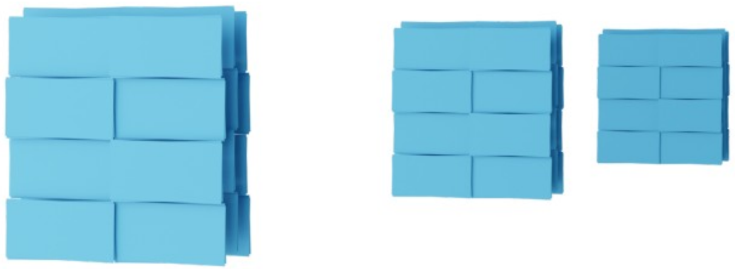
sPHENIX silicon





BELLE II

FASER

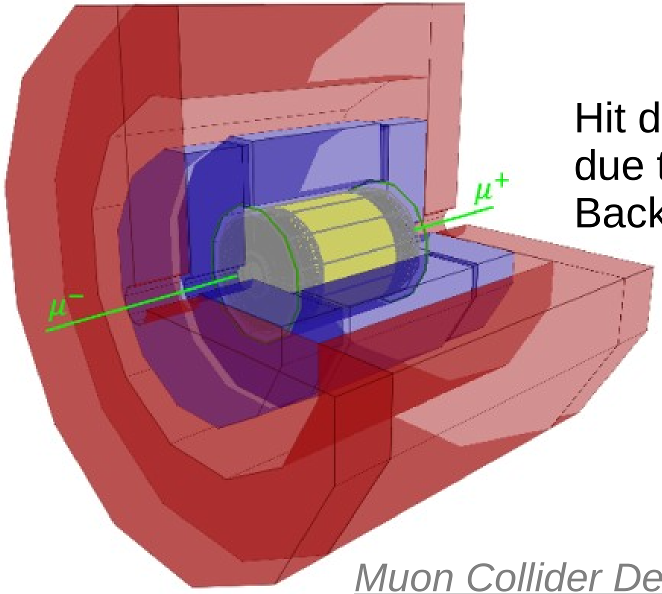


arXiv:2106.13593v1

ACTS application example: Muon Collider

Great potential for discovery in the multi-TeV energy range!

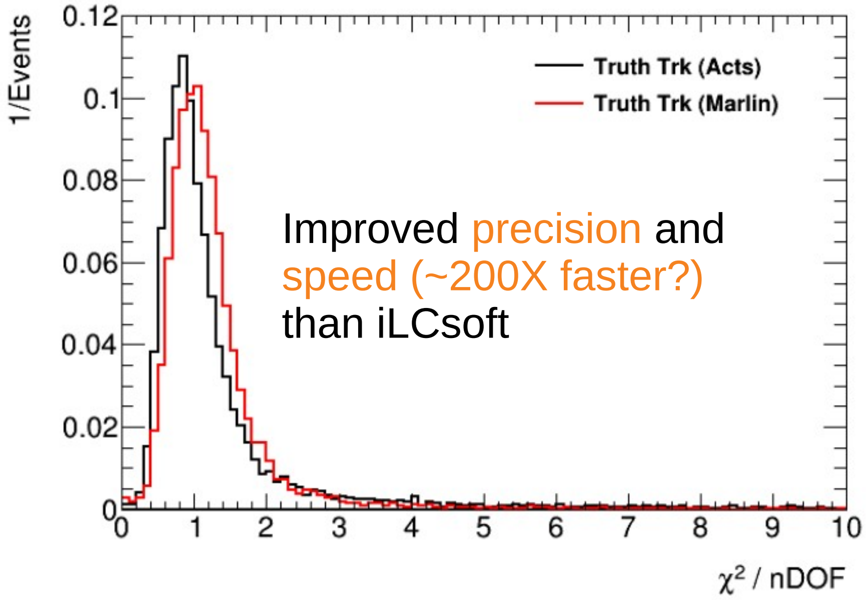
EW measurement, Higgs couplings, new resonances
DM search …



Muon Collider Detector

Hit density is 10x HL-LHC due to Beam Induced Background (BIB)

Plots from K. Krizka



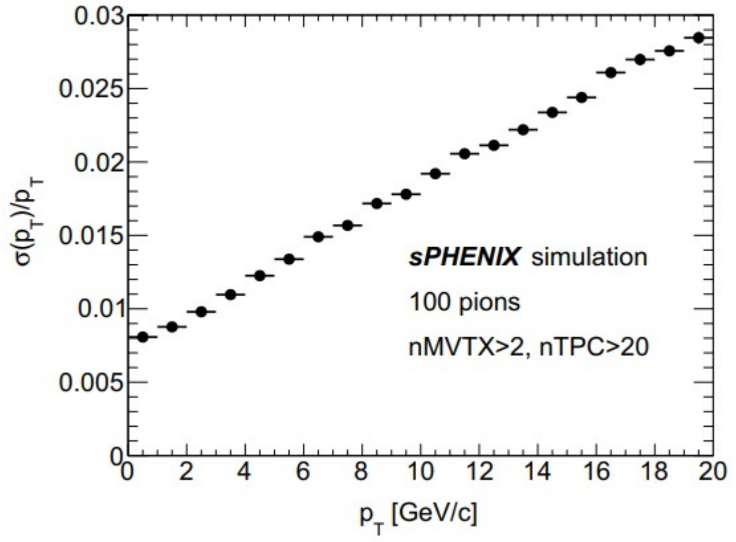
Improved precision and speed (~200X faster?) than iLCsoft

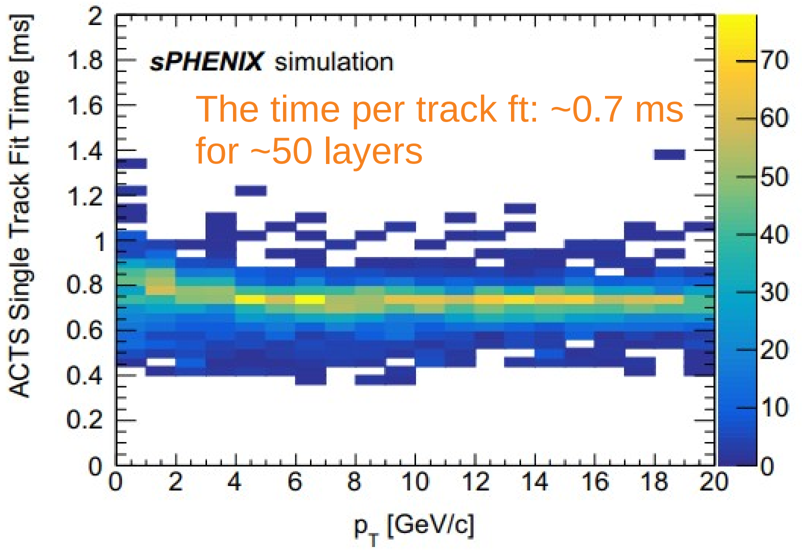
Fit Library	Execution Time
ACTS	0.5 ms / evt
iLCsoft	100 ms / evt

ACTS application example: sPHENIX

- Study quark-gluon-plasma, partonic structure of protons and nuclei in p+p, p+Au, Au+Au
  - Au+Au collision produces ~1000 tracks/ event
- ACTS provides good tracking resolution needed to resolve high momentum jets
  - $\Delta p/p \lesssim 0.2\% \cdot p$ (GeV) for pT > 10 GeV tracks
- ACTS provides **X8 faster** tracking than GenFit package
  - Total tracking time is 10 s/event (fitting time: ~1 s/event)

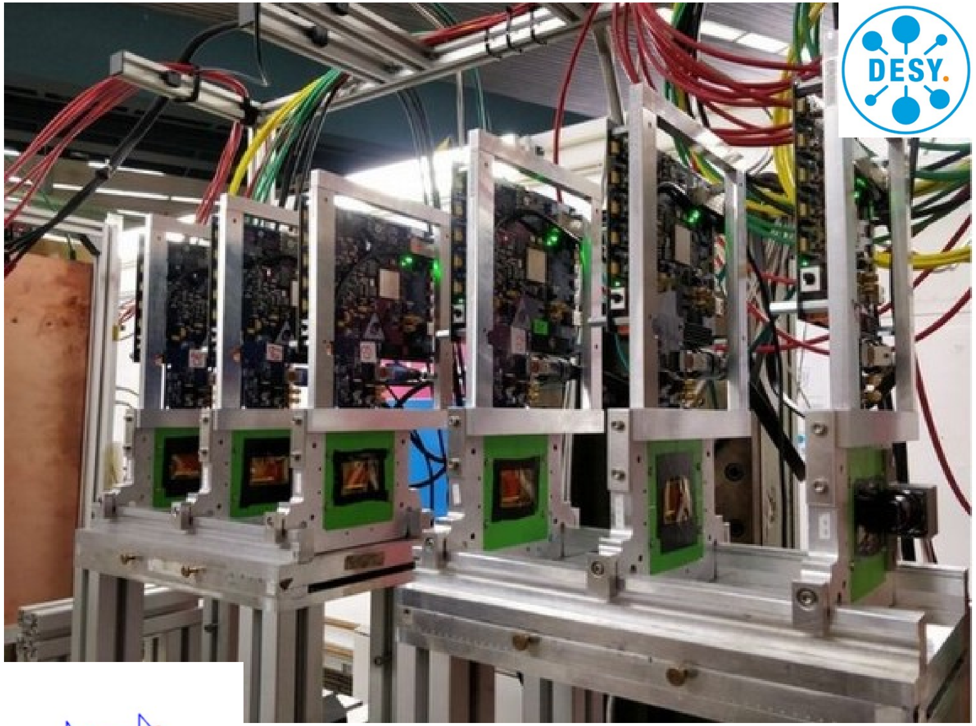
Comput Softw Big Sci 5, 23 (2021)



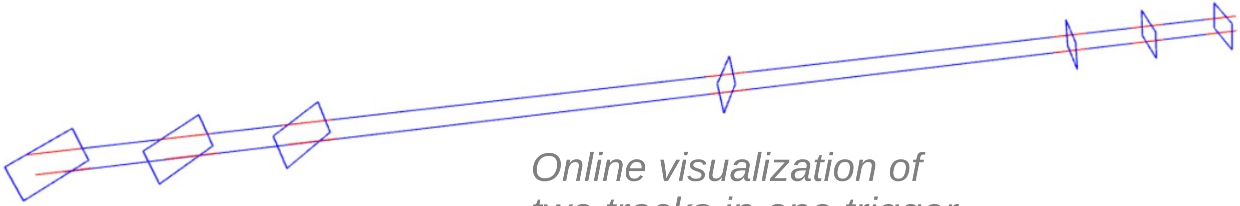


ACTS application example: ADENIUM beam telescope

- Beam telescope is a key instrumental tool for particle detector prototyping
- Combined tracking fitting and finding with CKF much ease the tracking process
- Good time performance allows **online track reconstruction and visualization**
  - Event processing rate up to 20 kHz in a single thread!



ADENIUM beam telescope developed by Y. Liu



Online visualization of two tracks in one trigger

How to achieve accurate, efficient and fast tracking for various detectors?

I. Develop a new high performant common tracking toolkit

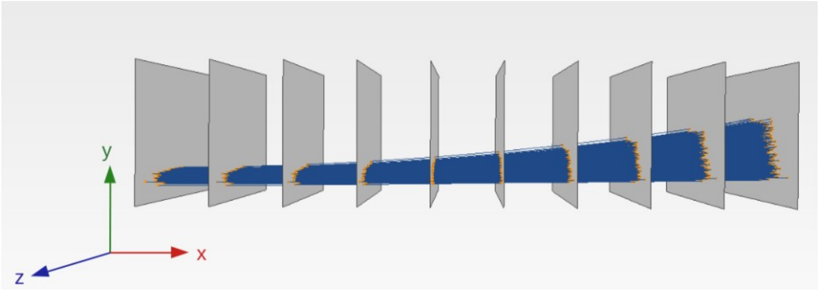
II. Parallelization, acceleration and ML

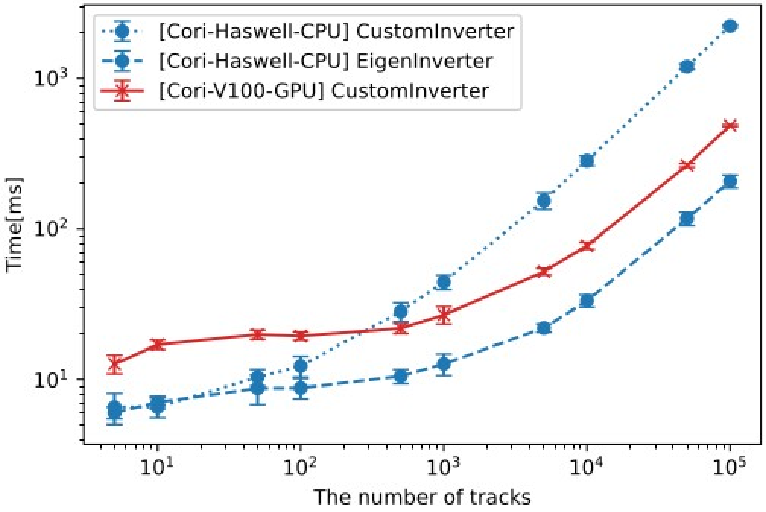
The GPU-accelerated track fitting

- ALICE, LHCb. CMS are exploring GPU for HLT trigger
- The first look at heterogeneous computing in ACTS was porting a full KF to GPU
  - **~ 4.5X speed gain for >1000 tracks!**
  - But not detector agnostic yet

Comput Softw Big Sci 5, 20

See Poster at ACAT21





CPU

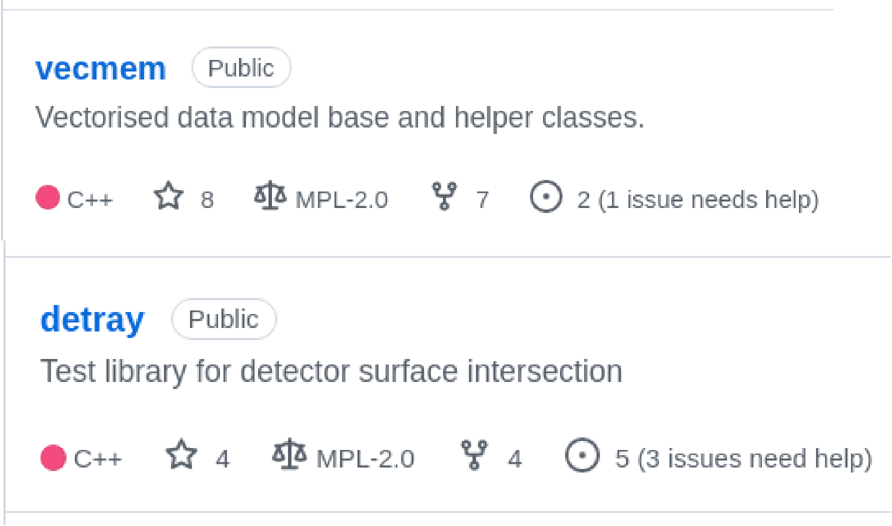
Sys.		S×C×T	Clock rate (GHz)	Mem. (GB)
1	Intel Xeon E5-2698 v3 (Cori-Haswell-CPU)	2×16×2	2.30	128
1	Intel Xeon Phi 7250 (Cori-KNL-CPU)	1×68×4	1.40	96
2	Intel Xeon Gold 5115 (NAF-SL-CPU)	2×10×2	2.40	376

GPU

Sys.		FP32 cores	FP64 cores	Clock rate (GHz)	Mem. (GB)
1	GV100-SXM2 (Cori-V100-GPU)	5120	2560	1.53	16
2	GP100-PCIe (NAF-P100-GPU)	3584	1792	1.48	16
2	GV100-SXM2 (NAF-V100-GPU)	5120	2560	1.53	32

The current ACTS parallelization R&D landscape

- Active on-going development **towards a full track chain on GPU** in ACTS community
  - Needs general solutions for difficulties with GPU: C++ STL containers and algorithms not usable, polymorphism not supported ...



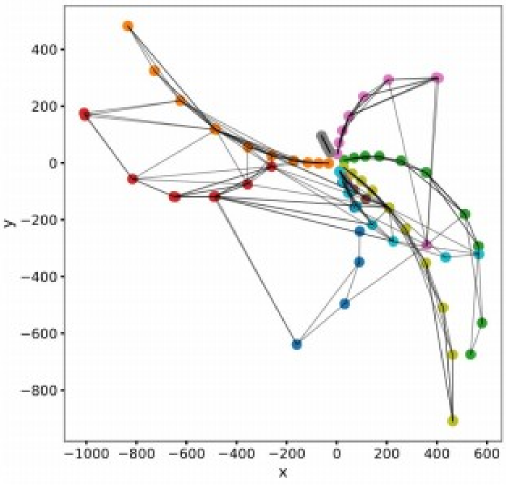
Designing **STL-like containers** for both CPU and GPU

Geometry navigation **without runtime polymorphism** based on indiced surfaces (boundaries, transformations, material...)

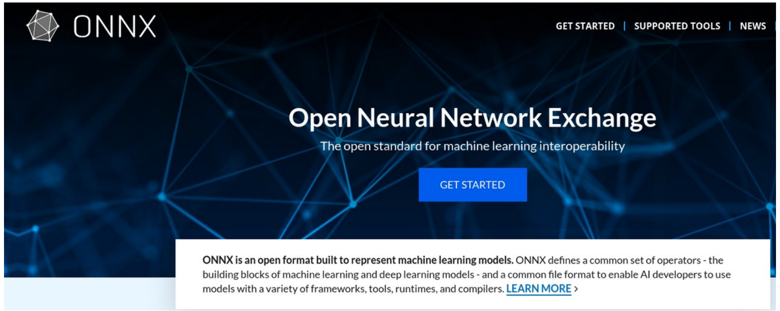
Machine Learning-based tracking algorithms

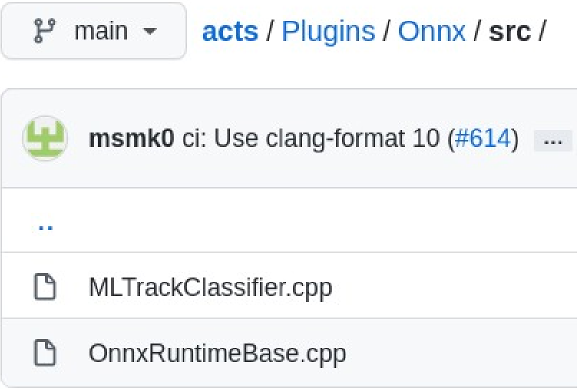
- ML is widely deployed in tracking domain
  - GNN for track finding (e.g. Exa.TrkX)
  - Evolutionary algorithm for parameters tuning
  - DNN based track classification
  - KNN for surface prediction
  - …

See poster at ACAT21



- The microsoft ONNX plugin was implemented in ACTS to allow **deployment of ML solutions**



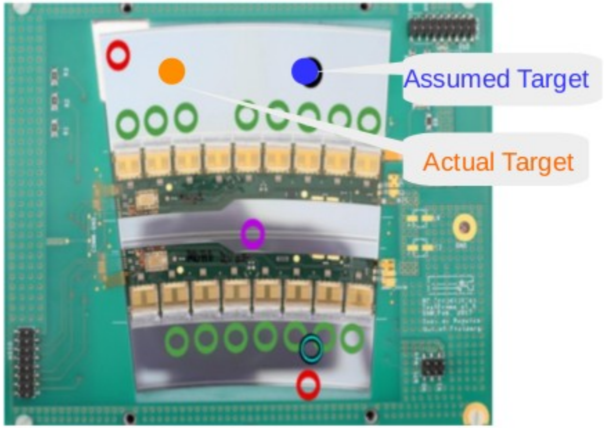


Track-based detector alignment

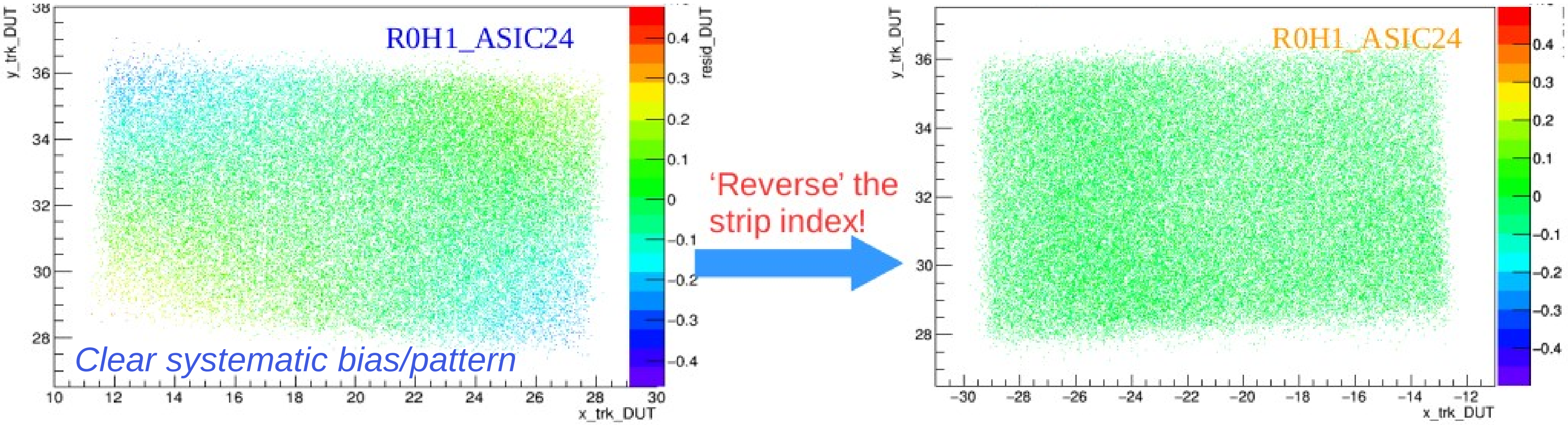
Tracking can help with detector alignment

- Alignment is necessary to provide accurate description of detector placement (translation + rotation)
- Misalignment will deteriorate the track resolution, but **tracking can notice and correct the misalignment**
  - e.g. a mistake of moving DUT to wrong target, detected by the dedicated tracking for radial strips

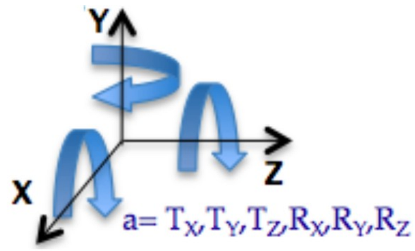
First ATLAS ITk Endcap strip R0 module at testbeam in 2017



Angular residual vs. local position



How to do track-based alignment?



- Estimation of alignment parameter α that is common for a sample of tracks by minimizing the total χ^2 w.r.t. to track parameters x_i and α:

$$\chi^2 = \sum_i {\chi_i}^2 = \sum_i [\vec{m}_i - \vec{h}_i(\vec{x}_i(\vec{\alpha}), \vec{\alpha})]^T V^{-1} [\vec{m}_i - \vec{h}_i(\vec{x}_i(\vec{\alpha}), \vec{\alpha})]$$

- Found the best α by minimizing the χ^2, i.e.
  - $0 \equiv \frac{d\chi^2}{d\vec{\alpha}}$ with $\forall_i \frac{\partial \chi_i^2}{\partial \vec{x}_i} = 0$
  - Needs to obtain $\Delta\alpha$ via solving:

$$\frac{d^2\chi^2}{d^2\vec{\alpha}} |_{\vec{\alpha}_0} \Delta\vec{\alpha} = -\frac{d\chi^2}{d\vec{\alpha}} |_{\vec{\alpha}_0}$$

How to get $\Delta\alpha$

```
// Solve the linear equation to get alignment parameters change
alignResult.deltaAlignmentParameters =
    -sumChi2SecondDerivative.fullPivLu().solve(sumChi2Derivative);
```

$$\left.\frac{d^2\chi^2}{d\alpha^2}\right|_{\alpha_0} \Delta\alpha = -\left.\frac{d\chi^2}{d\alpha}\right|_{\alpha_0}$$

Solved with LU decomposition

claimed stable and well tested with large matrices

$$\frac{d\chi^2}{d\alpha} = 2A^T V^{-1}\left(V - HCH^T\right)V^{-1}r,$$

$$\frac{d^2\chi^2}{d\alpha^2} = 2A^T V^{-1}\left(V - HCH^T\right)V^{-1}A.$$

The first and second derivatives for a single track is calculated in Acts::detail::trackAlignmentState (#234)

$r = m - h(x, \alpha)$ The track residual

V The measurement covariance

$H = \left.\frac{\partial h(x)}{\partial x}\right|_{x_0}$ The projection matrix from (bound) track parameters to measurement

C The global (bound) track parameters covariance with correlation between track parameters at different measurement surface taken into account

Calculated by Acts::detail::globalTrackParametersCovariance

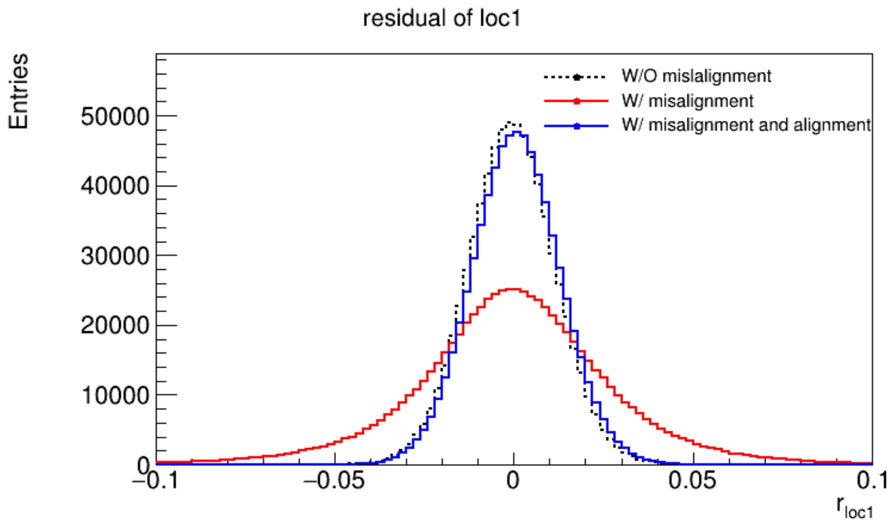
$A_{k\ell} \equiv \frac{\partial r_k}{\partial \alpha_\ell}$ The derivative of residual w.r.t. alignment parameters

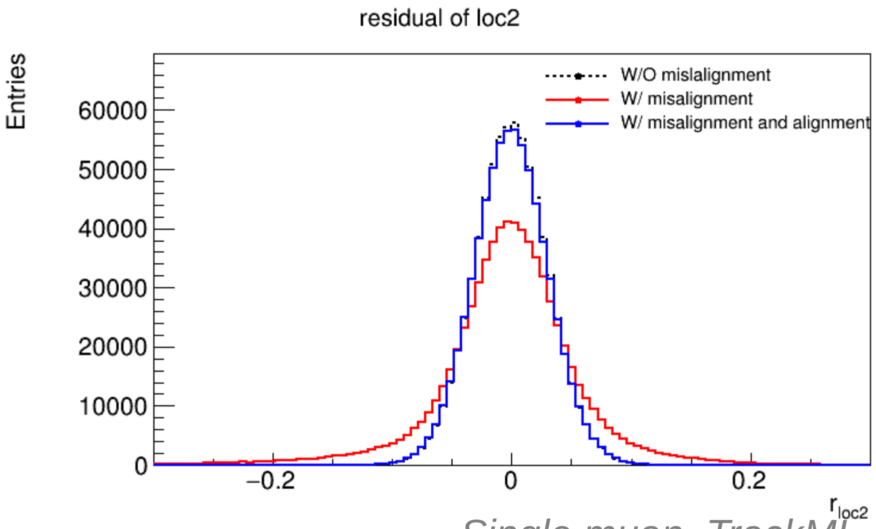
Calculated by Surface::alignmentToBoundDerivative

4

Using KF for track-based detector alignment

- KF-based tracking is commonly used, e.g. ATLAS and CMS experiments
  - KF-based alignment is more straightforward than χ^2 fitter based alignment
- **A KF-based alignment prototype in ACTS has been developed and looks promising!**





Single muon, TrackML detector, Bz= 2T

Summary

Summary

- Tracking is pivotal to reconstruction in HEP
- Tracking has direct and indirect impact to physics precision
- Tracking is non-trivial and can be more challenging at future HEP experiments
- Modern performant common tracking software (ACTS) is being developed and gets worldwide users from >10 experiments

Summary

- Tracking is pivotal to reconstruction in HEP
- Tracking has direct and indirect impact to physics precision
- Tracking is non-trivial and can be more challenging at future HEP experiments
- Modern performant common tracking software (ACTS) is being developed and gets worldwide users from >10 experiments
- Outlook
  - There are still tools to develop and optimize in ACTS
  - Interplay with experiment frameworks is the key to make a real generic toolkit

Collaboration is always welcome!

GitHub https://github.com/acts-project

Mattermost https://mattermost.web.cern.ch/acts/channels/town-square

acts-developers@cern.ch

acts-users@cern.ch

acts-parallelization@cern.ch

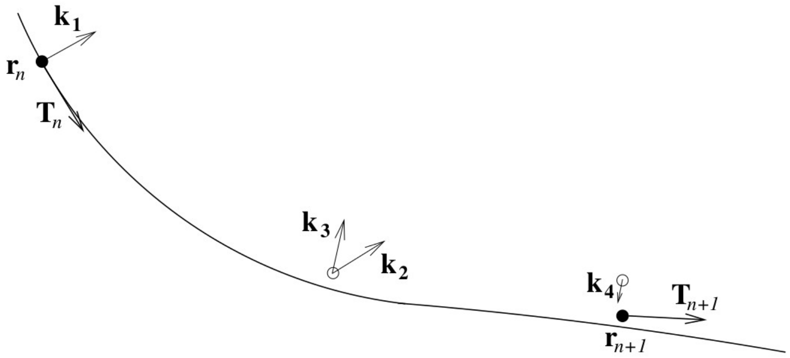
Backup

Q&A

Track parameter propagation

- Magnetic field is usually non-homogeneous.
  - The particle motion equation needs to be solved numerically
- The fourth-order Runge-Kutta-Nyström method for

$$\frac{d^2y(x)}{dx^2} = f(x, y, y')$$



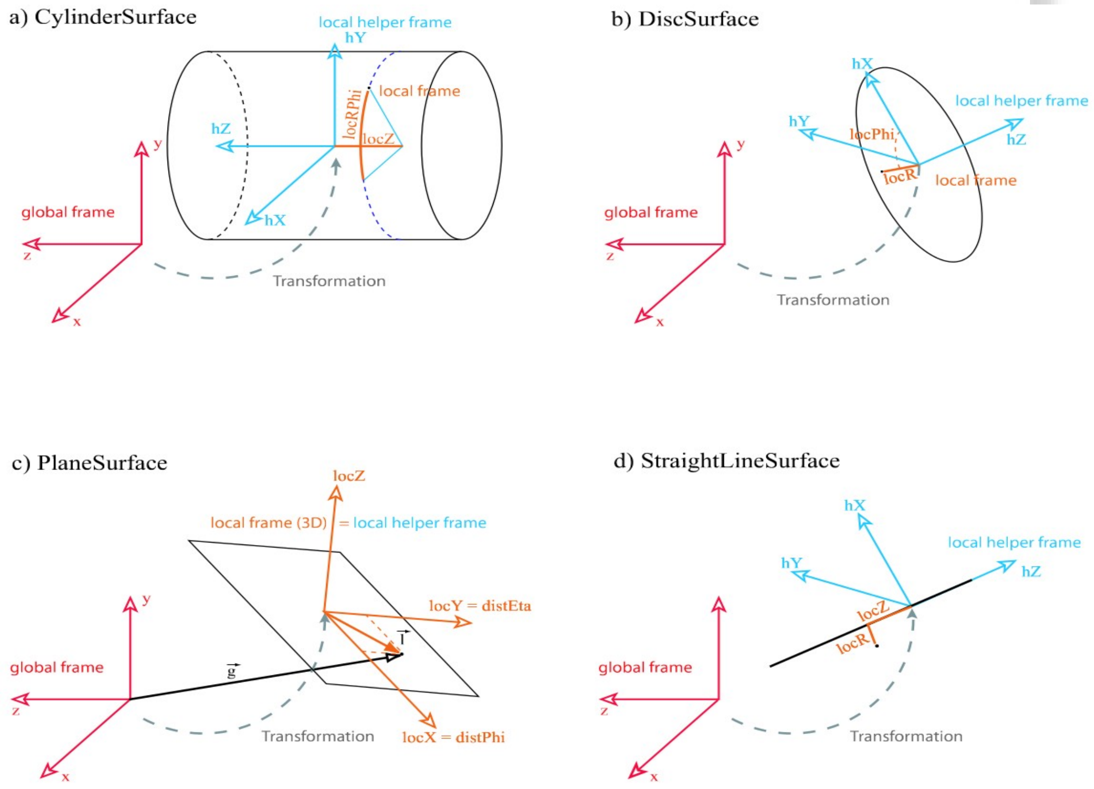
E. Lund et al. 2009 JINST 4 P04001

$$\begin{aligned}
k_1 &= f(x_n, y_n, y'_n) \\
k_2 &= f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1, y'_n + \frac{h}{2}k_1\right) \\
k_3 &= f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}y'_n + \frac{h^2}{8}k_1, y'_n + \frac{h}{2}k_2\right) \\
k_4 &= f\left(x_n + h, y_n + hy'_n + \frac{h^2}{2}k_3, y'_n + hk_3\right) \\
y'_{n+1} &= y'_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
y_{n+1} &= y_n + hy'_n + \frac{h^2}{6}(k_1 + k_2 + k_3)
\end{aligned}$$

ACTS Surface

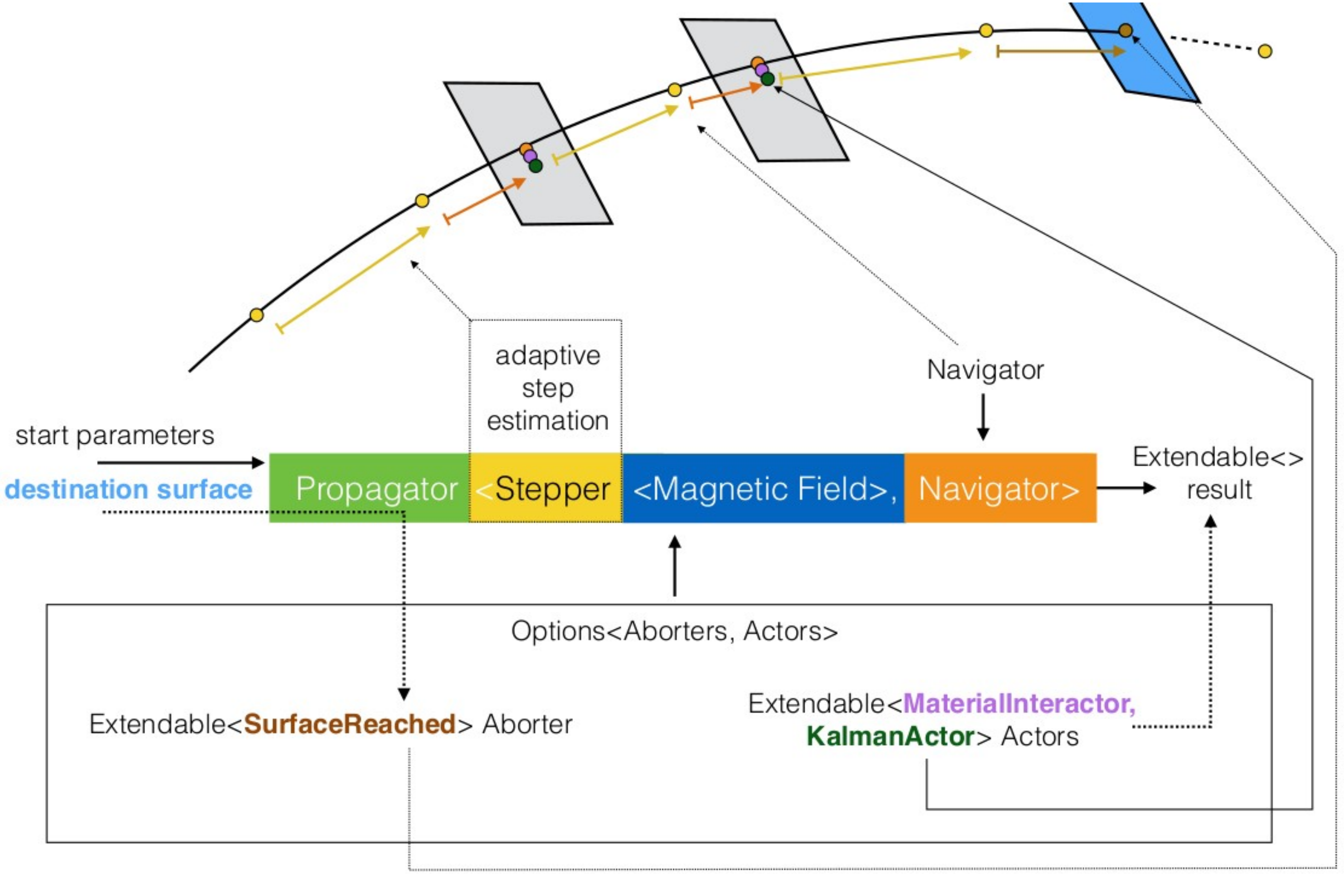
- Acts::Surface is the key component of tracking geometry
  - Surface concepts are largely transcribed from ATLAS SW
- Different concrete surfaces have different local coordinate definitions and shapes
  - Shape is described by Acts::SurfaceBounds

Surface types in ATLAS SW



ACTS Track parameter propagator interface

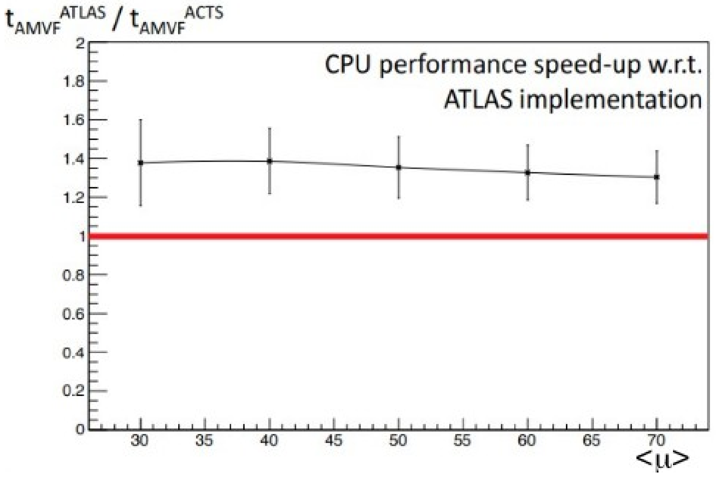
Integrating particle transport & geometry navigation
Highly-templated design emphasizing on speed and customizability

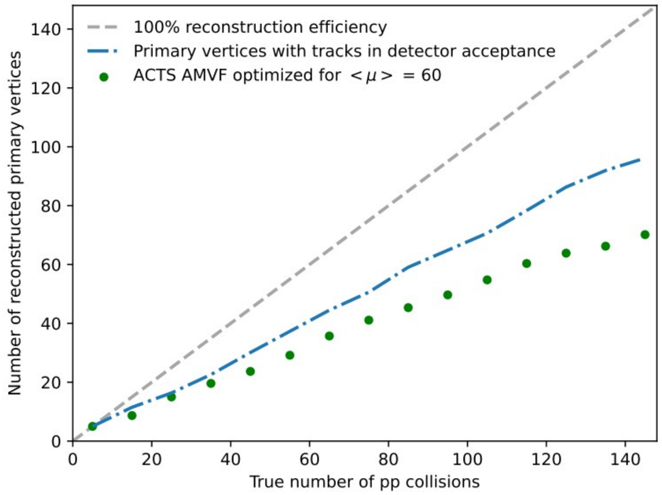


ACTS vertexing

- Iterative fitting-after-finding
  - Iterative Vertex Finder (IVF) (used at ATLAS Run-2)
- Finding-through-fitting
  - Adaptive Multi-Vertex Finder (AMVF) (to be used at ATLAS Run-3)

AMVF timing performance





See B. Schlag’s slides