

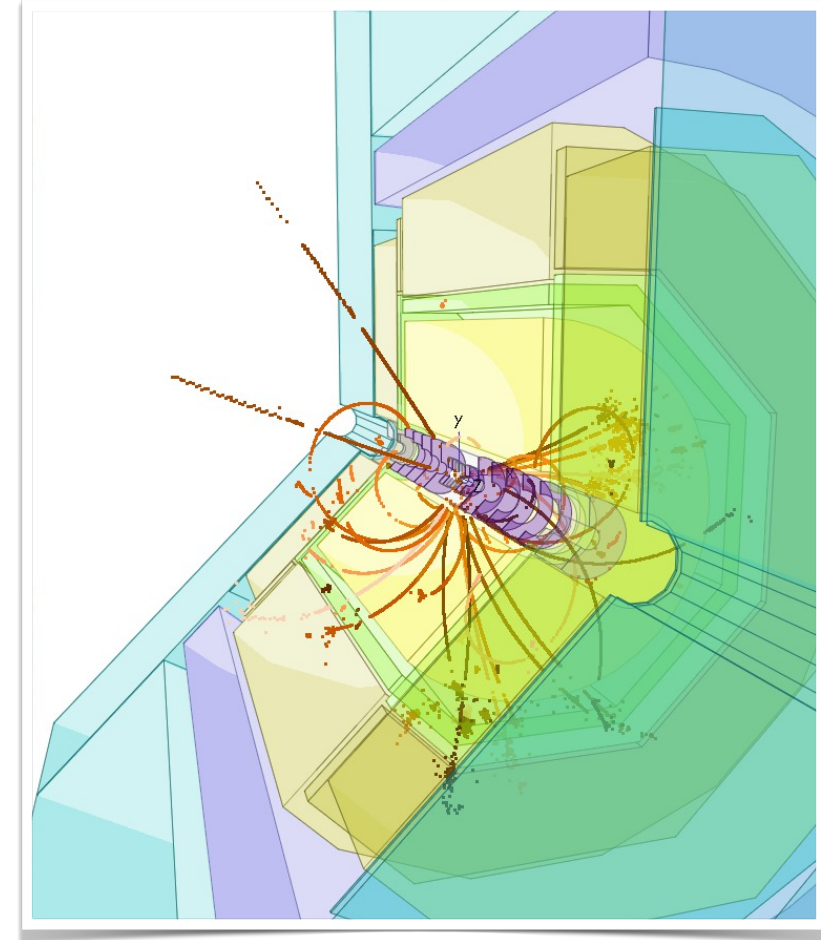
Software for future collider studies: ILD and beyond

ILD Meeting connected to ECFA 2024

Frank Gaede, DESY
LPHNE, Paris
Oct 8, 2024

Outline

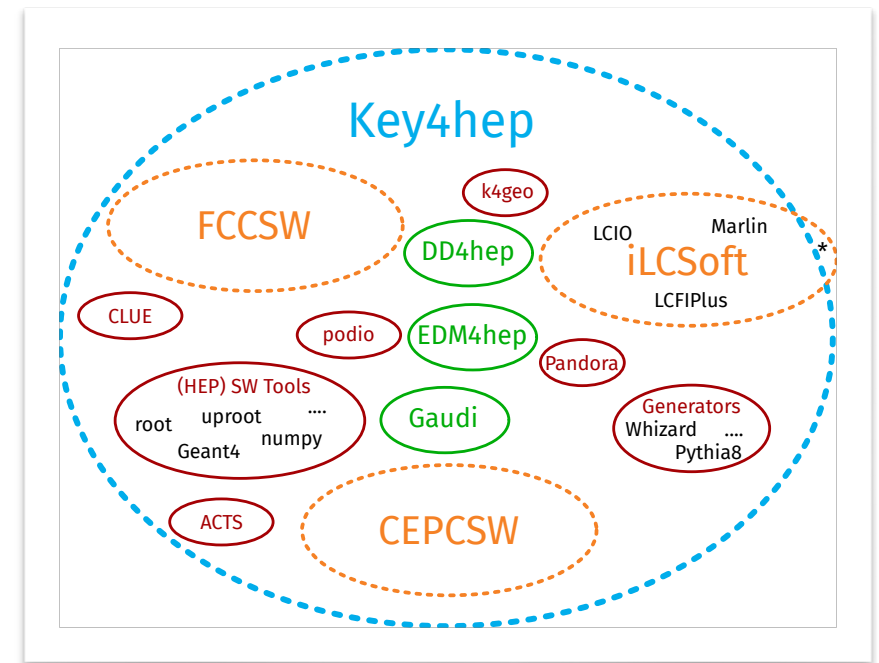
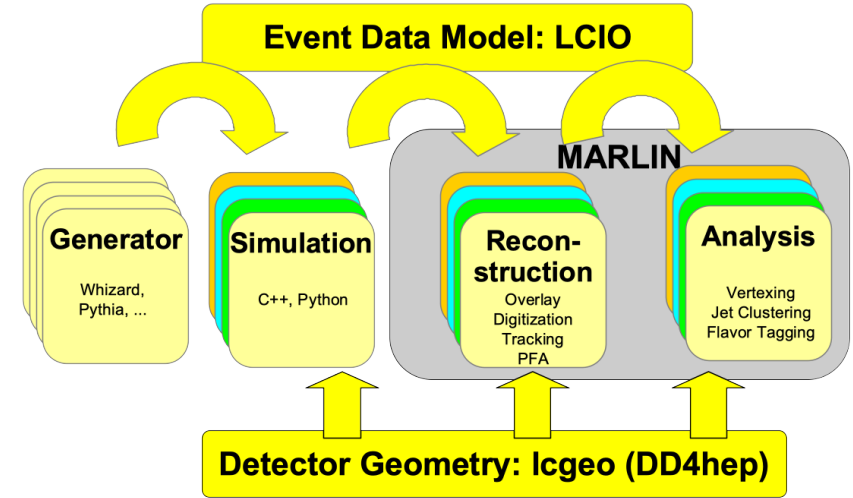
- Introduction and Reminder: Key4hep
- DD4hep detector models and reconstruction
- Standard ILD reconstruction algorithms and “Transition” to Key4hep
- **AI/ML in Key4hep:**
 - fast simulation and
 - recent HLR algorithms w/ ML
- Summary and Outlook



The common software vision

iLCSoft as integral part of Key4hep

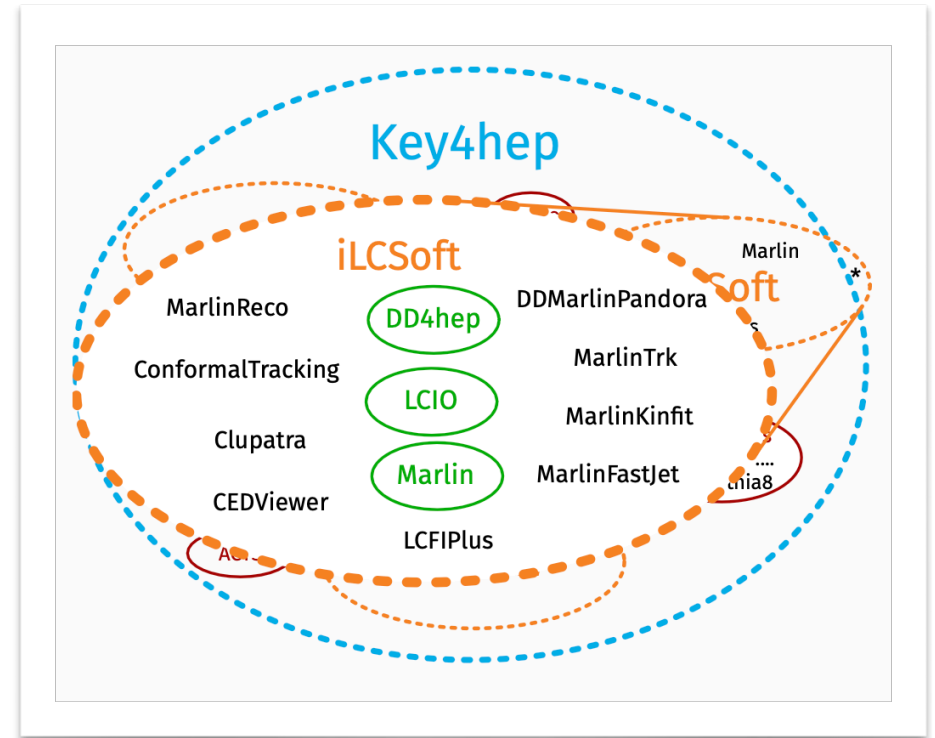
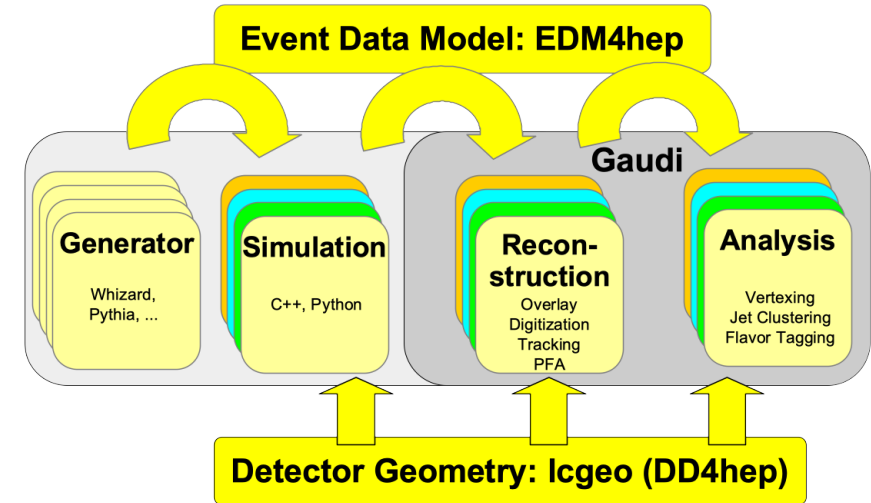
- complete set of tools for
 - **generation, simulation, reconstruction, analysis**
 - build, package, test, deploy
- core ingredients of current **Key4hep**
 - **PODIO** for **EDM4hep** (based on LCIO and FCC-edm)
 - **Gaudi** framework, devel/used for (HL-)LHC
 - **DD4hep** for geometry
 - originally developed for LC now adopted by community
 - **spack** package manager



The common software vision

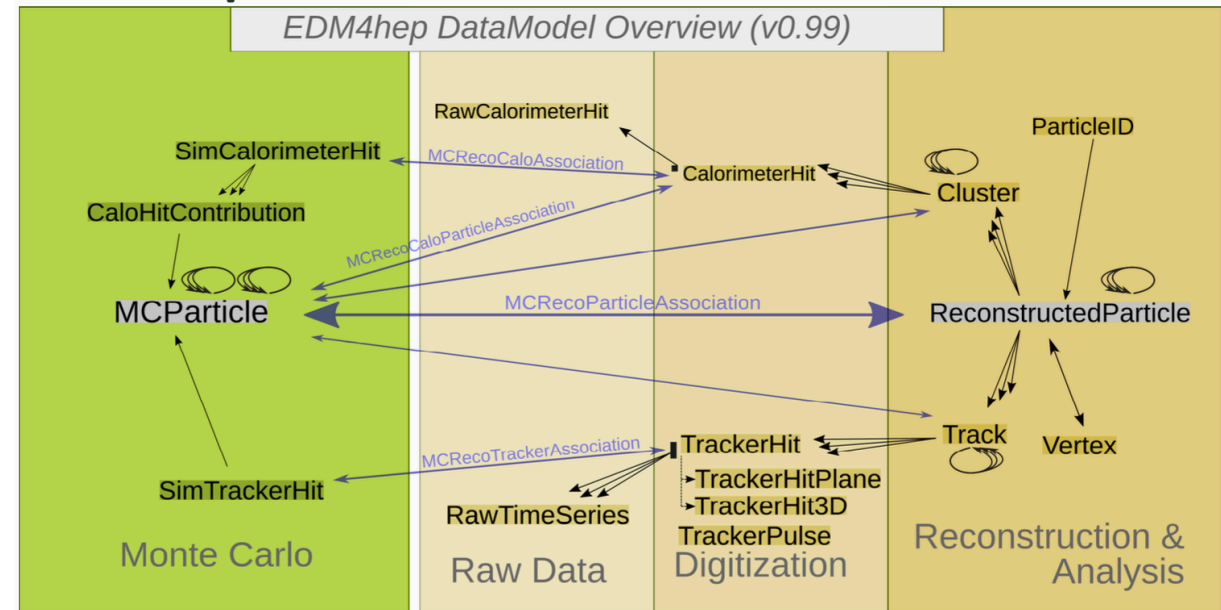
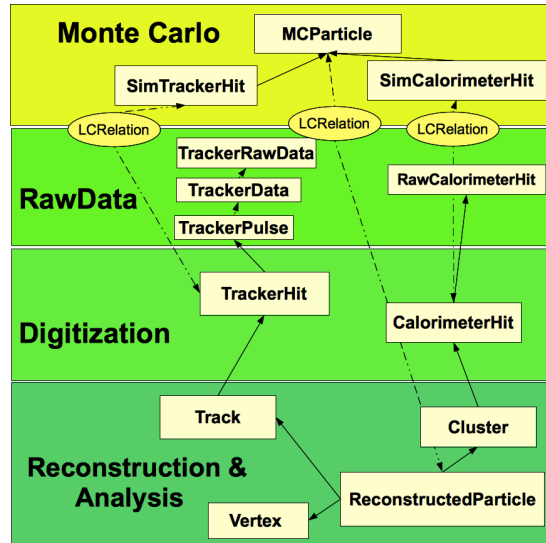
iLCSoft as integral part of Key4hep

- complete set of tools for
 - **generation, simulation, reconstruction, analysis**
 - build, package, test, deploy
- core ingredients of current **Key4hep**
 - **PODIO** for **EDM4hep** (based on LCIO and FCC-edm)
 - **Gaudi** framework, devel/used for (HL-)LHC
 - **DD4hep** for geometry
 - originally developed for LC now adopted by community
 - **spack** package manager

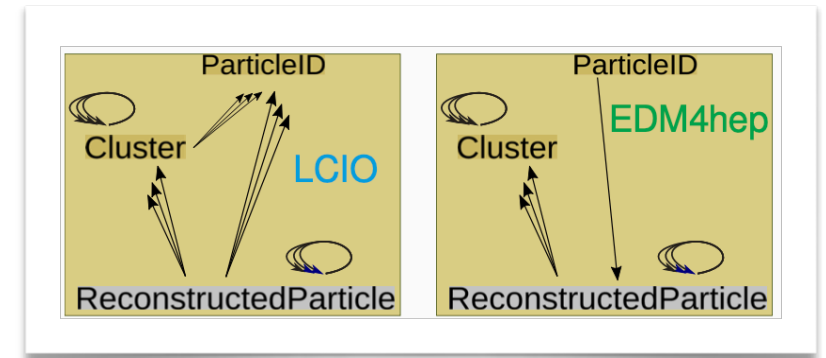


EDM4hep - Event Data Model in Key4hep

the designated successor of LCIO

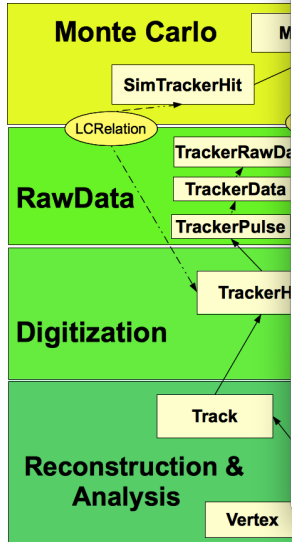


- EDM4hep to large extend reproducing (battle-proven) LCIO EDM - yet with
 - more strict **mutability** concept (**thread safety**)
 - more **consistent** linking to *lower level* data objects
 - more performant reading (ROOT files)
- **k4EDM4hep2LcioConv** provides consistent conversion between the two
 - **production version v01-00** to be released very soon



EDM4hep - Event Data Model in Key4hep

the designated success



```

edm4hep_plot_hrecoil.jl
using EDM4hep
using EDM4hep.RootIO
using Plots
using Statistics

file = "/Users/gaede/data/rv02-02_sv02-02_mILD_L5_o1_v02.E250-SetA.I402003.Pe2e2h.el.pR.n000_d_0
stn_15089_0.edm4hep_v02-03-02.root"

reader = RootIO.Reader(file)

dimuonmass = Float64[]
recoilmass = Float64[]

mass4v(a) = sqrt(a[1]^2 - a[2]^2 - a[3]^2 - a[4]^2)
momCMS = [250., 0., 0., 0.]

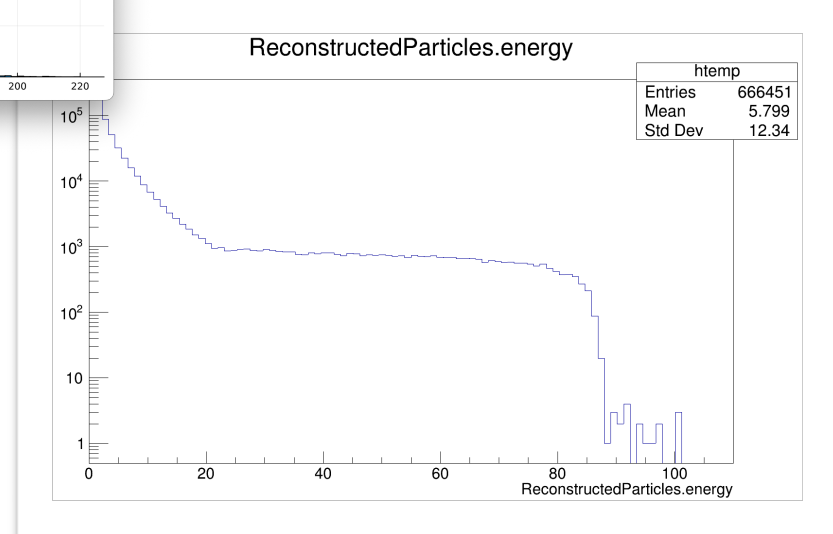
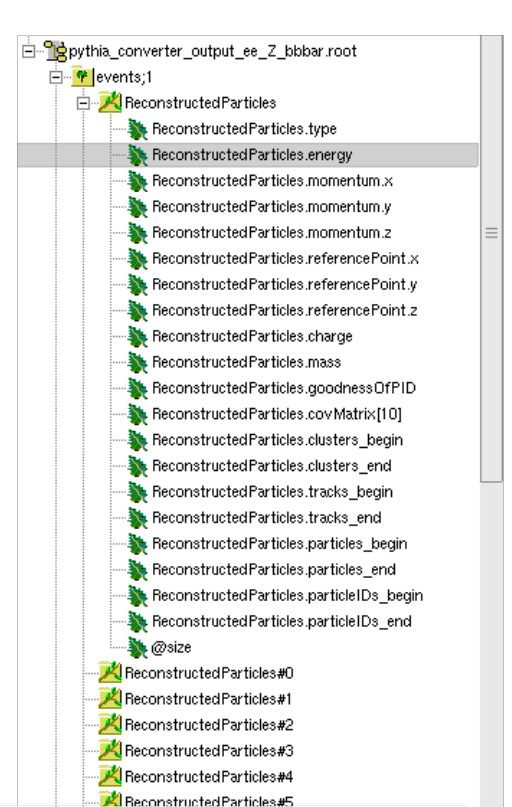
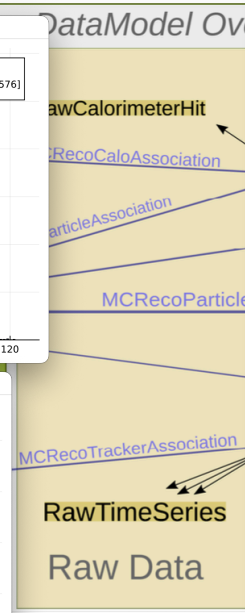
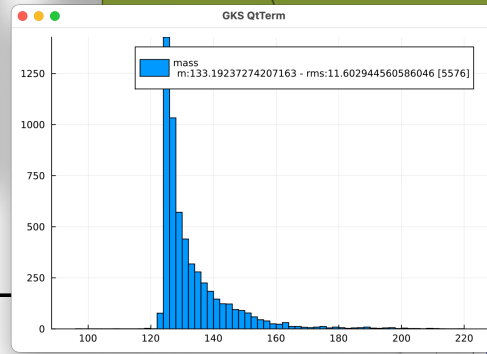
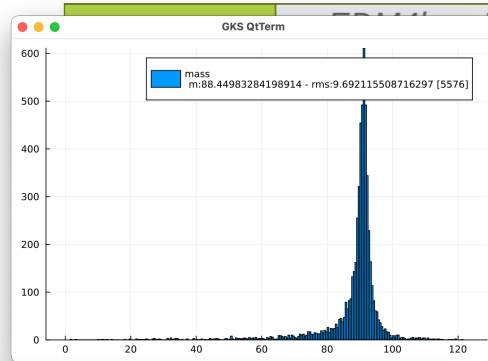
events = RootIO.get(reader, "events");
get_pfos = RootIO.create_getter(reader, "PandoraPFOs"; selection=[:energy,:type,:momentum])

for evt in events
    pfos = get_pfos(evt)
    muons = []
    for pfo in pfos
        if abs(pfo.type) == 13 # find muons
            push!(muons, [pfo.energy, pfo.momentum.x, pfo.momentum.y, pfo.momentum.z])
        end
    end
    if length(muons) == 2 # exactly two muons
        push!(dimuonmass, mass4v(muons[1] + muons[2]))
        push!(recoilmass, mass4v(momCMS - muons[1] - muons[2]))
    end
end

# ----- now plot histogram w/ recoil mass -----
m = mean(dimuonmass)
v = std(dimuonmass)
n = length(dimuonmass)
hist = histogram(dimuonmass, label="mass \n m:$m - rms:$v [$n]")

m = mean(recoilmass)
v = std(recoilmass)
n = length(recoilmass)
hist2 = histogram(recoilmass, label="mass \n m:$m - rms:$v [$n]")

edm4hep_plot_hrecoil.jl Top L42 (Julia)
  
```



- EDM4hep to large extend reproducing (battle-proven) L
- more strict mutability concept (thread safety)

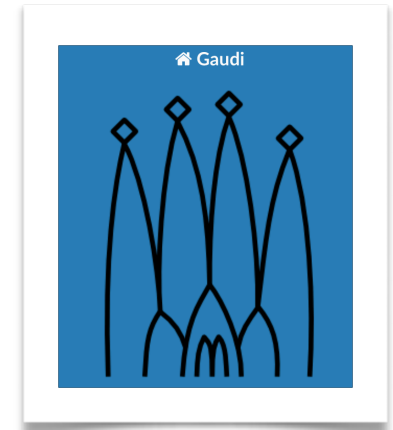
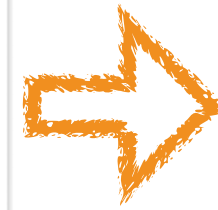
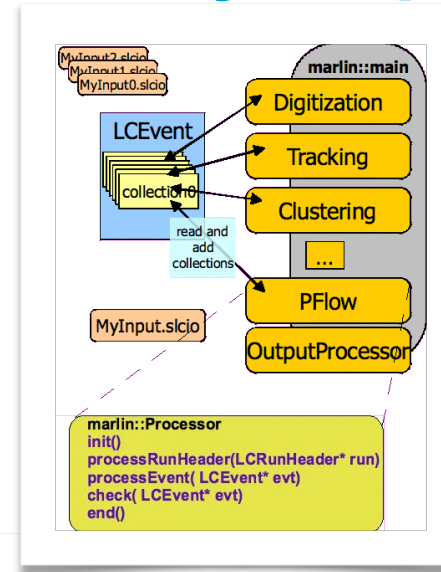
using ROOT files w/ columnar storage (TTree or RNTuple) - (optionally SIO files)
 fast analysis w/ C++ or Python using EDM4hep
 RDataFrame + EDM4hep (FCCAnalysis)
 Python or Julia reading ROOT files directly

the two

Gaudi - application framework in Key4hep

designated successor of Marlin

- C++ application framework for HEP
- developed at CERN
- used in production for
 - LHCb and ATLAS (*battle-proven*)
 - FCC-SW and smaller experiments
 - and now in Key4HEP
- highly configurable
 - EDM, workflows (algorithms)
- allows parallelisation through multi-threading
- integration of heterogeneous resources
 - CPUs, GPUs, FPGAs,...



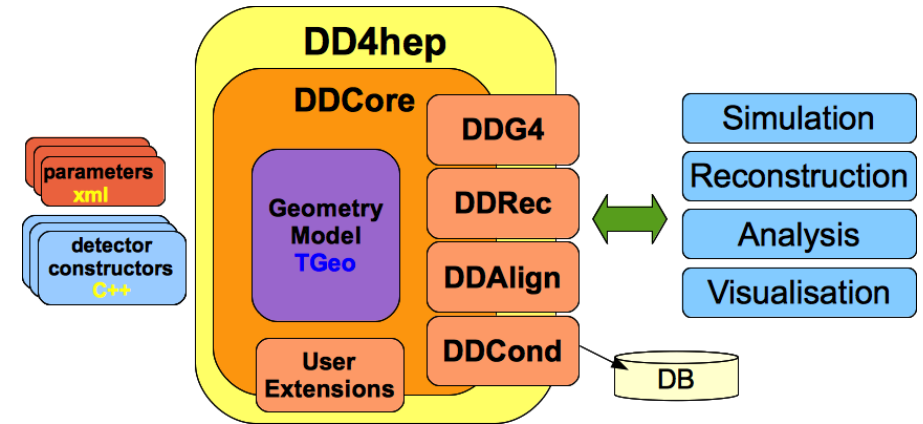
	Marlin	Gaudi
language	C++	C++
working unit	Processor	Algorithm
config language	XML	Python
transient data format	LCIO	anything
set up function	init	initialize
work function	processEvent	execute
wrap up function	end	finalize

GAUDI similar to MARLIN framework
yet more powerful and larger user basis

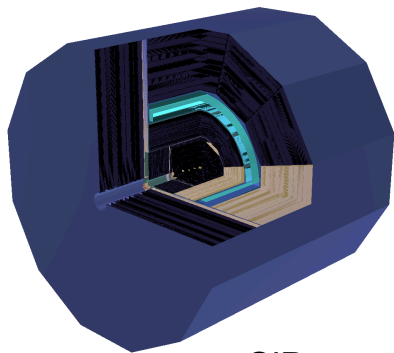
DD4hep geometry toolkit

defining the detector geometry and different views on it

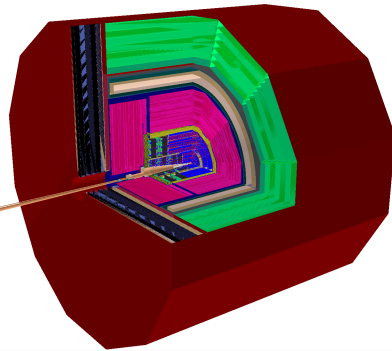
- supporting the full life cycle of the experiment
- **single source** of information for full **simulation**, **reconstruction**, **conditions**, **alignment**, **visualisation** and **analysis**
 - used by CEPC, CLIC, CMS, EIC, FCC, ILC, LHCb, ...



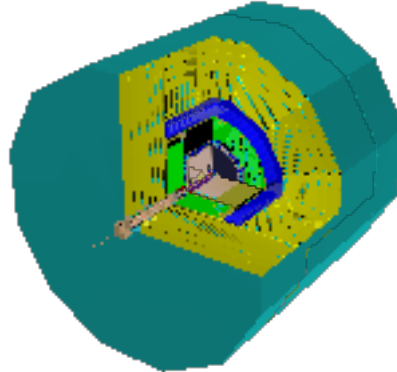
DD4hep: de facto industry standard



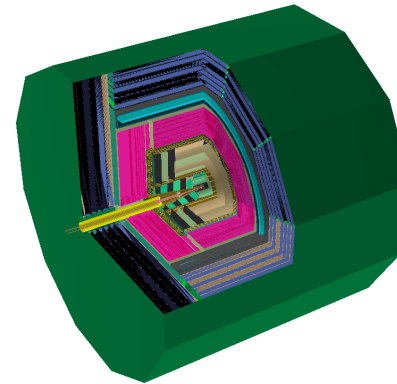
SiD



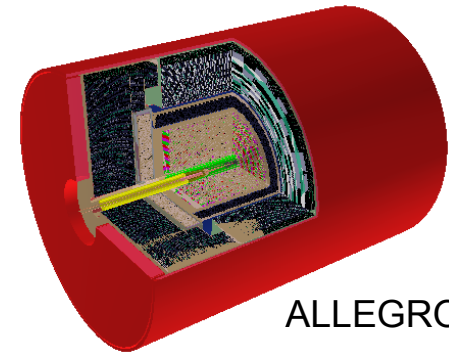
CLICdet



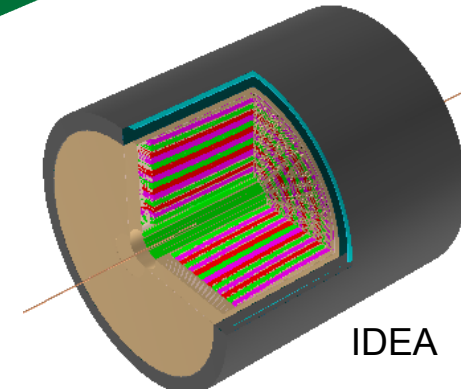
ILD



CLD



ALLEGRO



IDEA

all future Higgs factory detector simulation models in **one package**

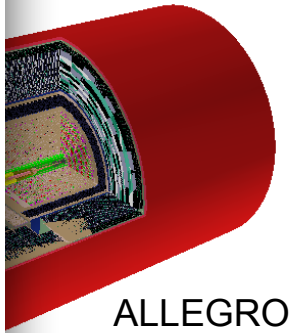
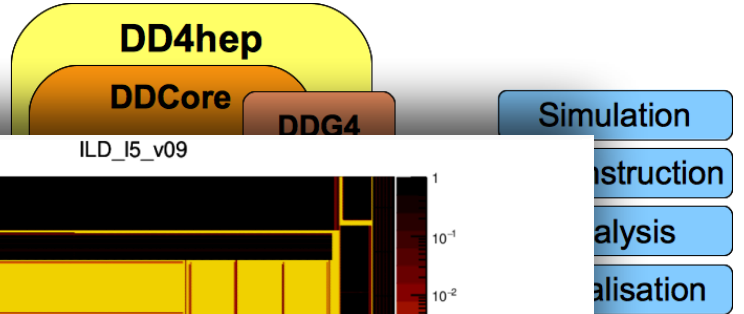
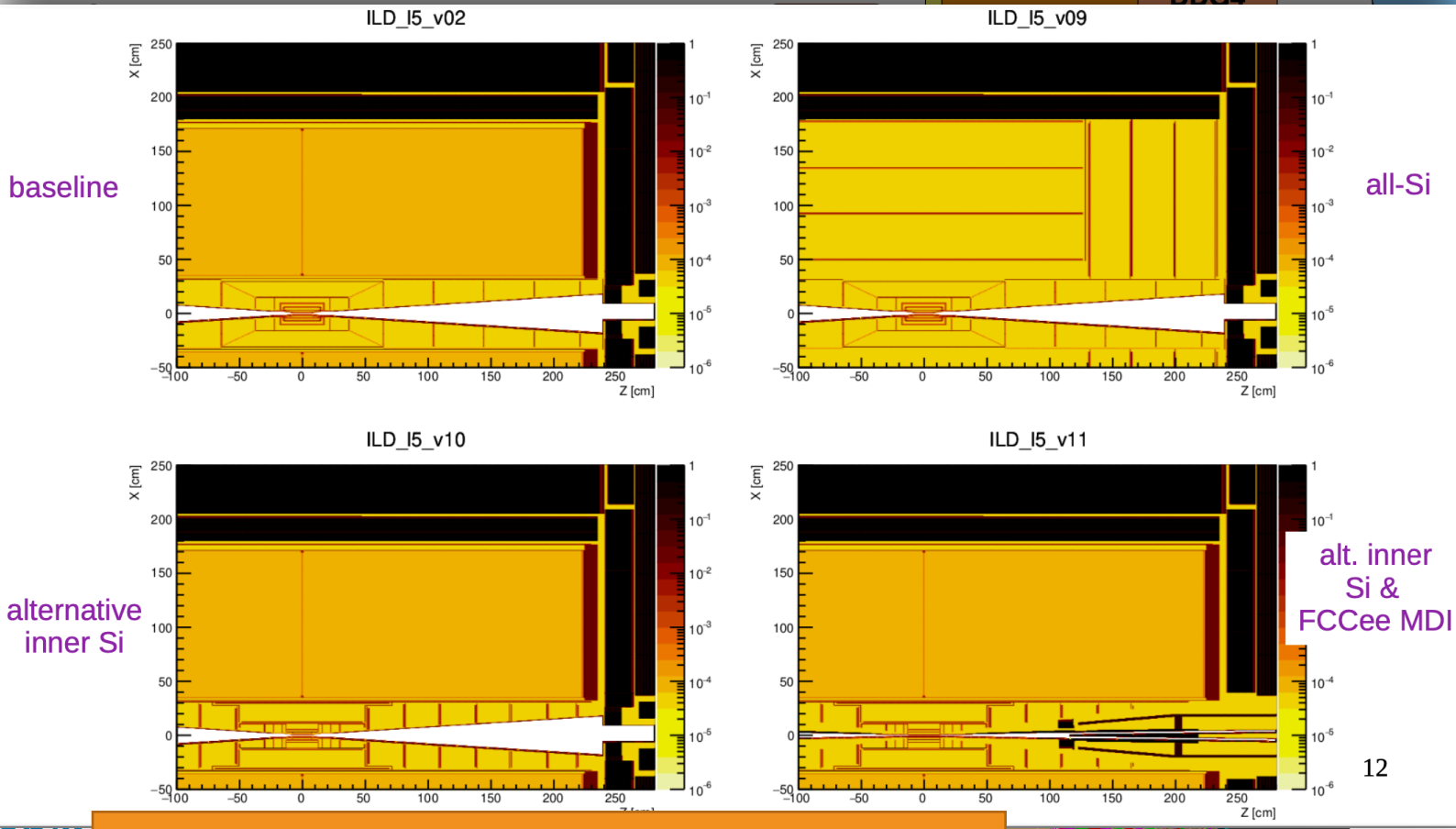
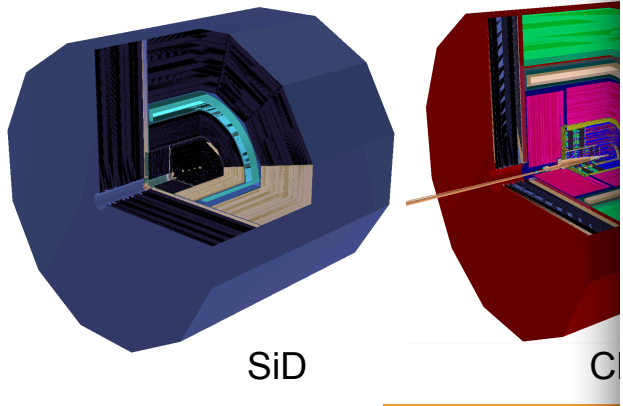
<https://github.com/key4hep/k4geo>

=> allows to **re-use reconstruction code** (for similar sub detectors) across detector concepts

DD4hep geometry toolkit

defining the detector geometry and different views on it

- supporting the full life cycle
- **single source** of information for **reconstruction**, **condition monitoring**, and **analysis**
- used by CEPC, CLIC

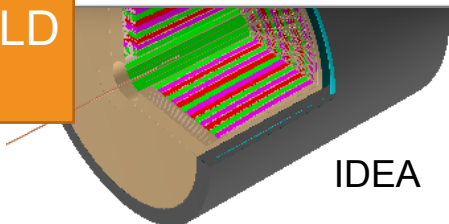


all future Higgs factory detectors

<https://github.com/key4hep/key4hep>

=> allows to re-use reconstruction across detector concepts

allows to study many detector variants for ILDC see: talk by D.Jeans



large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
 - generic API for fitting algorithms
 - large number of pattern recognition algorithms

Tracking in iLCSoft

pattern recognition and Kalman-Filter

- generic tracking API MarlinTrk based on DDRec material surfaces
- many pattern recognition algorithms exist, e.g.
 - ConformalTracking:**
 - generic algorithm that works for all Si-Trackers
 - used by CLICdet and SiD (also works for ILD inner)

achieve excellent tracking efficiencies and resolution w/ realistic tracking codes

CLICdp
Tracking efficiency vs p_T [GeV].
Conditions: $\sqrt{s} = 3$ TeV, $10^\circ < \theta < 170^\circ$, vertex R < 50 mm, $\Delta_{IC} > 0.02$ rad.
Series: No background (blue), 3 TeV $\gamma\gamma \rightarrow$ hadrons background (red).

ILD
 ϵ_{trk} vs p_T / GeV .
Conditions: $\sqrt{s} @ 500$ GeV - $p_T > 100$ MeV, $\cos(\theta) < 0.99$.
Series: IDR-L (blue), IDR-S (red).

CLICdp
 $\sigma(Dp_T/p_{T,true})$ [GeV⁻¹] vs θ [°].
Single μ^+ .
Series: $p_T = 1$ GeV (black), $p_T = 10$ GeV (red), $p_T = 100$ GeV (blue).

Momentum Resolution
 σ_{Dp_T} [GeV⁻¹] vs Momentum (GeV).
Series: $\sqrt{s} = 3$ TeV, $10^\circ < \theta < 170^\circ$, vertex R < 50 mm, $\Delta_{IC} > 0.02$ rad, $p_T > 100$ MeV, $\cos(\theta) < 0.99$.

DESY. Frank Gaede, LCWS 2021, 17.03.21

large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
 - generic API for fitting algorithms
 - large number of pattern recognition algorithms
- particle flow algorithms
 - PandoraPFA and Arbor, AprilPFA

Tracking in iLCSoft

pattern recognition and Kalman-Filter

Clupetra

→

LCIO

DDRec

→

MarlinKalTest

DDKalTest

Particle Flow Algorithms

highly granular calorimeter reconstruction

- all current detector concepts for LC are based on highly granular calorimeters
 - optimised for the Particle Flow Algorithm
- **PandoraPFA** is the de facto standard used by ILD, SiD and CLICdp
- alternative PFA algorithms exist and provide possibility to cross check
 - Arbor (CEPC), April (SDHCAL prototype)

Pandora Algorithms

slide: J.Marshall

DESY. Frank Gaede, LCWS 2021, 17.03.21

large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
 - generic API for fitting algorithms
 - large number of pattern recognition algorithms
- particle flow algorithms
 - PandoraPFA and Arbor, AprilPFA
- high level reconstruction
 - jet finding, flavor tagging, PID, TOF, ...

Tracking in iLCSoft

pattern recognition and Kalman-Filter

Particle Flow Algorithms

High Level Reconstruction

analysing the Particle Flow Objects

$\delta\lambda_{HHH}$ improves by 40% w/ perfect jet clustering

DESY. Frank Gaede, LCWS 2021, 17.03.21

- **High-Level reconstruction** algorithms are crucial to achieve the ultimate physics reach of detectors
- vertex finding and flavor tagging: **LCFIPlus**
- PID tools: dE/dx, TOF, shower shapes, ...
- Jet clustering: Durham, Valencia, ...

- very active field of development
 - already good set of tools available
 - further improvement in HLR tools often directly impacts the final physics performance

large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
 - generic API for f
 - large number of recognition algo
- particle flow algorithm
 - PandoraPFA and
- high level reconstruction
 - jet finding, flavor tagging, PID, TOF,...

• it is vital for ILD to preserve this battle proven code in Key4hep for some time

• -> need a smooth transition from LCIO/Marlin to EDM4hep/Gaudi AND new algorithms

• will use MarlinWrapper/LCIO2EDM4hepConverter for some time for ILD production

Tracking in iLCSoft
pattern recognition and Kalman-Filter

iLCSoft

Clupetra → LCIO DDDRec MarlinKalTest DDKalTest

iLCSoft

iLCSoft

$\frac{dE}{dx}$ (GeV/nm) vs Momentum (GeV)

iLCSoft

ion algorithms are crucial to physics reach of detectors

tagging: LCFIPlus

• Jet clustering: Durham, Valencia, ...

• very active field of development

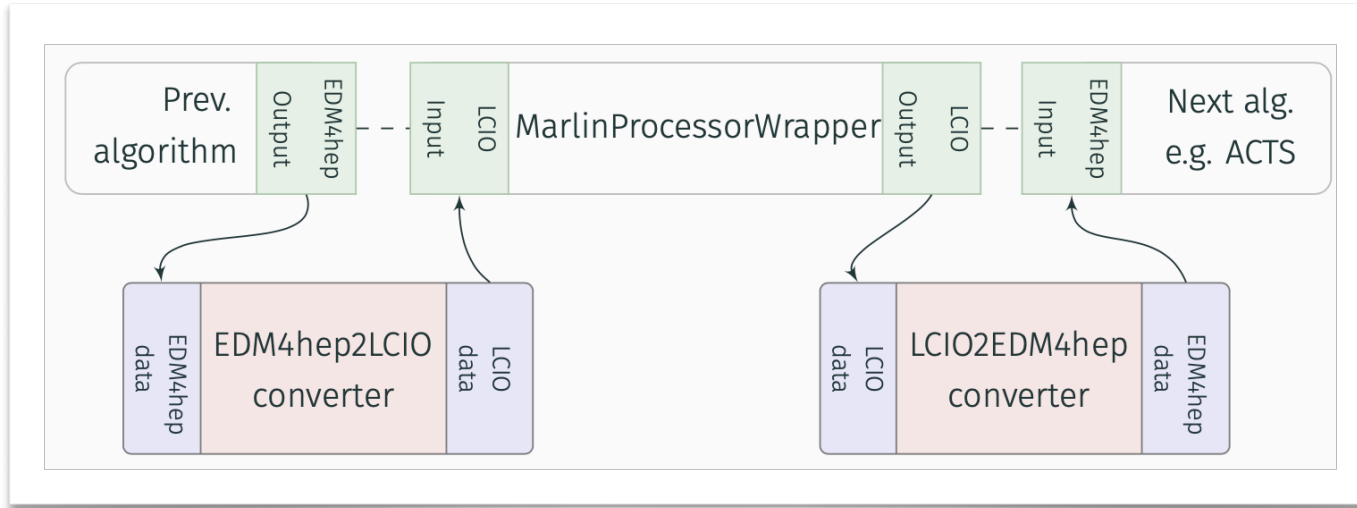
- already good set of tools available
- further improvement in HLR tools often directly impacts the final physics performance

$\delta\lambda_{HHH}$ improves by 40% w/ perfect jet clustering

DESY. Frank Gaede, LCWS 2021, 17.03.21

K4MarlinWrapppper

the vision: mix and match Marlin and Gaudi algorithms



- in a transition phase algorithms **developed in the new EDM4hep/Gaudi** world can gradually replace older algorithms
 - e.g. eventually one might want to replace track fitting with **ACTS** also for LC detectors
- some technicalities are address *under-the-hood* via **k4EDM4hep2LcioConv**



all existing (high level) reconstruction algorithms as Marlin processors fully available in Key4hep !

ILD Standard Reconstruction

now smoothly runs in in Key4hep

- translation of *MarlinReco.xml* to GAUDI python steering file *ILDReconstruction.py*
- plan to use this as new standard (soon ?)

ILD standard reconstruction in Key4hep

T.Madlener

- All configuration available from [iLCSoft/ILDConfig](https://github.com/iLCSoft/ILDConfig)
 - Everything that works in iLCSoft also works in Key4hep!
- ```
Marlin MarlinStdReco.xml --global.LCIOInputFiles=<input-file> [...]
```
- Now also with Gaudi
    - `k4run ILDReconstruction.py --inputFiles=<input-file> [...]`
      - Works with EDM4hep and LCIO inputs
      - EDM4hep output by default, LCIO output via `--lcioOutput=[true|only]`
  - Facilitates collaboration with other projects, e.g. CLD
  - Full migration of all workflows will take some time but process started
    - Some new developments already done exclusively in Gaudi configuration

The screenshot shows a web browser view of the GitHub repository `iLCSoft/ILDConfig`. The file browser on the left shows the directory structure, with `StandardConfig/production` expanded to show subdirectories like `BgOverlay`, `Calibration`, `CaloDigi`, `Config`, `Documentation`, `Examples/bbudsc_3evt`, `Gear`, `HighLevelReco`, `IsolatedLeptonTagging/weights`, `LCFIPlusConfig`, and `PandoraSettings`. The main content area shows the Python file `ILDReconstruction.py` with 383 lines. The code includes constants for detector technology, logic to identify the detector model from command-line arguments, and a sequence loader that dynamically loads configuration files based on the detector model and tracking requirements.

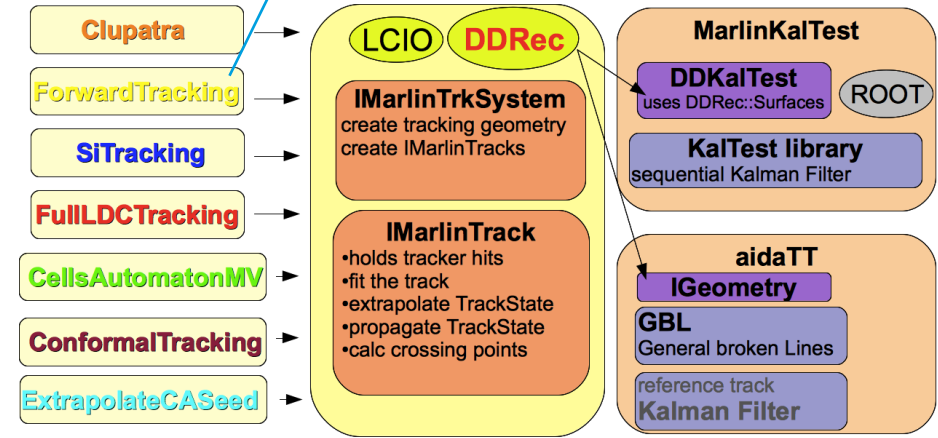
already started to develop new algorithms (or steering variants) in GAUDI only

# Reconstruction for ILD@FCCee

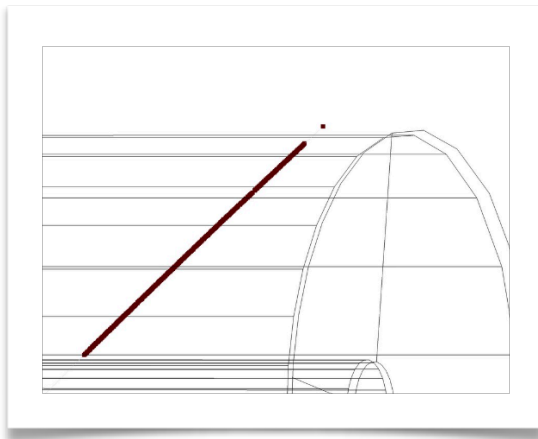
combining simulation models and tracking algorithms

- ILD models for FCCee w/ TPC, FCC-MDI (mask) and CLD inner Si-Tracking
- can run *ConformalTracking* for Si-tracking and *Clupatra* for TPC tracking
- done using **MarlinWrapper** and **ILDReconstruction.py**
  - combining tracks is work in progress...
  - => maybe write this in GAUDI ?

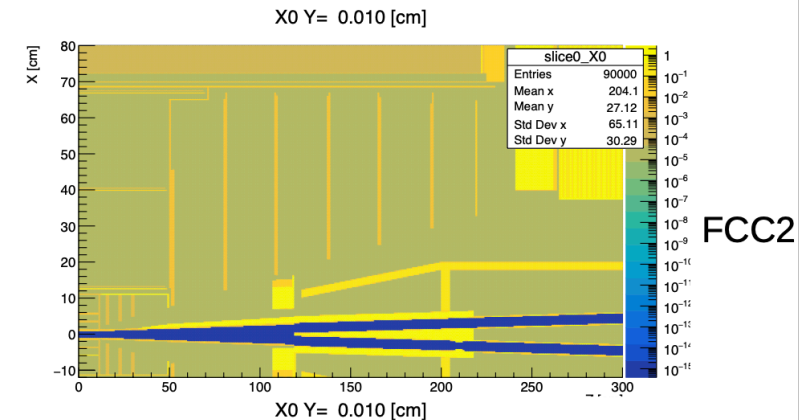
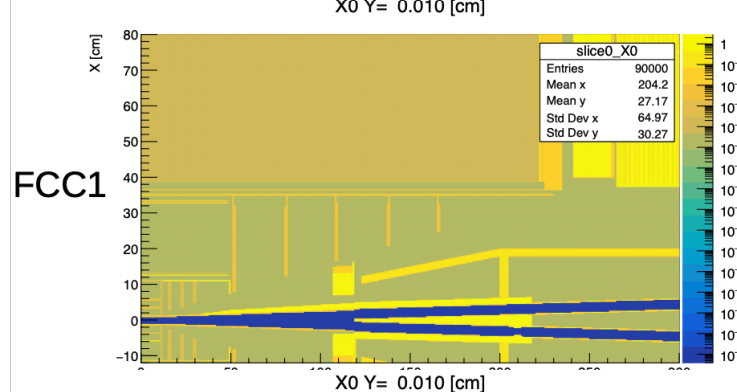
Forward Tracking uses a Hopfield Network ;-)



V.Schwan, D.Jeans



## Overview of FCCee Models



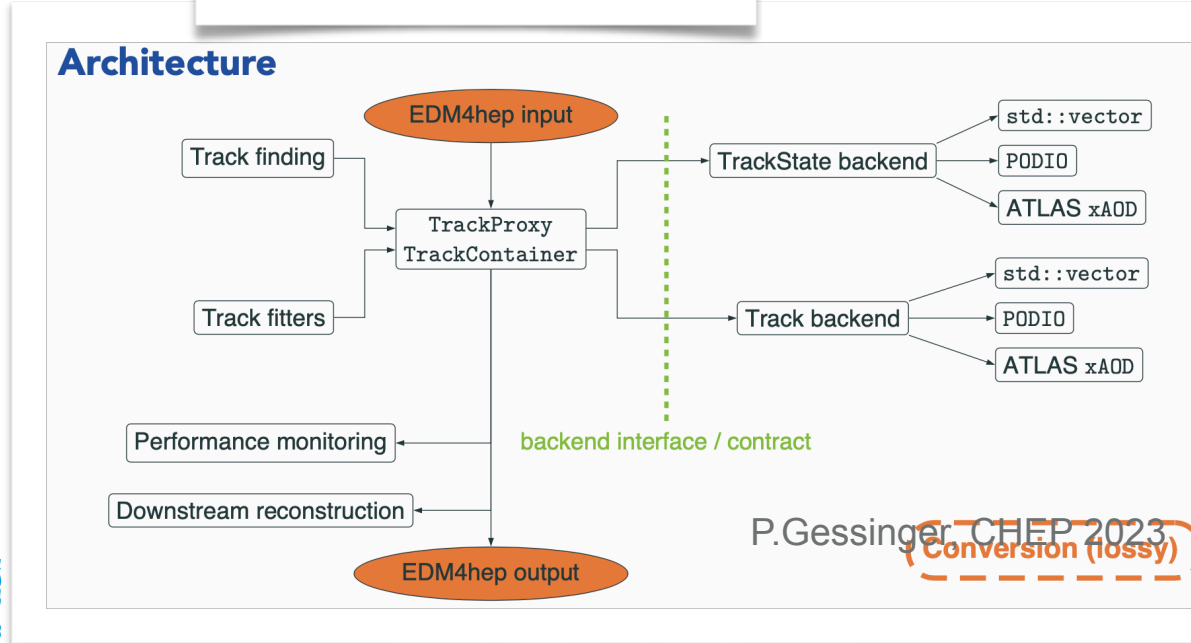
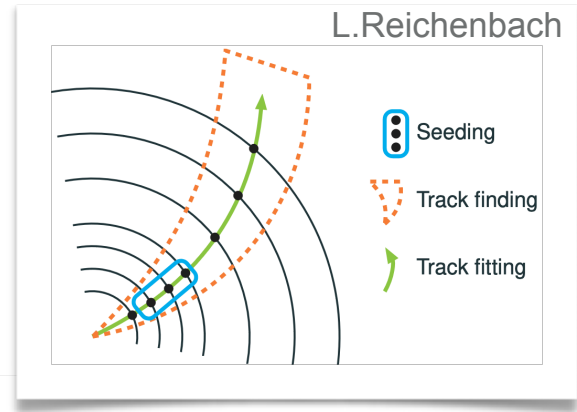


# ACTS - A Common Tracking Toolkit

used in Key4hep

- **ACTS** tracking toolkit is the the current choice for track fitting (and finding) in Key4hep
- new package **k4ACTSTracking** provides ACTS fitter
  - implemented as GAUDI algorithm
  - uses DD4hep geometry
- first implementation of TrueTrackfinder for CERN OpenDetector
- successfully used for MuonCollider w/ handcrafted tracking geometry

- ongoing work on **automatic** and **transparent** construction of **tracking geometry** for all detectors in k4geo (Key4hep)

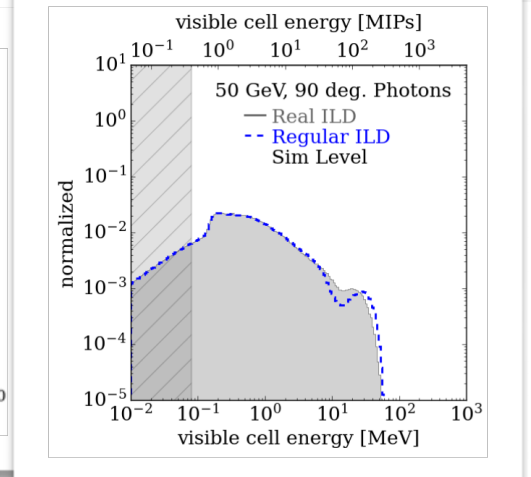
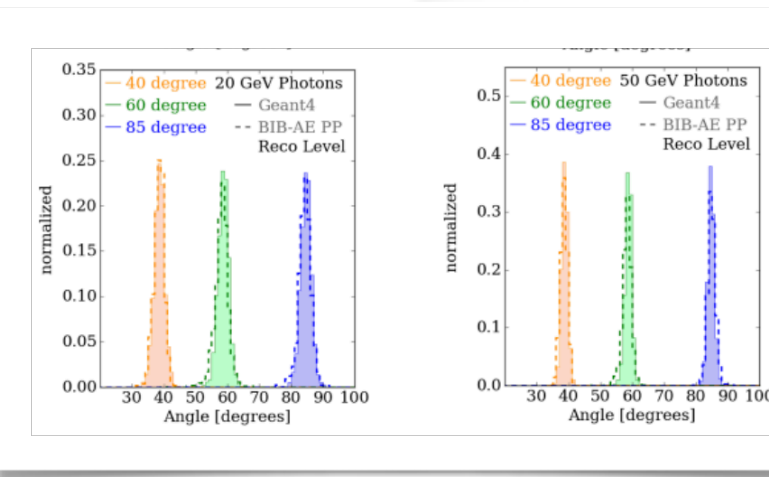
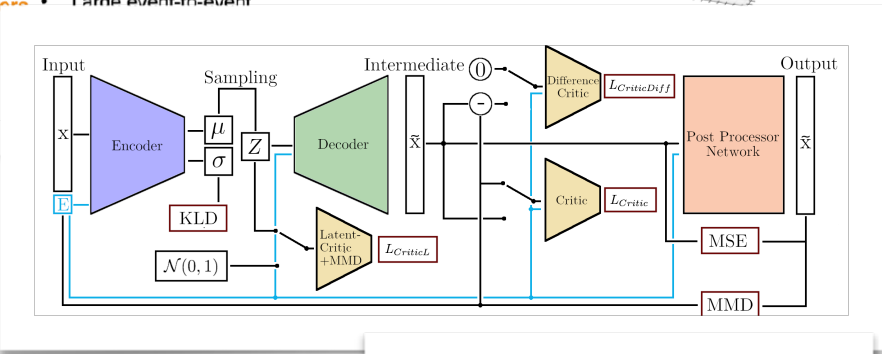
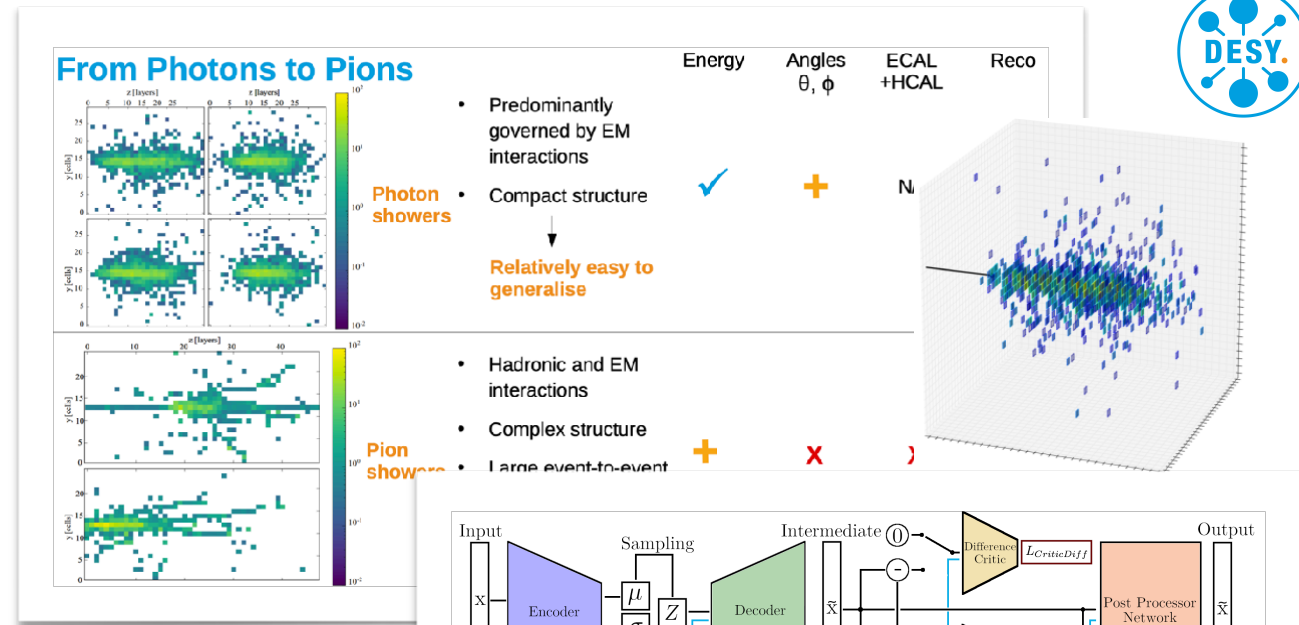


- might eventually benefit from this implementation
- expect significant effort to adapt to ILD tracking (w. TPC - cylinders) and all pattern recognition
- probably more a **midterm** project ...

# AI/ML in Key4hep

## generative ML for fast shower simulation

- shower generation with ML offers great potential  $O(10e3)$  for faster and more sustainable simulation in HEP
- DESY/UHH QU group studying many generative models: (W)GANs, VAE, BIB-AEs, Normalising Flows, Point Cloud Diffusion Models,...
- achieve high fidelity on individual physics distributions



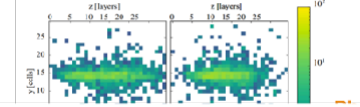
# AI/ML in Key4hep

## generative ML for fast shower simulation

- shower generation with ML offers great potential  $O(10e3)$  for faster and more sustainable simulation in HEP
- DESY/UHH QU group studying many generative models: (W)GANs, VAE, BIB-AEs, Normalising Flows, Point Cloud Diffusion Models,...
- achieve high fidelity on individual physics distributions
- developed *DDFastShowerML* library in Key4hep to transparently call ML inference during full simulation with DDG4/Geant4

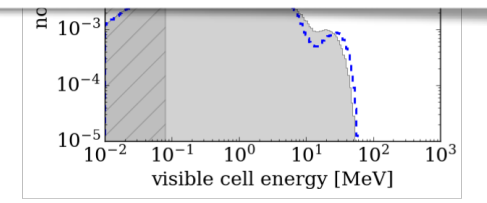
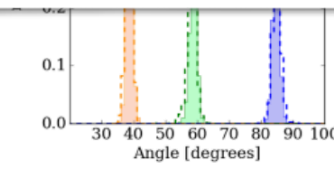
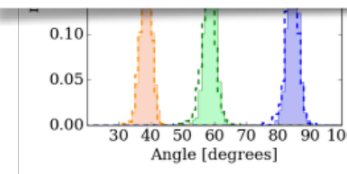
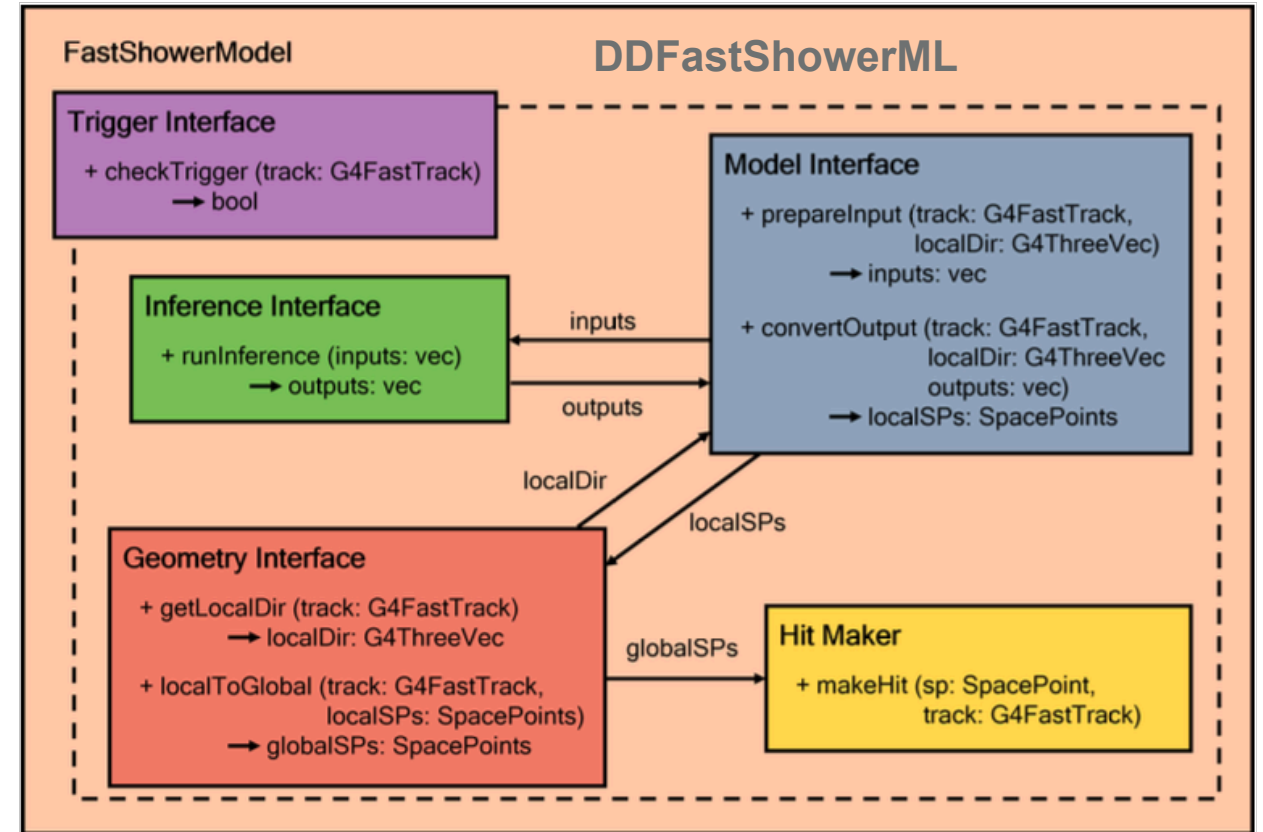


### From Photons to Pions



Energy Angles  $\theta, \phi$  ECAL +HCAL Reco

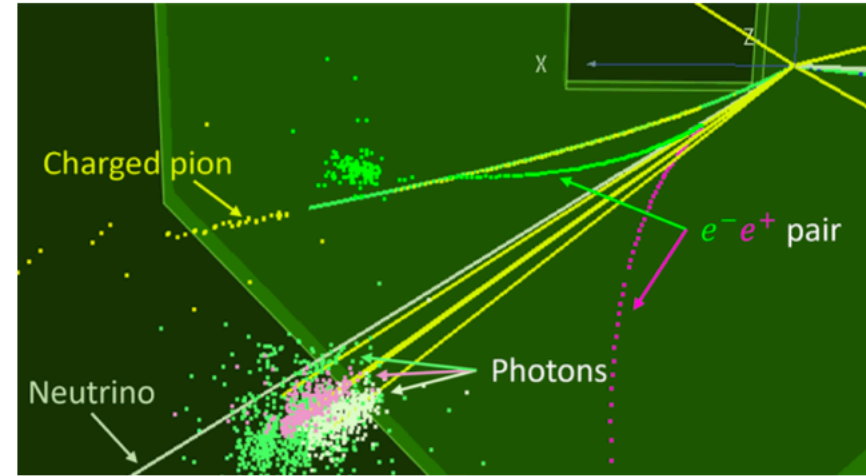
- Predominantly governed by EM interactions



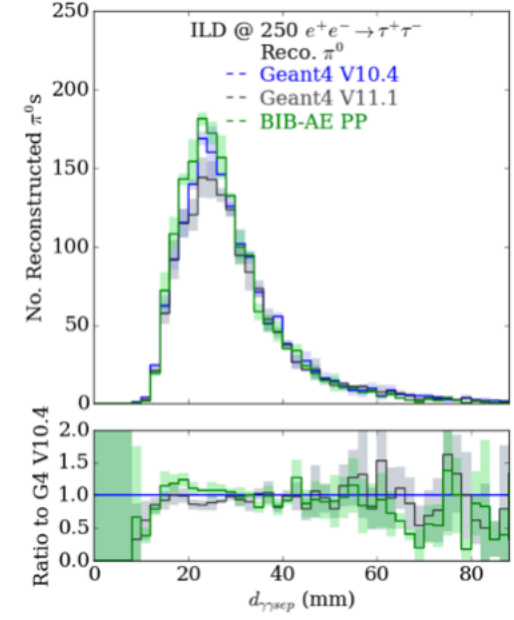
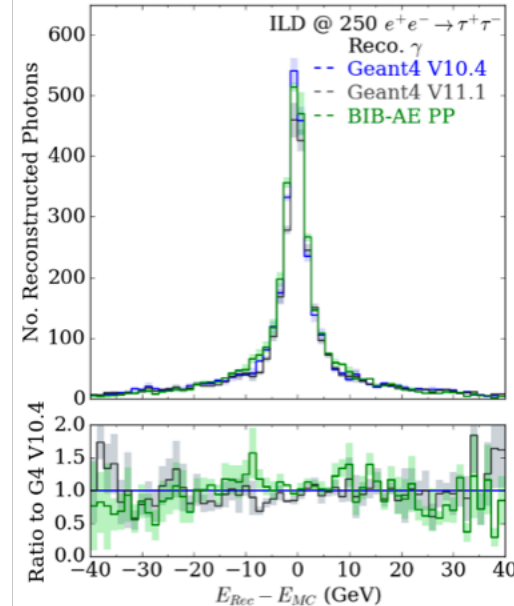
# AI/ML in Key4hep

## generative ML for fast shower simulation

- shower generation with ML offers great potential  $O(10e3)$  for faster and more sustainable simulation in HEP
- DESY/UHH QU group studying many generative models: (W)GANs, VAE, BIB-AEs, Normalising Flows, Point Cloud Diffusion Models,...
- achieve high fidelity on individual physics distributions
- developed *DDFastShowerML* library in Key4hep to transparently call ML inference during full simulation with DDG4/Geant4
- allows for **realistic benchmarking** of ML fastsim on **real physics observables** after full **reconstruction**



P.McKeown: PhD thesis 2024





# AI/ML in Key4hep

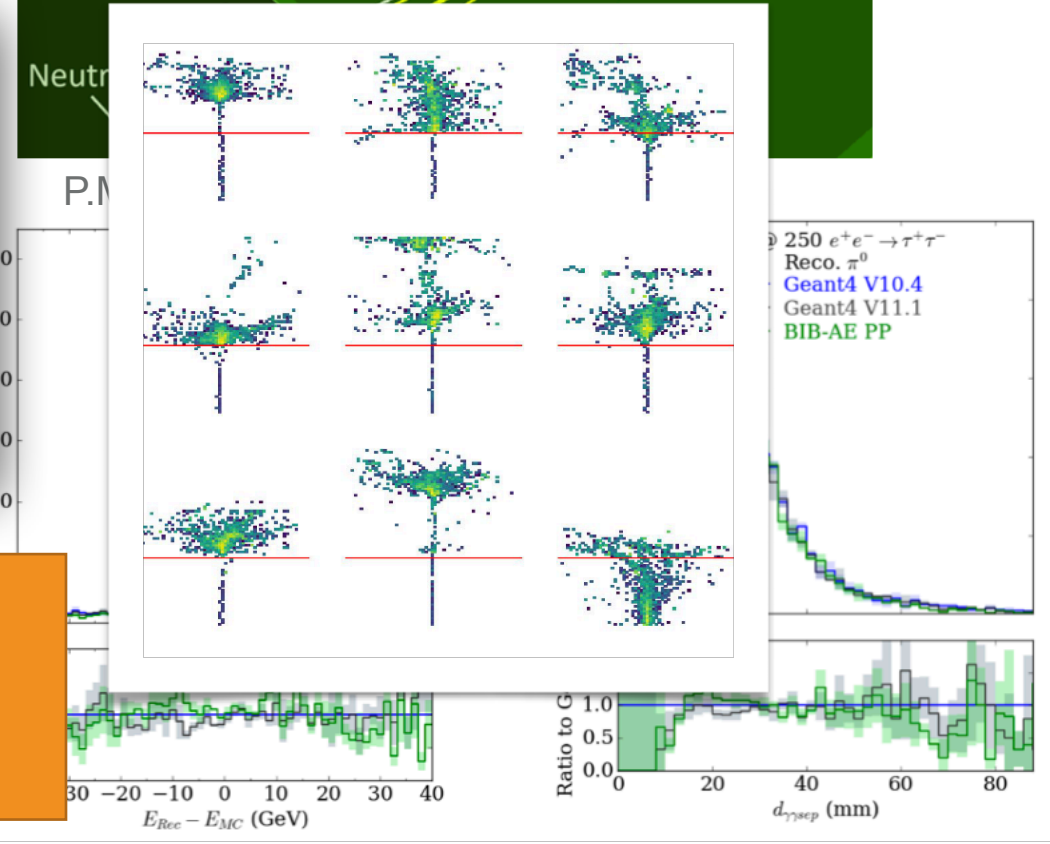
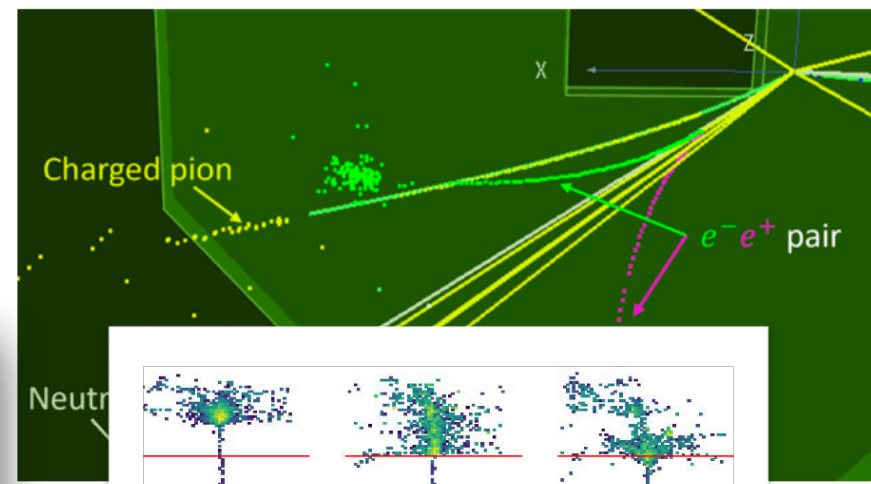
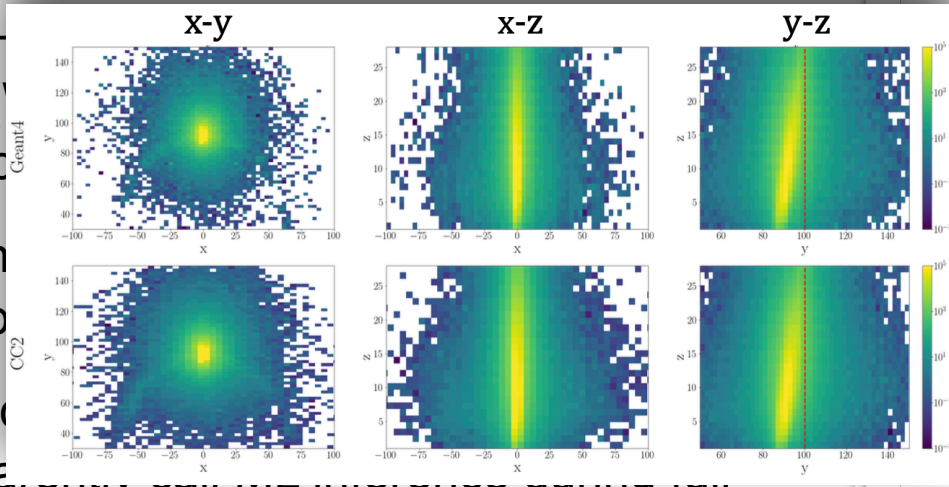
## generative ML for fast shower simulation

- shower generation with ML offers great potential  $O(10e3)$  for faster and more sustainable simulation in HEP

- DESY/UH
- models: (
- Flows, Po

- achieve h
- distributio

- developed
- to transpar



ongoing work:  
 • apply CaloClouds diffusion model to **CMS HGCal** and use for **hadronic showers** in **ILD**

# AI/ML in Key4hep

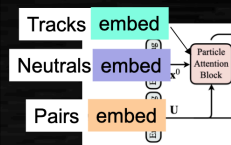
## Flavour tagging with deep learning methods

- implemented ParticleTransformer flavour tagging for ILD
  - achieve **dramatically better** results than LCFIPlus
- observe strange improvement w/ more training data at FCC  
-> to be studied !
- framework (Marlin/Gaudi?) inference work in progress
- started to look into **strange tagging**

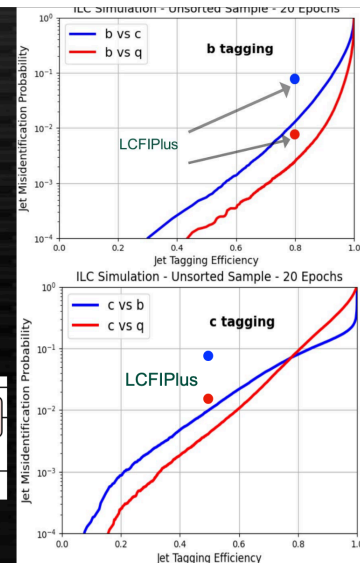
### Improvements wrt. LCFIPlus

- Factor (3-9) improvement at ParT from LCFIPlus without any tuning
- Another factor (max 3) improvement by tuning
  - Optimizing input variables
  - Separate embedding for tracks/neutrals

| background        | b-tag 80% eff. |          | c-tag 50% eff. |          |
|-------------------|----------------|----------|----------------|----------|
|                   | c jets         | uds jets | b jets         | uds jets |
| +LCFIPlus (BDT)   | 6.3%           | 0.79%    | 7.4%           | 1.2%     |
| *ParT (initial)   | 1.3%           | 0.25%    | 1.0%           | 0.43%    |
| **ParT (improved) | 0.48%          | 0.14%    | 0.86%          | 0.34%    |

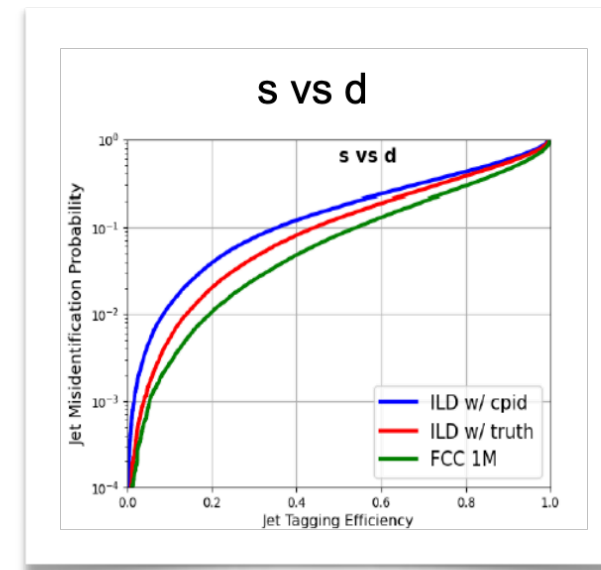
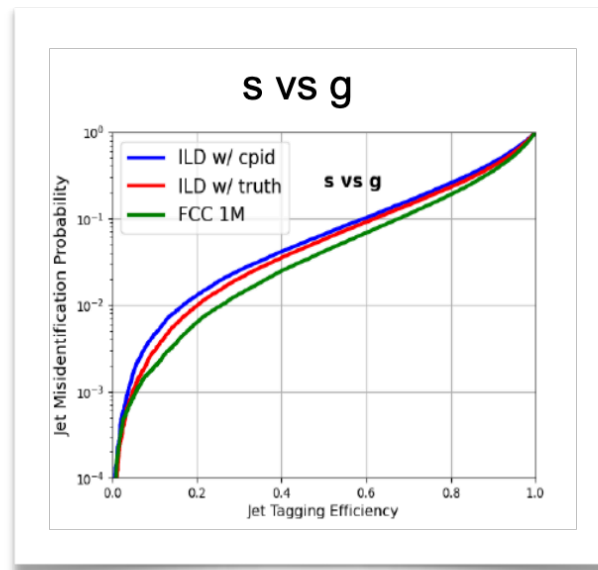


+LCFIPlus (BDT) 250 GeV nnqq  
 \*ParT (initial) 91 GeV qq, default settings  
 \*\*ParT (improved) 250 GeV nnqq, b/c/d separation  
 T.Suehara et al., ILD Software and Analysis meeting, 2 Oct. 2024, page 7



| Sample / sample size        | b-tag 80% eff. |          | c-tag 50% eff. |          |
|-----------------------------|----------------|----------|----------------|----------|
|                             | c jets         | uds jets | b jets         | uds jets |
| ILD full-sim 1M (optimized) | 0.48%          | 0.14%    | 0.86%          | 0.34%    |
| FCCee Delphes 1M (reduced)  | 0.47%          | 0.12%    | 0.64%          | 0.10%    |
| FCCee Delphes 1M (full)     | 0.21%          | 0.054%   | 0.36%          | 0.059%   |
| FCCee Delphes 4M            | 0.045%         | 0.025%   | 0.20%          | 0.033%   |
| FCCee Delphes 6M            | 0.014%         | 0.010%   | 0.13%          | 0.022%   |
| FCCee Delphes 8M            | 0.007%         | 0.006%   | 0.076%         | 0.021%   |

**We see mild consistency between ILD and FCC!**



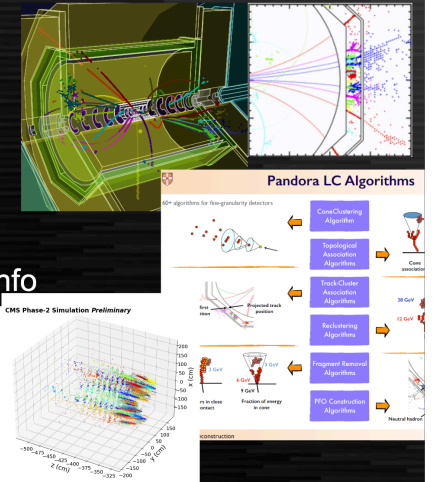
# AI/ML in Key4hep

## PFA with ML

- started to develop deep neural networks for particle flow - partly based on CMS HGCal
- using GravNet and Object Condensation
- after early, promising results **made significant progress**

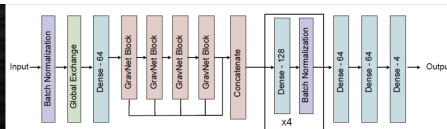
## Particle flow with DNN: introduction

- Separation of cluster at calorimeter
  - Charged or neutral cluster
- Essential for jet energy resolution
- Current algorithm: PandoraPFA
  - Combination of various process
  - Not easy to optimize or adding more info
- CMS HGCal clustering
  - Similar to ILD calo
  - Good for starting point



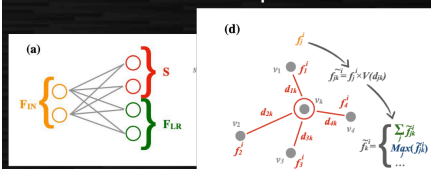
## PFA: clustering algorithm

- Input: position/energy/timing of each hit
- Output: virtual coordinate and  $\beta$  for each hit



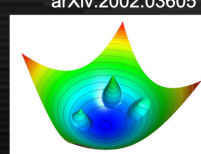
### GravNet [arXiv:1902.07987](https://arxiv.org/abs/1902.07987)

- The virtual coordinate (S) is derived from input variables with simple MLP
- Convolution using "distance" at S (bigger convolution with nearer hits)
- Concatenate the output with MLP

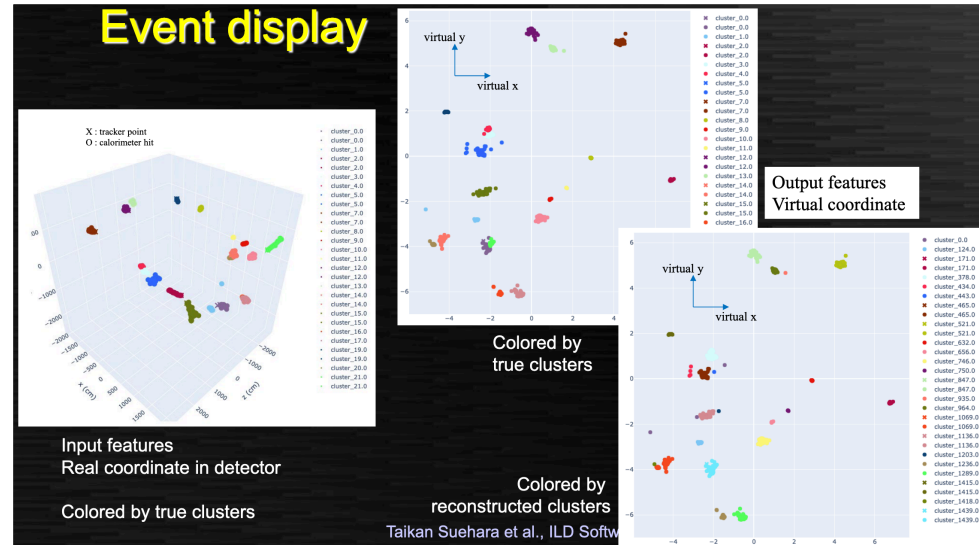


### Object Condensation (loss function) [arXiv:2002.03605](https://arxiv.org/abs/2002.03605)

- Condensation point: The hit with largest  $\beta$  at each (MC) cluster
- $L_V$ : Attractive potential to the condensation point of the same cluster and repulsive potential to the condensation point of different clusters
- $L_\beta$ : Pulling up  $\beta$  of the condensation point
- $L_p$ : Regression to output features



## Event display



## PFA with ML

- started to develop flow - partly based on
- using GravNet
- after early, promising progress

## Particle flow with DNN: introduction

### Results on efficiency and purity

| Algorithm train/test           | Electron eff. | Pion eff. | Photon eff. | Electron pur. | Pion pur. | Photon pur. |
|--------------------------------|---------------|-----------|-------------|---------------|-----------|-------------|
| GravNet 10 taus/10 taus        | 99.1%         | 96.5%     | 99.0%       | 91.8%         | 98.9%     | 97.1%       |
| PandoraPFA 10 taus             | 99.3%         | 94.0%     | 99.1%       | 91.8%         | 94.6%     | 97.2%       |
| GravNet jets/jets              | 94.5%         | 93.1%     | 95.2%       | 94.6%         | 93.2%     | 92.4%       |
| PandoraPFA jets                | 80.2%         | 90.4%     | 79.0%       | 75.0%         | 90.6%     | 77.7%       |
| PandoraPFA jets (ILCSof truth) | 96.7%         | 95.5%     | 96.4%       | 97.1%         | 90.4%     | 97.7%       |

At least in our measure, performance of GravNet-based algorithm **exceeds PandoraPFA**  
 → **Promising as full PFA (but energy regression to be done)**  
 Definition of MC truth clusters needs to be tuned (see ILCSof truth)

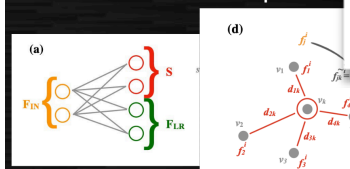
Taikana Suehara et al., ILD Software and Analysis meeting, 2 Oct. 2024, page 21

### PFA: clustering and

- Input: position/energy/timing
- Output: virtual coordinate and

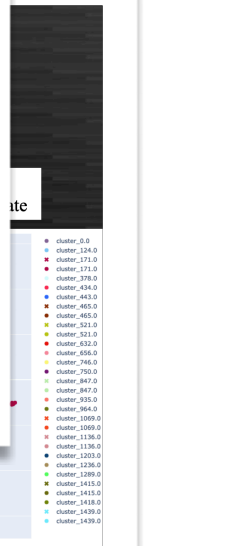
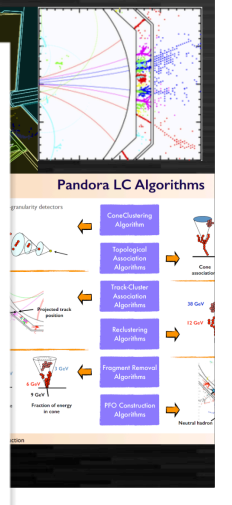
GravNet arXiv:1902.07987

- The virtual coordinate (S) is derived from input variables with simple
- Convolution using "distance" (bigger convolution with nearest neighbors)
- Concatenate the output with



- point of different clusters
- $L_{\beta}$ : Pulling up  $\beta$  of the condensation point
- $L_p$ : Regression to output features

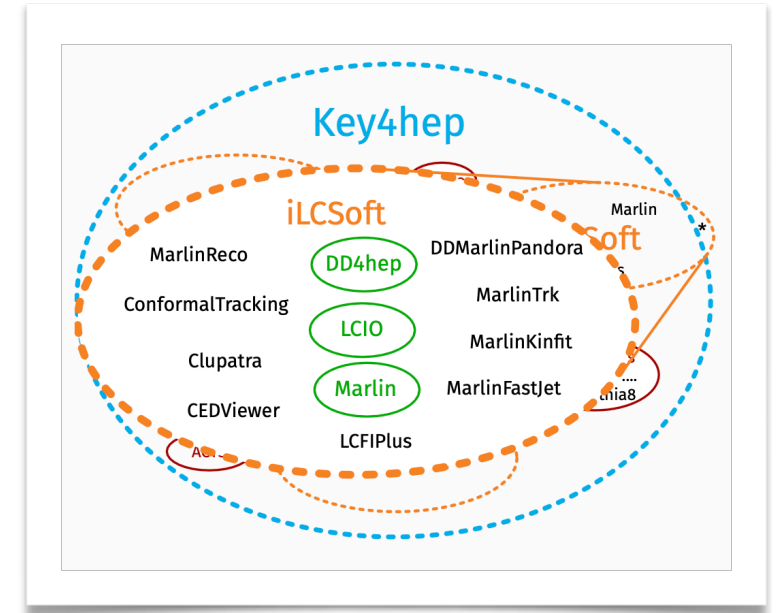
input features  
 Real coordinate in detector  
 Colored by true clusters  
 Colored by reconstructed clusters  
 Taikana Suehara et al., ILD Software





# Summary and Outlook

- **Key4hep** started as a new future collider community wide effort in 2020 to put together a modern turnkey software stack
  - contributors: CEPC, CLIC, FCC, EIC, ILC, LUXE, Muon Collider ...
- **iLCSoft and ILD software integral part of Key4hep from the start**
- **battle proven ILD standard reconstruction** can be run in Key4hep w/ **MarlinWrapper** as before - or w/ EDM4hep output
- many **AI/ML activities ongoing** (fast sim and (high level) reconstruction)
  - eventually need to validate and integrate in ILD standard reconstruction

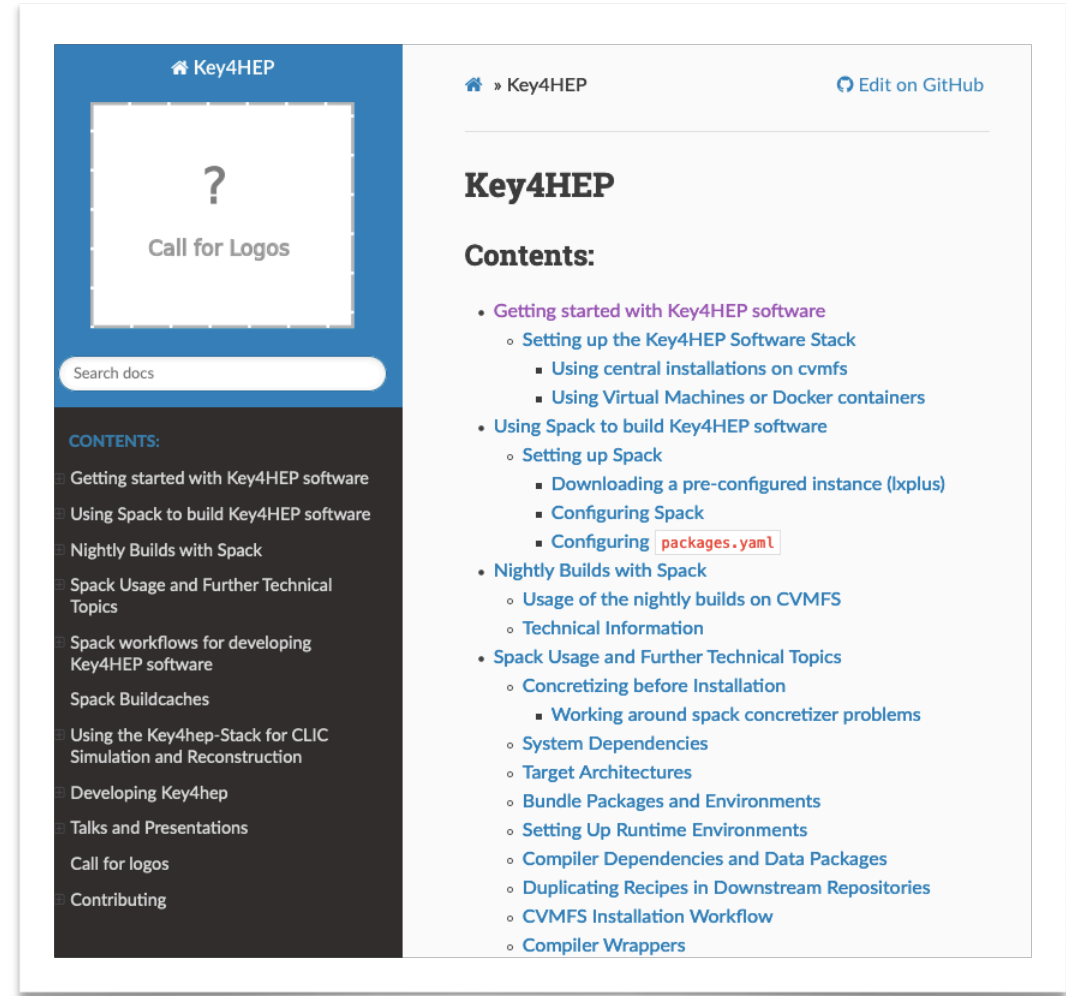


- Key4hep offers great opportunity to **modernise ILD software stack AND collaborate w/ other Higgs factories** - when studying ILD for FCC
- yet, limiting factor for all software developments: **manpower**
- need to somewhat balance the work on core tools and novel (AI) algorithms w/ ILD detector optimisation for FCC ( and ILC)

# pointers to documentation

## entry points to Key4hep

- Key4hep GitHub Project
  - <https://github.com/key4hep>
- Key4hep main documentation page
  - <https://key4hep.github.io/key4hep-doc/>
- Doxygen available., e.g. for EDM4hep
  - <https://edm4hep.web.cern.ch/>
- iLCSoft Github Project
  - <https://github.com/ilcsoft>



The screenshot shows the Key4HEP documentation website. The top navigation bar includes a home icon, the text 'Key4HEP', and a link to 'Edit on GitHub'. The main content area features a large white box with a question mark and the text 'Call for Logos'. Below this is a search bar labeled 'Search docs'. A dark sidebar on the left contains a 'CONTENTS:' section with a list of topics. The main content area on the right has a 'Key4HEP' heading and a 'Contents:' section with a detailed list of topics and sub-topics.

**Key4HEP**

**Contents:**

- Getting started with Key4HEP software
  - Setting up the Key4HEP Software Stack
    - Using central installations on cvmfs
    - Using Virtual Machines or Docker containers
- Using Spack to build Key4HEP software
  - Setting up Spack
    - Downloading a pre-configured instance (lxdplus)
    - Configuring Spack
    - Configuring `packages.yaml`
- Nightly Builds with Spack
  - Usage of the nightly builds on CVMFS
  - Technical Information
- Spack Usage and Further Technical Topics
  - Concretizing before Installation
    - Working around spack concretizer problems
  - System Dependencies
  - Target Architectures
  - Bundle Packages and Environments
  - Setting Up Runtime Environments
  - Compiler Dependencies and Data Packages
  - Duplicating Recipes in Downstream Repositories
  - CVMFS Installation Workflow
  - Compiler Wrappers