Progress in CEPCSW Core Software

Xingtao Huang, Teng Li, Weidong Li, Tao Lin, Jiaheng Zou
representing CEPC software team

2024 CEPC Workshop, Hangzhou, Zhejiang
22-27 October 2024

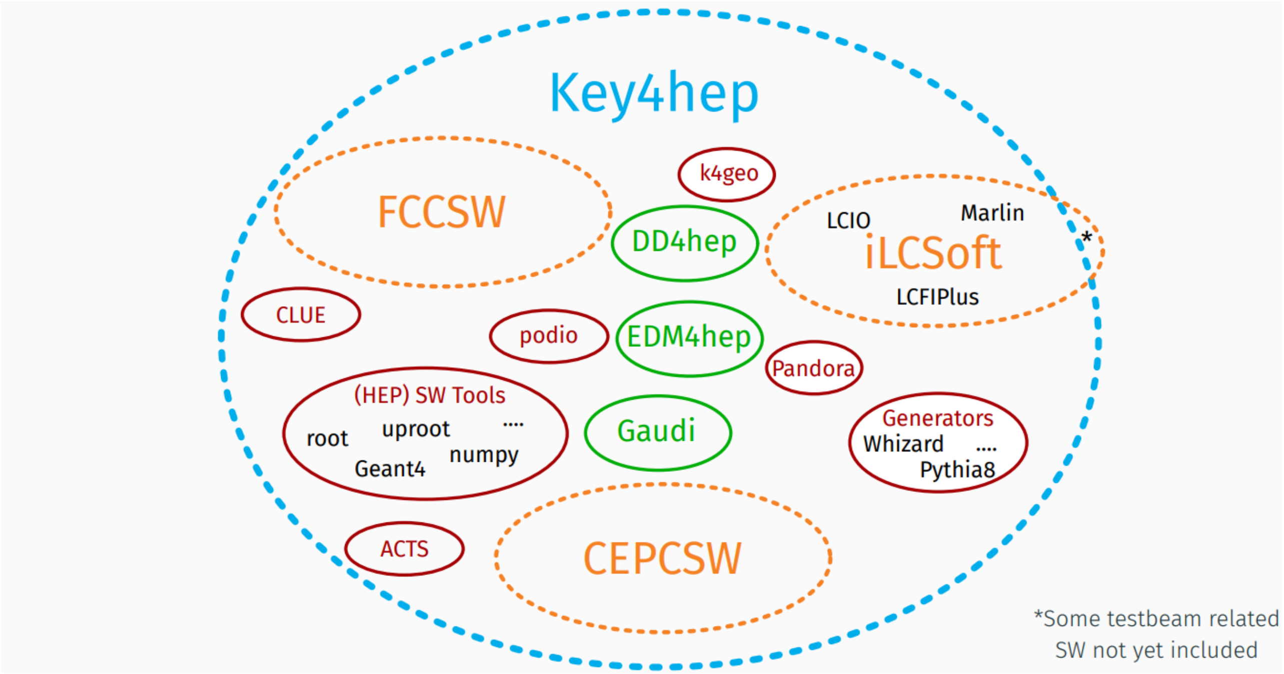
Outline

- Introduction
- Progresses in CEPCSW
  - Software releases towards RefTDR
  - Beam background simulation
  - Gaussino based simulation framework
  - RDataFrame based physics analysis tool
- R&D activities and future developments
  - ML-based fast calorimeter simulation
  - Application of TRACCC to seeding algorithm
  - Consideration on migration to latest Key4hep
- Summary

Introduction

- The development of CEPC software started with the iLCSoft
  - Developed CEPC components for simulation and reconstruction
  - Generated M.C. data for detector design and physics potential studies
  - Particularly, CEPC CDR studies done with the iLCSoft
- The consensus among CEPC, CLIC, FCC, ILC and other future experiments was reached at the Bologna workshop in June, 2019.
  - Develop a Common Turnkey Software Stack (Key4hep) for future collider experiments
  - Maximize the sharing of software components among different experiments

T.Madlener | Key4hep & EDM4hep
CEPC workshop, Edinburgh

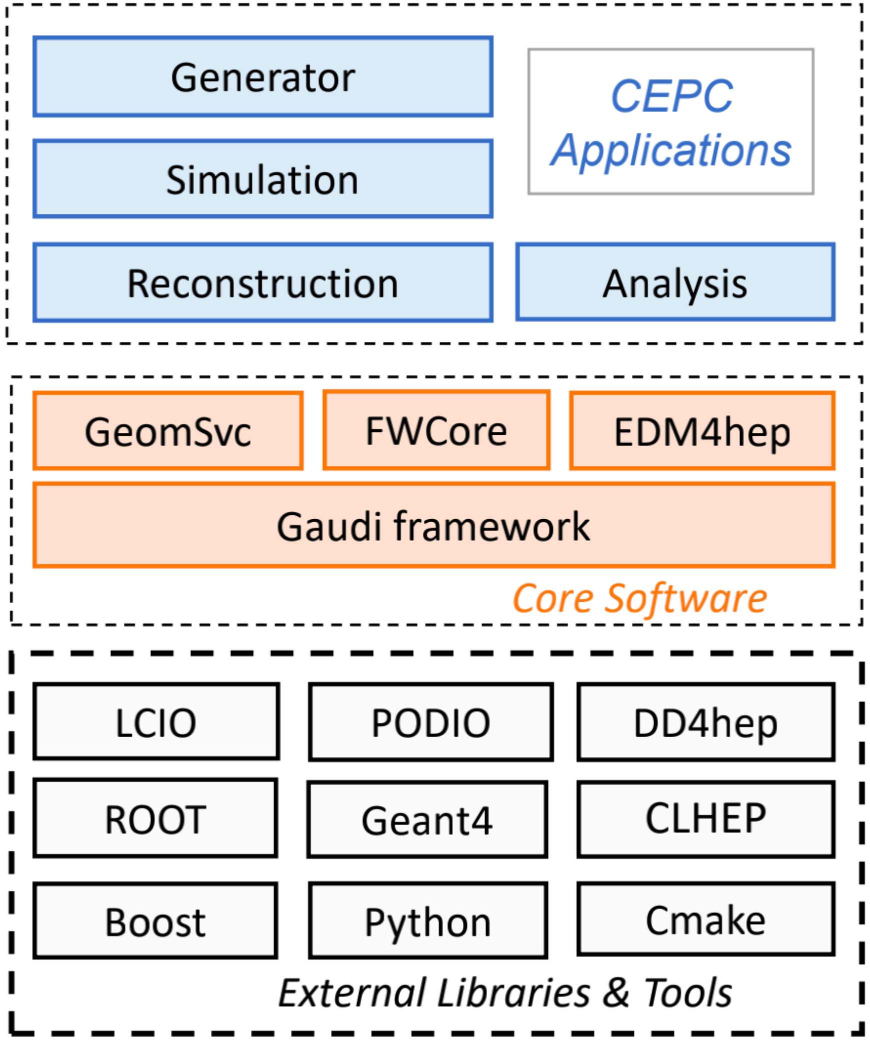


Key4hep project: https://github.com/key4hep
CEPCSW is the first application based on Key4hep.

Architecture of CEPCSW

- CEPCSW is organized as a multi-layer structure
  - Applications: simulation, reconstruction and analysis
  - Core software
  - External libraries

- The key components of core software include:
  - Gaudi: defines interfaces to all software components and controls their execution
  - EDM4hep: generic event data model
  - K4FWCore: manages the event data
  - DD4hep: geometry description
  - CEPC-specific framework software: generator, Geant4 simulation, beam background mixing, fast simulation, machine learning interface, etc.



https://code.ihep.ac.cn/cepc/CEPCSW

Software releases towards RefTDR (1)

- Motivation
  - Support the fast iterations of the reference detector design.
  - Release the latest versions of detectors to support physics and performance studies.
- Software development
  - Freeze the versions of external libraries (LCG 103 at centos7)
  - New version scheme: tdr*YY.MM*

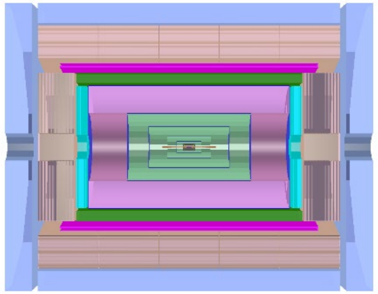
Mechanical

Vertex

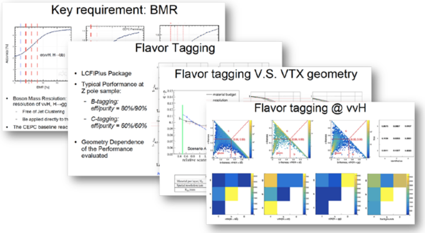
Tracker

Calorimetry

Muon



CEPCSW



Physics & Performance studies

feedbacks

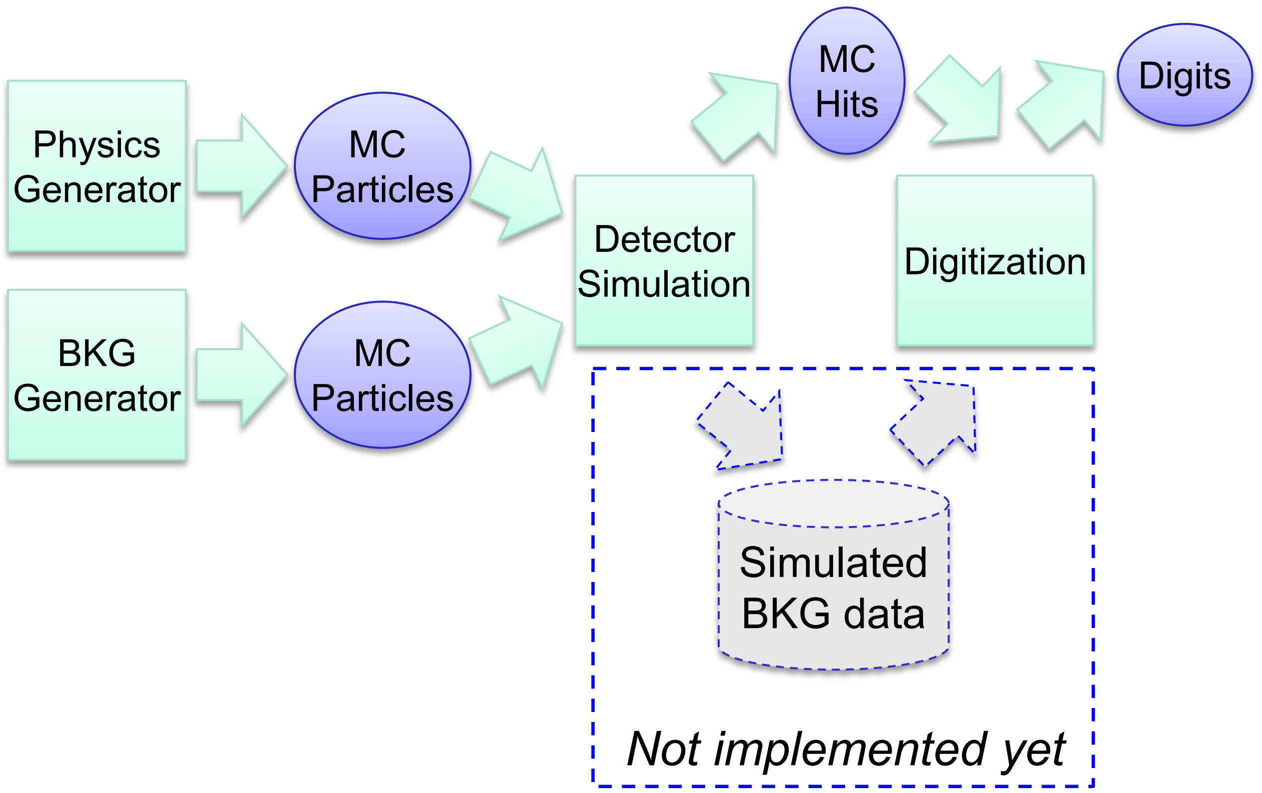
Release	Timeline	Features
tdr24.3 √	March	Core software
tdr24.4 √	April	Tracking and Background mixing
tdr24.5 √	May	PID and muon
tdr24.9 √	Sept	Calorimeters and speed optimization
tdr24.10	Oct	For physics performance

Software releases towards RefTDR (2)

Sub Detector	Options	Detector Description/Simulation	Digitization	Reconstruction
MDI+LumiCal		Implemented	None	None
VTX	RefTDR	Implemented	Smearing	
VTX	Backup	Cooling, electronics, part of support structure	Smearing	Clusters are formed and then converted into space points. Track finding starts from the most outer layers in the ITK and searches for space points of a track from outside to inside.
ITK		Equivalent material for sensitive detector and support structure		
FTK				
OTK				After adding the OTK hits, track fitting will be executed to produce track parameters.
OTK_PID		Generation of TOF through a parametric model	None	None
TPC	RefTDR	Implemented	Model based Garfield simulation	Searching for tracks in TPC first and then performing a combined fit to all the hits from both TPC and silicon trackers
TPC_PID		Generation of dEdx(or dN/dx) through a parametric model	None	None
ECAL-Barrel	RefTDR	Materials and geometry from the preliminary design	Model based on test beam data	New PFA algorithm
ECAL-Endcap			None	Being validated
HCAL-Barrel			Model based on test beam data	Being developed
HCAL-Endcap			None	
MUON-Barrel		Added materials and geometry	Model based lab measurement	The reconstructed tracks are extrapolated to the Muon Detector and matched with the muon track according to the truth information.
MUON-Endcap			None	

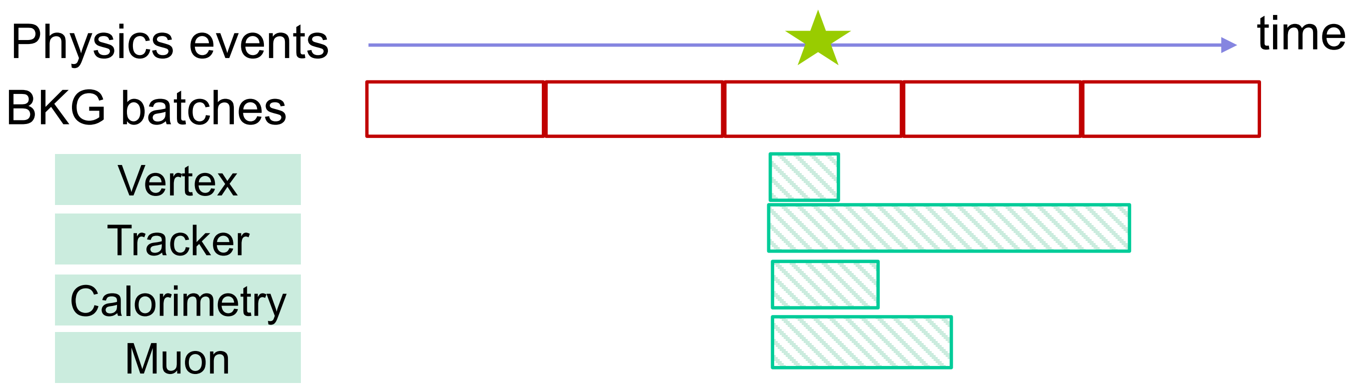
Beam induced background simulation (1)

- There are several different ways to mix the backgrounds.
  - ✓ Primary particle level (mixing before detector simulation)
  - Hit level (mixing after detector simulation)
  - Digit level (mixing after digitization)
- Need balances in CPU, memory and I/O.
- Currently, the MDI group uses the first way.
- For the physics performance studies, it is really time consuming.



Beam induced background simulation (2)

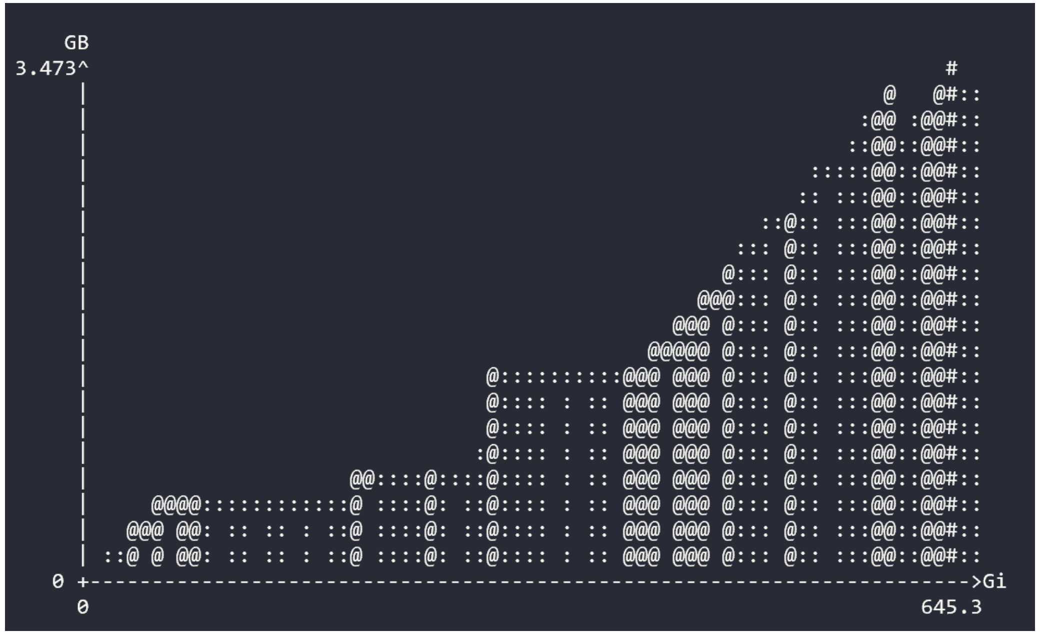
- Plan: support hit level mixing for physics studies, with BKG produced beforehand.
- Challenge: readout windows of sub-detectors are different.
- Possible solution:
  - First, simulating a batch of background events in fixed time-window. Backgrounds in multiple bunch crossings are included in one batch.
  - When simulating a physics event, pick the enough batches.
  - Finally, use the hits within corresponding readout time windows for different sub-detectors.



The readout time windows could be configurable.

Gaussino based simulation framework

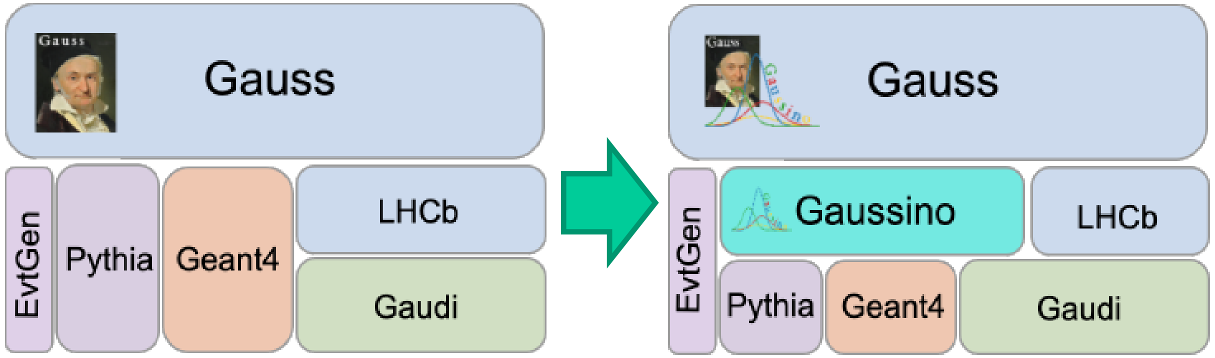
- Multi-threading simulation mandatory
  - Reduce memory footprint



Simulation setup:

- Software version: tdr24.9.1
- Detector: TDR_o1_v01
- Generation: single muons
- N events: 5

- Use Gaussino simulation framework from LHCb
  - Planned by Key4hep. [arXiv:2312.08152]

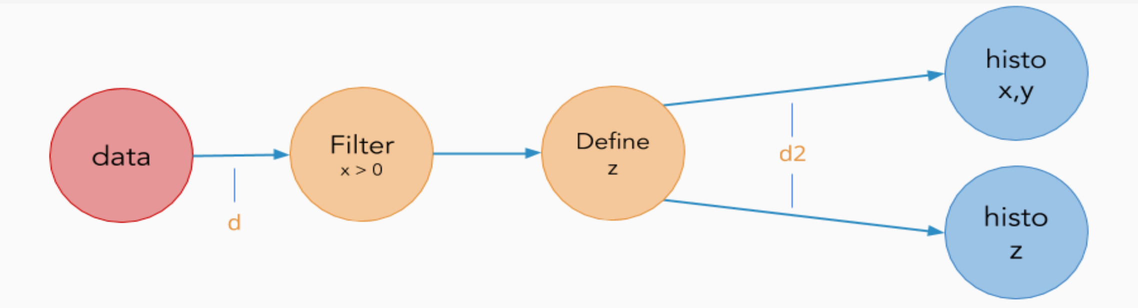


- Implement CEPC-on-Gaussino prototype
  - As a demo, VTX simulation was implemented.

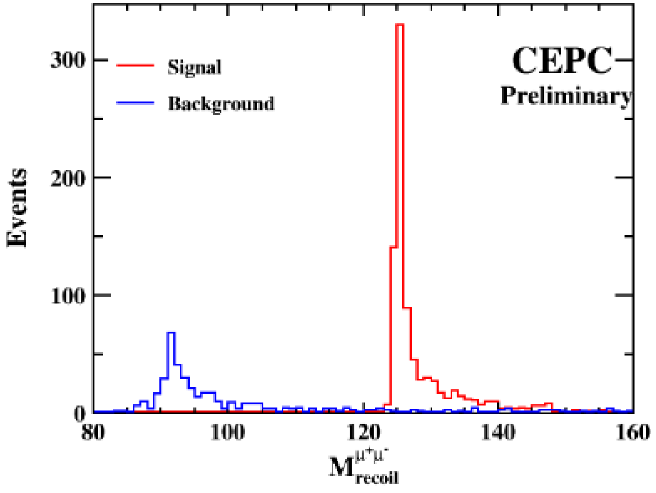
```
montool = giga.addTool(GenericTrackerMonTool, name="GenericTrackerM
montool.OutputLevel = DEBUG
montool.CollectionName = "VXDCollection"

giga.MonitorTools = ["GenericTrackerMonTool"]

GaussinoGeometry().GeometryService = "DD4hepCnvSvc"

DD4hepCnvSvc().DescriptionLocation = "CEPCSW/Detector/DetCRD/compac
DD4hepCnvSvc().SensDetMappings = {"VXD": "VXD"}
# DD4hepCnvSvc().OutputLevel = DEBUG
```

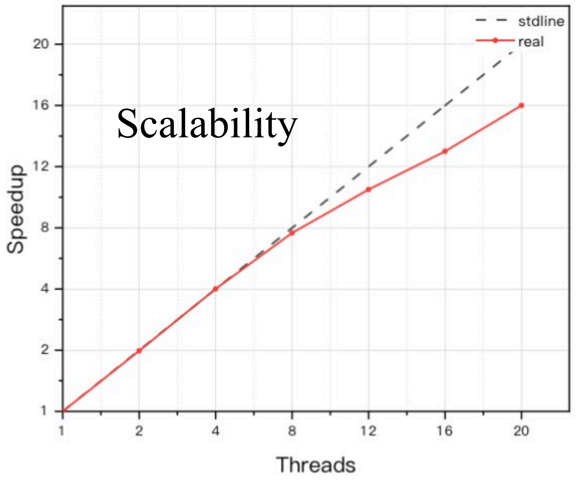
RDataFrame based physics analysis tool

- RDataFrame is a powerful tool for parallel data analysis
  - Programming language: Python and C++
  - Declarative programming and parallel processing
  - Used by many experiments such as FCC-ee
- New developments since last meeting
  - Developed common data input interfaces to support both LCIO data and EDM4hep data
  - Several algorithms were ported from Marlin
    - JetClustering , KinematicFit
  - More are being implemented
    - VertexFit, JetTagging, PID etc.
  - Performance test with two analysis channels
    - e+e- -> Z(mumu)H
    - e+e- ->H(2jet) mumu



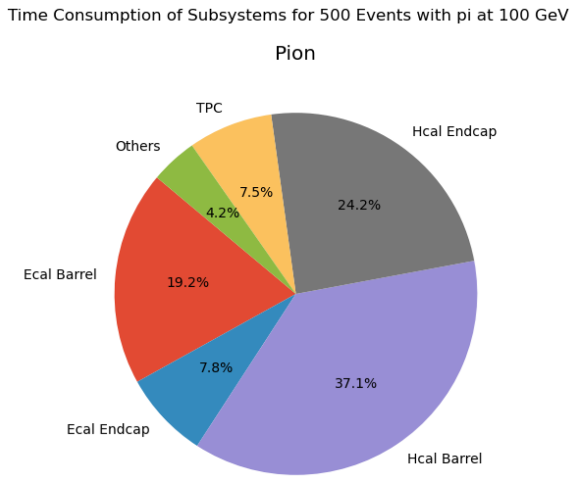

```
// d2 is a new data-frame, a transformed version of d
auto d2 = d.Filter("x > 0")
           .Define("z","x*x + y*y");
// make histograms out of it
auto hz = d2.Histo1D("z");
auto hxy = d2.Histo2D({"hxy","hxy",16,-1,1,64,-1,1},"x","y");
```

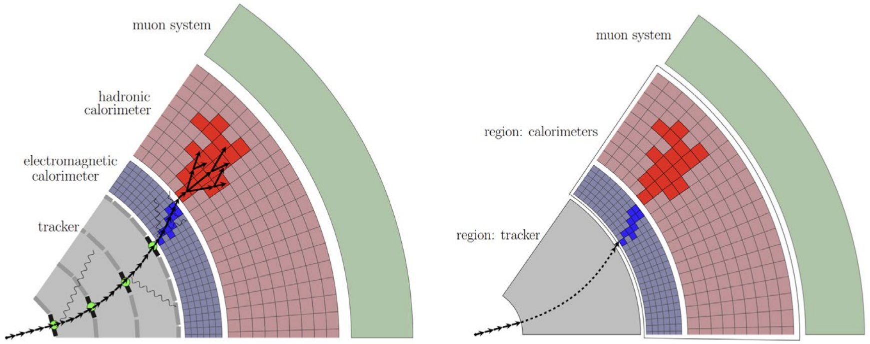




ML-based fast calorimeter simulation

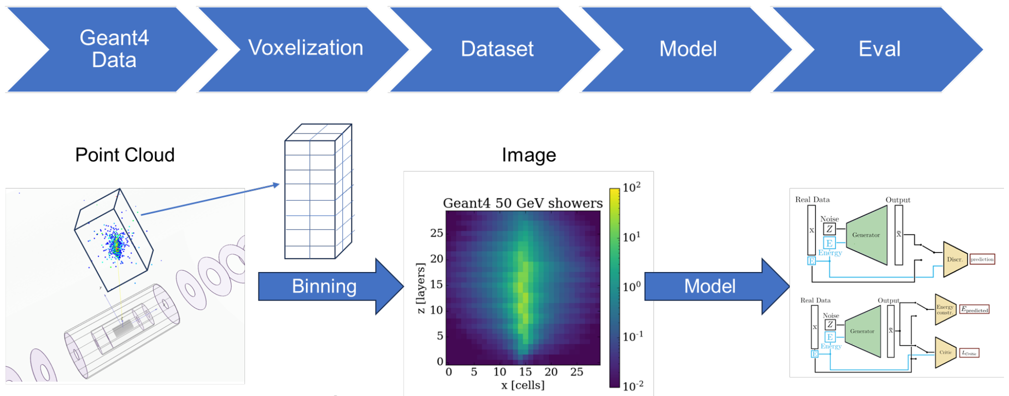
- Simulating calorimeter is the most CPU consuming part.
- Fast simulation is essential.



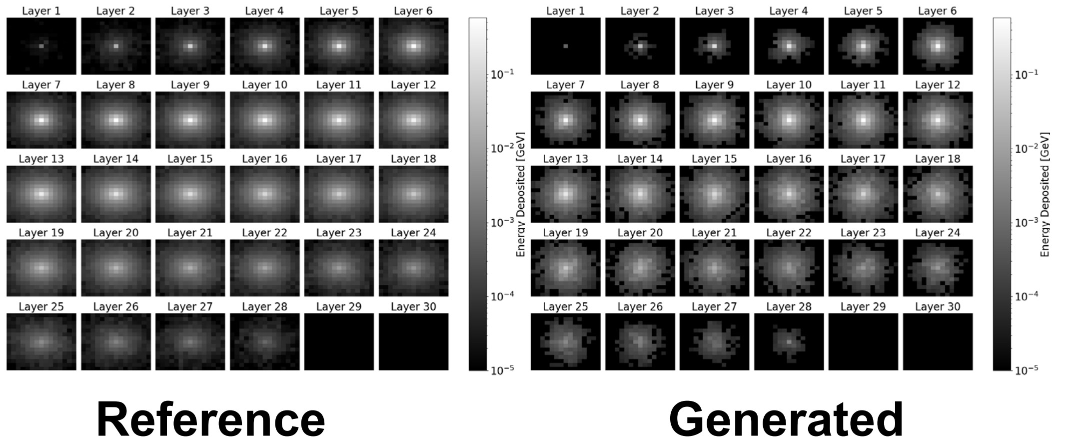


A Zaborowska 2017 J. Phys.: Conf. Ser. 898 042053

- Use machine learning method to replace the Geant4 simulation.



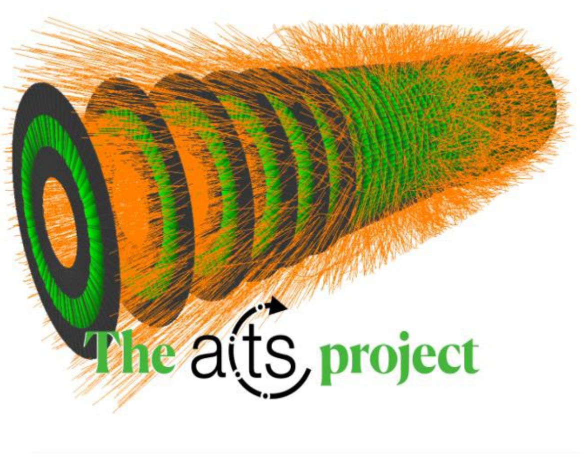
- The simulation of deposited energy in voxel is studied
  - For simplicity, only the ECAL barrel is used.



Application of TRACCC to seeding algorithm

❖ ACTS and TRACCC

- ACTS is an experiment-independent toolkit for track reconstruction
- TRACCC, one of R&D projects of ACTS, is a GPU tracking demonstrator.

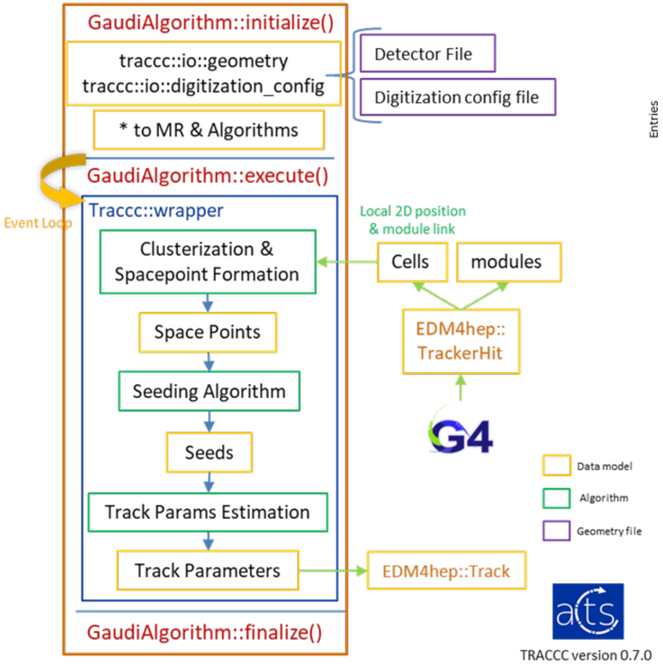


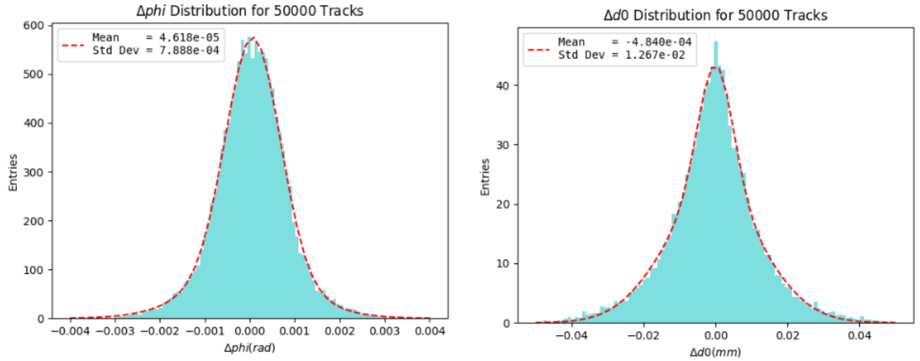
supported by

cooperations

❖ Developing CEPC seeding algorithm based on TRACCC

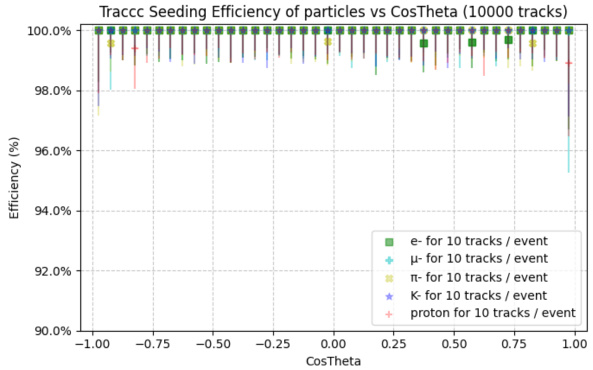
- Integration of TRACCC with the CEPCSW.
- New seeding algorithm of VTX was implemented, which can be run on both CPU and GPU





Difference between rec and sim track param

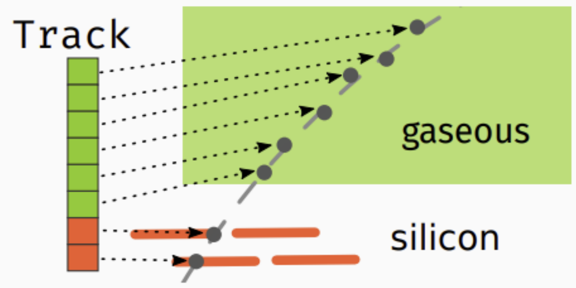
Track parameters include d0, phi (particle: mu-)



Yizhou Zhang, CHEP 2024

Consideration on migration to latest Key4hep (1)

- External libraries of CEPCSW are frozen.
- New developments in Key4hep are not used.
  - Some changes in EDM4hep could break the current CEPCSW. Need to adapt the updates.
  - Drop support for C++17 (#343, #354)
- Interface types are introduced in EDM4hep.
  - Solve the problem that no base class to inherit from.




```
interfaces:
  edm4hep::TrackerHit:
    Types: [edm4hep::TrackerHit3D, edm4hep::TrackerHitPlane]
    Members:
      - edm4hep::Vector3f position [mm] // hit position

datatypes:
  edm4hep::Track:
    OneToManyRelations:
      - edm4hep::TrackerHit trackerHits // hits of this track
```

```
auto track = edm4hep::Track{};
track.addHit(edm4hep::TrackerHit3D{});
track.addHit(edm4hep::TrackerHitPlane{});

const auto hits = track.getHits();
hits[0].isA<edm4hep::TrackerHit3D>();  // <-- true
hits[0].as<edm4hep::TrackerHit3D>();   // <-- "cast back"
hits[1].isA<edm4hep::TrackerHit3D>();  // <-- false
hits[1].as<edm4hep::TrackerHit3D>();   // <-- exception!
```

Comment: gcc13 and clang16 or newer have c++20 enabled. We are still using gcc11 on centos7.

Comment: need to update CEPCSW to use the interface types.

Thomas Madlener, CHEP 2024

Consideration on migration to latest Key4hep (2)

- External libraries of CEPCSW are frozen.
- New developments in Key4hep are not used.
  - Some changes in EDM4hep could break the current CEPCSW. Need to adapt the updates.
  - Drop support for C++17 (#343, #354)
- Interface types are introduced in EDM4hep.
  - Solve the problem that no base class to inherit from.
- Links (formerly known as Associations) are introduced.
  - Switch to C++ template.

Comment: need to use "auto" or the new link type.

```yaml
datatypes:
  MCRecoParticleLink:
    Description: "..."
    Authors: "..."
    Members:
      - float weight // weight
    OneToOneRelations:
      - RecoParticle from // rec
      - MCParticle to     // sim
```

```yaml
links:
  MCRecoParticleLink:
    Description: "..."
    Authors: "..."
    From: RecoParticle
    To: MCParticle
```

```cpp
#include <podio/LinkCollection.h>

// Link arbitrary podio generated datatypes
using McRecoParticlLinkCollection = podio::LinkCollection<
    edm4hep::ReconstructedParticle,
    edm4hep::MCParticle>;

// Enable I/O
PODIO_DECLARE_LINK(edm4hep::ReconstructedParticle,
                   edm4hep::MCParticle)

// Conventional access
auto mcP = link.getFrom();

// Templated / tuple like access
mcP = link.get<edm4hep::MCParticle>();
mcP = link.get<2>();
auto& [rp, mp, w] = link; // <-- structured bindings!
```

Consideration on migration to latest Key4hep (3)

- External libraries of CEPCSW are frozen.
- New developments in Key4hep are not used.
  - Some changes in EDM4hep could break the current CEPCSW. Need to adapt the updates.
  - Drop support for C++17 (#343, #354)
- Interface types are introduced in EDM4hep.
  - Solve the problem that no base class to inherit from.
- Links (formerly known as Associations) are introduced.
  - Switch to C++ template.
- RDataSource for podio generated EDMs
- For CEPCSW, we should follow these changes.

```
auto get_mothers(RVec<MCParticleData> mcps, RVec<int> idcs) {
  RVec<RVec<MCParticleData>> result{};
  for (const auto& mc : mcps) {
    RVec<MCParticleData> mothers{}
    for (auto i = mc.parents_begin; i != mc.parents_end; ++i) {
      mothers.push_back(mcps[idcs[i]]);
    }
    result.push_back(mothers);
  }
  return result;
}
```

```
rdf = RDataFrame("events", "input-file.root")
rdf.Define("mc_mothers",
           "get_mothers(MCParticles, _MCParticles_parents.index)")
```

```
auto get_mothers(MCParticleCollection mcps) {
  RVec<RVec<MCParticle>> result;
  for (const auto mc : mcps) {
    RVec<MCParticle> mothers(mc.getParents().begin(),
                             mc.getParents().end());
    result.push_back(mothers);
  }
}
```

```
rdf = podio.CreateDataFrame("input-file.root")
rdf.Define("mc_mothers", "get_mothers(MCParticles)")
```

Thomas Madlener, CHEP 2024

Summary

- The CEPCSW has been developed to support detector design, algorithm development and physics performance studies.
  - Release early, release often. Work tight with detector groups and performance group.
  - Bottleneck in simulation is identified, and multi-threading simulation is based on Gaussino.
  - Overlaying the background into physics events is under development.
  - Declarative physics analysis tool is developed based on RDataFrame to simplify the physics analysis.
- Long-term R&D in software is on going, the application of new technologies will be focus.
  - ML-based fast calorimeter simulation is under development to reduce the simulation time.
  - TRACCC based seeding algorithm is developed to demonstrate the usage of GPU.
  - There are new developments from Key4hep side. Need to migrate to the CEPCSW in the near future.

Thank you for your attention!