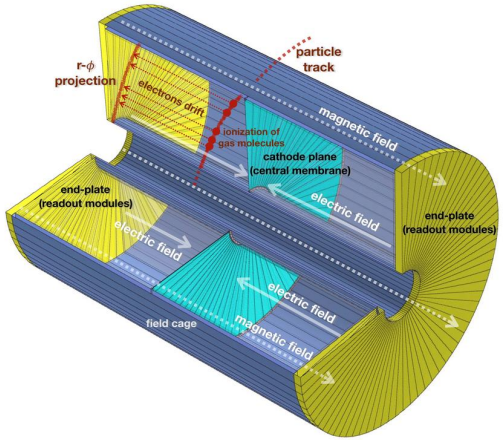
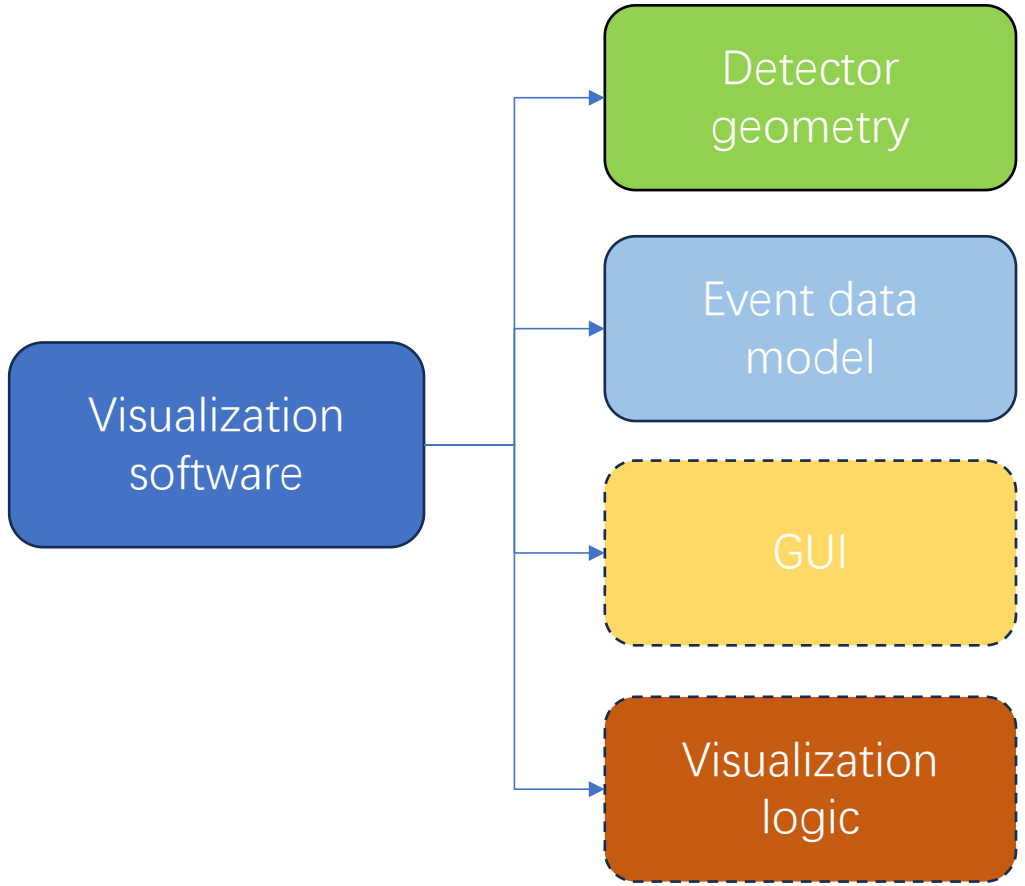
Detector and event visualization software for CEPC

Yujie Zeng（曾宇杰）, Tianzi Song（宋天资）, Zhaoyang Yuan（袁朝阳）,
Yumei Zhang（张玉美）, Zhengyun You（尤郑昀）
Sun Yat-sen University（中山大学）



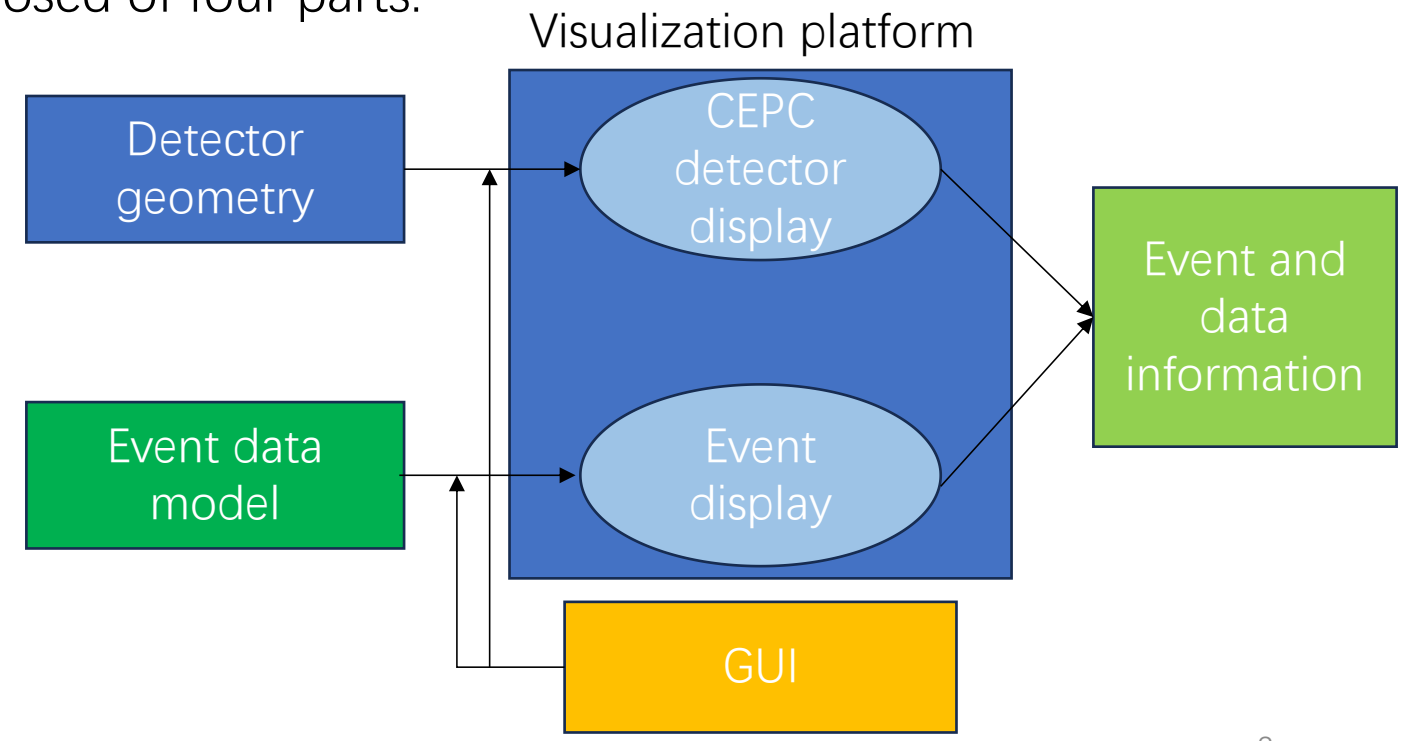
Contents

- Introduction
- Software structure
- Visualization in Phoenix
  - Detector visualization
  - Event display
- Visualization in Unity
  - FBX format
  - Detector visualization
  - Event display



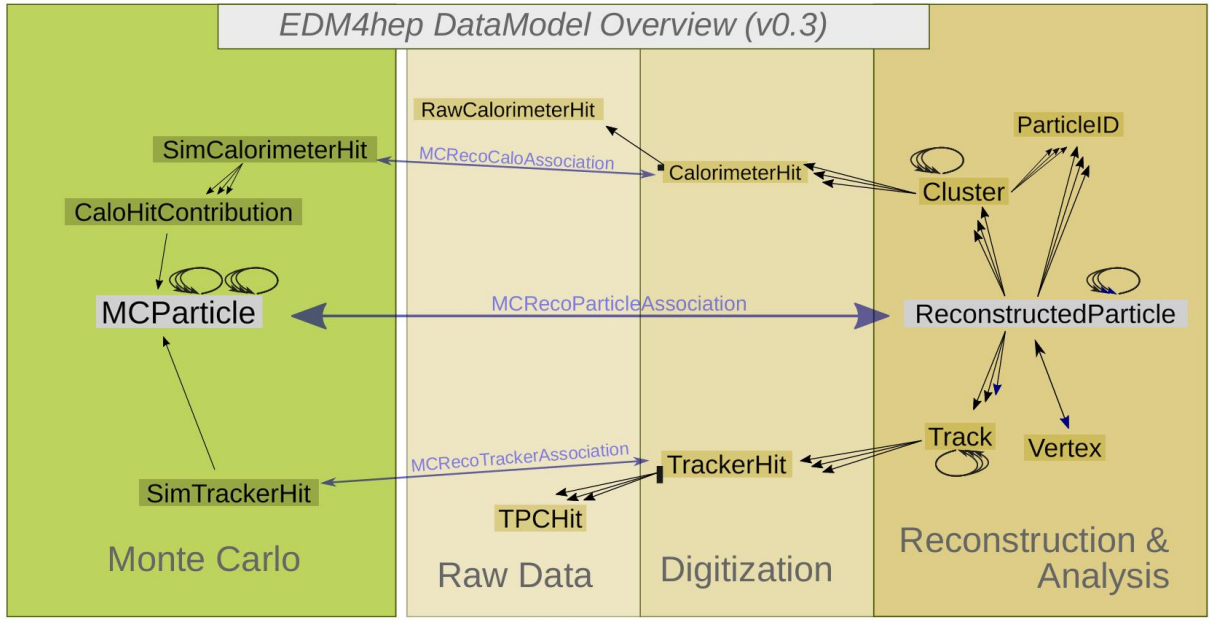
Structure of a visualization software

- An event display software should provide visualization functions of the detector geometry, display of different views of detectors, event hits distribution, and interactive control between the display and the users.
- The display software should be composed of four parts:
  - Detector geometry
  - Event data model
  - Visualization
  - GUI



Event data model

- The event data model contains three types of information: simulation, calibration, and reconstruction, which is generated by CEPC offline software.
- The EDM information is then read by the event manager and displayed together with detector geometry.
- EDM4hep is the common EDM that can be used by all communities in the Key4Hep project such as ILC, CLIC, FCC-ee & FCC-hh, CEPC, ...

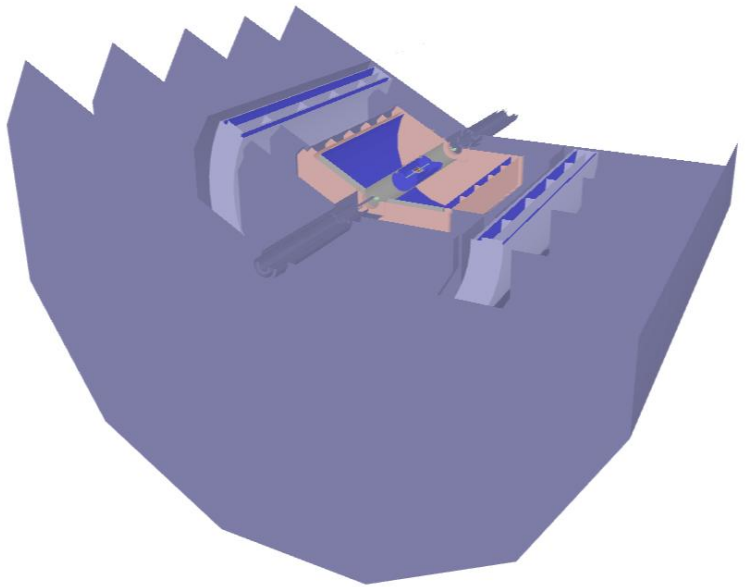


CEPCSW introduction

- CEPCSW is the CEPC offline software prototype based on Key4hep, it widely reused existing components (e.g. edm4hep, dd4hep, ROOT…)
- It provides a ready-to-work environment to algorithm developers and physicists
- Installation:
  - git clone git@code.ihep.ac.cn:cepc/CEPCSW.git
  - cd CEPCSW
  - git checkout master
  - source setup.sh
  - ./build.sh
- Generating a small MC data sample:
  - ./run.sh RecPFACyber/script/sim.py
  - ./run.sh RecPFACyber/script/digi.py
  - ./run.sh RecPFACyber/script/tracking.py
  - ./run.sh RecPFACyber/script/rec.py

Contents

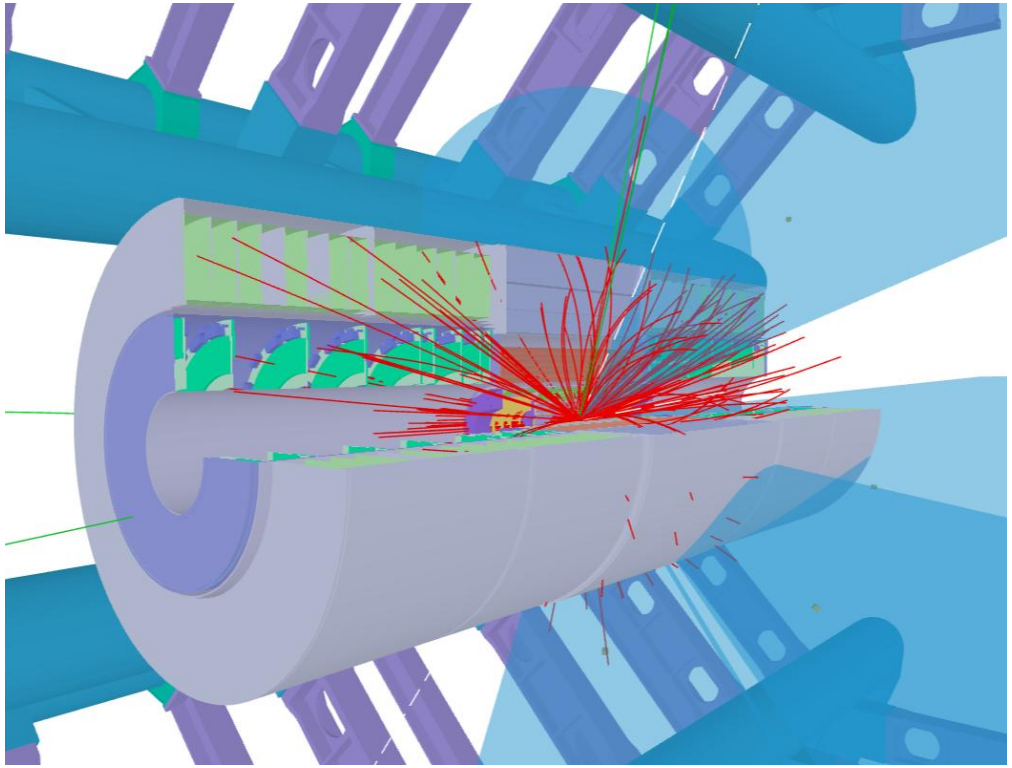
- Introduction
- Software structure
- Visualization in Phoenix
  - Detector visualization
  - Event display
- Visualization in Unity
  - FBX format
  - Detector visualization
  - Event display



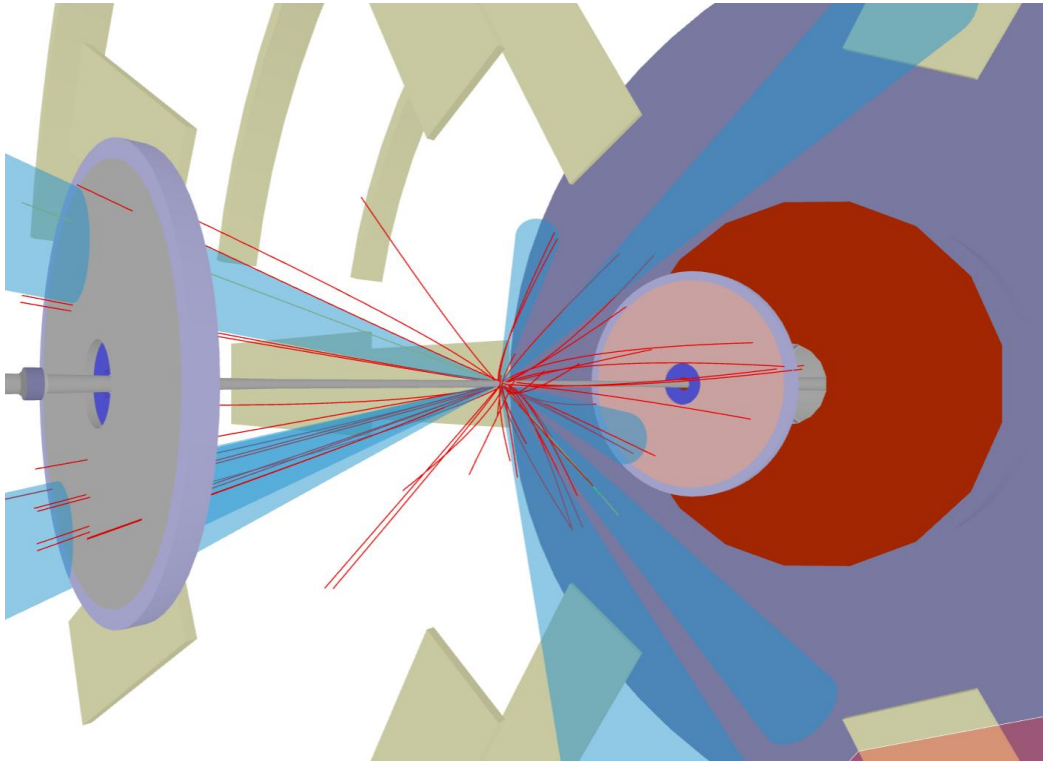
Phoenix, an web-based event display system

- Phoenix is an extensible, experiment-agnostic framework for event and geometry visualization.
- Can be used to visualize detector geometry in ROOT format, and uploaded events in JSON.
- TypeScript-based, which is an extended version of JavaScript, developed using Angular.js.
- Use the popular three.js library for 3D support, can easily run the application in a web browser or in a web server through Node.js.

Application of Phoenix in LHC experiments



ATLAS detector and events in Phoenix.

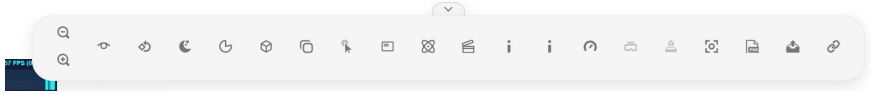


CMS detector and events in Phoenix.

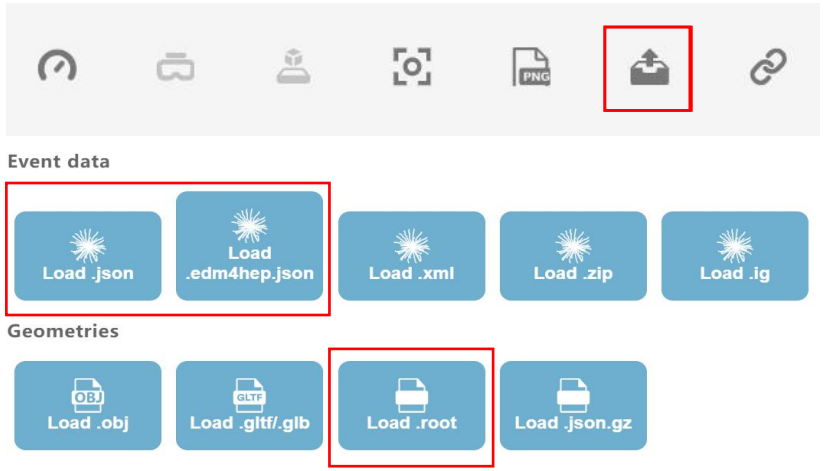
- Phoenix has been the official web event display of the ATLAS experiment. It can be used to visualize different versions of ATLAS geometry, and uploaded events.
- Also used in web event display of LHCb experiment.

Phoenix UI

- Phoenix official website: https://hepsoftwarefoundation.org/phoenix/#/
- The official Phoenix web page is shown on the right

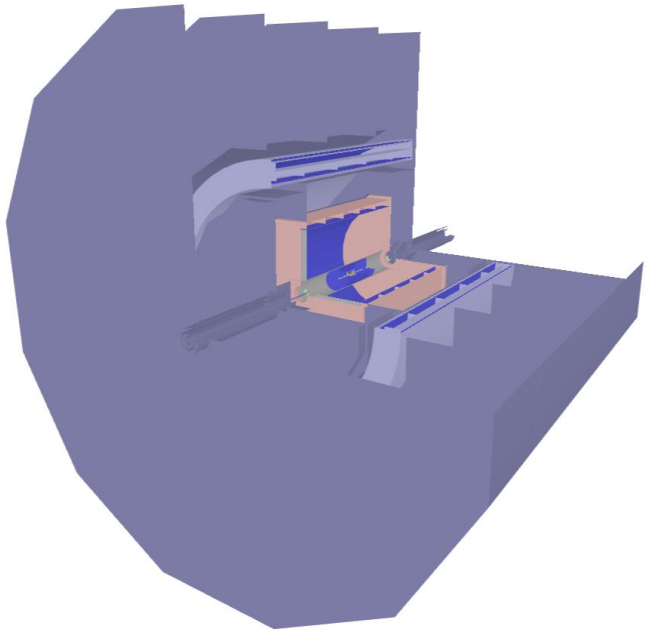


- To visualize detector and events, press Upload button in the memu tool bar.
- To visualize events, use upload buttons on the event data list. To visualize detector, press the “Load .root” button to upload detector geometry file



File formats supported by Phoenix

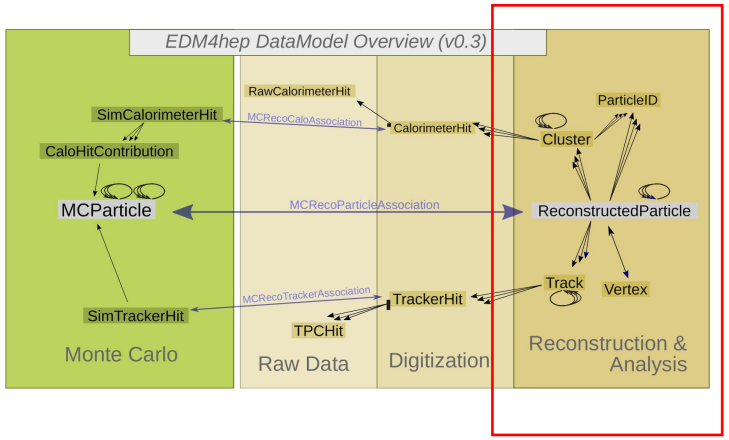
- Phoenix supports ROOT geometry file for detector visualization. With CEPCSW and ROOT software one can convert the detector geometry description to .root format.



- Phoenix makes use of a JSON format to represent event data.
  - A JSON file for event visualization should contain multiple visual components, i.e. tracks, jets, hits, clusters
  - Phoenix also supports using .edm4hep.json format to load event data.

Procedure to dump event data to JSON

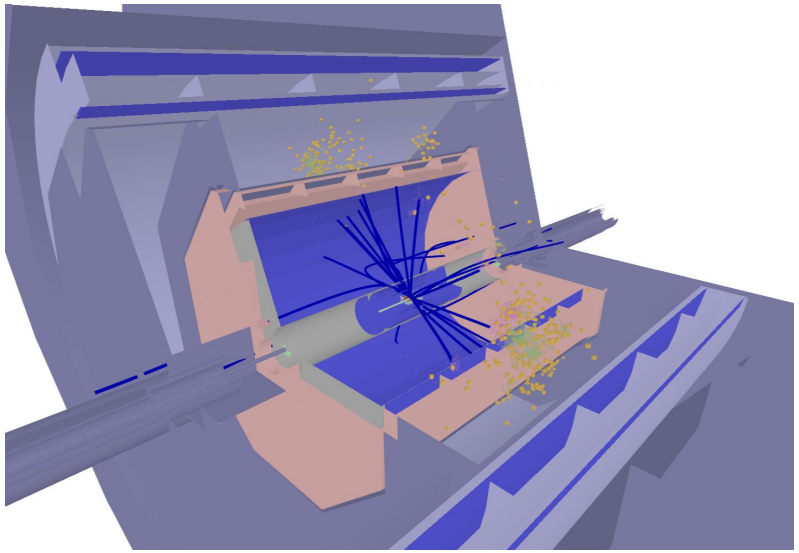
- edm4hep2json is a toolkit that dumps the contents of an EMD4hep file in JSON format.
  - Usage $edm4hep2json –e 0,1 Rec_TDR_o1_v01.root
  - -e option is used to specify the events will be dumped to JSON



edm4hep2json

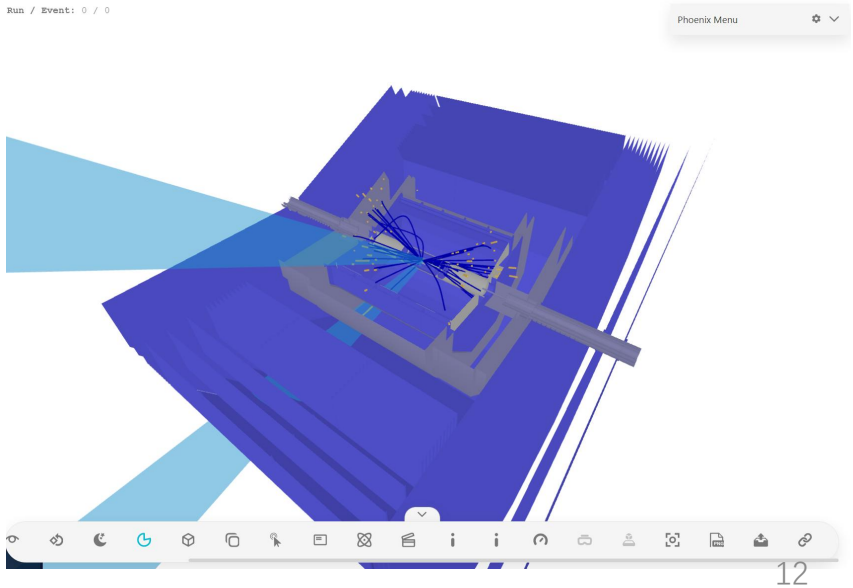
.edm4hep.json file

- Phoenix can only use reconstructed event files for visualization. The visualized detector and event are shown in the right



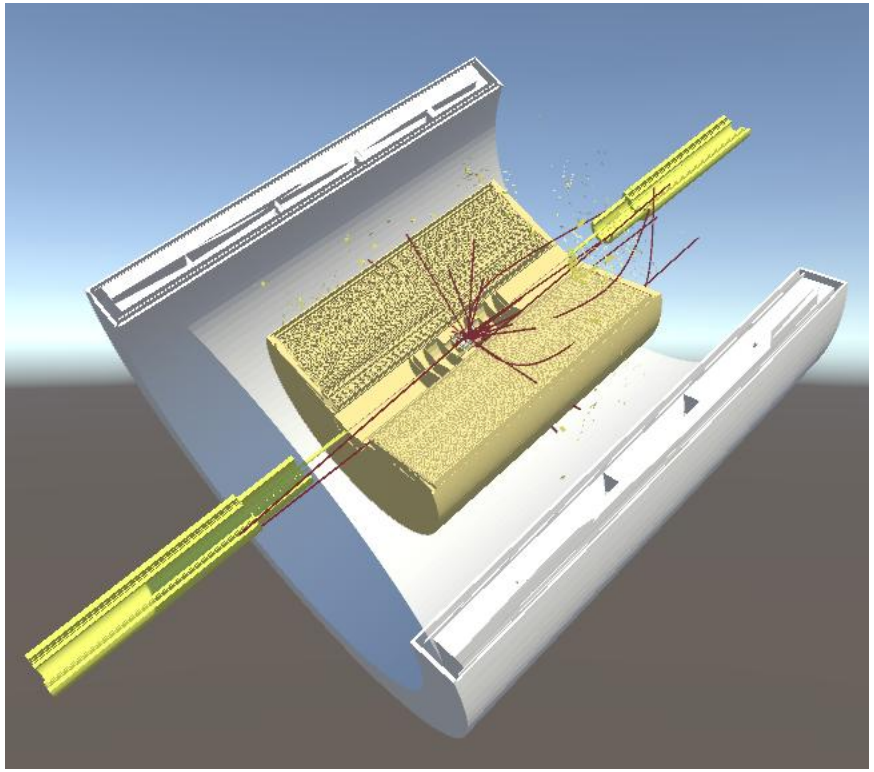
Run & Customize phoenix locally

- To develop our own phoenix application, one can download essential packages and run the applications locally.
- Essential packages:
  - Node.js: provides Runtime environment for Javascript
  - Angular.js: A web development framework that is used to develop Phoenix
  - python/Apache... e.t.c.: Softwares that can be used to start a simple website
- Build application:
  - ng build phoenix-app
- Run application:
  - python -m http.server 8080 --directory=dist/phoenix-app
  - Open http://localhost:8080 to see the web page
- After installing the packages, one can create a simple web application and edit the source code to realize more features



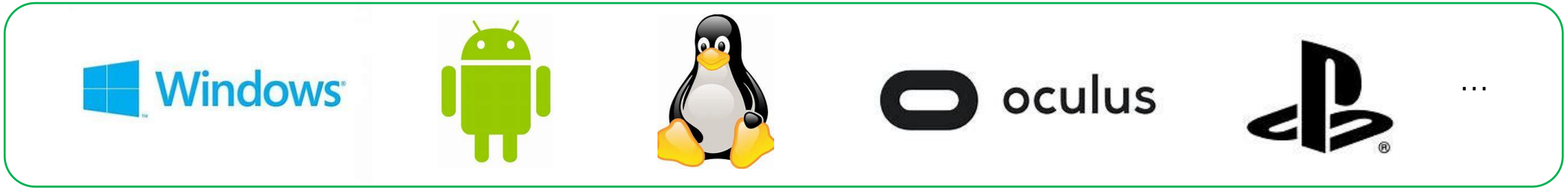
Contents

- Introduction
- Software structure
- Visualization in Phoenix
  - Detector visualization
  - Event display
- Visualization in Unity
  - FBX format
  - Detector visualization
  - Event display



Unity engine

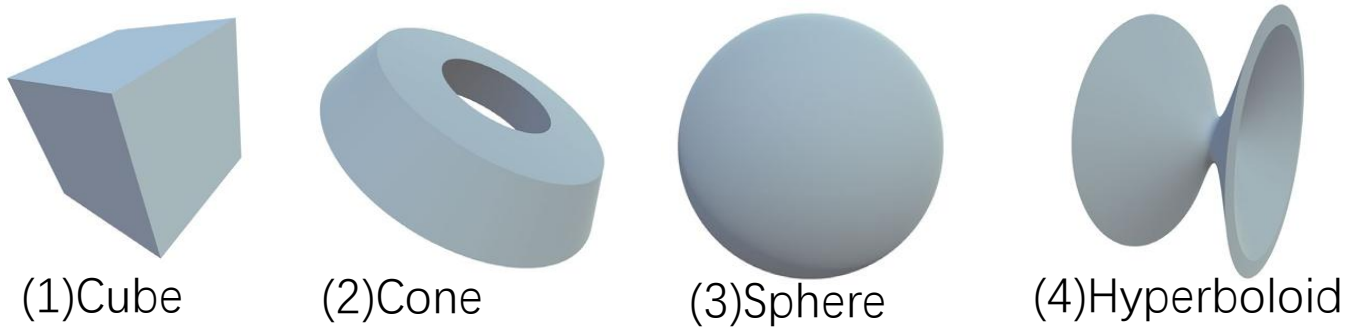
- **Unity** is the world's most popular development platform for creating multiplatform games and interactive experiences.
- Unity has the following advantages:
  - **Easy to use:** Unity engine has a intuitive interface for beginners to get started quickly. Additionally, it provides comprehensive documentation for users.
  - **High-Quality Graphics:** Unity supports advanced graphics technologies like physically-based rendering.
  - **Cross Platform:** With Unity, we can develop once and deploy the application across multiple platforms (Windows, Android, Linux, MacOS, ···)



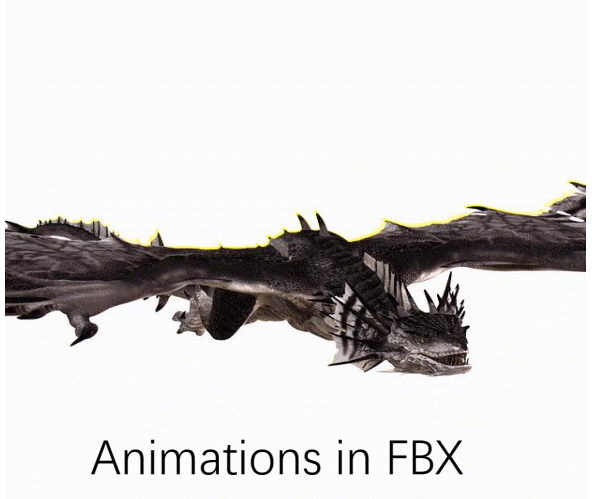
Platforms that Unity supports

FBX format

- FBX (Filmbox) is a 3D mesh file format used for exchanging 3D models and animation data.
- Contain geometry, materials, skeletons, **animations**, and others, making it a powerful file format.
- In Unity, FBX files can be directly imported for building game scenes, character models, animations, and more.



Some basic shapes in FBX format and shown in Unity

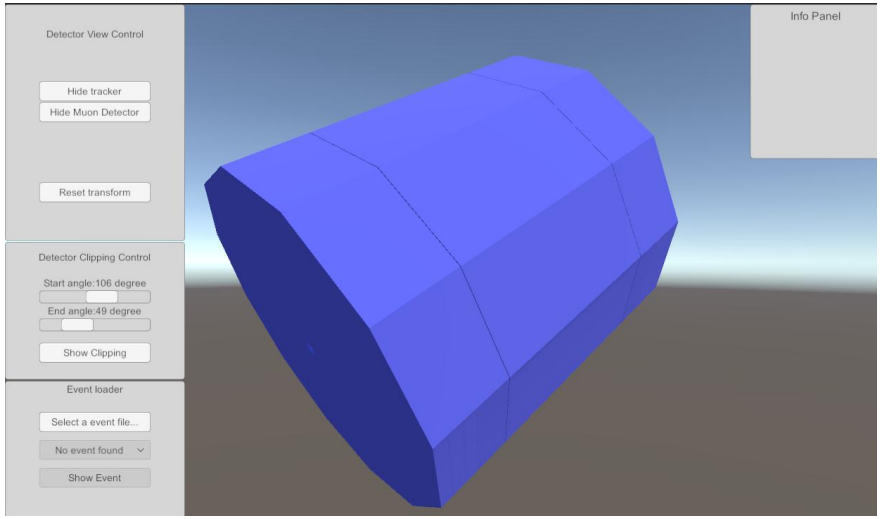


Animations in FBX

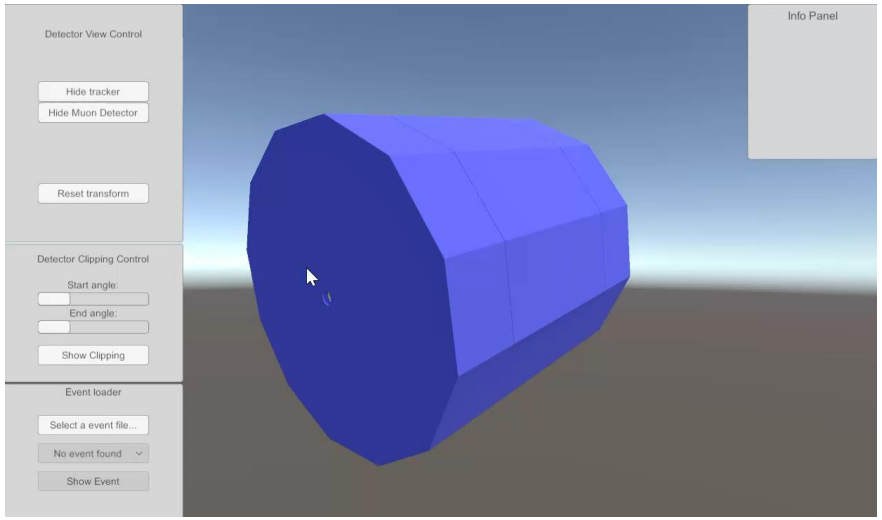
- To convert detector geometry to FBX format, **we developed DD4hep2FBX converter** that could convert DD4hep geometry description to FBX format.
- DD4hep2FBX directly converts detector geometry description to FBX format, while preserving *Shapes, Materials, transformations, Geometry hierarchy* properties

CEPC Visualization in Unity

- Currently we realized a few simple features for visualization. The GUI of the software is shown on the right:

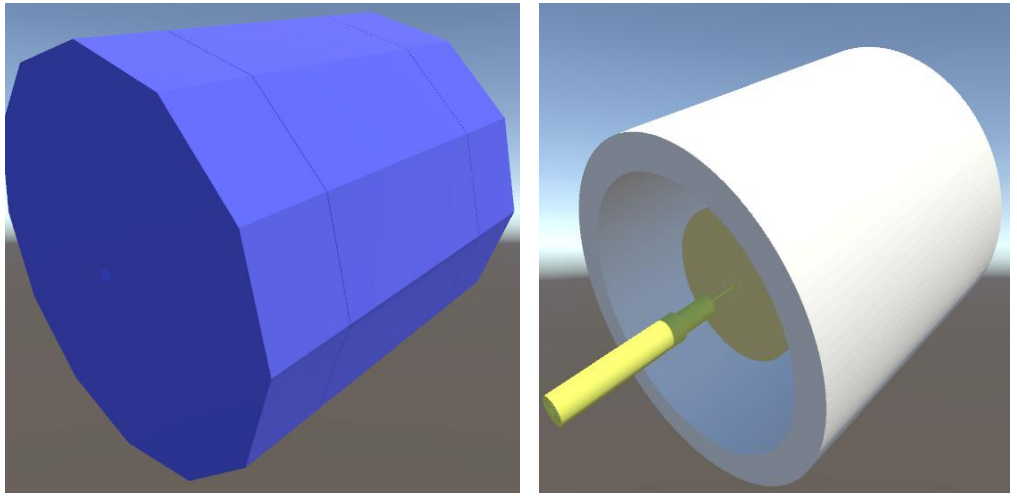


- You can drag the mouse to change the angle the viewing angle. Slide the mouse wheel to zoom in and zoom out:

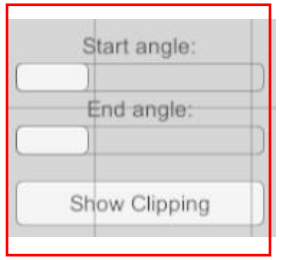


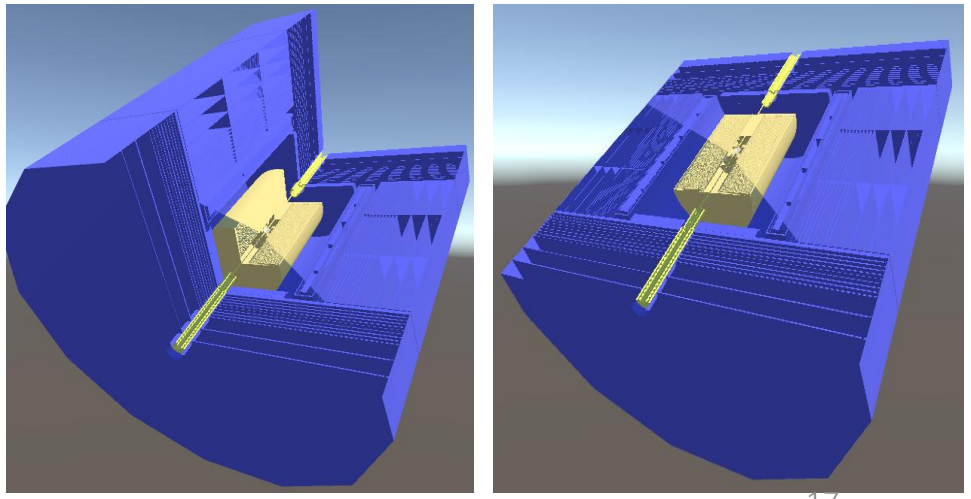
CEPC Visualization in Unity

- You can press button in the “Detector View” panel to show and hide subdetector of CEPC:



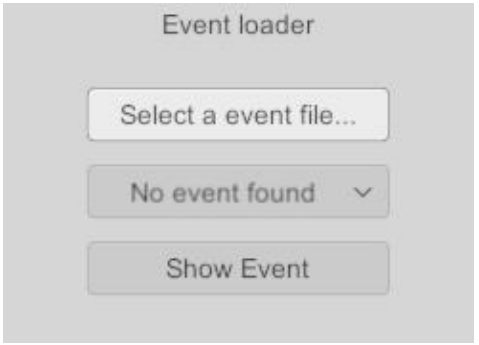
- Use the scrollbars and button to obtain a clipping view of the detector from different angles:

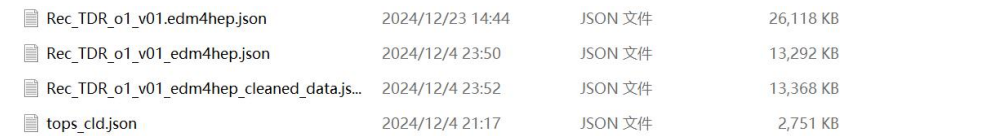


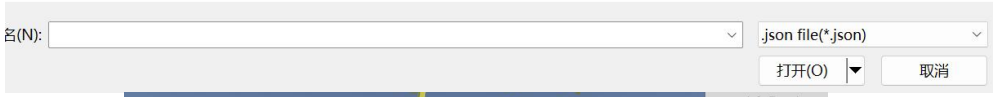


Event Visualization in Unity

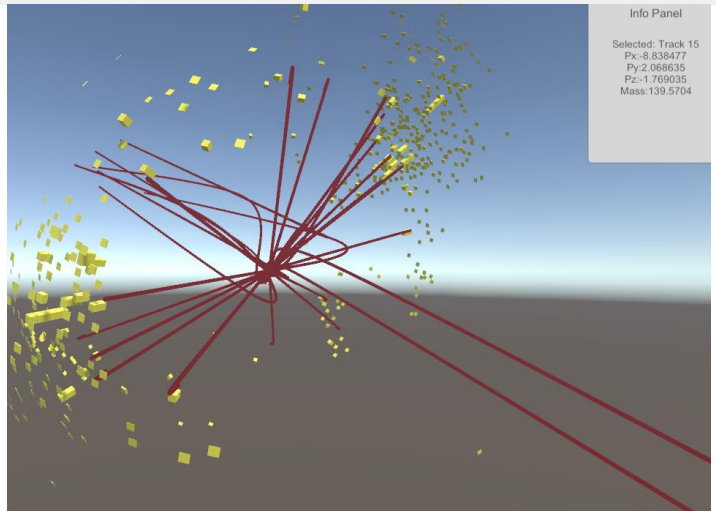
- Temporarily this application only supports .edm4hep.json file for event visualization.
- Select event file by clicking the button. Select the event to be visualized in the dropdown







- Press “show event button” to visualize the event. The visual components like tracks, clusters can be selected, and their information are shown in the info panel



Summary

Phoenix:

- Pros:
  - future official web display platform of HEP experiments
  - Highly cross platform. Can open in any web browser
  - Already has comprehensive functions
- Cons:
  - Lacked documentation support
  - Flawed functionality. Inflexible

Unity:

- Pros:
  - Support many platforms
  - Comprehensive community and documentation support
  - Highly flexible and customizable. Supports high-level 3D display effects.
- Cons:
  - Generated executables can only run locally. Not cross platform

Ongoing Work:

- Add more functions to current visualization software. i.e. Add event animation, add menu to change the rendering effect…
- Improve existing functions of the software
- Port the software to a VR platform such as Oculus Quest or Apple Vision Pro.

Thank you for listening!