Druid, displaying root module used for linear collider detectors

M Ruan

Laboratoire Leprince-Ringuet,
École polytechnique, CNRS/IN2P3, 91128 Palaiseau, France

E-mail: Manqi.ruan@llr.in2p3.fr

Abstract. Based on the ROOT TEve/TGeo classes and the standard linear collider data structure, a dedicated linear collider event display has been developed. It supports the latest detector models for both International Linear Collider and Compact Linear Collider as well as the CALICE test beam prototypes. It can be used to visualise event information at the generation, simulation and reconstruction levels. Many options are provided in an intuitive interface. It has been heavily employed in a variety of analyses.

1. Introduction

Visualization plays an important role in various aspect of experimental particle physics, including the verification of detector geometry, debugging of simulation/reconstruction codes, physics analysis and online detector monitoring. Druid has been developed as a dedicated event display for the linear collider.

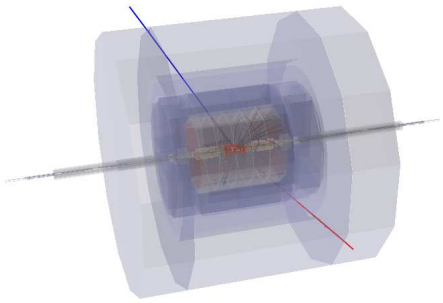

Figure 1. 230GeV ZH event at ILD detector simulation.

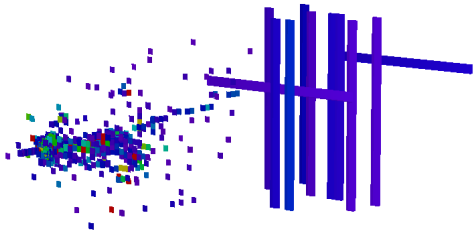

Figure 2. CALICE test beam event.

In this paper we present the dependency, development history, objects, display patterns and options for Druid, as well as an example on its utilization for debugging the reconstruction software.

2. TEve, gdml and Development history of Druid

Reading LCIO[1] data files and gdml[2] geometry files, Druid can visualize both event data and the detector geometry. For the event, Druid generates ROOT[3] TEve[4] objects according to the LCIO data file. It has been intensively tested on CALICE[5] test beam data and simulated/reconstructed full events, see figure 1, 2. For the detector geometry, Druid visualizes the geometry to different level of details according to the gdml file and the user's setting. The gdml file can be written by the official GEANT4[6] based simulation software Mokka[7]. The later release of Druid includes the gdml files for 5 most recent full detector concepts and CALICE test beam prototypes.

TEve is a framework for object management, providing hierarchical data organization, object interaction, and visualization through ROOT GUI and OpenGL. It can visualize detector geometry, simulated and reconstructed data such as hits, tracks, clusters. TEve is heavily used in LHC event displays. More details can be found in [4].

LCIO is the standard data format for the linear collider. The detector geometry is described with gear [8] or gdml file. Gear is a geometry description toolkit for ILC reconstruction software. Gear file records the large scale detector geometries, for example the radius and the half length of the TPC and the calorimeters. Gdml is an application-independent geometry description format based on XML. It can be used as the primary geometry implementation language and it provides a geometry data exchange format. In the Monte-Carlo study for linear collider, the gdml file is used to record all geometry information known to the simulation: the size, material, orientation and shape of every volume.

Druid has been optimized to have minimal dependencies: only ROOT (with version 5.28.00 or higher) and LCIO is requested. Druid has been integrated into the standard ilcsoft, which allows an automatic installation using ilcinstall [9].

The development of Druid was launched at the summer of 2009. The first prototype is a root macro, reading root files converted from LCIO files with the event information. Detector geometries was hard coded into cylinders and polygons with tunable parameters.

Soon Druid was modularized into an independent application compiled with LCIO and gear. This leads to the first release of Druid in November 2009.

In May 2010, after discussing with American colleagues working on the Silicon Detector (SiD), gdml support was added to Druid. This option allows the visualization of detector geometry up to the simulation level.

At November 2010, multiple windows option was enabled in Druid, which allows a synchronized display with R-phi and YZ plane projection (see figure 3). To minimize package dependency, gear support was dropped. Druid was stabilized since then, with minimal upgrading from time to time.

Testing on Dell E6500 (CPU: Intel Core 2 Duo T9600, 2.8GHz) with linux system, it takes about 5 second to start Druid on any data file. The time to display a event depends on the event size, for example, it takes about 3 seconds to display a 1 TeV ttH event at Compact Linear Collider (CLIC) SiD detector (see figure 3).

3. Objects

LCIO files contain event information at different levels and organize them into different collections. Druid accordingly organizes the event information into different groups.

Monte Carlo truth level:

MCParticle: displayed as helix.
SimCalorimeterHit: displayed as cuboid with different size, orientation and colors.
SimTrackerHit: displayed as points and colored according to track charge.

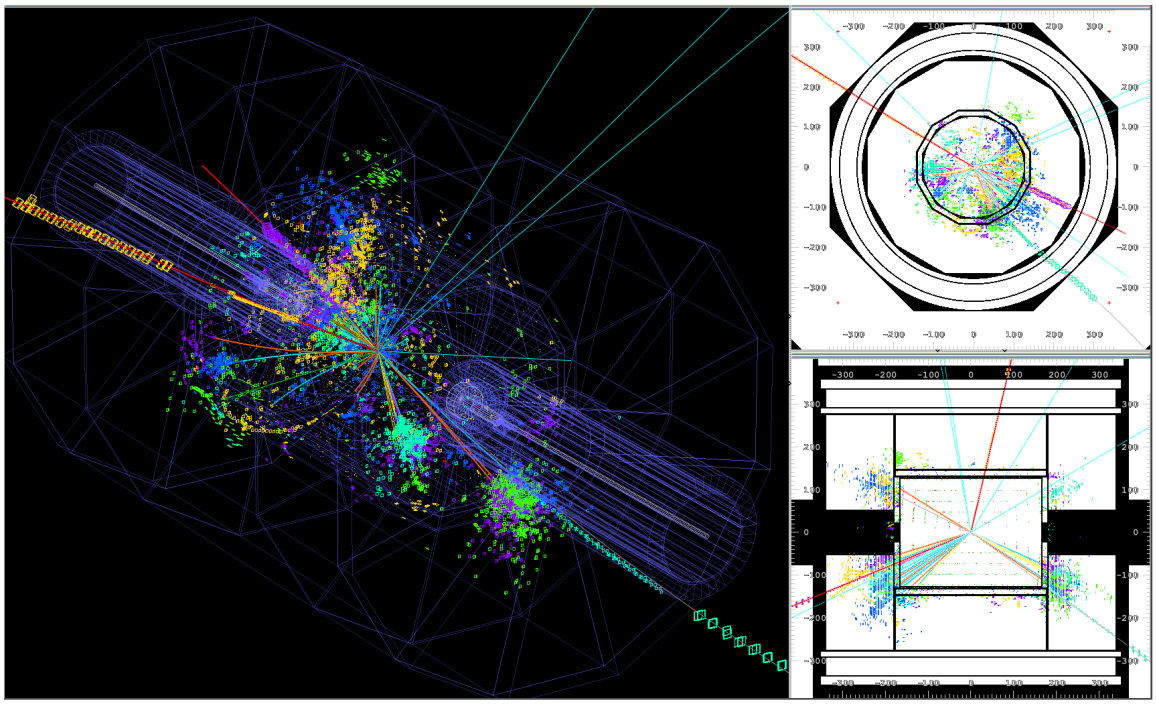

Figure 3. 1 TeV ttH event at the CLIC SID with projections.

Digitized level:

CalorimeterHit: displayed as cuboid with different size, orientation and color.
TrackerHit: displayed as point with uniform green color.

Reconstruction level:

Intermediate reconstructed objects: TrackAssignedHit, Vertex, ClusterHit. Displayed as points or cuboid and colored according to their index (the order in corresponding collection).
ReconstructedParticle: displayed as helix, colored according to the type of particle.

MCParticle and ReconstructedParticle collections are divided into subgroups according to the particle type/energy, while the detector hits are divided into subgroups according to the subdetectors.

For the detector geometry, each volume is displayed as a polyhedron, whose color and transparency is defined according to the material or to the user's setting. The full geometry information is usually too dense for a display focus on the event information, Druid therefore uses two options to reduce the workload of geometry display. First, for any "unwanted" volume, Druid allows not to display it. Secondly, a tunable parameter called display depth is employed, which refers to the level of hierarchy in the geometry organization. More details will be displayed with larger display depth. For the event display focus at event topology and debugging reconstruction code, staying at the default display depth, at which the contour of sub detectors are displayed, should be sufficient, see figure 4, 5.

4. Display Options

4.1. Options inherited from TEve and ROOT

From TEve, Druid inherits many display options with hotkey access, for example zoom, rotation, drawn back to the original orientation and scale, and switch the color of the background.

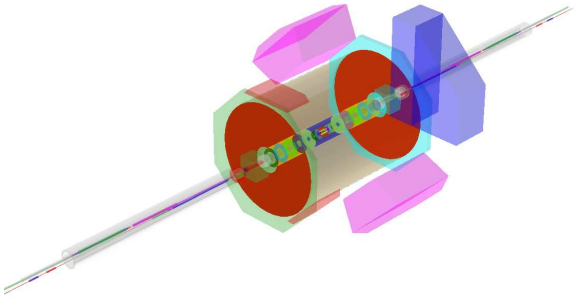

Figure 4. ILD detector with Yoke, Coil and part of Calo dismounted.

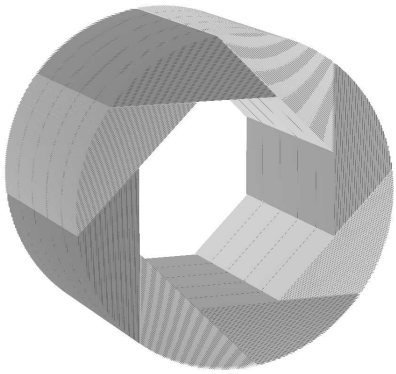

Figure 5. ILD HCAL Barrel.

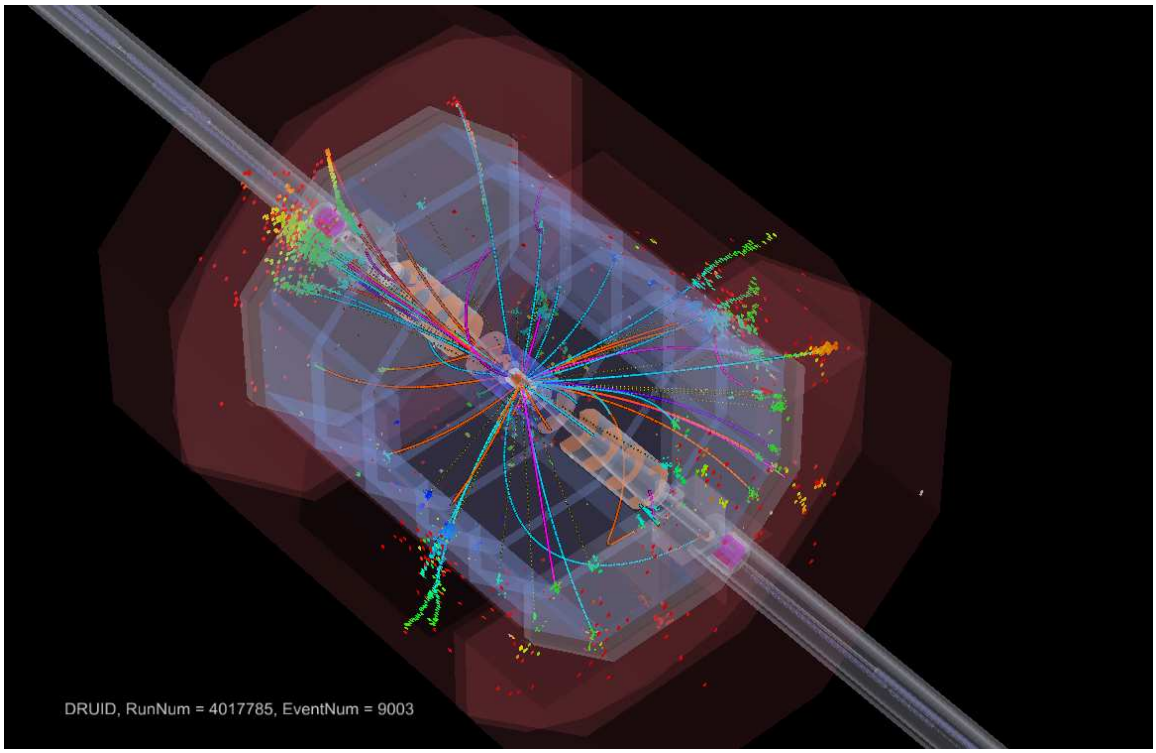

Figure 6. 500GeV ttbar event at ILD.

TEve allows a cut away view of the event display, which removes part of the display as defined by a box or a plane with interactively tunable parameters. An example is given in figure 6, where one eighth of the detector is removed.

Another useful function provided by TEve is the text attachement for each object. By picking up an object with the cursor, the attached text will be printed in the display, see figure 7.

Besides these options, there are lots of interactive actions at the ROOT GUI interface. The GUI interface is divided into three pages: the Eve page and file page inherited from TEve, as well as the option panel page defined by Druid. The file page browses the file directory while the Eve page contains a browser to all the displayed/hidden objects. By checking the appropriate boxes, any object can be hidden/displayed individually or by groups. The hide/display status will be

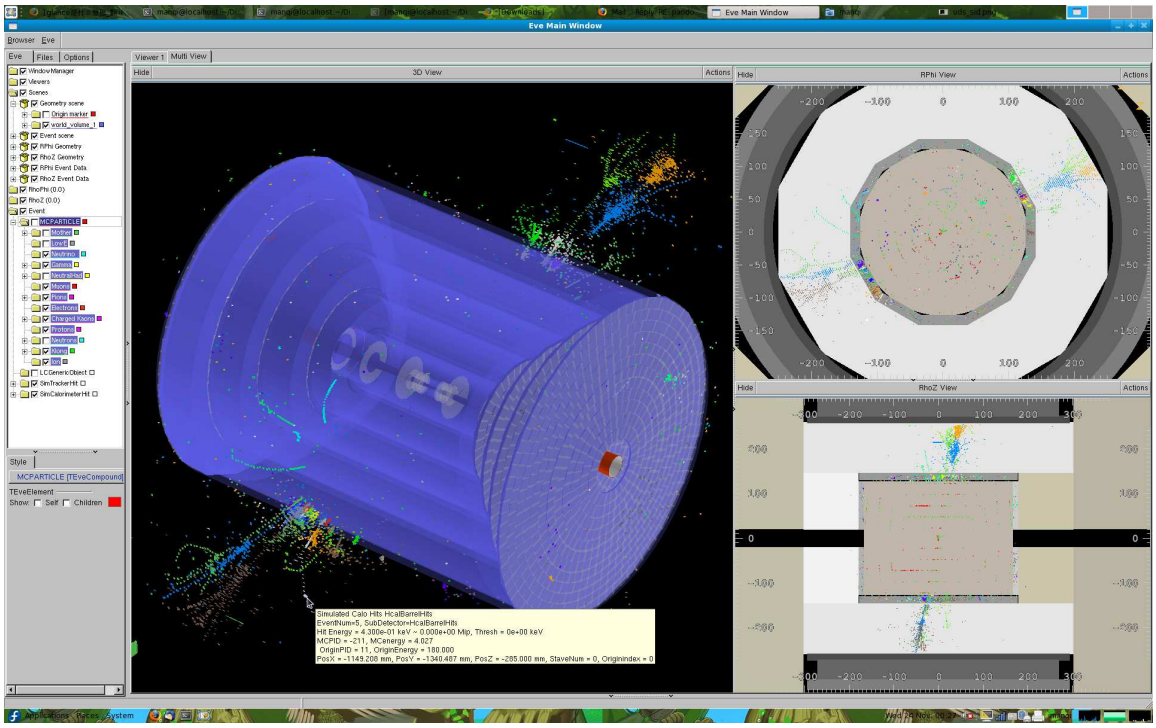

Figure 7. Z Threshold uds event at SID, with part of detector dismounted.

kept (by collection name) when navigating to a new event. Other options such as changing background color and illumination setting, tuning geometry display depth, setting reference point/frame are also available.

4.2. Event navigation and cuts

Many practical options are defined in the Druid option panel, for example:

Event navigation: to display the previous/next event, or specify the event number.

Druid defined options with buttons, there are (from left to right):

(1), Select rotation center.

(2), Regenerate the color: generate another color according to the object index. Here index means the order of a given object in its collection, for example the clusters. This option allows to a straightforward visual separation of different objects in one collection.

(3), Switch between scenarios: the minimal scenario which ignores all the intermediate reconstructed objects (as digitized hits, tracks and clusters) and the maximal scenario that displays every possible collection. To give fastest performance, the minimal scenario is set as the default.

Cuts:

The MCParticle list can be very long for the simulated event. Display all of the MCParticle may froze the display. a cut on the minimal transverse momentum (with default value 1.5GeV) is therefore applied.

A default energy cut of 0.2 mip (the most probable value of energy deposited by a minimal ionization particle in a certain volume) has been applied to both simulated and digitized calorimeter hits. The low energy hits can be shown (with default grey color) by clicking on

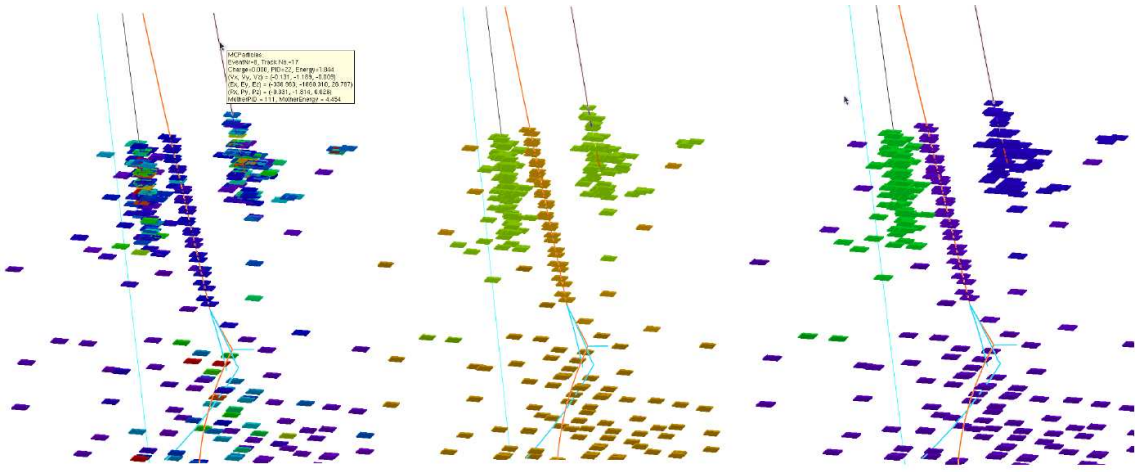

Figure 8. Tau jet $\tau \rightarrow \nu + \pi^0 + \pi^+$ with different color options: from left to right, energy, track ID and random index on track.

the box marked with 'show lowE cell'. The value of hit energy threshold can be interactively adjusted. Each time the color or size option is changed, the names and statistics of affected collections are printed on the terminal.

4.3. Color/size option for calorimeter hits

There may be three different types of calorimeter hit collections in the lcio file: the simulated, digitized and clustered calorimeter hits. Colors and sizes for these hits can be specified in the option panel according to different information.

Simulated hits can be colored according to the energy (calibrated to mips), the type or index of the particle (direct or origin) that creates this hit, the time, or a uniform color. figure 8 shows a τ jet displayed with different color option. The digitized calorimeter hits are colored to the energy calibrated to mips. A global color factor could be tuned when colored with the energy or time. Clustered hits are colored according to the cluster index.

The default size and orientation of calorimeter hits has been tuned to the cell sizes corresponding to different models and subdetectors. The hit size can be adjusted independently for simulated, digitized and clustered hits on the option panel.

5. Example application: debugging reconstruction code

One of the most important applications for Druid is the debugging of reconstruction code. Here we explain how it works with one example.

Figure 9 shows the same τ jet at simulation and reconstruction level. The left view is the reconstructed particle tracks with assigned calorimeter hits, where you can see two photons and one pion is reconstructed. The middle view overlays the MCParticle information, thus we know that reconstruction module successfully found the gammas and pions, some secondary particles generated in the shower and the neutrino are also displayed. The right view overlays the simulated calorimeter hits: some simulated calorimeter hits are dropped during the digitization/reconstruction.

6. Summary

Druid, a dedicated event display for linear collider has been developed. It has been heavily used in the geometry verification, the data analysis and the reconstruction algorithm developments.

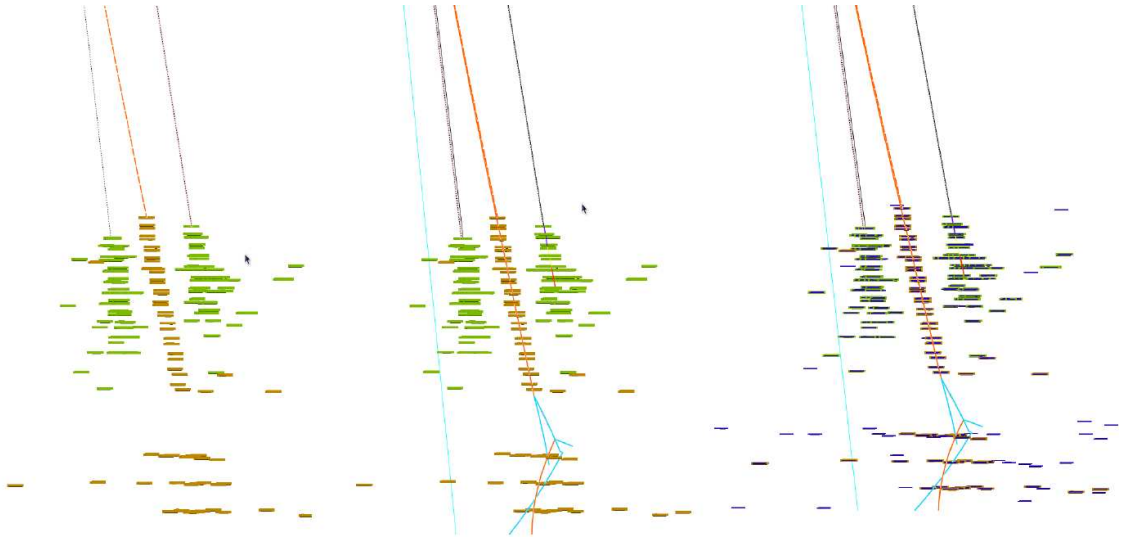

Figure 9. Tau jet $\tau \rightarrow \nu + \pi^0 + \pi^+$ at MCTruth and Reconstruction level.

Acknowledgments

I am grateful to Gabriel Musat, Daniel Jeans for the polishing of the codes, to Vincent Boudry, Jean-Claude Brient and Henri Videau for the valuable discussion and their continuous support. Thanks to Jayant Pande for his hard work in the early developing phase.

Special Thanks goes to Matevz and Alja, for all the nice discussions and beer we had.

The research leading to these results has received funding from the European Commission under the FP7 Research Infrastructures project AIDA, grant agreement no. 262025.

References

[1] Gaede F, *et al.* 2003 LCIO-A persistency framework for linear collider simulation studies *LC-TOOL-2003-053*

[2] Chytracek R, McCormick J, Pokorski W, Santin G 2006 Geometry Description Markup Language for Physics Simulation and Analysis Applications *IEEE Trans. Nucl. Sci.*, Vol.**53**, Issue: 5, Part 2, 2892-2896

[3] Brun R and Rademakers F 1997 ROOT - An Object Oriented Data Analysis Framework, Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, *Nucl. Inst. Meth. in Phys. Res. A* **389** 81-86

[4] Tadel M 2010 Overview of Eve - the Event Visualization Environment of the ROOT *J. Phys.: Conf. Series.* **219** 042055

[5] The CALICE collaboration 2008 Design and electronics commissioning of the physics prototype of a Si-W electromagnetic calorimeter for the International Linear Collider *Journal of Instrumentation* **3(08)**: P08001, arXiv:0805.4833v1 [physics.ins-det].

[6] Agostinelli S, *et al.* 2003 Geant4 - a simulation toolkit *Nucl. Instrum. Meth.* A **506**, 250-303

[7] http://polzope.in2p3.fr:8081/MOKKA

[8] http://ilcsoft.desy.de/portal/software_packages/gear

[9] http://ilcsoft.desy.de/portal/software_packages/ilcinstall