17th International Conference on Computing in High Energy and Nuclear Physics (CHEP09) IOP Publishing
Journal of Physics: Conference Series **219** (2010) 032014 doi:10.1088/1742-6596/219/3/032014

Fireworks: A Physics Event Display for CMS

D Kovalskyi[1], M Tadel[2], A Mrak-Tadel[2], B Bellenot[2], V Kuznetsov[3], C D Jones[4], L Bauerdick[4], M Case[5], J Mülmenstädt[6], A Yagil[6]

[1] University of California, Santa Barbara, USA
[2] CERN - European Organization for Nuclear Research, Switzerland
[3] Cornell University, USA
[4] Fermi National Accelerator Laboratory, USA
[5] University of California, Davis, USA
[6] University of California, San Diego, USA

E-mail: dmytro@physics.ucsb.edu

Abstract. Fireworks is a CMS event display which is specialized for the physics studies case. This specialization allows us to use a stylized rather than 3D-accurate representation when appropriate. Data handling is greatly simplified by using only reconstructed information and ideal geometry. Fireworks provides an easy-to-use interface which allows a physicist to concentrate only on the data in which he is interested. Data is presented via graphical and textual views. Fireworks is built using the Eve subsystem of the CERN ROOT project and CMS's FWLite project. The FWLite project was part of CMS's recent code redesign which separates data classes into libraries separate from algorithms producing the data and uses ROOT directly for C++ object storage, thereby allowing the data classes to be used directly in ROOT.

1. Introduction

Event displays represent an essential tool for any high energy physics experiment. They help to understand how the detector works by visualizing physics processes and particle interactions with various sub-detectors. Different tasks require different levels of detail. Generic event displays capable of doing all tasks in a single application tend to be quite complex with many limitations such as single platform support, network and database connectivity requirements, intermediate data formats etc.

Fireworks is an event display that was designed primarily as a physics analysis tool that can be effectively used on laptops. A typical use case is a physicist traveling between his home institution and CERN. In most cases a network is either unavailable or a connection has very limited bandwidth and high latency. Limited power and CPU resources require the application to be light. Limited storage space and performance limit the size of the application and demand efficient data access. All these aspects were carefully considered in the design of the event display and solutions were found to satisfy the requirements and achieve the primary goal: make a useful physics analysis tool.

2. Design

Fireworks relies on the CMS Event Data Model [1, 2]. In this model an event is made of collections of objects such as tracks, electrons, muons etc. Each object can be of arbitrary C++ type and its persistent and transient representations are identical. ROOT I/O is chosen as the

underlying technology for implementing the event store, and Reflex dictionaries are provided for every stored object type [3].

In the CMS software the object data types are factorized from the remaining software in separate libraries with limited external dependencies. Data types can only depend on other data types and a very limited number of special service classes. This makes it possible to build a light ROOT-based application capable of reading the standard CMS data files without the overhead of the full CMS software framework and reconstruction code. One such application is FWLite, an interactive ROOT environment with on-demand loading of needed CMS data format libraries. FWLite also provides a read-only framework to access collections of objects in the event similar to the full CMS framework.

Fireworks is built on top of FWLite and therefore by design can show only what is stored in data. The standard CMS data tiers (RECO and AOD) provide enough information to visualize events, which makes reconstruction on the fly unnecessary. This design choice allows us to make the event display much lighter than the full CMS software framework.

In order to simplify the event display and to avoid dependence on a condition database, we use the ideal CMS geometry used for Monte Carlo event simulations. This detector description is adequate for physics analysis needs and allows the event display to be used off-line.

Event visualization is performed by Eve, a new ROOT package which provides a comprehensive framework for data visualization [4]. Objects are rendered using OpenGL with details of implementation hidden from users. The Eve interface consists of a set of standard object types which simplify visualization of tracks and calorimeter energy depositions and allow to render arbitrary shapes to represent information.

For tracks Eve takes care of proper track propagation in the user defined magnetic field using information from the reconstructed track trajectories as reference points. The accuracy of such visualization depends on the amount of available information about the reconstructed track trajectory and the magnetic field model. In practice even with the most basic magnetic field model, quite different from the true magnetic field model, we are able to have accurate trajectory visualization with just a few reference points taken from the result of the full trajectory fit during the event reconstruction.

Eve makes it possible to implement a number of different views. A generic 3D view can be automatically projected to 2D views: $\rho - \phi$ and $\rho - z$, where the former represents the transverse projection with respect to the beam direction and the latter is a projection with integrated out ϕ angle information.

One of the key design principles for Fireworks is the event display usability. Event displays are not everyday tools for most physicists, so the user interface should be intuitive such that the event display can be used with little prior knowledge.

3. Display overview

Figure 1 is an overview of the event display which illustrates the features of immediate interest to most users.

3.1. Event navigation

At the top of the event display window, the user is presented with an *event navigation interface* that allows him to go forward or backward between events or to jump to the event with the desired run and event number. It is also possible to display only a subset of events by entering a *filter expression* in the corresponding entry field. For example to look only at events that have at least one muon with transverse momentum greater than 20 GeV, a user would type in the filter field: `$Muons.pt()>20`. The navigation interface is also used to display information about the currently selected event: the run and event number, luminosity block id, time stamp, and the event filtering currently in effect.

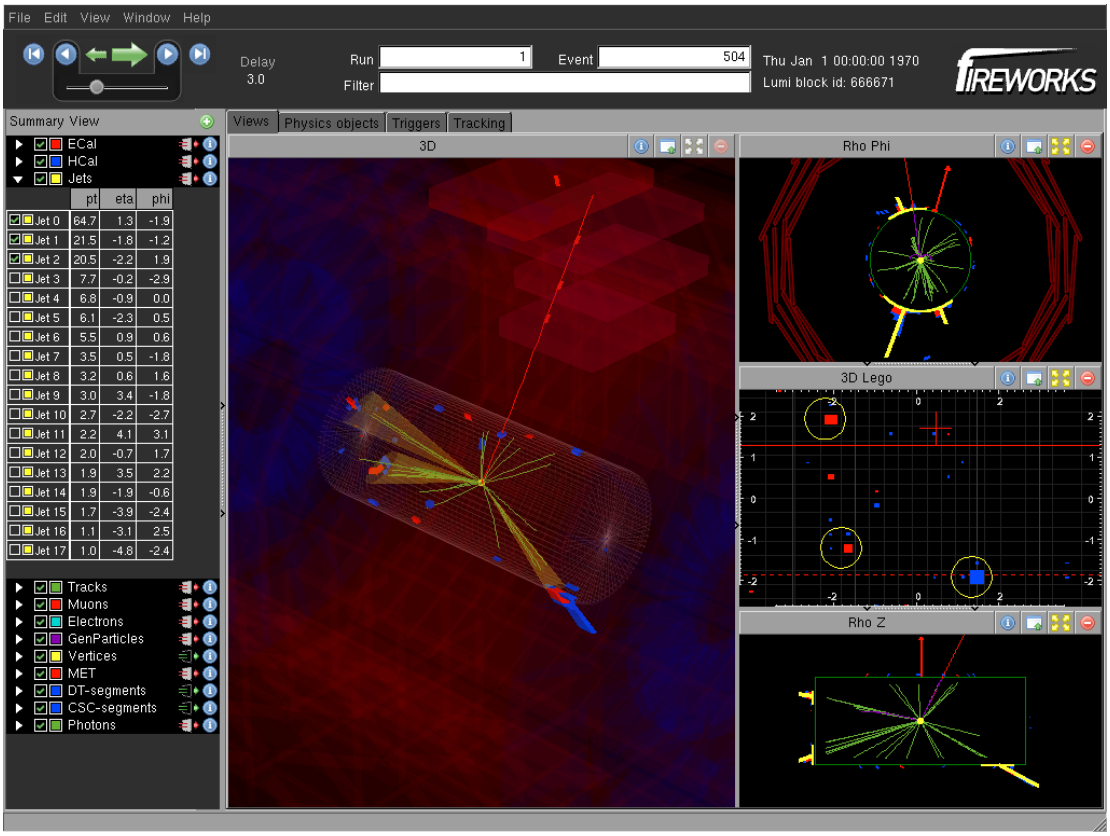

Figure 1. Fireworks event display with a typical configuration, showing some Monte Carlo simulated event.

A special feature of the event display, *playback mode*, can also be activated from the event navigation interface. This is useful if the event display is used to monitor a stream of events by switching between events automatically at a user-specified interval.

3.2. Summary view

To the left of the event display window, the *summary view* contains a list of all object collections currently known to the event display. This list can be used to control which objects are visible and to select objects. From this view, it is also possible to add other event collections contained in the data file (for which the user is presented with a list of all available collections), and to invoke interfaces that control how data are displayed.

3.3. Three-dimensional graphical view

A three-dimensional view of all objects in the event and an outline of the detector is provided. By making the detector outline transparent, the display provides a frame of reference for the displayed objects without obstructing the user's view. Together with the ability to rotate, translate, and zoom, this gives the user the tools to visualize the three-dimensional structure of the event.

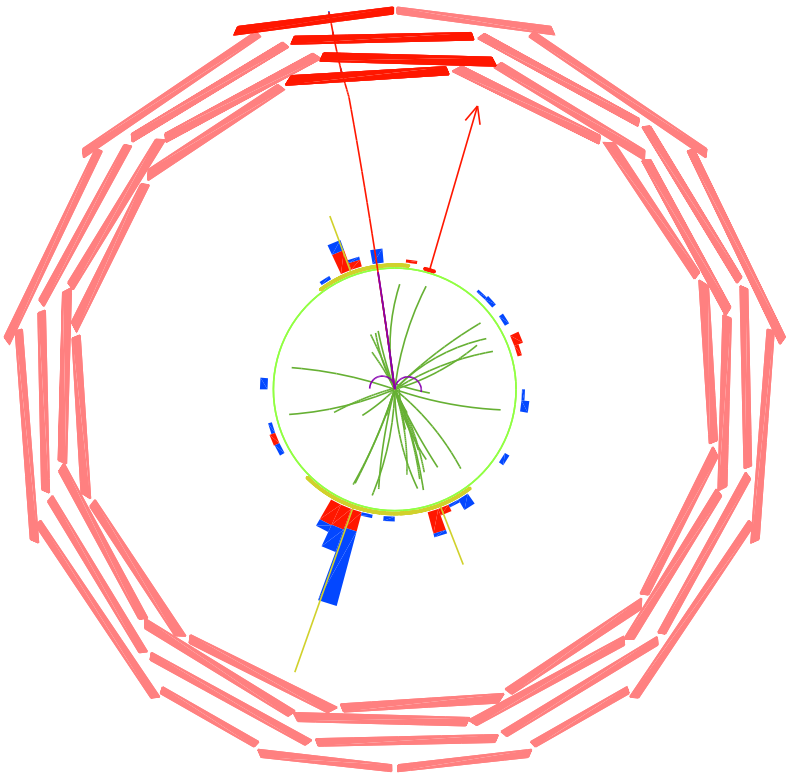

Figure 2. $\rho - \phi$ view. Compression of the muon system allows to see more details in the tracker and devote more space to the calorimeter area, which is used to show energy of the reconstructed objects such as jets (yellow) and missing transverse energy (red) along with the energy depositions in the electro-magnetic (red) and hadronic (blue) calorimeters.

3.4. Projected views

Two-dimensional projected views in ρ–z, ρ–ϕ, and η–ϕ show tracks, calorimeter deposits, jets, electrons, photons, muons, and missing E_T. Trigger-level reconstructed objects, generator particles, primary and secondary vertices, as well as other user-specified objects can also be displayed. To aid in the optimal display of the very different length scales between the inner detector and the muon system, distortion can be used to compress the representation of the muon chambers (Fig. 2).

The η–ϕ (Fig. 3) view shows the energy deposition in the electromagnetic and hadronic calorimeters in two different display modes: a two-dimensional mode in which the area of the graphical representation is proportional to the energy deposit, and a three-dimensional lego mode in which the height of the graphical representation is proportional to the energy. The displayed segmentation of the calorimeter adapts to the zoom level. Jet outlines, muons, and

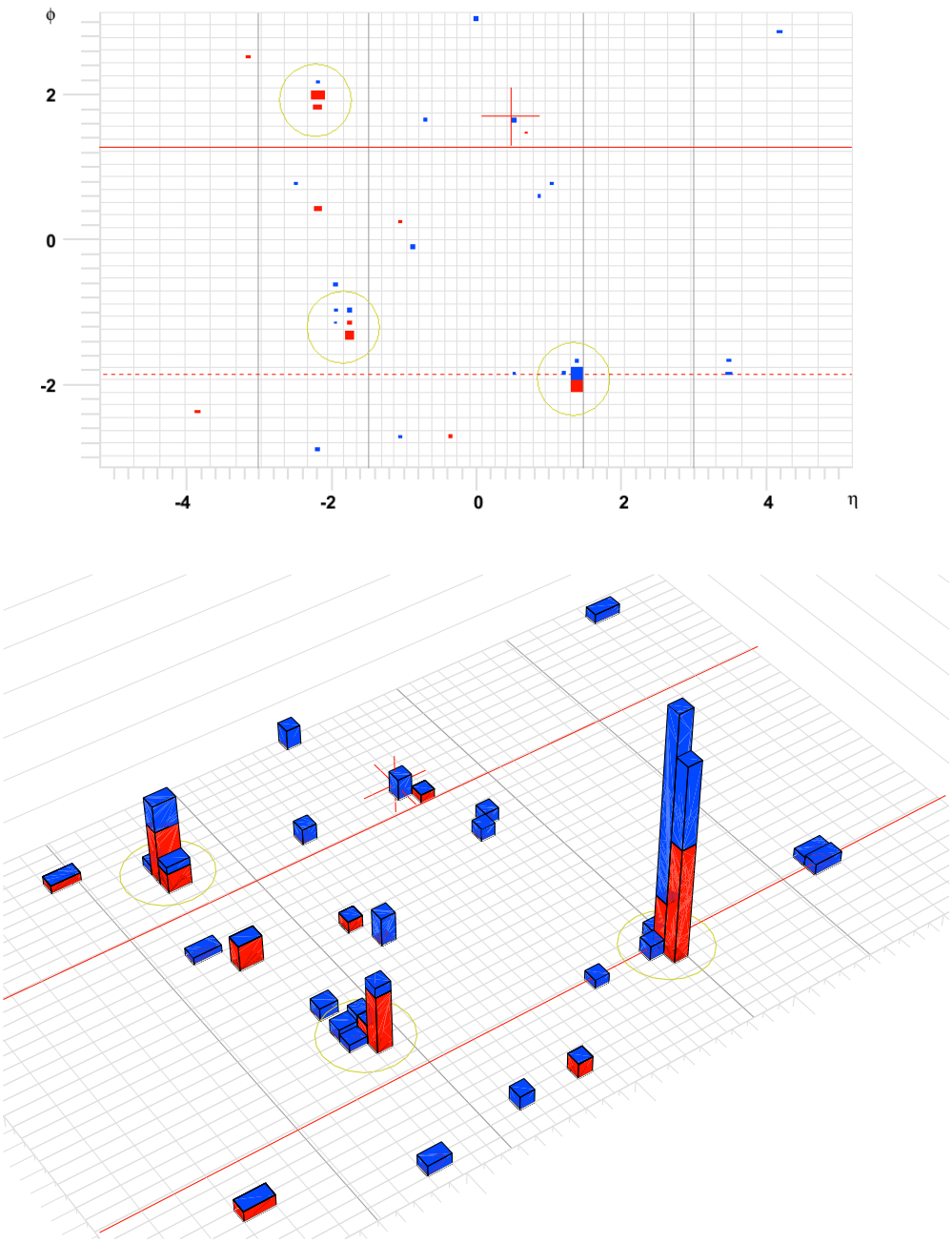

Figure 3. $\eta - \phi$ view. Top pictures shows a 2D option of the view, where cell size represent energy and color corresponds to the calorimeter that made the biggest contribution. Bottom picture shows a standard 3D Lego option.

missing E_T are also shown on the η–ϕ grid.

3.5. Table views
Tables containing detailed information about physics objects, triggers and tracking are available in *table views*. Any number of table views can be open at a given time. The user can choose interactively which properties to display for a given data type with an interface that allows adding, deleting, and modifying table columns. For the user's convenience, this interface, like the filtering interfaces, provides a list of available methods through tab completion. The tables can be sorted according to any of the displayed variables.

3.6. Cross-view consistency
Cross-view data interpretation is easy since the same object is shown using the same color in all views and if the object is selected it is highlighted in all views.

4. Feature overview
Following the overview of views in the previous section, this section will discuss the features that are provided across views to provide clear and easy interaction with the display.

4.1. What data to show
To derive physical insight from the event display, the user must be able to remove clutter that would distract from the relevant event content. Selective data display is a key design consideration in Fireworks. Several mechanisms exist:

(i) The user can choose which object collections the event display should consider
(ii) Using the summary view, individual objects or entire collections can be made invisible
(iii) Filters can be applied to make objects invisible
(iv) One or more objects can be selected (by hand or automatically with a selection requirement), causing them to be highlighted
(v) Objects of particular interest can be emphasized by choosing special colors.

4.2. Object filtering
Objects can be made invisible using a filter expression. Any `const` method that takes no arguments can be evaluated. Objects for which the expression evaluates to `false` are made invisible. Tab completion is provided. Methods of the data objects are evaluated using the Reflex dictionaries.

4.3. Object selection
Objects can be made selected using a filter expression. All comments in the previous section apply.

4.4. Tooltip
Some important identifying information about an object is displayed immediately as a mouse pointer tooltip box when the pointer hovers over the object.

4.5. Configuration
Once the physicist has configured Fireworks to his liking he can save the configuration. The configuration stores the collections to be displayed; the active filters; the list of views, their options and their placement; and the list of columns to be displayed for each object type in table views. A new configuration file can be created, or the existing one can be reused, at the discretion of the user.

4.6. Space management
Using the limited available desktop space efficiently is of paramount importance. Resizing of views, swapping a view between windows, and *undocking* a window from the main event display window all help the user make the most effective use of his desktop real estate.

4.7. Detailed views
Objects can be further studied by displaying a detailed view. Hit-level information, currently implemented for electrons and photons, where the individual crystals are displayed. This capability will be extended to other object types in the future.

5. Distribution
Two distribution mechanisms are used. Stand-alone distribution as a tarball for Linux and Mac OS X allows users to install the event display without having to install any other packages. This distribution mechanism was designed to be as simple as possible for the end-user: he only needs to download the tarball, untar, and run; no setting of environmental variables and no local configuration are necessary.

The event display is also distributed as packages in CMSSW. Thus, on any machine that already has CMSSW installed, the event display is automatically available.

6. Extensibility
Rendering of data in the event display is implemented in a form of a plugin system, which hides details of the event display internals from users. For each data type and for each view there is a plugin that defines how the data should be presented. The event display has plugins to render most data type in CMS software. If some data type is not supported, a user may write his own plugin.

Each plugin can render a collection of objects of a given data type and all other types that inherit from it. In the CMS software majority of data objects inherit from a common base class which effectively represents an abstraction of a particle with 4-momentum and particle type. Therefore in practice it is hardly ever necessary to write any code to visualize new data.

7. Conclusion
We have presented a new physics analysis oriented CMS event display - Fireworks. It was implemented with a focus on the display of physics object level information, optimizing use of desktop space to show the relevant information with the emphasis on usability on a typical laptop without access to remote services. An overview of the different ways of viewing information was given, as was a list of the features that enable a user to display the available information selectively.

References
[1] The CMS Collaboration, "CMS technical design report, volume II: Physics performance,", J. Phys. G **34**, 995 (2007).
[2] Jones C D et al., "The New CMS Event Data Model and Framework", Proc. CHEP 2006 (Mumbai, India,13-17 February 2006)

[3] Brun R and Rademakers F, "ROOT - An Object Oriented Data Analysis Framework", Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, Nucl. Inst. & Meth. in Phys. Res. A 389 (1997) 81-86. See also http://root.cern.ch/.
[4] M. Tadel, "EVE - Event Visualization Environment of the ROOT framework", PoS ACAT (2008) 103.
[5] L Lista, C D Jones, G Petrucciani, "Expression and cut parser for CMS event data", Proc. CHEP 2009