FERMILAB-CONF-11-877-CD

International Conference on Computing in High Energy and Nuclear Physics (CHEP 2010) IOP Publishing
Journal of Physics: Conference Series **331** (2011) 072039 doi:10.1088/1742-6596/331/7/072039

Event Display for the Visualization of CMS Events

L A T Bauerdick[1], G Eulisse[1], C D Jones[1], D Kovalskyi[2], T McCauley[1], A Mrak Tadel[3], J Muelmenstaedt[3], I Osborne[1], M Tadel[3,4], Y Tu[3], A Yagil[3]

[1] Fermi National Accelerator Laboratory, USA
[2] University of California, Santa Barbara, USA
[3] University of California, San Diego, USA
[4] CERN, Switzerland

E-mail: fireworks@cern.ch

Abstract. During the last year the CMS experiment engaged in consolidation of its existing event display programs. The core of the new system is based on the Fireworks event display program which was by-design directly integrated with the CMS Event Data Model (EDM) and the light version of the software framework (FWLite). The Event Visualization Environment (EVE) of the ROOT framework is used to manage a consistent set of 3D and 2D views, selection, user-feedback and user-interaction with the graphics windows; several EVE components were developed by CMS in collaboration with the ROOT project. In event display operation simple plugins are registered into the system to perform conversion from EDM collections into their visual representations which are then managed by the application. Full event navigation and filtering as well as collection-level filtering is supported. The same data-extraction principle can also be applied when Fireworks will eventually operate as a service within the full software framework.

1. Introduction

Event displays are essential tools for any high-energy physics experiment. They find their use in online and offline monitoring and viewing of detector operation, as aids in physics analysis, and in production of displays showcasing physics results.

As originally described in [1], Fireworks is the event display used by the CMS experiment. Fireworks is directly integrated with the CMS Event Data Model (EDM) [2, 3] and FWLite, the light version of the CMSSW (CMS software) framework. ROOT I/O is used for implementing the event store, with Reflex dictionaries provided for every stored object type [4]. In addition, the Event Visualization Environment (EVE) [5] of the ROOT framework is used to implement the GUI.

We describe here improvements to the core of Fireworks, its integration with the full CMSSW framework, use of geometry, and online operations.

2. Improvements of Fireworks core

2.1. Supported EDM objects

Fireworks now supports almost the full complement of objects in CMSSW suitable for visualization. This includes objects from the RAW data tier, so-called digis, which contain ADC signal information: an example are the signals for the cathode strip chamber strips. Also

now supported are rec hits objects, which are constructed from the digital signals: this includes for example ECAL rec hits which show the energy deposit in each calorimeter crystal. Simulated data is now supported as well, including generated particles, tracks, and hits.

2.2. Plugin system extensions
Fireworks uses a plugin system for registration of code fragments responsible for conversion of EDM collections into visual representations. Recent improvements to the infrastructure for plugin management and execution allow the same plugin to produce visual objects for arbitrary number of view-types. The visual objects can be shared between several views. This also allows for easy addition of new view types without adversely affecting memory or CPU usage and is the base for other new features (such as global energy scaling and cross-view highlighting).

2.3. Application configuration
Previous configuration of Fireworks relied on a ROOT macro format and required a restart of the program in order to implement changes. This configuration system was adapted so that reloading the configuration at runtime is now possible. In addition the configuration format itself was modified and is now stored in a human-readable XML format, while keeping compatibility with the old ROOT macro format. In order to minimize dependency on external tools an XML parser ($\sim$ 500 lines of code) was written.

2.4. Cross-view object highlight and feedback
With a large amount of visual objects presented in multiple views it is extremely important to provide fast and succinct feedback to a user inspecting an event. Fireworks provides two feedback methods triggered by hovering the mouse over an object. First, the object is highlighted (*i.e.* drawn with a surrounding halo) in all graphical views, and second, a tool-tip is displayed below the mouse cursor detailing the object provenance information (EDM collection, internal index, and other relevant information associated with the object, typically p_T, E_T, or E).

There are cases when a highlighted object represents more than one element in a collection. For example, calorimeter towers in projected views get summed up along η ($R-\varphi$ projection) or φ ($R-z$ projection). If such an object is selected, all corresponding objects in other views are highlighted. Figure 1 shows cross-view highlighting of calorimeter towers in $R-\varphi$ and 3D views.

2.5. Energy scales
When viewing a collection of events the maximum energy of event objects changes from event to event. As there are several objects that change their size depending on their energy, it is important to provide a consistent means for calculation of object size based on its associated energy. Fireworks provides three different modes. In *automatic scale mode* energy-to-size conversion is defined for each event so that the largest drawn object reaches a given surface of the detector (typically the inner radius of the muon system). This allows for best possible use of screen space, but can be misleading as an additional piece of information, the automatically computed scale, has to be processed by the user. In *fixed scale mode* the energy to size conversion factor is defined by the user. The last mode, *combined scale mode*, is a combination of the other two: the scale is fixed up to a given energy and auto-scaled above it. The energy-scaling parameters are defined globally for all views. However, they can be overridden for each individual view.

2.6. Performance optimizations
Fireworks can process and draw more than 10 proton-proton events per second. In preparation for heavy-ion data taking, which began on November 8, 2010 (see Figure 2) significant

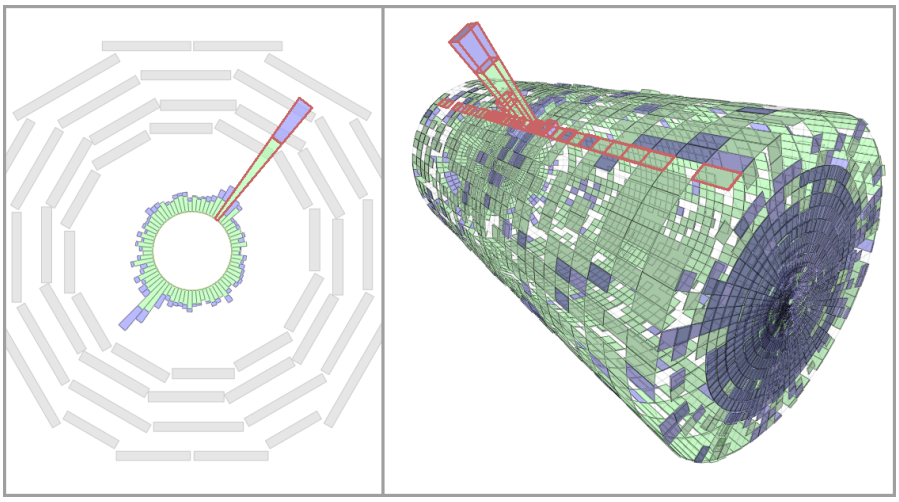

Figure 1. Screenshot demonstrating the cross-view selection of calo-towers: selecting one phi slice in $R-\varphi$ view (left) leads to selection of a whole row of towers in the 3D view (right).

performance optimizations were made by using simulated heavy-ion events as a benchmark (*e.g.* a central heavy-ion event typically contains 5000 tracks, 60k primary particles and 20k ECAL digits). Interestingly, loading of a new event spent more than 90% of the time in destruction of old visual representations. Several changes aimed towards a better reuse of existing objects lead to a 100-fold speed-up of event processing.

Destruction of collections had to be optimized as well: removing a large EDM collection from event display could take up to several minutes. A dedicated method for fast recursive destruction of objects was implemented in TEveElement class (the base-class of ROOT-EVE). Compared to standard TEveElement recursive destroy, this method skips reference checks and presumes all sub-elements are enclosed in one simple hierarchical tree. When this method is used in Fireworks removal of collections becomes instantaneous.

3. Full-framework integration

Standard Fireworks distribution and usage is stand-alone: the user downloads a zipped tar file, unzips, and runs on a local machine. Alternatively, the software is also distributed as a package in CMSSW, allowing for running of the display on any machine upon which CMSSW is installed. However, in CMSSW the usage was the same: access only to the RECO and AOD data tiers, no access to schedule information (*i.e.* paths, modules, and parameter sets), and no interaction with the framework itself.

A recent effort to integrate Fireworks in the CMSSW framework now allows for additional functionality and usage beyond stand-alone mode. A specialized looper interface has been implemented that allows for the event display to run with a standard CMSSW python configuration file as its input, akin to a standard cmsRun job. In this mode the event display has access to Run and Event information, access to all data tiers, the ability to visualize the schedule information, and the ability to modify module parameters (path and module information is immutable).

The last two features are implemented in a separate GUI window. Full hierarchical path, module, and module parameter set information is displayed in a collapsible table view, an example of which can be seen in Figure 3. Additional features include:

- a search dialog for path, module, and parameter names

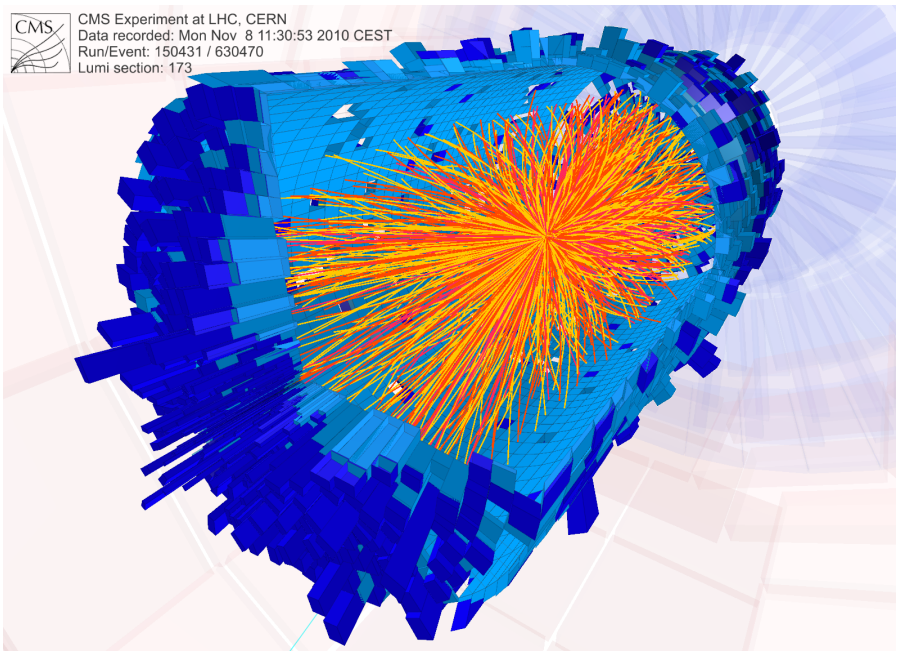

Figure 2. One of the first heavy-ion collisions in CMS as displayed by Fireworks.

- a color-coded view of the paths and modules, indicating which paths "passed" or "failed" and which module (in most cases a filter module) made the "decision"
- in-line editing of parameter set values

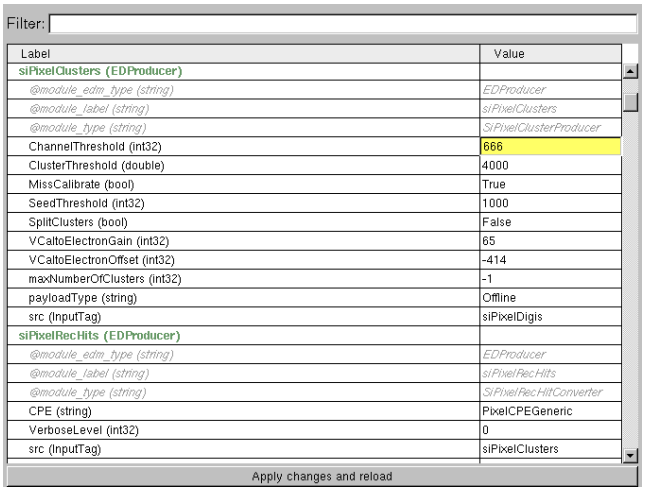

Figure 3. Screenshot of the so-called Path GUI indicating a search dialoque, modules in the paths, and editable parameters.

Besides being a useful tool for visualization of a full CMSSW sequence, this interface now allows for CMSSW to be run in an interactive mode via the event display. One can, for example, begin a reconstruction on input data, visualize the resulting event, modify a cut interactively, and view the results on the display.

As a debug aid, it is also possible to use ROOT-EVE directly. Developers can create EVE objects from within their modules and register them for display via a simple framework service (EveService) providing a thin layer over EVE object registration API and implementing a simple GUI for event navigation and user-feedback. It is also possible to register objects of a single event in a step-by-step mode, allowing the developer to inspect intermediate results of a complex computation.

4. Usage of geometry

The stand-alone distribution of Fireworks does not have access to the CMSSW EventSetup object, which provides access to detailed local coordinates for visualizable objects as well as transformations from local to global coordinates. Instead, a ROOT file containing this information is distributed with a release. This file contains the information (extracted using an analyzer in a CMSSW job) needed from EventSetup to visualize the data objects and geometry. A FWGeometry object provides the functionality and an interface to allow for, given a detector unit: local (in the unit frame) to global transformations and determination of shape and topological information. Additional parameters that are not associated to a given detector unit but are needed for visualization of objects (*e.g.* strip angles and spacing in the cathode strip chambers) are also extracted and stored in the ROOT file. In full-framework mode the event display utilizes the same interface but loads the geometry from a transient record.

In both stand-alone and full-framework mode the event display is capable of handling any possible changes in the geometry from for example, changes in alignment conditions. In stand-alone mode a new geometry is simply generated, reflecting the new conditions, while in full-framework mode the transient record from EventSetup will already reflect the changes.

With each new generation of geometry a validation suite is run. The validation confirms that the information extracted from the CMSSW geometry and written to the file is the same. It also aids in determining what information is needed in the FWGeometry in order to properly draw physics and geometry objects.

5. On-line operations

Fireworks event displays have been deployed for on-line operations in both the CMS control room in Cessy and in the CMS Centre at CERN in Meyrin. Centrally installed services provide the event displays with promptly reconstructed data and png snapshots from display stations in the CMS control room are produced and published on an online server [1]. The event displays run in auto-play mode and the time delay between events can be changed interactively. Locally installed scripts and cron jobs assure automatic recovery of the displays, clean up of the ROOT data files and png images, and automatic e-mail notification in case of problems.

The shift crew located in the control room can interact with the display, and can for example pause it, change visualization cuts and filters, add new collections to be displayed, and scan previous events. The event display configuration is restored after its restart.

6. Conclusion

The past year has seen the first p-p collisions at $\sqrt{s} = 7$ TeV and heavy ion collisions (see Figure 2) at $\sqrt{s} = 2.76$ TeV at the LHC. Event displays have proven to be essential tools for on-line monitoring of detector operation, aiding in physics analysis, and communicating results to the public during this exciting time. In all of these tasks the Fireworks event display program was used with success by the CMS collaboration. We have described here the design and new features that have made this possible.

[1] see for example `http://cmsdoc.cern.ch/cmscc/cmstv/cmstv.jsp?channel=12`

Development of Fireworks core is now complete. Future work will focus on extensions required for expert use and, in collaboration with the ROOT project, on improvements of the GUI and graphics systems.

7. Acknowledgments

We gratefully ackowledge the support of our CMS collaborators as well as that of the US DOE and NSF.

References

[1] Kovalskyi D *et al.*, 2010 J. Phys.: Conf Ser. **219** 032014
[2] CMS Collaboration, 2007 J. Phys. G **34** 995
[3] Jones C D *et al.*, 2006 Proc. CHEP
[4] Antcheva I *et al.*, 2009 Comput. Phys. Commun. **180** 2499
[5] Tadel M, 2008 PoS **ACAT08** 103