Computer Physics Communications 183 (2012) 245–250

Contents lists available at SciVerse ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

Three-dimensional event visualization for the ATLAS calorimeter

Luciano M. de Andrade Filho [a,*], José Manoel de Seixas [b], Roberto Agostino Vitillo [c], Brian Thomas Martin [d]

[a] Federal University of Juiz de Fora, Department of Electrical Circuits, Brazil
[b] Federal University of Rio de Janeiro, Signal Processing Laboratory, Brazil
[c] National Institute of Nuclear Physics, Pisa, Italy
[d] Michigan State University, MI, United States

ARTICLE INFO

Article history:
Received 10 August 2010
Received in revised form 6 September 2011
Accepted 15 September 2011
Available online 22 September 2011

Keywords:
Event display
Cosmic rays
Calorimetry
ATLAS detector
ROOT framework

ABSTRACT

The ATLAS detector was commissioned with cosmic rays. For such commissioning, a number of software tools has been developed to support data analysis. Among ATLAS sub-detectors, commissioning the calorimeter system demanded a considerable effort due to its segmentation into seven detection layers, which produce more than a hundred thousand readout channels. Tasks like performance evaluation of the calorimeter, calibration and handling noisy or dead channels benefit a lot from cosmic muon track visualization, which facilitates the identification of the activated cells in the calorimeter. The coherence of the reconstructed data can be visually checked and potential problems can be detected in a easier way. This work presents a 3D visualization tool for the ATLAS calorimeter system, the CaloGeoView. The tool was built using ROOT Framework, which provides a smooth integration with analyses currently performed by the ATLAS community. The CaloGeoView structure and some applications with reconstructed data are presented. Due to its 3D graphical interface, the CaloGeoView has been providing a simple and intuitive way to analyze results of event reconstruction.

1. Introduction

LHC [1] will collide protons with 14 TeV in the centre of mass at a 25 nanosecond period. As one of the main LHC detectors, ATLAS [2] will explore the fundamental nature of matter and the basic forces that shape our universe. The detector has a total length of 42 m and a radius of 11 m (see Fig. 1). Details about ATLAS sub-detectors and integration can be seen in [3–6].

Due to the extremely high interaction rate and the huge background noise generated by LHC, ATLAS requires the design of a sophisticated on-line triggering system, for which calorimeters provide an essential information. In ATLAS, the triggering system operation is split into three cascaded levels and should reduce dramatically the background noise rate, providing an output frequency of the order of 100 Hz [7].

Synchronization of all the ATLAS sub-detectors with the trigger system can be considered as a real challenge, when complexity of the detector system, the huge number of channels and the variety of technologies used in the different parts are taken into account.

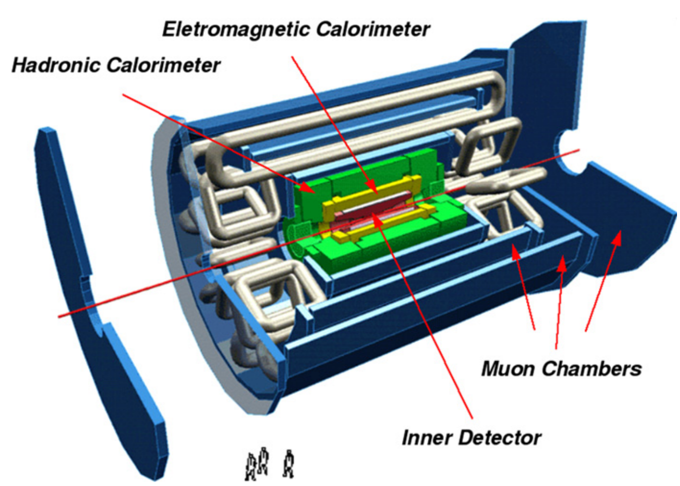

Fig. 1. The ATLAS detector.

In order to achieve this goal, several commissioning procedures have been put into place. For commissioning the whole system with physical events, before the LHC start, exhaustive tests with cosmic rays have been performed since 2006 [8–10]. These tests have allowed to check the integrity of each sub-detector, identifying local problems, and testing the intercommunication of the sub-systems as well.

* Corresponding author.
E-mail address: luciano.andrade@engenharia.ufjf.br (L. M. de Andrade Filho).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2011.09.004

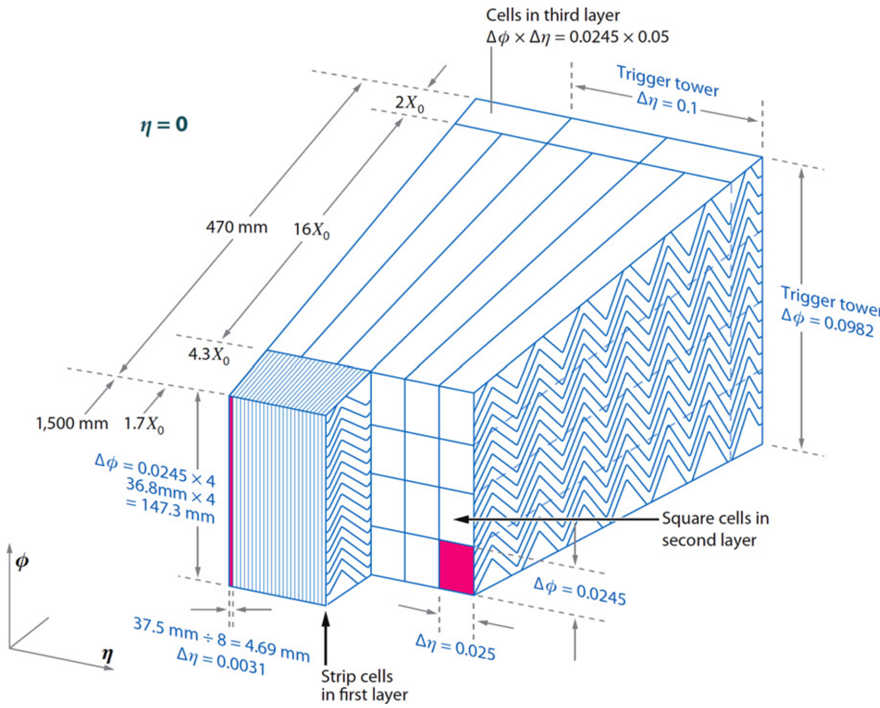

Fig. 2. Electromagnetic calorimeter segmentation.

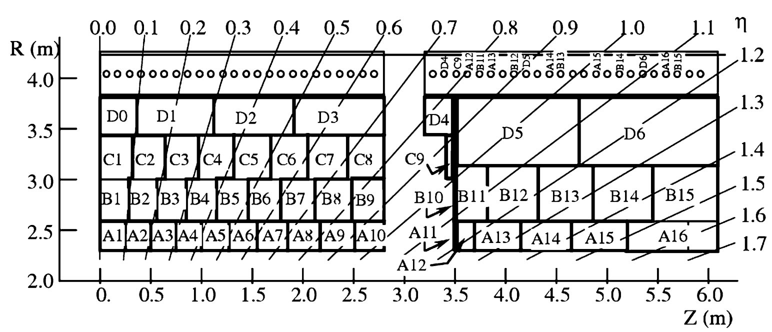

Fig. 3. TileCal longitudinal segmentation.

Different algorithms are executed to analyze cosmic runs. One of the most efficient ways to identify local problems is the visual inspection of the reconstructed data, using an event display tool. This work presents a software tool for 3D visualization of the calorimeter cells, the CaloGeoView (CGV). While standard event displays in ATLAS – like Atlantis [11], Persint [12] and VP1 [13] – were developed to visualize as much information as possible in LHC proton–proton collisions, the CGV provides specific calorimeter cell level information that can be very usefull during the comissioning phase with cosmic rays, like actual 3D cell geometry, energy-to-color palette and track visualization of cosmic muons based on calorimeter information.

The CGV has been developed using the ROOT framework [14] and its embedded geometry package [15]. Since the framework for data reconstruction in ATLAS, Athena [16], generates specific files to be loaded in the ROOT environment, the integration between reconstructed data and the CaloGeoView can be done smoothly. Due to its easy and intuitive way to analyze and identify problems, this tool has become part of the Online Hadronic Calorimeter Monitoring System and was operational during ATLAS commissioning phase. Besides, for offline analysis, CaloGeoView can be used to check interesting LHC proton–proton collision events as well.

This paper is organized as it follows. Details about implementation are given in Section 2. Section 3 presents the graphical interface and features of this 3D event display tool. Instructions for installing and running are given in Section 4. Then, Section 5 presents some practical examples. Section 6 addresses issues about performance. Finally, conclusions are derived in Section 7.

2. Architecture

The CaloGeoView is based on ROOT framework. The ROOT system is an Object Oriented (OO) framework for large scale data analysis [14]. Written in C++, ROOT contains an efficient hierarchical OO database, a C++ interpreter, advanced statistical analysis (multi dimensional histogramming, fitting, minimization, cluster finding algorithms) and visualization tools. The user interacts with ROOT via a graphical user interface, the command line or batch scripts. The scripting language is C++ (using the interpreter) and large scripts can be compiled and dynamically linked in. The OO database design has been optimized for parallel access (reading as well as writing) by multiple processes.

2.1. The geometry package

Figs. 2 and 3 show details about ATLAS calorimeter segmentations for the electromagnetic (e.m.) and the hadronic barrel part (Tilecal) respectively. The e.m. calorimeter amounts to more than

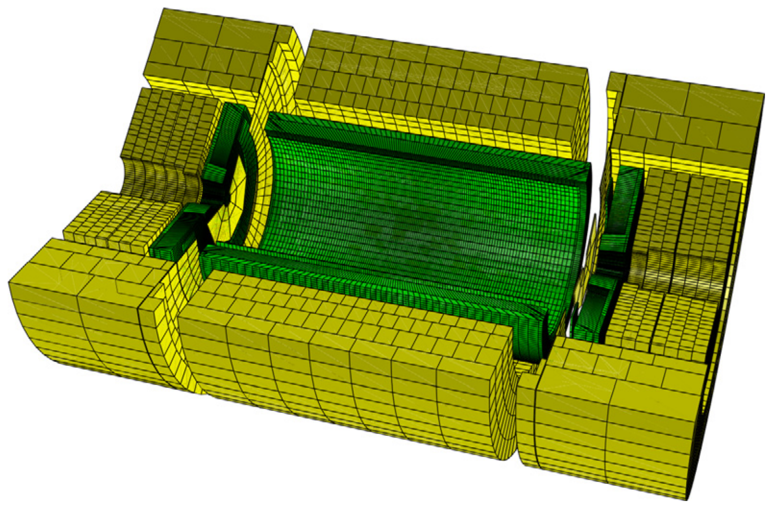

Fig. 4. ATLAS calorimeter cell segmentation implemented with the ROOT geometry package. (For interpretation of the references to color in the text, the reader is referred to the web version of this article.)

```
void         cgvInit                (TGCompositeFrame*); // CaloGeoView Initialization
void         cgvSetOnline                     (bool on); // Start online event loading (to Control Room)
int          cgvGetRunNumber                         (); // Extract Run Number
int          cgvGetEvNumber                          (); // Extract Event Number
const char*  cgvGetEvTime                            (); // Extract Event Time
const char*  cgvGetTriggerType                       (); // Extract Trigger Type
void         cgvSetTileThreshMin            (float val); // Set Minimal Threshold to visualize cell in TileCal
void         cgvSetTileThreshMax            (float val); // Set Maximal Threshold to visualize cell in TileCal
void         cgvSetPaletteTile             (int palette); // Set Color palette in TileCal
void         cgvSetTileActivated       (bool activated); // Set TileCal visibility
void         cgvEBVisible   (bool visible, bool Aside); // Set Extended barrel visibility
void         cgvSetBounds              (int ba, int eb); // Set kind of Calorimeter boundery
void         cgvSetBoundTransparency      (char transp); // Set boundery transparency level
void         cgvSetBoundColor          (Pixel_t color); // Set boundery color
void         cgvSetGraphicStyle             (int style); // Set boundery style (wireframe, solid, flat)
void         cgvSetBckGroundColor      (Pixel_t color); // Set Background color
void         cgvAnimate                       (bool ok); // Redraw the scene
void         cgvSetShowTracks               (bool show); // Set track visibility
```

Fig. 5. A small part of commands that can be called to interact with the CGV.

170,000 readout channels. The hadronic end-cap (not shown separately) has a structure similar to the e.m. calorimeter, but coarser granularity and about 6000 readout channels. Finally, the hadronic barrel calorimeter has about 5000 cells, but unlike other calorimeters in ATLAS it has two readout channels per cell. The dimensions are given in terms of ATLAS coordinates (pseudorapidity and azimuthal angle).

The presented visualization tool uses the ROOT Geometry Package [15], which is a package for building, browsing, navigating and visualizing detector geometries. The first step is to build the detector. This is done using primitive shapes like boxes, spheres, trapezoids, etc, and positioning each one according to the detector geometry. For the CGV, the detector geometry is built running the *CaloBuild.C* macro as a separate step, which produces a .root file with the ATLAS calorimeter geometry. Once the geometry is assembled, the user can run the visualization tool over this .root file. Fig. 4 shows the whole ATLAS calorimeter cell geometry assembled with this package, where one can identify the hadronic (in yellow) and electromagnetic (in green) calorimeters. More than 170,000 primitive shapes were needed to build the ATLAS calorimeter segmentation.

After loading the geometry file, one can navigate in the geometry and change colors and visibility of each primitive shape (the calorimeter cells). This is done through a Geometry Manager, the TGeoManager ROOT class. Finally, the Visualization Manager (TGLViewer) is responsible to draw the 3D graphics on the screen. The TGLViewer uses a standard and powerful library for 3D graphical visualization, the OpenGL package [17]. The entire C++ code responsible to build the ATLAS Calorimeter geometry in ROOT is in the file *CaloBuild.C* on the ATLAS repository at CERN [18].

2.2. The graphical user interface

The Graphical User Interface (GUI) of CGV was built using the ROOT GUI Builder [19]. With this package, one can build custom windows, with buttons and text box to input user information and to control the operation of the program. Through the GUI, the user can choose the reconstructed event files, energy threshold, navigate through the events, etc. File *CaloGeoGui.C* in repository contains the complete source code of the GUI for this tool.

2.3. The control interface

The commands sent by the user trough the GUI should be translated into commands of the TGeoManager class. This is done by a layer of software called the Interface. Actually most of the specific code of this tool is inside this layer. The file called *CaloGeoInterface.C* contains the code necessary to execute all possible commands within the tool. The respective header file (*CaloGeoInterface.h*) has the list of commands one can perform to interact with the CGV. Those commands start with the letter “cgv” (from CaloGeoView). See Fig. 5 for a sample of commands. Those instructions can be called directly by the ROOT command line or using the GUI, which implements calls for these functions graphically.

Fig. 6 shows the CaloGeoView architecture, where one can identify two software layers. As said before, the outer layer, the GUI, is responsible for receiving commands and sending status to the user,

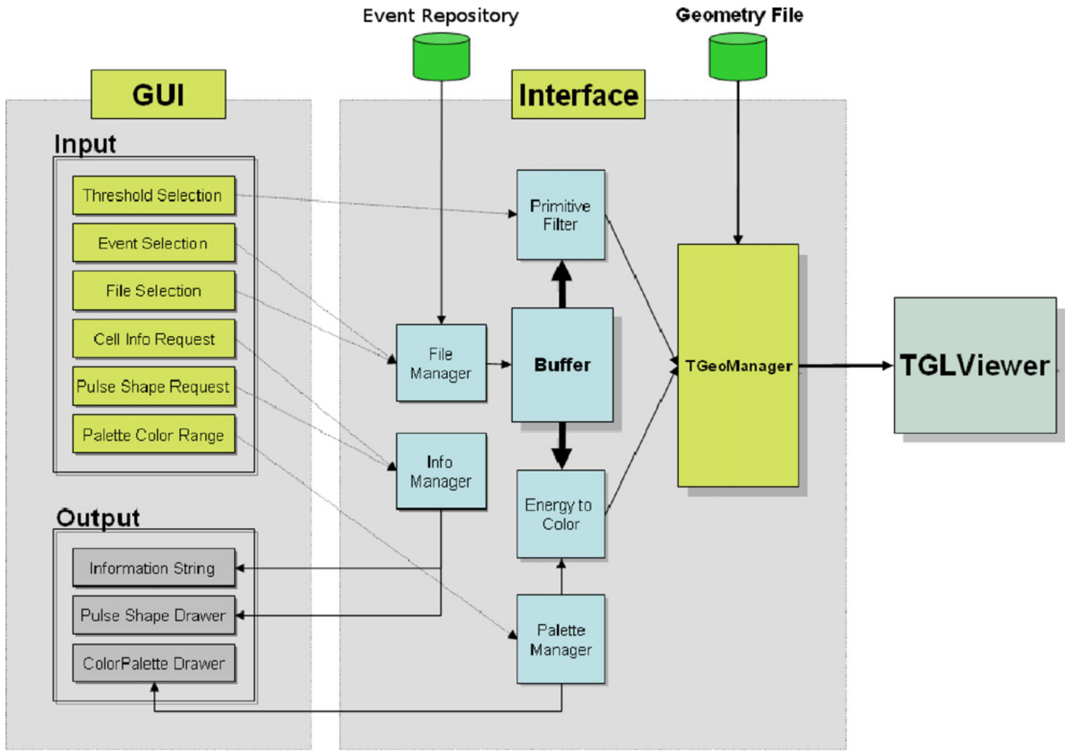

Fig. 6. Visualization tool architecture.

providing a user-friendly interface. An inner layer, the Interface, translates the user commands to TGeoManager commands. The geometry file is loaded by the TGeoManager as soon as the program starts. Information from a specific event is stored in a local *Buffer* for further accesses, increasing the tool speed response. A *Primitive Filter* block decides whether a specific calorimeter cell (a primitive shape in the geometry) will be shown on the screen, depending on the threshold. The color of the primitives is controlled by the *Energy to Color* component, according to the cell energy and the color palette used. The communication between the TGeoManager and the TGLViewer is straightforward and the primitive shapes are drawn on the screen.

3. Features and graphical interface

Fig. 7 shows the Graphical User Interface (GUI) for the CGV. Three windows can be observed. The left-back window is the standard viewer for 3D objects in ROOT (ROOT's GL viewer). A cosmic ray event is displayed in the center. In this central window, one can identify the Calorimeter structure, which is shown in a transparent gray scale, the activated cells with colors depending on an energy scale and the corresponding track of the reconstructed cosmic muon. The right most window offers user controls to open, access and visualize the information for the reconstructed data. Among the information available, one can see the ADC counts for the selected cell, as it is shown in the left-front window. The line connecting the ADC counts is a standard interpolation featuring in ROOT.

The main features of the CGV are:

- Color Palette adjustment. The colors of the activated cells depend on the energy deposited in the cell. A bright to dark green palette is assigned to the EM calorimeter and a yellow to dark red palette is assigned to the hadronic calorimeter, with 256 color levels. The full energy scale may be set by the user and can accommodate the full energy range.
- Threshold by cell energy. The whole calorimeter has about 200,000 cells; in order to display only the desired cells for a given application, a cut value in MeV can be applied.
- Cell selection with mouse. Desired information such as energy, position and PMT channels (TileCal only) for a specific cell is obtained by selecting the cell with the mouse.
- Rotate and zoom. Basic visualization commands, such as rotation, translation and zooming, can be achieved with the mouse; the desired transformation is displayed on the screen almost instantaneously.
- Show ADC counts. When pulse shape samples are available in the reconstruction file, this tool can show graphics with this information for the selected cell.

4. Package installation and running

In order to run the tool, one should have a ROOT version greater then 5.12 with OpenGL support. The source code is in ATLAS repository at CERN [18]. Before running the first time, the file containing the calorimeter geometry should be created. This is done typing

root [0] .x CaloBuild.C+

in the ROOT command line. This command will create the file *CaloGeometry.root*. Once this file is in the directory, one can start ROOT typing

$ root -l run.c

This command will initiate ROOT and start the CaloGeoView automatically.

5. Application examples

This section presents some applications of this visualization tool. Several commissioning aspects, like simulation validation, online monitoring and offline reconstruction were tested using the visualization facilities provided by this tool.

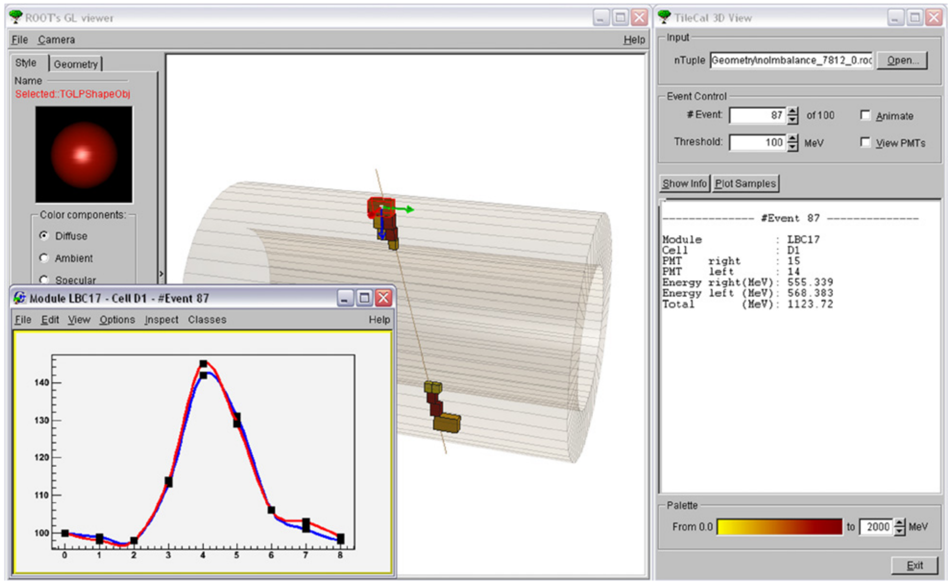

Fig. 7. The Graphical interface of the visualization tool.

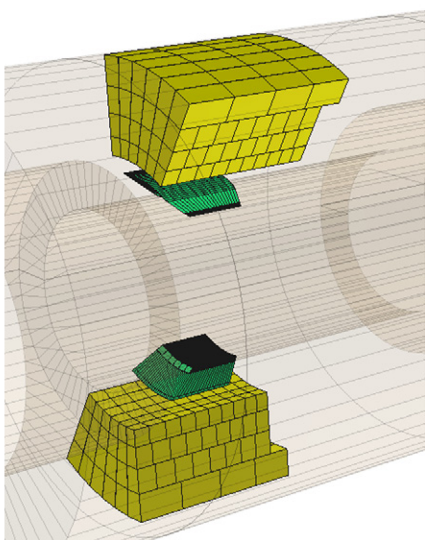

Fig. 8. High level trigger RoI seen with CaloGeoView.

5.1. High level trigger RoI visualization

Fig. 8 shows the CaloGeoView graphical interface where one can identify a level-two trigger Region of Interest (RoI) simulation. This figure represents a typical LHC collision simulation resulting in two electrons. Thanks to this kind of visualization, a reconstruction problem (Fig. 9) related to the RoI border cells, in some events with a higher pseudorapidity angle, where outer calorimeter cells are used, has been found. One can see a smooth cell energy decrease from the center cell (most energetic cell) after fixing the problem. This kind of problem is hard to detect without 3D visualization.

5.2. Level-two trigger feature extraction algorithm

The job of the Level-Two Trigger's calorimeter baseline algorithm, named T2Calo [20], consists of evaluating a more precise interaction point using the available calorimeter granularity and, based on that, extracting cluster features that may lead to a reasonable discrimination accuracy. This kind of algorithm is followed by a hypothesis making algorithm, which takes the extracted features from the target cluster and, with a simple set of cuts, is able to distinguish interesting objects with better accuracy than Level-one Trigger.

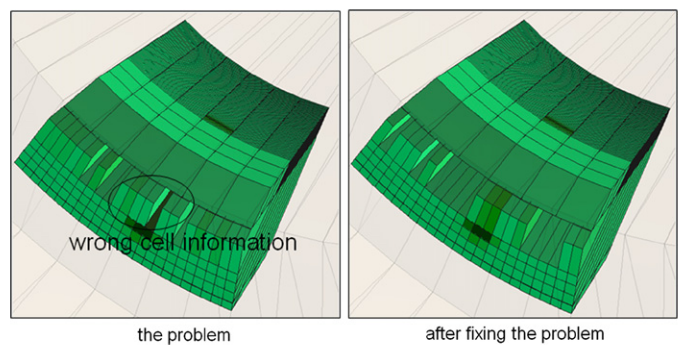

Fig. 9. Problem in RoI border cells detected with CaloGeoView.

A *Ring-Like* topological strategy has been proposed as an alternative for RoI feature extraction [21]. It consists of forming, for each calorimeter layer, concentric rings around the most energetic cell of the RoI. Then, the energy deposited in cells that belong to the same ring is added up. The algorithm implementation in the Athena framework has been evaluated and tuned with the CaloGeoView, using simulation, where different ring layers can be seen applying different colors.

5.3. Cosmic ray simulation in ATLAS

Fig. 10 shows an example of a cosmic ray simulation event without noise. One can see the perfect match between the truth track, generated by the Monte Carlo simulation, and the activated cells for both the hadronic and electromagnetic calorimeters. Both the track truth and activated cell list are read from the same reconstruction file. This kind of visualization has become important to attest and to get confidence with the simulation of cosmic rays in ATLAS.

6. Performance

In terms of performance, the computational cost strongly depends on the chosen energy threshold, since this value defines how

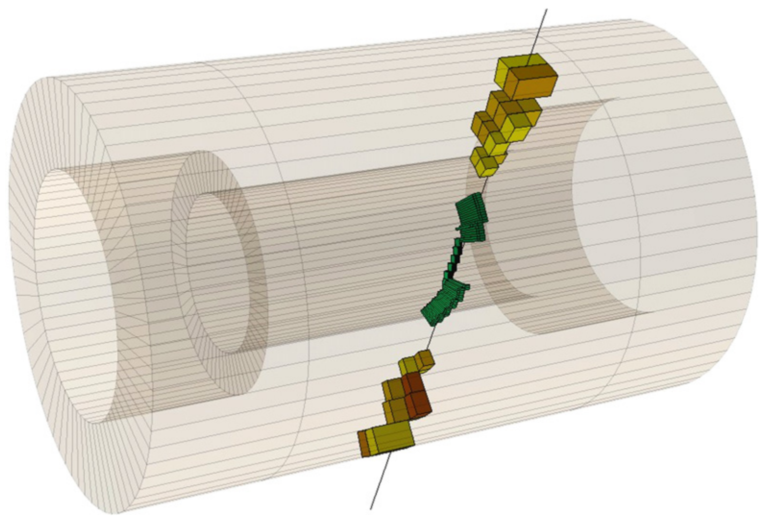

Fig. 10. Cosmic simulation seen with CaloGeoView.

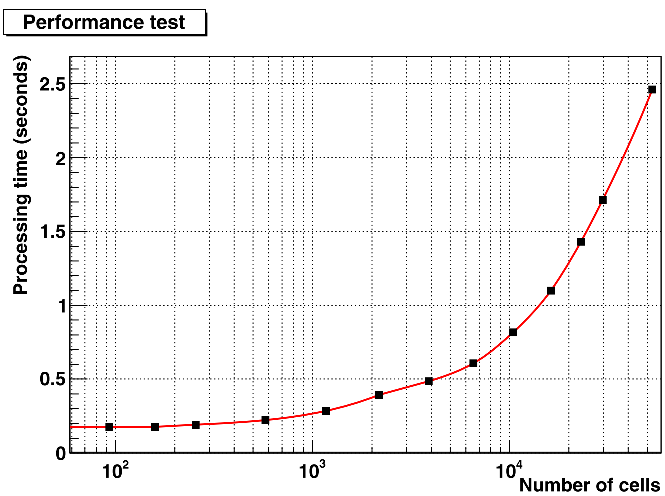

Fig. 11. Processing time versus number of drawn cells (Processor Intel Centrino – 1.7 GHz).

many cells should be visualized. In other words, the time to read an entire event, assign the correct primitive colors and draw the desired cells grows with the number of plotted cells. Fig. 11 shows how the processing time varies with the number of drawn cells, performed on a regular Personal Computer (Processor Intel Centrino – 1.7 GHz). One can notice that up to a thousand of cells, the processing time is lower than 300 milliseconds. Most of the interesting events in the commissioning phase was under this limit. In fact, event visualization becomes useless for a big amount of drawn cells.

7. Conclusions

In the whole chain of an ATLAS event data processing, one of the final analyses is the reconstructed event visualization. With a simple visual inspection, one can check the agreement of all subsequent data processing steps, starting from trigger acceptance, passing trough the readout of all detector channels, up to decodification and reconstruction of the event in a readable way.

The presented 3D visualization tool has been extensively used in the commissioning phase of the ATLAS calorimeter system. Different kind of problems, either detected from real data acquisition with cosmics or simulated cosmic rays and L2 RoI, have been discovered thanks to the CaloGeoView.

Acknowledgements

The authors would like to thank CAPES, CNPq, FINEP and FAPERJ (Brazil), CERN (Switzerland) and European Union for their support to this work. We would also like to thank our colleagues from the ATLAS TileCal, LAr and Trigger Collaborations for providing the data and for fruitful discussions concerning this work.

References

[1] L. Evans, P. Bryant (Eds.), LHC Machine, JINST 3 S08001, 2008.
[2] ATLAS Collaboration, The ATLAS experiment at the CERN large hadron collider, JINST 3 (2008) S08003.
[3] ATLAS Collaboration, ATLAS Inner Detector, Technical Design Report, 1997.
[4] P. Puzo, ATLAS calorimetry, Nuclear Instruments and Methods in Physics Research A 494 (2002) 340–345.
[5] R. Wigmans, Calorimetry. Energy Measurement in Particle Physics, Oxford University Press, 2000.
[6] ATLAS Collaboration, Muon Spectrometer, Technical Design Report, 1997.
[7] S. Falciano, The ATLAS Level-l and Level-2 trigger, Nuclear Instruments and Methods in Physics Research A 384 (1996) 136–142.
[8] ATLAS Collaboration, Readiness of the ATLAS liquid argon calorimeter for LHC collisions, EPJC 70 (2010) 723–753.
[9] ATLAS Collaboration, Readiness of the ATLAS tile calorimeter for LHC collisions, EPJC 70 (2010) 1193–1236.
[10] ATLAS Collaboration, Commissioning of the ATLAS muon spectrometer with cosmic rays, EPJC 70 (2010) 875–916.
[11] http://www.hep.ucl.ac.uk/atlas/atlantis, accessed on March 29th, 2011.
[12] D. Pomarede, M. Virchaux, The Persint visualization program for the ATLAS experiment, in: Computing in High Energy and Nuclear Physics, 24–28 March 2003, La Jolla, California.
[13] http://cern.ch/atlas-vp1, accessed on March 29th, 2011.
[14] R. Brun, F. Rademakers, ROOT – an object oriented data analysis framework, in: Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, Nuclear Instruments and Methods in Physics Research A 389 (1997) 81–86.
[15] R. Brun, A. Gheata, M. Gheata, The ROOT geometry package, Nuclear Instruments and Methods in Physics Research A 502 (2003) 676–680.
[16] K.S. Cranmer, The ATLAS analysis architecture, Nuclear Physics B (Proc. Suppl.) 126–130 (2008).
[17] http://www.opengl.org, accessed on March 29th, 2011.
[18] https://svnweb.cern.ch/trac/atlasoff/browser/Calorimeter/CaloGeoViewer, accessed on June 27th, 2011.
[19] I. Antcheva, R. Brun, C. Hof, F. Rademakers, The graphics editor in ROOT, Nuclear Instruments and Methods in Physics Research A 559 (2006) 17–21.
[20] ATLAS Collaboration, ATLAS high-level trigger, DAQ and DCS Technical Design Report, 2003.
[21] A. dos Anjos, R.C. Torres, J.M. Seixas, B.C. Ferreira, T.C. Xavier, Neural triggering system operating on high resolution calorimetry information, Nuclear Instruments and Methods in Physics Research A 559 (2006) 134–138.