DELPHES 3: A modular framework for fast-simulation of generic collider experiments

Michele Selvaggi

Centre for Cosmology, Particle Physics and Phenomenology (CP3),
Université Catholique de Louvain,
Chemin du Cyclotron 2, B-1348 Louvain-la-Neuve, Belgium

E-mail: michele.selvaggi@uclouvain.be

Abstract. The new version of the DELPHES C++ fast-simulation framework is presented. The tool is written in C++ and is interfaced with the most common Monte Carlo file formats (LHEF, HepMC, STDHEP). Its purpose is the simulation of a multipurpose detector response, which includes a track propagation system embedded in a magnetic field, electromagnetic and hadronic calorimeters, and a muon system. The new modular version allows to easily produce the collections that are needed for later analysis, from low level objects such as tracks and calorimeter deposits up to high level collections such as isolated electrons, jets, taus, and missing energy.

1. Introduction

Sophisticated detectors are designed in order to detect and precisely measure particles originating from high energy collisions. For most phenomenological studies, a complete detector description is often not needed and a simplified approach based on the parametrisation of the detector response is in general good enough. In 2009, the DELPHES framework was designed to achieve such a goal [1].

DELPHES takes as input the most common event generator output data-formats and performs a fast and realistic simulation of a general purpose collider detector. To do so, long-lived particles emerging from the hard scattering are propagated to the calorimeters within a uniform magnetic field. Photons and charged leptons final observables are computed by smearing the initial momenta according to the resolution of the relevant sub-detectors. High-level reconstructed quantities such as jets and missing energy can be computed either starting from simple calorimeter deposits, or with the so-called particle-flow algorithm.

With respect to its predecessor [1], the present DELPHES version includes a more accurate particle-flow description allowing to combine and optimally use the information of all sub-detectors. This approach is particularly suitable to the treatment of pile-up, which has also been included in DELPHES 3.0. From a technical perspective, the code structure is now fully modular, providing a greater flexibility to the user. Finally, an event display has been added to the DELPHES package.

2. Software implementation

2.1. Data-Flow

DELPHES is a modular framework written in C++ and is based on the ROOT analysis framework [2]. It accesses data from different file formats HEPMC [3], STDHEP [4] and the LesHouches event format (LHEF) [5]). Event files coming from external Monte-Carlo generators are first processed by a Reader. The Reader converts stable particles into a collection of universal objects. This collection is then processed by a series of modules. The jet clustering procedure is performed by means of the FastJet package [6]. Finally, DELPHES allows to store and analyze events in a ROOT tree format [2]. The modular system allows to configure and schedule modules via a configuration file, add modules, change data-flow, alter output information.

2.2. Technical Performance

The main motivation for a tool like DELPHES is to minimize the resources needed on top of those used for event generation: small memory footprint, efficient usage of CPU and small file size.

Figure 1 illustrates how well DELPHES achieves these goals. Memory usage does not exceed a few hundreds of megabytes and remains constant after the initial memory allocation. Processing time can be as low as a few milliseconds and is expected to follow the scaling law of the underlying jet reconstruction algorithm.

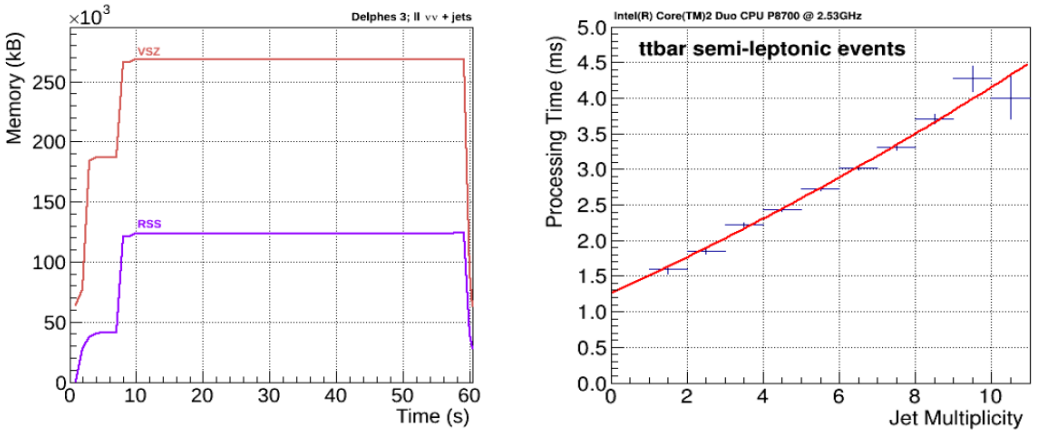

Figure 1. Left: memory usage as function of time after the program start. Right: processing time per event as function of jet multiplicity on inclusive $t\bar{t}$ events.

3. New features

3.1. Particle-Flow

The philosophy of the particle-flow approach is to maximally make use of the information provided by the various sub-detectors for reconstructing the event. This modus operandi is adopted by several collaborations (see for example [7]) but intrinsically depends on the specificity of the experimental device. In DELPHES, we opted for a simplified approach based on the tracking system and the calorimeters for implementing the particle-flow event reconstruction. If the momentum resolution of the tracking system is higher than the energy resolution of calorimeters, it can be convenient to use the tracking information within the tracker acceptance for estimating the charged particles momenta. The particle-flow algorithm produces two collections of 4-vectors — particle-flow tracks and particle-flow towers — that can serve later as input for reconstructing jets and missing transverse energy with a higher resolution.

The comparison of the performance achieved with DELPHES fast-simulation tool against the CMS experiment full-simulation [7] is shown in Figure 2. Jets were clustered with the help of the anti-k_T algorithm [8] and the events were generated with MADGRAPH5 [9]. The comparison is shown for quantities that have been reconstructed using calorimeter information only, and using the particle-flow approach. The resolutions obtained with DELPHES and the CMS full-simulation show a good agreement.

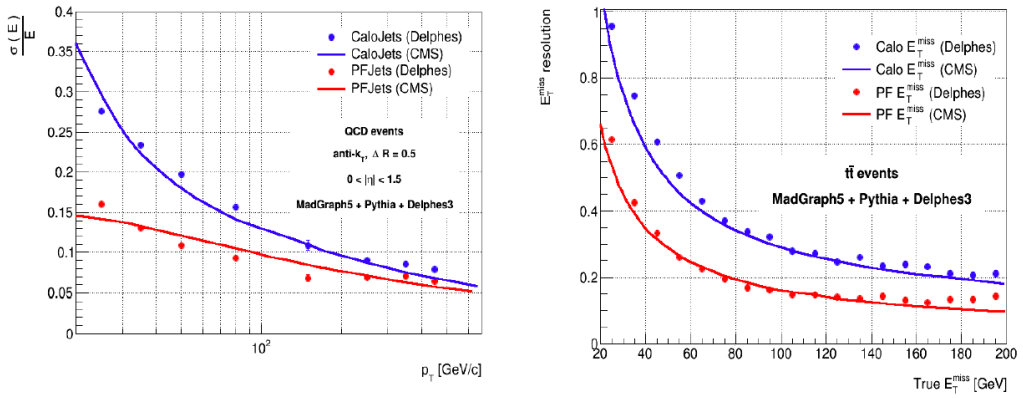

Figure 2. Comparison of the performance achieved with DELPHES fast-simulation tool against the CMS experiment full-simulation. Left: Jet energy resolution as a function of the jet p_T in QCD events. Right: Missing transverse energy resolution as a function a of the true event missing transverse energy in top-pair events.

3.2. Pile-up simulation

At the LHC, several collisions per bunch-crossing will occur in high luminosity conditions. Such additional *pile-up events*, can have the effect of deteriorating the detector response. In DELPHES, pile-up interactions are extracted from a pre-generated low-Q^2 QCD sample. These minimum-bias interactions are randomly placed along the beam axis according to some longitudinal spread that can be set by the user (see Figure 3). The actual number of pile-up interactions per bunch-crossing is randomly extracted from a Poisson distribution.

Pile-up directly affects the performance of jets, E_T^{miss} and isolation. In real experiments, pile-up interactions are identified by means of vertex reconstruction. If such interactions occur far enough from the hard interaction, a precise vertexing algorithm is able to detect them. Combining vertexing and tracking information allows to identify contaminating charged particles from pile-up. On the other hand, since neutral particles do not produce tracks, neutral pile-up contamination can only be estimated *on average*. In DELPHES, pile-up subtraction can be done in two steps:

3.2.1. Charged pile-up subtraction We assume that vertices corresponding to pile-up interactions occurring at a distance $|z| > Z_{vtx}$ can be reconstructed. The parameter Z_{vtx} is obviously related to the spatial vertex resolution of the detector. We assume that pile-up interactions occurring at $|z| < Z_{vtx}$ cannot be disentangled from those originating from the high-Q^2 process. Therefore every particle-flow track originating from such vertices cannot be subtracted from the event, while every particle-flow track originating from a vertex positioned at $|z| > Z_{vtx}$ can be removed from the event.

3.2.2. Residual pile-up subtraction Other techniques are needed in order to extract and remove residual contributions: these include particles that are too close to the hard interaction vertex to be identified as pile-up products with tracking information, charged particles that failed track reconstruction and neutral particles. In DELPHES we have opted for the Jet Area method [10, 11]. This approach, widely used in present collider experiments, allows to extract an average contamination density ρ on an event-by-event basis. The residual pile-up density ρ, can be used to correct observables that are sensitive to the residual contamination, mainly the jet energies and the isolation.

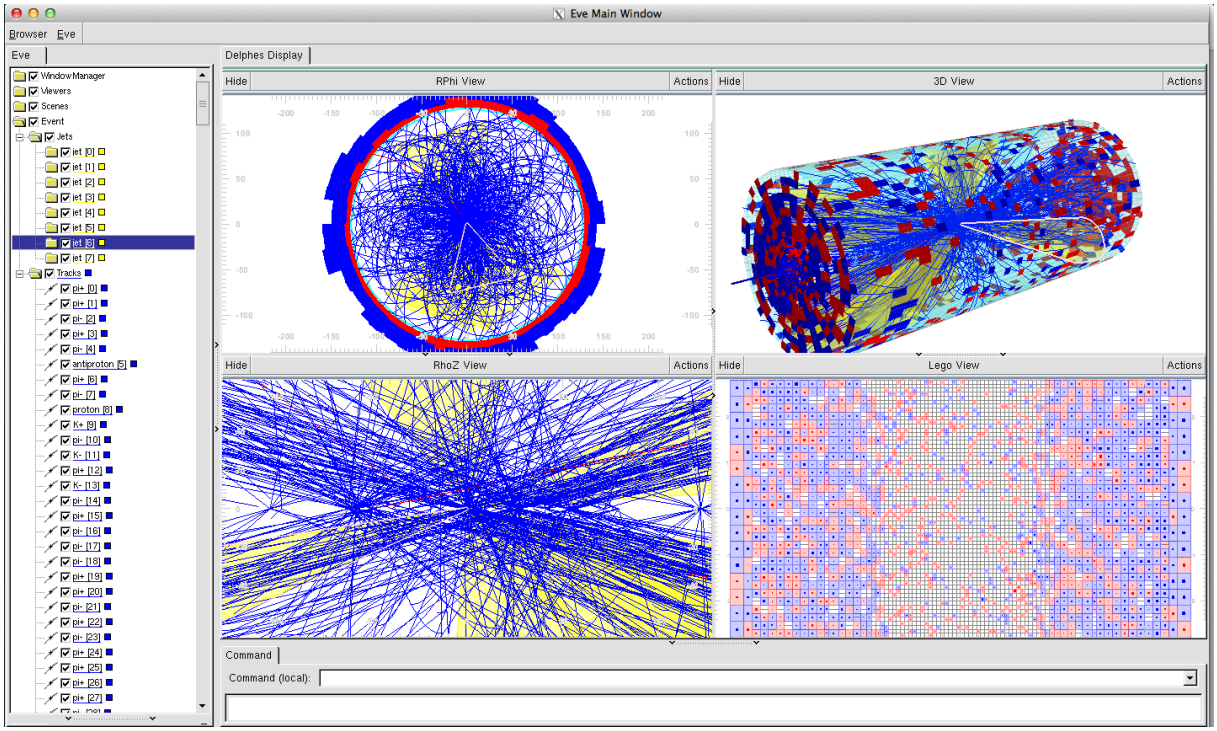

Figure 3. QCD event with 50 pile-up interactions shown with the DELPHES event display based on the ROOTEVE libraries [12]. Transverse view (top left), longitudinal view (bottom left), 3D view (top right), (η,ϕ) view (bottom right).

4. Conclusion

The version 3.0 of DELPHES, a framework designed to perform a fast and realistic simulation of a general purpose collider experiment, was presented. New features such as the particle-flow event reconstruction, and pile-up simulation have been included and validated.

Acknowledgments

First we wish to thank the ACAT 2013 workshop organizers for giving the opportunity to present the new version of DELPHES. We would like to thank Séverine Ovyn and Xavier Rouby for their invaluable contribution to the early DELPHES development. We wish to thank the DELPHES users for continuously providing feedback and deep insights. We acknowledge the support from the FNRS and the IAP Program, BELSPO VII/37. This work is partly supported by the IISN convention 4.4503.13.

References

[1] Ovyn S, Rouby X and Lemaitre V 2009 DELPHES, a framework for fast simulation of a generic collider experiment *Preprint* hep-ph/09032225

[2] Antcheva I *et al* 2009 A C++ framework for petabyte data storage, statistical analysis and visualization *Comput. Phys. Commun.* **180** 2499

[3] Dobbs M *et al* 2001 *Comput. Phys. Commun.* **134** 41

[4] Garren L and Lebrun P 2006 StdHep User Manual `http://cepa.fnal.gov/psm/stdhep/`

[5] Alwall J *et al* 2007 A Standard format for Les Houches event files *Comput. Phys. Commun.* **176** 300

[6] Cacciari M, Salam G P and Soyez G 2012 FastJet User Manual *Eur. Phys. J.* C **72** 1896

[7] CMS Collaboration 2009 Particle-Flow Event Reconstruction in CMS and Performance for Jets, Taus, and MET CMS-PAS-PFT-09-001

[8] Cacciari M, Salam G P and Soyez G 2008 The Anti-k(t) jet clustering algorithm *JHEP* **0804** 063

[9] Alwall J, Herquet M, Maltoni F, Mattelaer O and Stelzer T 2011 MadGraph 5 : Going Beyond *JHEP* **1106** 128

[10] Cacciari M, Salam G P and Soyez G 2008 The Catchment Area of Jets *JHEP* **0804** 005

[11] Cacciari M, Salam G P and Soyez G 2008 Pileup subtraction using jet areas *Phys. Lett.* B **659** 119

[12] Tadel M 2008 Raw-data display and visual reconstruction validation in ALICE *J. Phys. Conf. Ser.* **119** 032036