

CERN-LHCC-2005-018
ALICE TDR 012
15 June 2005

ALICE

Technical Design Report

of the

Computing

Cover design by CERN Desktop Publishing Service.
Visualisation of ALICE Experiment as simulated by AliRoot.

Printed at CERN
June 2005.

ISBN 92-9083-247-9

Contents

4	Simulation	4
4.1	Event generators	4
4.2	Afterburner processors and correlation analysis	7
4.3	Detector response simulation	7
4.3.1	Simulation framework	10
4.3.2	Geometry of structural elements	10
4.3.3	Geometry of detectors	11
4.4	Fast simulation	13
4.5	Event merging and embedding	13
	References	15

4 Simulation

4.1 Event generators

Heavy-ion collisions produce a very large number of particles in the final state. This is a challenge for the reconstruction and analysis algorithms which require a predictive and precise simulation of the detector response.

The ALICE experiment was designed when the highest nucleon–nucleon center-of-mass energy in heavy-ion interactions was at 20 GeV per nucleon–nucleon pair at the CERN SPS, i.e. a factor of about 300 less than the LHC energy. Model predictions, discussed in Volume 1 of the the ALICE Physics Performance Report [1], for the particle multiplicity in Pb–Pb collisions at LHC vary from 1400 to 8000 charged particles per rapidity unit at mid-rapidity. In summer 2000 the RHIC collider came online. The RHIC data seem to suggest that the LHC multiplicity will be on the lower side of the predictions. However, the RHIC top energy of 200 GeV per nucleon–nucleon pair is still 30 times less than the LHC energy. The extrapolation is so large that both the hardware and software of ALICE had to be designed to cope with the highest predicted multiplicity. On the other hand, we have to use different generators for the primary interaction, since their predictions are quite different at LHC energies.

The simulations of physical processes are confronted with several issues:

- Existing event generators give different predictions for the expected particle multiplicity, p_t and rapidity distributions, and the dependence of different observables on p_t and rapidity at LHC energies.
- Most of the physics signals, like hyperon production, high- p_t observables, open charm and beauty, quarkonia, etc. even at lower energies, are not exactly reproduced by the existing event generators.
- Simulation of small cross-section observables would demand prohibitively long runs to simulate a number of events that is commensurable with the expected number of detected events in the experiment.
- The existing generators do not simulate correctly some features like momentum correlations, flow, etc.

Nevertheless, to allow for efficient simulations we have developed the offline framework such that it allows for a number of options:

- The simulation framework provides an interface to several external generators, like for example HIJING [2] and DPMJET [3].
- A simple event generator based on parametrized η and p_t distributions can provide a signal-free event with multiplicity as a parameter.
- Rare signals can be generated using the interface to external generators like PYTHIA [4] or simple parametrizations of transverse momentum and rapidity spectra defined in function libraries.
- The framework provides a tool to assemble events from different signal generators (event cocktails).
- The framework provides tools to combine underlying events and signal events on the primary particle level (cocktail) and on the digit level (merging).
- Afterburners are used to introduce particle correlations in a controlled way.

The implementation of these strategies is described below. The theoretical uncertainty on the description of heavy-ion collisions at LHC has several consequences for our simulation strategy. A large part of the physics analysis will be the search for rare signals over an essentially uncorrelated background of emitted particles. To avoid being dependent on a specific model, and to gain in efficiency and flexibility, we generate events from a specially developed parametrization of a signal-free final state. This is based on a parametrization of the HIJING pseudo-rapidity (η) distribution and of the transverse momentum (p_t) distribution of CDF [5] data. To simulate the highest anticipated multiplicities we scale the η -distribution so that up to 8000 charged particles per event are produced in the range $|\eta| < 0.5$. Events generated from this parametrization are sufficient for a large number of studies, such as optimization of detector and algorithms performance, e.g. studies of track reconstruction efficiency as a function of particle multiplicity and occupancy. For physics performance studies, we have to simulate more realistic Pb–Pb collisions. HIJING, which yields charged particle multiplicities of up to $dN/d\eta \approx 6000$, is used to simulate the underlying event that is subsequently merged with different signals like jets or heavy flavour.

In order to facilitate the usage of different generators we have developed a generator base class called `AliGenerator`, see Fig. 4.1. Each `AliGenerator` implementation has to provide a basic set of configuration methods such as kinematics and vertex cuts. A high degree of flexibility is reached by providing several pointers as data members to:

- `TGenerator` in order to use external generators (see below);
- `AliVertexGenerator` for the possibility to obtain the event vertex from an external source needed for event merging;
- `AliCollisionGeometry` to provide a collision geometry (impact parameter, number of participants, etc.) to the event header or to other generators;
- `AliStack` to allow for the stand-alone use of a generator.

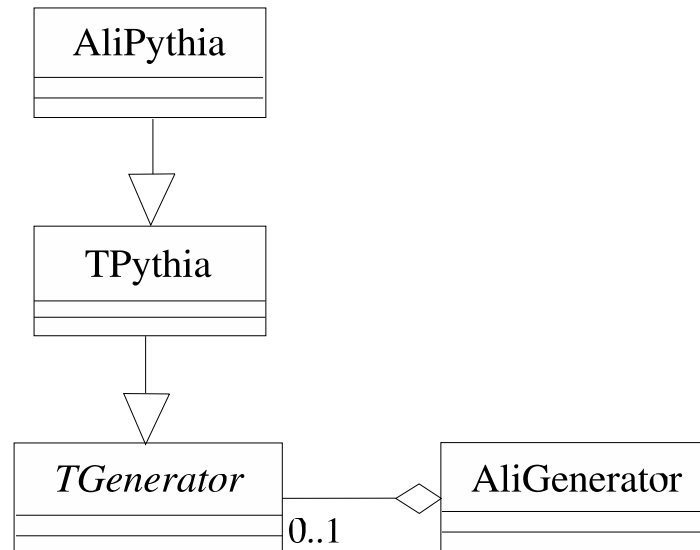


Figure 4.1: `AliGenerator` is the base class that has the responsibility of generating the primary particles of an event. Some realizations of this class do not generate the particles themselves but delegate the task to an external generator like PYTHIA through the `TGenerator` interface.

Several event generators are available via the abstract ROOT class that implements the generic generator interface, `TGenerator`. Through implementations of this abstract base class we wrap FORTRAN Monte Carlo codes like PYTHIA, HIJING, etc. that are thus accessible from the AliRoot classes. In particular, the interface to PYTHIA, `AliPythia` deriving from `TPythia`, includes the use of nuclear structure functions of PDFLIB and a large set of preconfigured processes such as jet-production, heavy flavours, and pp minimum bias.

In many cases, the expected transverse momentum and rapidity distributions of particles are known. In other cases the effect of variations in these distributions must be investigated. In both situations it is appropriate to use generators that produce primary particles and their decays sampling from parametrized spectra. To meet the different physics requirements in a modular way, the parametrizations are stored in independent function libraries wrapped into classes that can be plugged into the generator. This is schematically illustrated in Fig. 4.2 where four different generator libraries can be loaded via the abstract generator interface.

It is customary in heavy-ion event generation to superimpose different signals on an event to tune the reconstruction algorithms. This is possible in AliRoot via the so-called cocktail generator (Fig. 4.3). This creates events from user-defined particle cocktails by choosing as ingredients a list of particle generators.

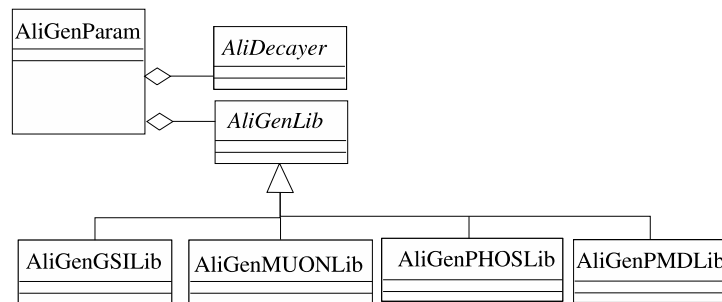


Figure 4.2: `AliGenParam` is a realization of `AliGenerator` that generates particles using parametrized p_t and pseudo rapidity distributions. Instead of coding a fixed number of parametrizations directly into the class implementations, user-defined parametrization libraries (`AliGenLib`) can be connected at run time allowing for maximum flexibility.

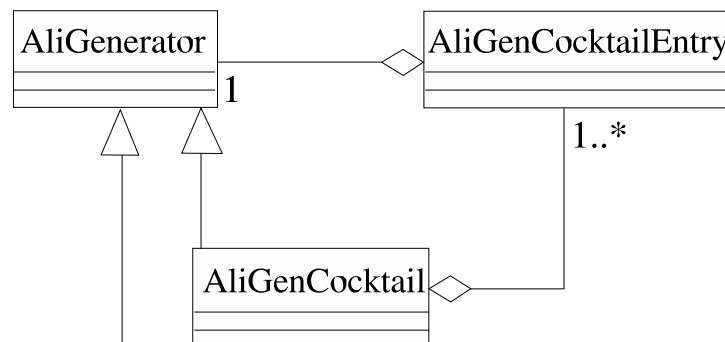


Figure 4.3: The `AliCocktail` generator is a realization of `AliGenerator` which does not generate particles itself but delegates this task to a list of objects of type `AliGenerator` that can be connected as entries (`AliGenCocktailEntry`) at run time. In this way different physics channels can be combined in one event.

4.2 Afterburner processors and correlation analysis

The modularity of the event generator framework allows easy integration with the simulation steering class `AliRun` and with the objects that are responsible for changing the output of event generators or for assembling new events making use of the input of several events. These processors are generally called ‘afterburners’. They are especially needed to introduce a controlled (parametrized) particle correlation into an otherwise uncorrelated particle sample. In `AliRoot` this task is further simplified by the implementation of a stack class (`AliStack`) that can be connected to both `AliRun` and `AliGenerator`. Currently, afterburners are used for the simulation of the two-particle correlations, flow signals, and jet quenching.

4.3 Detector response simulation

To respond to the ALICE simulation requirements, it is important to have a high-quality and reliable detector response simulation code. One of the most common programs for full detector simulation is GEANT 3 [6] which, however, is a 20-year old FORTRAN program, not being (officially) further developed since 1993. GEANT 4 [7] is being developed by a large international collaboration with a strong component in CERN/IT as the OO simulation package for the LHC. We are also using FLUKA [8] as a full detector simulation program. These three programs have a very different user interface, therefore we decided to build an environment that could take advantage of the maturity and solidity of GEANT 3 and, at the same time, protect the investment in the user code when moving to a new Monte Carlo. In order to combine immediate needs and long term requirements into a single framework, we wrapped the GEANT 3 code in a C++ class (`TGeant3`) and we developed a Virtual Monte Carlo (VMC) abstract interface (now part of ROOT, see below). We have interfaced GEANT 4 and FLUKA with our virtual Monte Carlo interface. We will thus be able to change the simulation engine without any modification in the user detector description and signal generation code. This strategy has proved very satisfactory and we are able to assure coherence of the whole simulation process which includes the following steps regardless of the particle transport package in use:

- **Event generation of final-state particles:** The collision is simulated by a physics generator code or a parametrization and the final-state particles are fed to the transport program.
- **Particle transport:** The particles emerging from the interaction of the beam particles are transported in the material of the detector, simulating their interaction with it and the energy deposition that generates the detector response (hits). An event display is shown in Colour Figure II.
- **Signal generation and detector response:** During this phase the detector response is generated from the energy deposition of the particles traversing it. This is the ideal detector response, before the conversion to digital signals and the formatting of the front-end electronics is applied.
- **Digitization:** The detector response is digitized and formatted according to the output of the front-end electronics and the data acquisition system. The results resemble closely the real data that will be produced by the detector.
- **Fast simulation:** The detector response is simulated via appropriate parametrizations or other techniques that do not require the full particle transport.

Virtual Monte Carlo interface

As explained above, our strategy to isolate the user code from changes of the detector simulation package was to develop a virtual interface to the detector transport code. We call this interface Virtual Monte Carlo. It is implemented [9] via C++ virtual classes and is schematically shown in Figs. 4.4

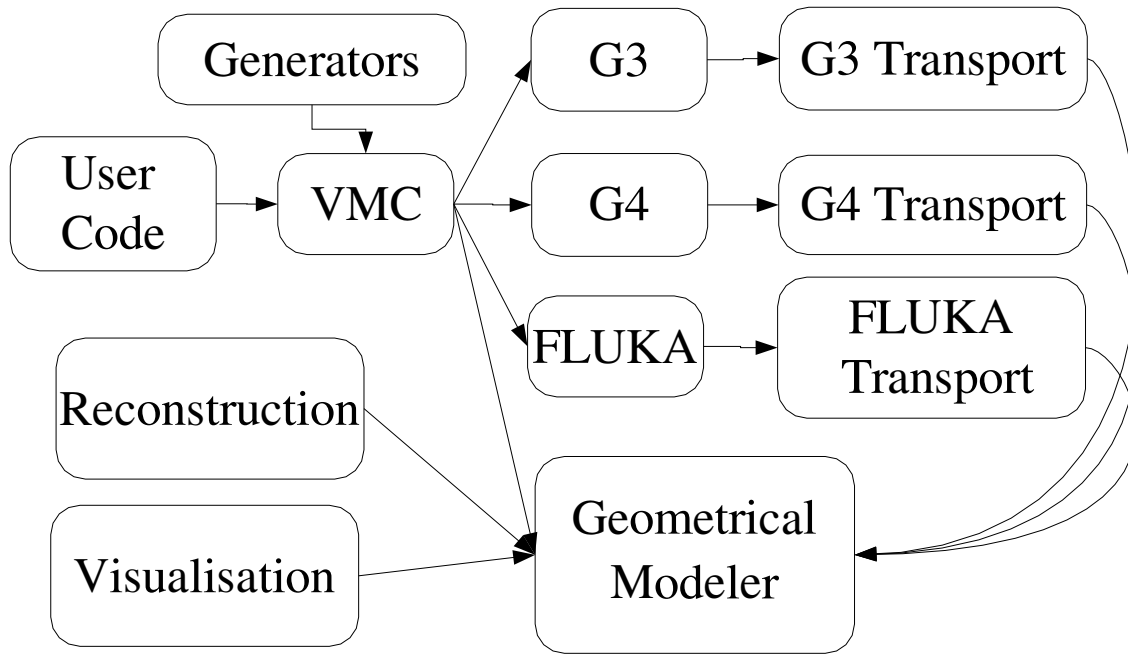


Figure 4.4: The Virtual Monte Carlo concept.

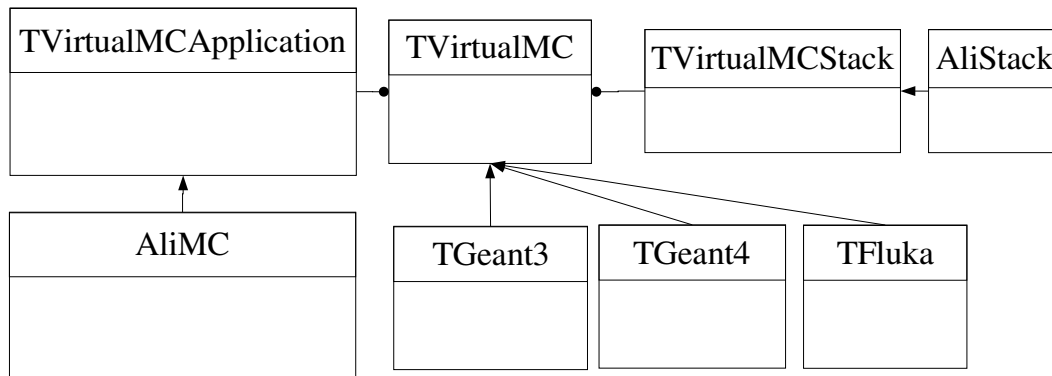


Figure 4.5: The Virtual Monte Carlo design and its realization within the AliRoot framework.

and. 4.5. The codes that implement the abstract classes are real C++ programs or wrapper classes that interface to FORTRAN programs.

An additional step is to replace the geometrical modeller of the different packages with a single one, independent from any specific simulation engine; the aim is to use the same geometrical modeller also for reconstruction and analysis. Thanks to the collaboration between the ALICE Computing project and the ROOT team, we have developed a geometrical modeller, `TGeo` [10], that is able to represent the ALICE detector, and to replace the GEANT 3 modeller for navigation in the detector. It has also been interfaced to FLUKA and discussions are under way with the GEANT 4 team to interface it to the GEANT 4 Monte Carlo. Using the `roottog4` converter, geometries defined for `TGeo` can be converted into GEANT 4 geometries.

Using the virtual Monte Carlo we have converted all FORTRAN user code developed for GEANT 3 into C++, including the geometry definition and the user scoring routines, `StepManager`. These have been integrated in the detector classes of the AliRoot framework. The output of the simulation is saved directly with ROOT I/O, simplifying the development of the digitization and reconstruction code in C++.

GEANT

GEANT 3 is the detector simulation Monte Carlo code used extensively so far by the HEP community for simulation of the detector response. However, it is no longer maintained and has several known drawbacks, both in the description of physics processes, particularly hadronic [11], and of the geometry. Its designated successor is GEANT 4. ALICE has spent considerable effort in evaluating GEANT 4 via several benchmarks; details on ALICE experience with GEANT 4 can be found in Ref. [12]. We were able to keep the same geometry definition using the `G3toG4` utility to translate from GEANT 3 to GEANT 4 geometry; in addition we have improved `G3toG4` and made it fully operational. The virtual Monte Carlo interface allows us to run full ALICE simulations also with GEANT 4 and to compare them with the GEANT 3 results; the advantage being that both simulation programs use the same geometry and the same scoring routines. An additional advantage is a substantial economy of effort. Using the same geometry description eliminates one of the major sources of uncertainty and errors in the comparison between different Monte Carlos, which comes from the fact that it is rather difficult to make sure that, when comparing two Monte Carlos on a particular experimental configuration, there are no differences in the geometry description and in the scoring.

This exercise has exposed the GEANT 4 code to a real production environment and we experienced several of its weaknesses. We faced several problems with its functionality that have required substantial user development. In particular, the definition of volume or material-specific energy thresholds and mechanism lists are not so straightforward as in GEANT 3. The strategy of GEANT 3 was to provide the user one state-of-the-art implementation of the physics processes together with a few configuration options essentially for performance optimization and debugging. In contrast, GEANT 4 has to be configured using so-called physics lists corresponding to different combinations of model implementation that in addition may vary from detector to detector. They need a great amount of inside knowledge to be used correctly.

We have also performed a number of benchmark tests of the hadronic [13] and low-energy neutron transport [14].

We are now planning to interface GEANT 4 with the ROOT geometrical modeller to avoid the conversion step via `G3toG4` and to take advantage from its advanced solid modelling capabilities.

FLUKA

FLUKA plays a very important role in ALICE for all the tasks where detailed and reliable physics simulation is vital, given its thorough physics validation and its almost unique capability to couple low-energy neutron transport with particle transport in a single program. These include background calculation, neutron fluence, dose rates, and beam-loss scenarios [15]. An example for a neutron fluence map obtained with FLUKA is shown in Colour Figure III. FLUKA has been particularly important for ALICE in the design of the front absorber and beam shield. To ease the input of the FLUKA geometry, ALICE has developed an interactive interface [16], called `ALIFE`, that allows setups described with FLUKA to be combined and modified easily. Figure 4.6 schematically describes the use of `ALIFE` to prepare the input for FLUKA.

To provide another alternative to GEANT 3 for full detector simulation, we have developed an interface with FLUKA, again via the Virtual Monte Carlo. The native FLUKA geometry modeller has been replaced by `TGeo` which allows us to run FLUKA with the same geometry as used for GEANT 3 and GEANT 4 simulations.

Rigorous tests have been performed to ensure that the FLUKA native geometry modeller and navigator and the FLUKA interfaces to `TGeo` give exactly the same results. Based on these tests, the FLUKA Project considers the FLUKA-`TGeo` as validated.

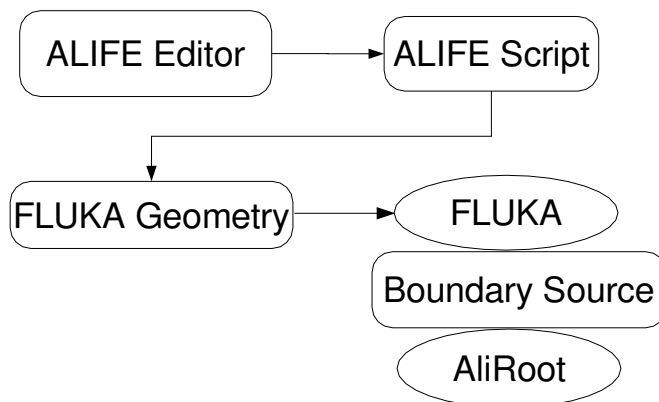


Figure 4.6: Example of the use of ALIFE. The ALIFE editor allows easy creation of an ALIFE script, which is in fact FLUKA input geometry. FLUKA is then used to transport particles, including low-energy neutrons, to a virtual boundary surface. The particles are then written to a file that is used as a source for a regular AliRoot simulation to evaluate the detector response to background.

4.3.1 Simulation framework

The AliRoot simulation framework can provide data at different stages of the simulation process [17], as described in Fig. ?? on page ?. Most of the terminology comes from GEANT 3. First, there are the so-called hits that represent the precise signal left by the particle in the detector before any kind of instrumental effect, i.e. precise energy deposition and position. These are then transformed into the signal produced by the detector, summable digits that correspond to the raw data before digitization and threshold subtraction. The introduction of summable digits is necessary because of the embedding simulation strategy elaborated for the studies in the Physics Performance Report. These summable digits are then transformed into digits that contain the same information as raw data, but in ROOT structures. The output of raw data in DATE (the ALICE data acquisition system [18]) format has already been done during the data challenges.

The ALICE detector is described in great detail, see Fig. 4.7, including services and support structures, beam pipe, flanges, and pumps. The AliRoot geometry follows the evolution of the baseline design of the detector in order to continuously provide the most reliable simulation of the detector response. AliRoot is also an active part of this process since it has been used to optimize the design, providing different geometry options for each detector. The studies that provided the results presented in the PPR were performed with the baseline geometry.

4.3.2 Geometry of structural elements

The description of the front- and small-angle absorber regions is very detailed on account of their importance to the muon spectrometer. The simulation has been instrumental in optimising their design and in saving costs without a negative impact on the physics performance. The material distribution and magnetic fields of the L3 solenoidal magnet and of the dipole magnets are also described in detail. The magnetic field description also includes the interference between the two fields. The field distributions are described by three independent maps for 0.2, 0.4 and 0.5 T solenoid L3 magnetic field strengths. Alternatively, it is possible to use simple parametrizations of the fields, i.e. constant solenoidal field in the barrel and a dipole field varying along the z direction for the muon spectrometer. The space frame, supporting the barrel detectors, is described according to its final design taking into account modifications to the initial design such that it allows the eventual addition of a proposed electromagnetic calorimeter [19].

The design of the ALICE beam pipe has also been finalized. All elements that could limit the detector performance (pumps, bellows, flanges) are represented in the simulation.

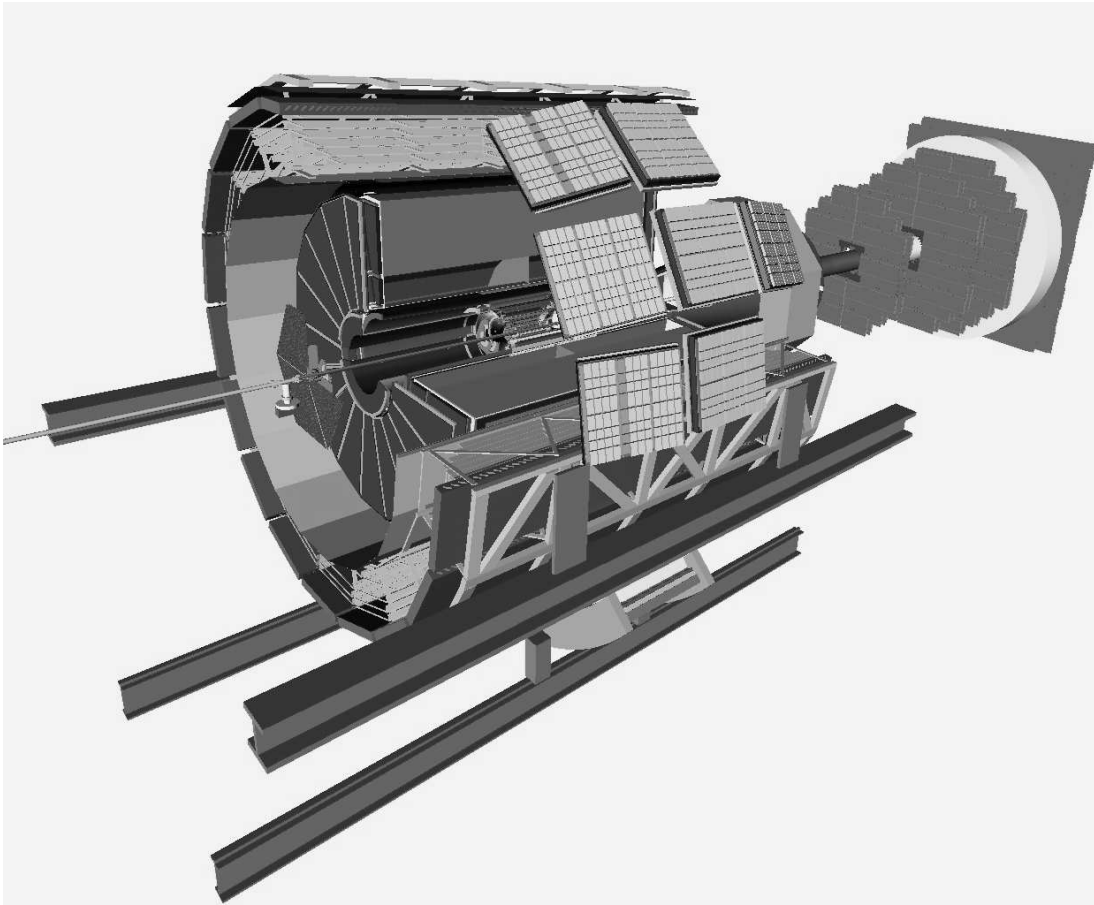


Figure 4.7: AliRoot simulation of the ALICE detector.

4.3.3 Geometry of detectors

Most of the detectors are described by two versions of their geometry; a detailed one, which is used to accurately simulate the detector response and study their performance, and a coarse version that provides the correct material budget with minimal details, and is used to study the effect of this material budget on other detectors. For some detectors, different versions of the geometry description corresponding to different geometry options are selectable via the input C++ script at run time. In the following we give some examples. We remind the reader that some of this information is still subject to rapid evolution.

Both a detailed and a coarse geometry are available for the ITS. The detailed geometry of the ITS is very complicated (see Colour Figure IV) and crucially affects the evaluation of impact parameter and electron bremsstrahlung. On the other hand, simulation of the coarse geometry is much faster when ITS hits are not needed.

Three configurations are available for the TPC. Version 0 is the coarse geometry, without any sensitive element specified. It is used for the material budget studies and is the version of interest for the outer detectors. Version 1 is the geometry version for the Fast Simulator. The sensitive volumes are thin gaseous strips placed in the Small (S) and Large (L) sectors at the pad-row centres. The hits are produced whenever a track crosses the sensitive volume (pad-row). The energy loss is not taken into account. Version 2 is the geometry version for the slow simulator. The sensitive volumes are S and L sectors. One can specify as sensitive volumes either all sectors or only a few of them, up to 6 S and 12 L sectors. The hits are produced in every ionizing collision. The transport step is calculated for every collision from an exponential distribution. The energy loss is calculated from an $1/E^2$ distribution and the response is parametrized by a Mathieson distribution.

The TRD geometry is simulated in great detail, including the correct material budget for electronics and cooling pipes. The full response and digitization have been implemented allowing studies of open questions such as the number of time-bins, the 9- or 10-bit ADC, the gas and electronics gain, the drift velocity, and maximum Lorentz angle. The transition-radiation photon yield is approximated by an analytical solution for a foil stack, with adjustment of the yield for a real radiator, including foam and fibre layers from test beam data. This is quite a challenging detector to simulate, as both normal energy loss in the gas and absorption of transition-radiation photons have to be taken into account. During the signal generation several effects are taken into account: diffusion, 1-dimensional pad response, gas gain and gain fluctuations, electronics gain and noise, as well as conversion to ADC values. Absorption and $\mathbf{E} \times \mathbf{B}$ effects will be introduced.

A detailed study of the background coming from slow neutron capture in Xe gas was performed [20] with FLUKA. The spectra of photons emitted after neutron capture are not included in standard neutron-reaction databases. An extensive literature search was necessary in order to simulate them. The resulting code is now part of the FLUKA Monte Carlo [21].

The TOF detector covers a cylindrical surface of polar acceptance $|\theta - 90^\circ| < 45^\circ$. It has a modular structure corresponding to 18 sectors in ϕ and to 5 segments in z . All modules have the same width of 128 cm and increasing lengths, adding up to an overall TOF barrel length of 750 cm. Inside each module the strips are tilted, thus minimizing the number of multiple partial-cell hits due to the obliqueness of the incidence angle. The double stack-strip arrangement, the cooling tubes, and the material for electronics have been described in detail. During the development of the TOF design several different geometry options were studied, all highly detailed.

The HMPID detector also poses a challenge in the simulation of the Cherenkov effect and the secondary emission of feedback photons. A detailed simulation has been introduced for all these effects and has been validated both by test-beam data and with the ALICE RICH prototype that has been operating in the STAR experiment. An event display with the typical rings is shown in Colour Figure V.

The PHOS has also been simulated in detail. The geometry includes the Charged Particle Veto (CPV), crystals (EMC), readout (APD) and support structures. Hits record the energy deposition in one CPV and one EMC cell per entering particle. In the digits the contribution from all particles per event are summed up and noise is added.

The simulation of the ZDC in AliRoot requires transport of spectator nucleons with Fermi spread, beam divergence, and crossing angle for over 100 m. The HIJING generator is used for these studies taking into account the correlations with transverse energy and multiplicity. Colour Figure VI shows the result of the simulation of the hadronic shower induced by a 2.7 TeV neutron.

The muon spectrometer is composed of five tracking stations and two trigger stations for which detailed geometries have been implemented. Supporting frames and support structures are coarsely described but they are not very important in the simulation of the signal. The muon chambers have a complicated segmentation that has been implemented during the signal generation via a set of virtual classes. This allows one to change the segmentation without modifying the geometry.

Summable digits (pad hits) are generated taking into account the Mathieson formalism for charge distribution, while work is ongoing on the angular dependence, Lorentz angles and charge correlation.

The complex T0–FMD–V0–PMD forward detector system is still under optimization. Several options are provided to study their performance.

The description of ALICE geometry and the generation of simulated data are in place. Hence the off-line framework allows the full event reconstruction including the main tracking devices. The framework also allows comparison with test-beam data. The early availability of a complete simulation has been an important point for the development of reconstruction and analysis code and user interfaces, which is now the focus of the development.

4.4 Fast simulation

Owing to the expected high particle multiplicity for heavy-ion collisions at the LHC, typical detector performance studies can be performed with a few thousand events. However, many types of physics analysis, in particular of low cross-section observables, such as D meson reconstruction from hadronic decay channels, have to make use of millions of events. Computing resources are in general not available for such high-statistics simulations.

To reach the required sample size, fast simulation methods based on meaningful parametrizations of the results from detailed and consequently slow simulations are applied. The systematic error introduced by the parametrizations is in general small compared with the reduction of the statistical error. This is particularly true for the studies of the invariant-mass continuum below a resonance (cocktail plots).

It is hard to find a common denominator for fast simulation methods since they are very specific to the analysis task. As a minimum abstraction, we have designed base classes that allow for a representation of the detector or detector systems as a set of parametrizations of acceptance, efficiency, and resolution. The Muon Spectrometer fast simulation has been implemented using these classes.

Another interesting development concerns the fast simulation of the resolution and efficiency of track reconstruction in the central barrel. In this approach, resolution and efficiency in TPC are obtained from the track parameters at the inner radius of the TPC, using a parametrization. After this, full track reconstruction is performed for the inner tracking system, which is needed for detailed secondary vertex reconstruction studies. For details see Ref. [22].

4.5 Event merging and embedding

The simulation of small cross-section observables would demand prohibitively long runs to simulate a number of events commensurable with the expected number of detected events in the experiment. To circumvent this problem we use an event merging procedure: during digitization we produce so called summable digits. These are digits before the addition of electronic noise and pedestal subtraction. Merged events are produced by adding the summable digits from background and signal events. Each background event is used n times for merging. Since computing time is dominated by the simulation of the background events, in this way the event statistics is increased by a factor of n .

A similar technique called embedding consists in mixing data from simulation with real events. This allows for the realistic evaluation of track reconstruction performance in a high-particle-density environment. Here, events are mixed on the level of ADCs and are subsequently passed to the standard reconstruction code.

References

Chapter 4

- [1] CERN/LHCC 2003-049, ALICE Physics Performance Report, Volume 1 (7 November 2003); ALICE Collaboration: F. Carminati *et al.*, J. Phys. G: Nucl. Part. Phys. **30** (2004) 1517–1763.
- [2] X. N. Wang and M. Gyulassy, Phys. Rev. **D44** (1991) 3501;
M. Gyulassy and X. N. Wang, Comput. Phys. Commun. **83** (1994) 307–331;
the code can be found in <http://www-nsdth.lbl.gov/~xnwang/hijing/>.
- [3] J. Ranft, Phys. Rev. **D51**, (1995) 64;
J. Ranft, New features in DPMJET version II.5, hep-ph/9911213;
J. Ranft, DPMJET version II.5, code manual, hep-ph/9911232.
- [4] H.-U. Bengtsson and T. Sjostrand, Comput. Phys. Commun. **46** (1987) 43;
the code can be found at <http://nimis.thep.lu.se/~torbjorn/Pythia.html> .
T. Sjostrand, Comput. Phys. Commun. **82** (1994) 74;
the code can be found at <http://www.thep.lu.se/~torbjorn/Pythia.html> .
- [5] F. Abe *et al.*, (CDF Collaboration), Phys. Rev. Lett. **61** (1988) 1819.
- [6] R. Brun, F. Bruyant, M. Maire, A.C. McPherson, P. Zancarini, GEANT3 User Guide, CERN Data Handling Division DD/EE/84–1 (1985);
<http://wwwinfo.cern.ch/asdoc/geantold/GEANTMAIN.html> .
- [7] S. Agostinelli *et al.*, Geant4 - A simulation toolkit, CERN-IT-20020003, KEK Preprint 2002-85, SLAC-PUB-9350, submitted to Nucl. Instrum. and Methods A;
<http://wwwinfo.cern.ch/asd/geant4/geant4.html> .
- [8] A. Fassò *et al.*, in *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003); <http://www.slac.stanford.edu/econf/C0303241/proc/papers/MOMT004.PDF> .
- [9] I. Hřivnáčová *et al.*, in *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003); <http://www.slac.stanford.edu/econf/C0303241/proc/papers/THJT006.PDF> .
- [10] R. Brun, A. Gheata, and M. Gheata, *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003);
<http://www.slac.stanford.edu/econf/C0303241/proc/papers/THMT001.PDF> .
- [11] See for instance the note ATL-PHYS-96-086
(<http://preprints.cern.ch/cgi-bin/setlink?base=atlnot&categ=Note&id=phys-96-086>).
- [12] I. González Caballero *et al.*, *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003);
<http://www.slac.stanford.edu/econf/C0303241/proc/papers/MOMT011.PDF> .
- [13] F. Carminati, I. González Caballero, Geant4: A benchmark of hadronic processes, ALICE-INT-2001-041.
- [14] Note in course of publication, see
<http://wwwinfo.cern.ch/asd/geant4/reviews/delta-2002/schedule.html> .
- [15] B. Pastircak and A. Morsch, ALICE-INT-2002-028.
- [16] A. Morsch, ALIFE: A geometry editor and parser for fLUKA, ALICE-INT-1998-029.
- [17] F. Carminati and A. Morsch, *Proc. of Computing in High Energy and Nuclear Physics*, La Jolla, California (2003);
<http://www.slac.stanford.edu/econf/C0303241/proc/papers/TUMT004.PDF> .
- [18] <http://aldwww.cern.ch> .
- [19] <http://nuclear.ucdavis.edu/~jklay/ALICE/> .
- [20] G. Tsiledakis *et al.*, ALICE-INT-2003-010.

- [21] A. Fassò, A. Ferrari, P.R. Sala, G. Tsileidakis, Implementation of xenon capture gammas in FLUKA for TRD background calculation, ALICE-INT-2001-28.
- [22] A. Dainese and R. Turrisi, ALICE-INT-2003-019;
A. Dainese and R. Turrisi, ALICE-INT-2003-028.