International Journal of Modern Physics A
© World Scientific Publishing Company

# LINEAR COLLIDER DETECOR SIMULATION OVERVIEW

NORMAN A. GRAF*

*SLAC*

*2575 Sand Hill Road, Menlo Park CA 94025, USA*

Received (Day Month Year)
Revised (Day Month Year)

We describe the ILC detector simulation environment, which features a flexible GEANT4-based detector response program and a Java toolkit for full event reconstruction and analysis. The toolkit is currently being used for detector design and physics analysis for a future $e^+e^-$ linear collider. The reconstruction components are fully modular and are available for tasks from digitization of tracking detector signals through to cluster finding, pattern recognition, fitting, jetfinding, and analysis.

*Keywords*: International Linear Collider; Detector Simulations

## 1. The International Linear Collider

Detectors are currently being designed to fully exploit the physics discovery potential of $e^+e^-$ collisions at $\sqrt{s}$=1TeV by performing precision measurements of complex final states. This will require exceptional momentum resolution, excellent vertexing capabilities, and highly segmented, hermetic calorimetry.

The Linear Collider Detector (LCD) Simulation and Reconstruction group is tasked to provide full simulation capabilities for the International Linear Collider (ILC) physics program, including physics simulations, detector designs, machine-detector interface and background studies. Due to the long leadtime of this project, we need flexibility for new detector geometries and technologies and innovative reconstruction algorithms. The goal is to develop a common simulation environment used in ILC studies which allows sharing of detector designs, reconstruction algorithms, and code. The system should be flexible, powerful, yet simple to install, maintain and use. Here we present the approach taken by the North American Linear Collider Detector (LCD) group.

## 2. Event Generation

Most of the legacy event generation programs are written in FORTRAN and target the HEPEVT common block. We have defined an abstract interface to the

*Norman.Graf@slac.stanford.edu

event generation sequence which allows us to create wrappers for most of the existing general-purpose packages (e.g. PYTHIA, HERWIG, ISAJET) and access the events through a consistent platform-independent interface. We use the Java Native Interface to call these event generators via C++. The use of shared-object or dynamic link libraries enables late binding and provides runtime flexibility. The user can select physics processes, beamstrahlung, decay channels, etc. at runtime simply by editing a text file. Events can be generated on the fly within a fast-simulation and analysis environment, or written out in binary form using the stdhep format. Any event generator targeting the HEPEVT structure can be easily accommodated within this framework.

## 3. Detector Simulation

The response of detector designs to the simulated events is modeled using the GEANT4 physics kernel with a thin layer of LCD-specific C++ code. Detector geometry is input at runtime using an XML file, allowing different detector designs to be simulated simply by editing a plain ASCII file. We have identified generic hit classes which encapsulate the information that is intrinsic to tracking detectors and calorimeters. This information, along with the Monte Carlo particle hierarchy, is written out in a format which is meant to be accessible to all Linear Collider reconstruction and analysis formats (LCIO[1]).

## 4. Event Reconstruction

The reconstruction software [2] runs either standalone or inside Java Analysis Studio (JAS3[3]). The GEANT4 full simulation program writes out the full Monte Carlo hit information; detector digitization is deferred until the reconstruction phase. This allows detector readout schemes to be varied (within reason) to study the effects of CCD pixel size, depletion depth, silicon microstrip pitch, TPC readout electronics, etc. Digitizing the hit information provides more realistic hits by simulating the effects of electronic noise and crosstalk, hit merging, ghost hits arising from strip pairing, and cluster dependent position measurement uncertainties. One can overlay arbitrary combinations of beam and physics background at the detector hit level, as well as adding salt and pepper noise hits, or simulating inefficiencies by dropping hits. The full reconstruction package features *ab initio* track finding and fitting, calorimeter clustering, individual particle reconstruction (e.g. cluster-track association). A fast, 4-vector smearing Monte Carlo provides a "best you can do" target towards which the full reconstruction aims.

### 4.1. *Detector Hit Digitization*

The GEANT4 full simulation package writes out the full Monte Carlo detector hit information. Since this would be a prohibitively large (and mostly useless) amount of information for each element of a particle shower in a calorimeter, the calorimeter hit

(SimCalorimeterHit) information is quantized into cells in the GEANT4 program. Studies of the impact of various readout options such as segmentation, number of readout bits and thresholds can be undertaken by ganging cells and digitizing at the reconstruction stage. Hits in the tracking detectors (SimTrackerHit) preserve the full MC information for each hit: position, time, dE/dx, MC parentage, etc. The hits are then digitized at the reconstruction level, allowing the effects of strip pitch, pixel size, charge sharing, electronic noise, etc. to be studied.

### 4.2. *Track Finding and Fitting*

The released reconstruction software features full pattern recognition in 3D detectors. For 2D outer trackers, such as silicon microstrip trackers, tracks found in the vertex detector can be extrapolated outwards. The current software is specifically tuned for two strawmen detectors, but work is ongoing to generalize the pattern recognition software. A generic Kalman filter approach to fitting, which accomodates the effects of multiple scattering and energy loss, has been implemented.

### 4.3. *Calorimeter Clustering*

To achieve the best dijet invariant mass resolution, we have adopted the "Particle Flow" approach to event reconstruction. The calorimeters are used to reconstruct and measure the energy of electromagnetic and neutral hadron showers. The momentum of charged hadrons is measured much more accurately in the tracking detectors. Associating energy depositions in the calorimeters to particle showers is, therefore, at the heart of the "Particle Flow" algorithm, and emphasizes granularity over intrinsic energy resolution. A generic cluster interface has been defined, and several clustering algorithms are currently implemented. A MC "cheater" is available to associate cells based on the known MC track depositing the energy to provide a target towards which to aim. A Nearest-Neighbor algorithm, with user-definable neighborhood domains for association is also available. A fixed-radius cone algorithm has shown promise as a fast, efficient clustering algorithm for finding both electromagnetic and neutral hadron showers. The fine granularity of the proposed calorimeters allows the particle identity of clusters to be identified with reasonable confidence based on neural net analyses of the cluster characteristics such as shape and energy moments.

### 4.4. *Reconstructed Particle*

"Particle Flow" algorithms are being developed with minimal coupling to specific detector designs. The photon-finding and muon reconstruction packages are fairly mature. Although several approaches are being investigated, the emphasis for charged hadron reconstruction has been placed on a track-following algorithm. Tracks found in the central trackers are extrapolated into the calorimeters and energy deposited in the readout cells is added to the track until a threshold is reached, either in the matching of energy to momentum of the track, or a distance from-the-track metric is exceeded. The high granularity of the calorimeter readout design allows the traces of minimum ionizing particles (either muons or charged hadrons before they shower) to be reconstructed with high efficiency and reasonable precision.

## 5. Analysis

The analysis suite includes a pure Java Neural Net implementation available for training and use, physics tools such as topological vertex finding, jet finding and jet flavor tagging. Analysis tools feature an LCD-specific event display based on WIRED[4], as well as a full suite of ntuple and histogram manipulation and fitting functions. The existing code and documentation (including examples and tutorials) can be downloaded from the hep.lcd homepage [2].

Systematic investigations of the dijet invariant mass resolution as a function of the central magnetic field, the calorimeter inner radius and barrel-endcap aspect ratio, calorimeter transverse readout cell area, longitudinal segmentation, material and readout technology are being undertaken, employing a "Particle Flow" approach.

## 6. Summary

Designing detectors for the International Linear Collider is an area of active development with many choices, requiring close coupling of design, simulation, and reconstruction. Ease of use and speed of development are essential for physicists conducting ILC studies part-time and in a heterogeneous environment. The GEANT4-based simulation of the detector response features a flexible runtime geometry based on an XML description, A fairly complete suite of reconstruction and analysis tools written in Java exists and is being successfully used for these studies. As part of an ongoing program to truly internationalize the ILC physics and detector simulation effort, the event data model for the simulation and reconstruction software is being rewritten to target the LCIO data model. Experience gained in the use of the existing packages is being used to refactor much of the software. The improved code will be released as a collection of packages in the org.lcsim namespace [5].

## Acknowledgments

I would like to acknowledge and thank all the LCD developers who have contributed to this project.

## References

1. http://lcio.desy.de
2. http://www-sldnt.slac.stanford.edu/jas/Documentation/lcd/
3. http://jas.freehep.org/jas3
4. http://wired.freehep.org
5. http://lcsim.org