

# THE NEW FAST ATLAS TRACK SIMULATION ENGINE (Fatras)

A. Salzburger, University of Innsbruck, Austria & CERN, Geneva, Switzerland\*

A. Wildauer, University of Innsbruck, Austria & CERN, Geneva, Switzerland

S. Fleischmann, Bergische Universität, Wuppertal, Germany

T. Lenz, Bergische Universität, Wuppertal, Germany

## Abstract

Various systematic physics and detector performance studies with the ATLAS detector require very large simulated event samples. Since the full detector simulation is a highly cpu time consuming operation, fast simulation techniques are widely used in such applications. The powerful ATLAS fast simulation package ATLFast, however, is based on the smearing of the initial track representation and does not allow tracking detector studies on hit level. Alternatively, the new ATLAS Fast Track Simulation (Fatras) is capable of producing full track information, including hits on track. Initially developed as a validation tool for the ATLAS offline track reconstruction, the new fast track simulation has become a powerful engine for various use cases. Fatras is based on common ATLAS offline track reconstruction code and is fully embedded in the ATLAS event data model. Evidently, it is realized within the ATLAS C++ based software framework ATHENA.

## INTRODUCTION

During the last three years the ATLAS offline track reconstruction has undergone a complete redesign[1], including the development of the new common ATLAS event data model (EDM). Modularity in terms of coherent interface definitions has been imposed, following strictly a component software pattern. As a consequence component libraries can be loaded dynamically at run setup and flexible run configuration via python scripts at the user front end is possible[2]. This modularity invoked a rapid development of various concrete implementations of each single component in track reconstruction: from first and second stage pattern recognition, ambiguity solving, track and vertex fitting, realized by different AlgTools and Algorithms of similar purpose. AlgTools, Algorithms and Services are implemented as extensions of the according ATHENA [3] framework interfaces and will be in the following referred to as such. The new extrapolation package [4] is a concrete example for the imposed software design structure. Together with its concept of an underlying connective reconstruction geometry, in the following referred to as TrackingGeometry, that enables full navigation through the detector, a new fast track simulation has been developed, initially to serve the needs of distinctive validation of the recently introduced tracking components. The ATLAS fast track simulation (Fatras) differs substantially from both existing simulation packages that have been used within the ATLAS

collaboration. It is, contrary to the widely used fast simulation package ATLFast[5] not based on a parametric smearing of the initial track parameters, but a full Monte-Carlo simulation technique. On the other hand, Fatras uses a more simplified detector geometry than the full detector simulation that is based on the Geant4 [6] simulation toolkit. Data objects of the offline reconstruction chain are directly used in the track creation process. These simplifications decrease the cpu time consumption by both, viewer propagation steps through the magnetic field and detector material, such as the omission of a hit digitization process.

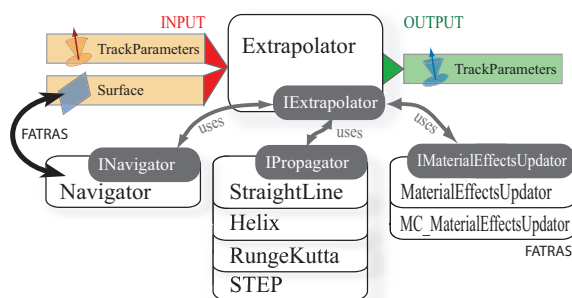


Figure 1: Schematic illustration of the track extrapolation structure: common interfaces allow interaction with the algorithms to be used for the extrapolation of a track representation to a destination surface. In the Fatras application, the Navigator is interfaced with the input surface, such that track predictions can be done.

## FATRAS SOFTWARE DESIGN

The extrapolation package and its associated TrackingGeometry build the core of the new fast track simulation. Fatras is a full monte carlo simulation using the TrackingGeometry as the simulation geometry. The trajectory is created by propagating through the magnetic field and intersecting sensitive detector elements. Consequently, navigation between volume and surface nodes in the TrackingGeometry has to be imposed to allow the prediction of the sequence of surfaces that are intersected by the trajectory.<sup>1</sup> During the propagation through the detector, interactions with the detector material according to different particle types is taken into account. The extrapolation package

<sup>1</sup>In the track reconstruction process, the navigation through the TrackingGeometry is used as a fast material and magnetic field access; both, magnetic field and material information are coupled to the Tracking volumes which are the main component blocks of the TrackingGeometry.

\* corresponding author: Andreas.Salzburger@cern.ch

and the concepts and realization of the TrackingGeometry are discussed in the following subsections. The created trajectory, i.e. a ordered vector of track representations on the intersected surfaces, is then processed by dedicated algorithms that introduce measurement smearing, followed by noise creation and optionally a refit with smeared input parameters.

### The Extrapolation package

The extrapolation process can be divided into three conceptually different realms:

- a) *mathematical propagation of parameters and associated errors*
- b) *navigation between volumes and layers*
- c) *material effects integration*

This three column structure is reflected by the design of the extrapolation package: a central AlgTool, the Extrapolator, steers and coordinates propagation, navigation and material integration AlgTools in a coherent way. For the usage in Fatras, the navigator provides additionally the input surfaces in an iterative way and the standard material effects updating AlgTool is exchanged by a Monte-Carlo based version. A more detailed description of the ATLAS track extrapolation package, emphasizing the navigation process, can be found in [4], Fig. 1 shows a schematic block diagram of the main components the extrapolation package consists of.

### The TrackingGeometry

The TrackingGeometry is the key to the navigation in track extrapolation processes. Volumes and Surfaces build the purely geometrical descriptions that are extended to the physical or logical entities TrackingVolume and Layer, which themselves refer to information about the material and the magnetic field. Volumes are implemented as containers of confining boundary surfaces, that again hold pointers to the attached volumes. Enhanced by the fully connective nature of the TrackingGeometry, boundary surfaces are used to predict the path through the detector volumes. Inside a volume, sensitive detector elements are grouped on layers. The layers are ordered within the natural volume frame to enable the navigation between them. Boundary surface and layer classes extend the geometrical surface classes and can therefore straightforwardly be used as an input to the Extrapolator AlgTool. The TrackingGeometry is created by parsing and simplifying the common ATLAS detector description GeoModel [7], which also serves as an input to the full Geant4 [6] based ATLAS detector description. Since the surfaces that represent sensitive detector elements are simply proxies for the according GeoModel objects misalignment is automatically taken into account when applied through the common atlas conditions database.

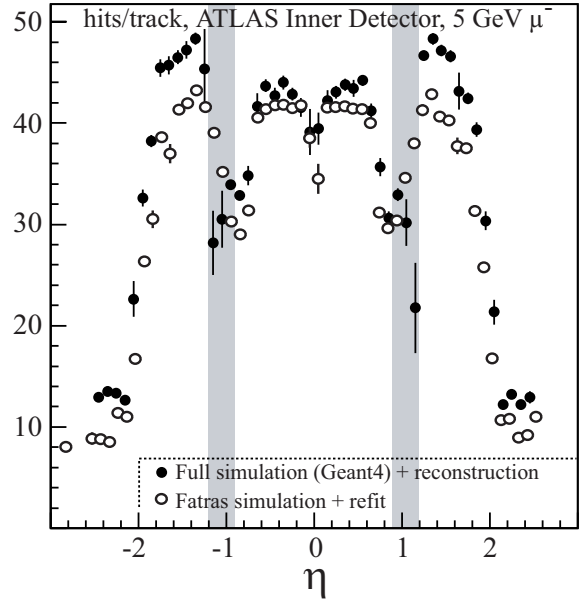


Figure 2: Comparison of the average number of hits on track for 5 GeV single  $\mu$  tracks in the ATLAS Inner Detector for full simulation (including reconstruction) with Fatras simulation (including refitting).

### Track creation

Trajectories created by the Extrapolator AlgTool are transformed into a track by another dedicated algorithm. A loop is performed over the track representations on the different surfaces intersected by the idealistic track and measurement smearing is applied according to the associated detector parameters. Common hit objects are created and filled in the EDM track structure.

Following the imposed Gaudi-Athena framework model of a strict separation between data model and algorithmic parts in the reconstruction process, the simulated tracks are written to the transient event store (StoreGate) to be retrieved by successive algorithms.

Figure 2 shows the average number of simulated hits per track in comparison to tracks that have been created by full detector simulation, digitization and standard reconstruction. The number of hits produced by the Fatras simulation is in general too low, which is due to an incorrect treatment of module overlap regions, mostly in the silicon strip detector. A more sophisticated strategy to detect overlap regions is been worked on currently. Additionally it can be seen that around the region of pseudo-rapidity  $|\eta| \sim 1$ , Fatras produces more hits on track, which indicates to be a problem in the track reconstruction from full simulated tracks.

### Post processing and noise creation

In the default setup, tracks created by the Fatras track creator are refitted with a smeared perigee input information to create a more realistic picture of a *reconstructed*

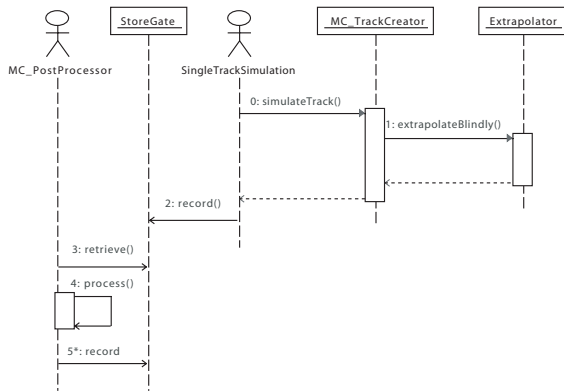


Figure 3: Simplified UML sequence diagram for the Fatras SingleTrackSimulation. The two framework algorithms are illustrated as UML actor nodes, the event data is written respectively accessed through the transient event store (StoreGate).

track.<sup>2</sup> Additionally, Fatras includes a post-processor, which prepares the simulated track data for the further reconstruction chain. As the input objects for the reconstruction in the ATLAS software are handled on an event-by-event basis using special data collections in the transient event store, the hits have to be extracted from the simulated tracks. In the same step noise measurements, according to user-defined levels for the various sub detectors, are simulated and added. This procedure allows Fatras to be used as an input engine of the standard track finding and pattern recognition procedures. Figure 3 shows a simplified UML sequence diagram including track creating, post-processing and the writing respectively retrieving of the various collections to the transient event store.

## FATRAS APPLICATIONS

Fatras was created as a debugging tool, following the idea of factorizing the complex procedure of track reconstruction into clearly defined modules that can be validated in a controlled environment. Fatras enables e.g. the validation of track fitters without introducing any dependencies on the pattern recognition process. Similarly, as the fast track simulation uses almost the identical geometrical setup for simulation and refitting, dependencies on the material description respectively the magnetic field can either be cancelled or estimated.

Three different simulation modes are available for the Fatras simulation:

- a The **SingleTrackSimulation** enhances the simulation of single tracks with different particle hypotheses and user-defined kinetic parameters.

<sup>2</sup>The refit of the created track with the initial input parameters for track creation would introduce a bias towards too narrow residual and pull distributions. However, for quality studies comparing reconstructed with created track origin, the unsmeared initial parameters are kept on the simulated track.

- b The **TracksFromVertexSimulation** allows to simulate tracks coming from a given vertex. The direction of the decay particle, the number and type of daughter particles, such as their directions and momenta can be defined by the user. A dedicated validation package has been developed to monitor the performance of the Vertex fitters. Again, the component software pattern automatically enhances the validation of all vertex fitters extending the a common vertex fitting interface.

- c The **GenEventSimulation** takes generated input files from any physics event generators. Similarly to the full detector simulation, only the stable particles are propagated to the track creator module, respecting their different particle type. Vertex smearing can be imposed, simply through job configuration parameters.

## FATRAS USAGE

Although introduced just recently, the new fast track simulation has been already heavily used, mainly for validation purposes. Several examples that do not claim completeness are given in the following:

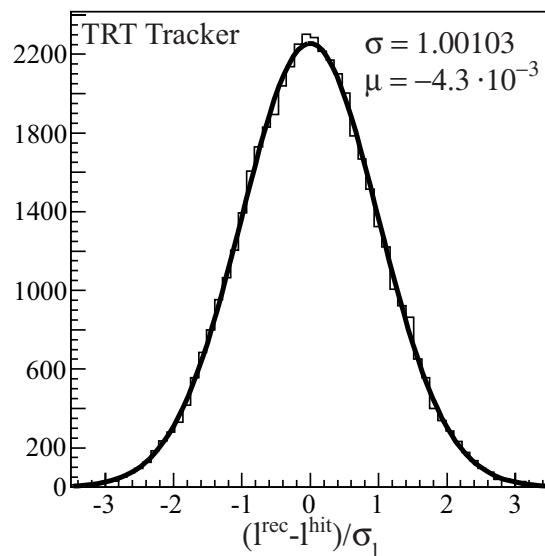


Figure 4: Pull distribution for the drift radius in the ATLAS TRT tracker for tracks simulated with Fatras and refitted with the standard Kalman fitter. The pull distribution shows a perfect standard gaussian shape.

### Fitter validation

The validation of the new track fitters implemented under a common interface has been the first usage of Fatras in the ATLAS Inner Detector. Large scale timing, reliability and performance tests have been done, Fig. 4 shows as an example the pull distribution of the drift radii in the TRT

Tracker for 5000 single  $\mu$  tracks at an energy of 5 GeV, that have been simulated by Fatras and refitted using the standard Kalman fitter. The processing time of this sample including both, simulation and refitting, has been about 96s on a 2GHZ Pentium PC.

### Momentum scale estimation

The estimation of the momentum scale of the Inner Detector reconstruction is a critical task that has a direct input on various physics results, e.g. the measurement of the W mass. Fatras has been used to verify a first-step procedure to tune the material description of the Inner Detector. Figure 5 illustrates the dependency of the reconstructed momentum on a global material scale factor; specifically in this example it shows that the initially introduced reconstruction material was overestimated. In a very similar way the effect of a wrongly scaled magnetic field description has been studied.

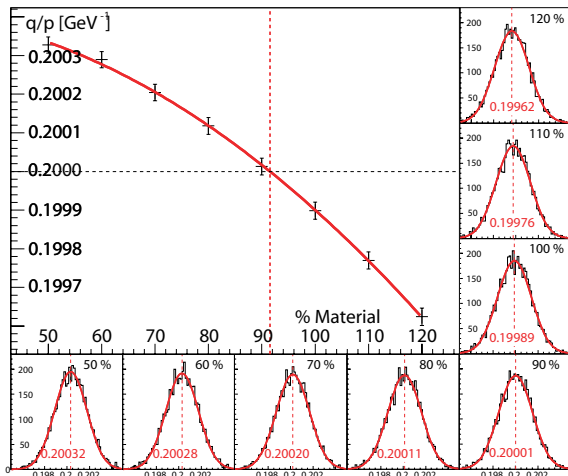


Figure 5: Simple application for the momentum scale estimation of the reconstruction. The reconstruction geometry has been simply scaled by a global scale factor in respect to the simulation setup.

## VISUALIZATION

As Fatras is fully embedded in the EDM, all ATLAS standard visualization tools can be used to display tracks, hits and noise created by the fast track simulation. An example of a simulated event from Fatras as shown by the event display ATLANTIS [8] can be seen in Fig. 6.

## CONCLUSION

A new fast track simulation has been deployed in the restructured ATLAS Inner Detector software. It is based on the extrapolation package used in the reconstruction software. The reconstruction geometry description is used as a simulation geometry, which gives a significant speed increase of up to three orders in comparison to the full de-

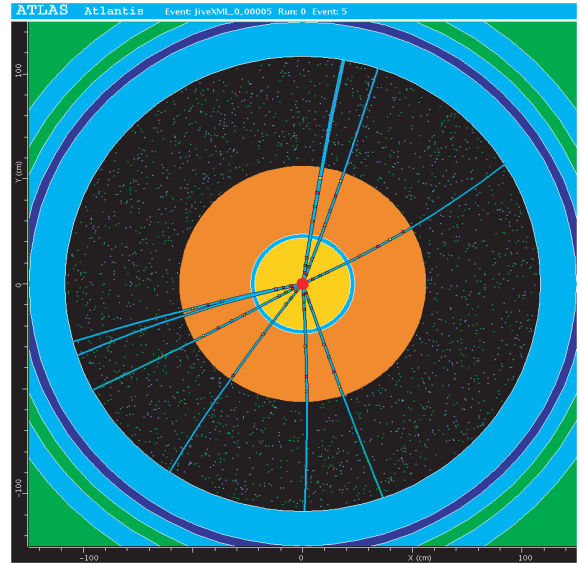


Figure 6: A sample event from the TrackFromVertexSimulation - initially intended for vertex fitter Validation - visualized with the ATLAS event display ATLANTIS

tor simulation, which is based on the Geant4 simulation toolkit. However, the fast track simulation remains to be a simplified simulation and should not be regarded as a concurrence but an addition to the fast or full detector simulation used in ATLAS. Moreover it imposes a fully controlled test bed, with great capability firstly but not only for validation applications. Such validation applications have been developed for track and vertex fitters and can be used with future software releases to guarantee a long term validation of the reconstruction software. Similar studies involving the pattern recognition process are ongoing.

## REFERENCES

- [1] D. Primor et al., Track Reconstruction with the ATLAS Detector, Proceedings of the CHEP06 Conference, 13-17 February 2006, T.I.F.R. Mumbai, India
- [2] W. Lavrijsen, Physics-level Job Configuration, Proceedings of the CHEP06 Conference, 13-17 February 2006, T.I.F.R. Mumbai, India
- [3] <https://uimon.cern.ch/twiki/bin/view/Atlas/AthenaFramework>
- [4] A. Salzburger, Track Extrapolation Package with Intrinsic Navigation in the new ATLAS Tracking Scheme, Proceedings of 9th ICATPP, 2005, Como, Italy
- [5] <http://www.hep.ucl.ac.uk/atlas/atlfast>
- [6] <http://geant4.cern.ch>
- [7] J. Boudreau, V. Tsulaia, The GeoModel Toolkit for Detector Description, Proceedings of the CHEP04 Conference, 27 Sep - 01 Oct 2004, Interlaken, Switzerland
- [8] <http://cern.ch/atlantiss>