The new ATLAS Track Reconstruction (NEWT)

T. Cornelissen, M. Elsing, I. Gavrilenko
CERN, Switzerland

W. Liebig
NIKHEF, Amsterdam, The Netherlands

E. Moyse
University of Massachusetts, USA

A. Salzburger[1]
Leopold Franzens Universität Innsbruck, Austria & CERN, Switzerland

on behalf of the ATLAS Inner Detector software group.

Abstract. The track reconstruction of modern high energy physics experiments is a very complex task that puts stringent requirements onto the software realisation. The ATLAS track reconstruction software has been in the past dominated by a collection of individual packages, each of which incorporating a different intrinsic event data model, different data flow sequences and calibration data. Recently, the ATLAS track reconstruction has undergone a major design revolution to ensure maintainability during the long lifetime of the ATLAS experiment and the flexibility needed for the startup phase. The entire software chain has been re-organised in modular components and a common event data model has been deployed. A complete new track reconstruction that concentrates on common tools aimed to be used by both ATLAS tracking devices, the Inner Detector and the Muon System, has been established. It has been already used during many large scale tests with data from Monte Carlo simulation and from detector commissioning projects such as the combined test beam 2004 and cosmic ray events. This document concentrates on the technical and conceptual details of the newly developed track reconstruction.

1. Introduction

The preparation of track reconstruction software has been an ongoing field of activity in ATLAS for more than 15 years, starting with first track reconstruction packages that have been written in FORTRAN and which were integrated in the former ATLAS reconstruction framework ATRECON. Several competing packages existed at that time for the two ATLAS tracking devices, the *Inner Detector* (ID) and the *Muon Spectrometer* (MS). These monolithic package have mostly been single-author driven and incorporated an individual event data model

[1] Corresponding author: Andreas.Salzburger@cern.ch

(EDM), a separate geometry description and specific algorithmic steering. When ATLAS moved to the new C++ based object-oriented software framework ATHENA [1], which is an enhanced version of the GAUDI [2] project that has been developed by the LHCb collaboration, these well performing and optimised applications have indeed been ported to C++, while the main block structure remained unchanged. In 2003, a dedicated reconstruction task force has realised the potential danger of this software model, in particular in conjunction with the necessary adaption of the ATLAS track reconstruction applications to a more realistic detector setup (including misalignment, distortions and an excessive use of detector conditions data), which is inevitable for the readiness of the reconstruction applications for first data taking. These extensions would have been necessary to be integrated individually in the several reconstruction packages, since a common interface level has not been compliant with their software model. During the last four years, a new track reconstruction — also referred to as *New Tracking* (NEWT) — has been deployed that is based on a common tracking EDM [3] and a coherent interface model that enhances a pure component software structure. A lot of well performing algorithmic code from the former reconstruction programs has been integrated into NEWT, but also new development has been facilitated. NEWT has been established as the default ATLAS ID reconstruction and many components of NEWT are also used in MS applications, and, furthermore, in combined reconstruction and physics analyses. In this sense, NEWT is not yet another track reconstruction program for the ATLAS experiment, but rather a collective term for the new philosophy deployed in the current track reconstruction algorithms, that are in ATLAS — for the first time — based on a common model for both tracking devices. NEWT runs in similar sequences that access mostly common tools in the reconstruction of test beam and commissioning data using cosmic rays, as well as in the offline reconstruction and the seeded third level trigger, the so-called *Event Filter* (EF). Figure 1 shows an event display for the same simulated $t\bar{t}$ event reconstructed with standard offline track reconstruction to the left, and with the region restricted EF version to the right.

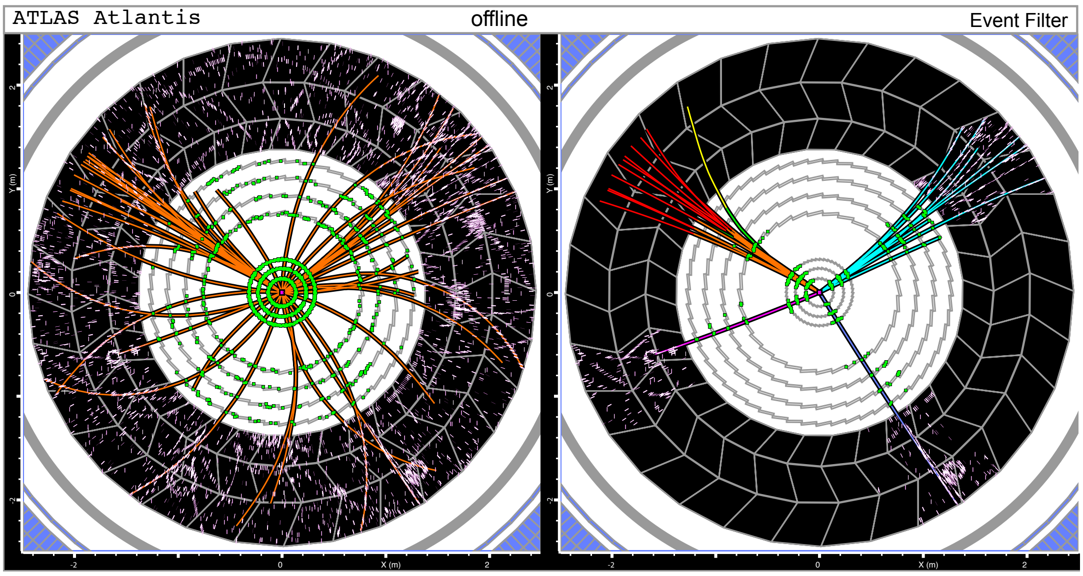

Figure 1. A simulated $t\bar{t}$ event reconstructed with the offline (left) and third level trigger versions (right) of the ATLAS New Tracking. Latter is restricted to dedicated regions that are, in general, provided by lower level trigger algorithms, but then runs an almost identical sequence and guarantees high track reconstruction performance already in the event triggering.

NEWT spans currently over about 250 different component packages and concentrates the work of many authors.

1.1. The ATHENA framework, common services and the event data model

ATHENA is a data-centered software framework that follows a *blackboard* model design: algorithms work as factories and read and write to a central transient event store (TES). Between algorithms, there is, in general, no direct dependency rather than the shared event data containers in the TES, which creates a *pseudo*-dataflow between the different algorithms. ATHENA provides interfaces for components of different levels: *services* exist during the entire program flow (such as the magnetic field service, or the TES itself), *algorithms* are executed once per event, and *tools* carry most of the algorithmic workload. The factory model requires to have a well defined event data model, such that the different components can *understand* the data on the TES written by prior algorithms. The new common ATLAS tracking EDM establishes a polymorphic structure of data classes, extending the main conceptual base classes needed for track reconstruction: a track parameterisation, a hit representation and for the convenience of the user a dedicated representation for physics analyses purposes. Many tools used in the NEWT algorithmic flow work directly on base class level, such as e.g. the extrapolation package or the various available track fitting algorithms. In the following sections we will present the main concepts of NEWT and concentrate on the actual implementation in the ATLAS ID, an exhaustive description, however, would go far beyond the scope of this document and will be presented elsewhere [4].

Common tools and interfaces Many repeating tasks and operations during track reconstruction can be performed without the need of any dedicated information about the underlying detector technology. This is in particular given for purely mathematical operations or higher level tasks that operate on base class level of the underlying tracking EDM. Track or vertex fitting is, in general, independent from the way the track information has been gathered as long as the EDM provides enough — and mathematical consistent — information about the track to perform the fit. In the ATLAS New Tracking realm, these tasks have been identified and concentrated in common tools, ordered in an intuitive package structure in a dedicated *Tracking* repository. Many tracking-relevant operations like the track fitting or the propagation algorithms are deployed in this scope, several of them in different actual realisations, but extending well defined common interfaces. This gives high flexibility to the user, who is e.g. able to switch between different track fitting techniques purely through the job configuration interface, but on the other hand stringently requires an intuitive and easy-to-use job configuration system in place[2].

2. New Tracking in the Inner Detector

The new track reconstruction of the ATLAS inner tracking device, the Inner Detector, has been the first to be completely transfered to the new component design and EDM. First tests have been carried out in the context of test beam setups in 2004, while an ongoing validation of NEWT in the ID takes place using Monte Carlo simulated events and cosmic ray runs for the detector commissioning.

2.1. The ATLAS Inner Detector

The ATLAS ID consists of three different tracking technologies:

[2] In ATLAS, this is done through auto-generated python equivalents of each tool, algorithm or service implementation, see [5].

- a silicon pixel detector — deployed as three barrel cylinders and three endcap disc structures on each side — at innermost radii that provides the most accurate track particle localisation of all measurement devices of the ATLAS detector and is mainly designed to achieve a high vertex resolution,
- a silicon strip detector (SCT) that consists of four barrel layers and 18 endcap discs in total,
- and a transition radiation tracker (TRT) at outermost radii that provides many tracking hits and a strong electron identification potential by using the transition radiation probability.

All together, the ATLAS ID provides about 87 million readout channels, 80 millions in the pixel detector, about 6 millions for the silicon strip sandwich modules, and about 400 000 channels for the TRT detector (one per each straw detector);

2.2. The ID reconstruction sequences
In modern track reconstruction there is no clear distinction between the classical modules *pattern finding* and *track fitting*. This is on the one hand due to the fact that many pattern finding strategies (contrary to a classical histogram based approach) nowadays incorporate a two stage pattern: a global pattern search, as well as a local pattern recognition where track fitting is already part of; on the other hand, many track fitters such as the combinatorial extension of the standard Kalman filter [6] or the deterministic annealing filter incorporate an intrinsic pattern recognition during the fitting process. Thus, the full chain of pattern recognition and track fitting will be in the following described as a single unit.

The ID New Tracking currently covers two sequences, the main *inside-out* track reconstruction and a consecutive *outside-in* tracking. The primary pattern search concepts for both sequences have been to a large extent adopted from the already existing ATLAS ID reconstruction program XKALMAN [7], but integrated and accomplished by additional components following the common NEWT approach. A third sequence, the *second stage pattern* recognition for the finding of V0 vertices, kink objects due to bremsstrahlung and their associated tracks has been also deployed using the common tracking tools and EDM, but is not particular to the New Tracking approach.

2.3. The Inside-out sequence
The primary ID pattern recognition starts with seeding in the inner silicon tracker and performs hit finding towards the outer border of the ID. The simplified sequence, showing only two levels of deployed tools that are used at the various stages, is illustrated in Fig. 2 as an extended UML sequence diagram. In a classical picture, many modules of this sequence can be divided into global pattern recognition and consecutive local pattern recognition that only works on the reduced output sample of the global search.

2.3.1. Global track seeding The first step in the inside-out track reconstruction is the creation of three-dimensional representations of the silicon detector measurements. Track seeds are built from these objects and by applying a window search given through the seed direction, track candidates are built. Hits from the detector elements that are within the road window are collected and judged on a simplified Kalman filtering and smoothing approach, these hits are added to the track candidates or rejected, respectively.

2.4. Ambiguity solving
The collection of track candidates that are found in the crude silicon pattern recognition contains usually many fake tracks or overlapping track segments with shared hits. A dedicated module for resolving and cleaning this initial track collection is therefore inserted as the successive step

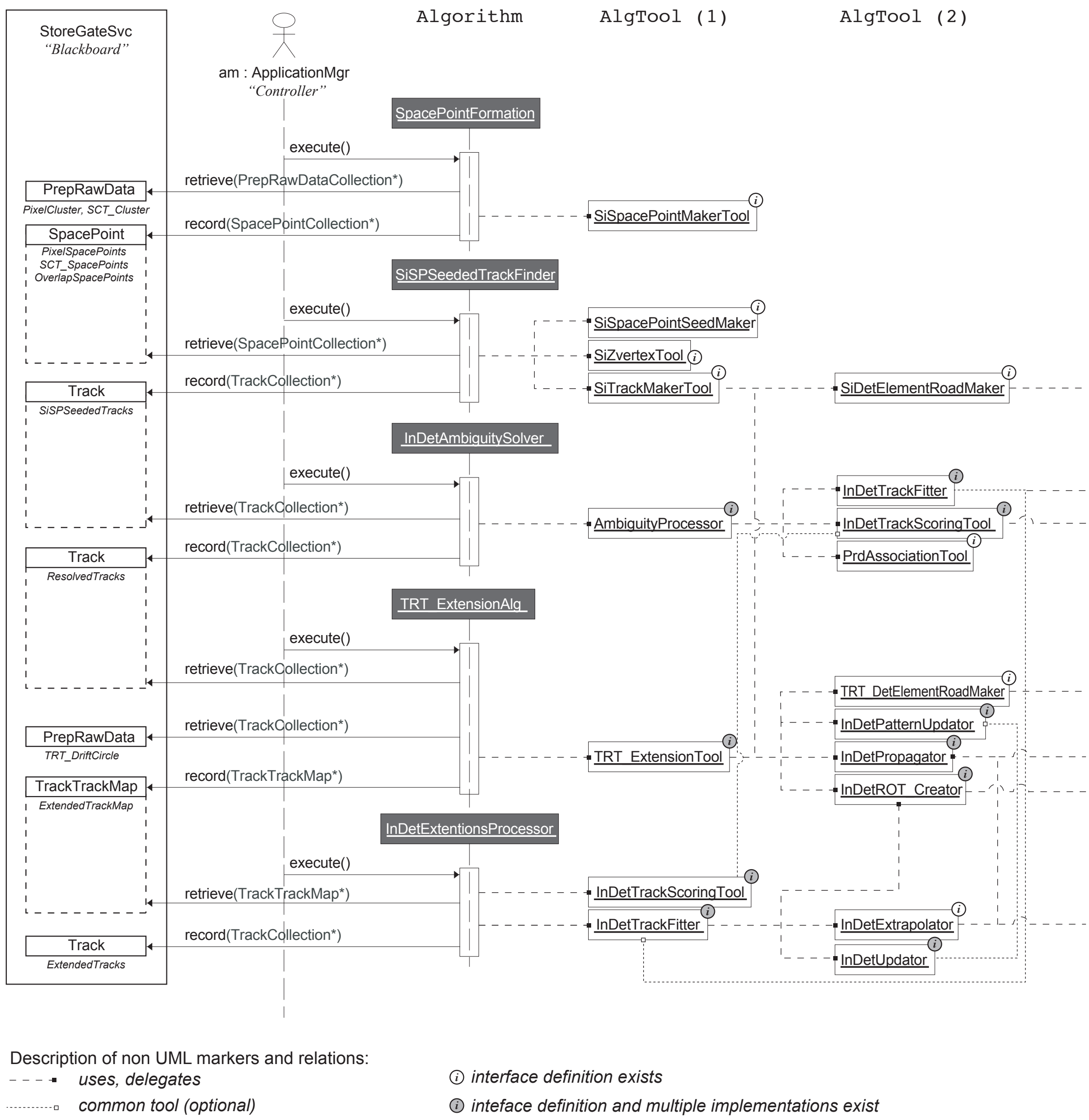

Figure 2. The main sequence of modules and some of the tool classes used in the ATLAS New Tracking for the Inner Detector illustrated as an extended UML sequence diagram. Following a blackboard architecture structure, the transient event store (StoreGateSvc) acts as the *blackboard* for reading and writing of event data, the `ApplicationMgr` as the *controller* of the system. Only two levels of embedded tool classes are shown, where dashed lines indicate a *uses* relationship, while the dotted line indicates possible sharing of tool instances.

in the algorithmic sequence. The ambiguity solving is hereby done by scoring the tracks in a reward/penalty schema with respect to another to finally achieve the best track collection.

In general, each hit associated with the track leads to a better score value to favor fully reconstructed tracks rather than small track segments. The measurements of different sub-detectors are, in general, weighted with different score additions, preferring the precision measurements (e.g. pixel clusters) and downgrading measurements from less precise detector parts. Figure 3 illustrates some of the track characteristics to be resolved in the SCT barrel detector. The ambiguity solving modules uses a scoring tool, but only refers to a tool interface, such that different scoring approaches can be in parallel be deployed and optimised. Currently, a hit-pattern based scoring and a maximum likelihood approach are available.

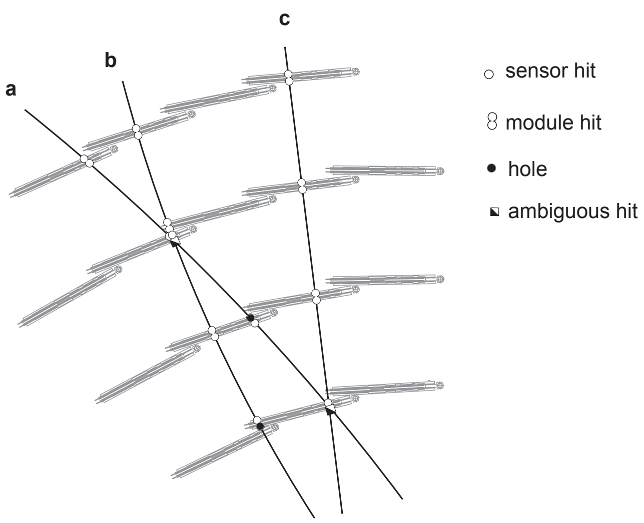

Figure 3. Simplified model of the ambiguity solving process, illustrated in the SCT barrel. Tracks **a**, **b**, and **c** have been found through the seeded track finding, but share several hits. The χ^2/n_{dof} may not be appropriate to distinguish a true from a fake track, therefore dedicated track scoring that is optimised for each sub-detector is used. In the shown example, a *module hit* representing measurements on both sides of the SCT silicon detector is scored relatively higher than two single hits without associated backside module. Hits in an overlap region as for track **b** are particularly high scored, while holes on track, i.e. an expected hit that has not been found, lead to a penalty in the track score.

2.4.1. TRT track extension The track (segment) extension from the silicon detectors into the outer TRT is split into two modules, an extension algorithm searching for seeded candidates and a second module that processes these extensions and evaluates them. Tracks found in the silicon detector and, of course, only those that have survived the first round of ambiguity solving are used as an input to find compatible sets of TRT measurements that are further processed as candidate extension. Following the component pattern, the extension algorithm simply delegates this task to a dedicated tool, represented through a pure abstract interface. The silicon-only track must hereby not be modified, the association of the TRT hits is therefore a pure extension and not done by a combination. Figure 4 shows the extensions of the silicon seeded tracks into the TRT detector for a sample $t\bar{t}$ event.

Two concrete implementations of the track extension tool exist:

- The standard implementation follows a classical approach starting with road finding through track extrapolation, and — using the hit coordinates expressed in $r-\phi$ in the barrel, and $r-z$ in the endcap region, respectively — performs a line fit to estimate whether the hit is compatible with the silicon track seed or not.
- a second implementation is available, that is based on the deterministic annealing filter (DAF) [8] — an extension of the standard Kalman filter formalism — and is optimised for very high hit occupancies. The DAF strategy also starts with the road building (and uses hereby the same tool as the first extension strategy), but follows a different philosophy for the hit finding and hit assignment; TRT measurements within the road are grouped together on the same readout element (or on planes perpendicular to the extrapolated track, depending of the initial configuration) and represented as one input object to the track fit. The different hits within the group are weighted by their likeliness to represent the true

hit, while the weights correspond mainly to the distance of the hit from the trajectory prediction. The hit assignment is then performed in a final annealing procedure.

The extension algorithm only provides candidates of TRT extensions for the seeded silicon tracks, which are written in associative containers to the TES. The second module, the TRT extension processor evaluates the given candidates then on basis of a track fit and a subsequent scoring mechanism that reuses the framework of the first ambiguity solving step, see Sec. 2.4. The resulting track collection marks the end of the inside-out chain of the ATLAS ID track reconstruction.

2.5. The Outside-in sequence

The inside-out sequence of the ID New Tracking relies on a track seed to be found in the silicon detector. Although being very efficient, see Sec. 3, not all tracks can be found through an inside-out procedure: ambiguous hits can shadow the track seed in the silicon and prevent the score of the silicon seeded track to survive the ambiguity processor on the one hand, and on the other hand, tracks coming from secondary decay vertices further inside the Inner Detector volume (e.g. K_s decays) or from photon conversions may not have any silicon hits (or only an insufficient number) to comply with the inside-out sequence. Clearly, when no candidate track in the silicon detector is found, the extension into the TRT is automatically lost. As a third source for missing the TRT extension, substantial energy loss — mostly of electrons — at outer radii of the silicon seeded track and not known to the road building may guide the extension search into the *wrong* direction. A second, reverse sequence has therefore been deployed that starts a global pattern recognition in the TRT. Track segments are identified using a standard Hough transform mechanism, while a dedicated association tool prevents hits that have already been assigned to tracks in the inside-out procedure to be used again (which saves a significant amount of CPU time).

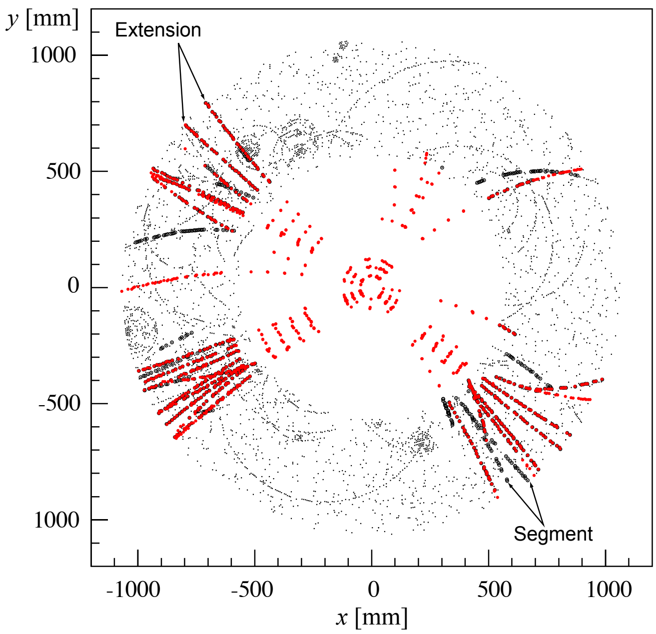

Figure 4. A sample $t\bar{t}$ event showing the two possible TRT hit associations (only the barrel measurements): the brighter colored hits show the extensions that follow the track found by seeds in the silicon, while the darker ones show TRT hits found through the stand alone track segment search. The TRT segments are then used for back tracking into the silicon, aimed at optimising the overall reconstruction efficiency.

The TRT segments are then followed back into the silicon detector (*backtracking*), which allows to find small track segments in the silicon part that have been missed in the initial inside-out stage.

3. Performance

A complete overview of the ATLAS track reconstruction performance would go far beyond the scope of this document and will be presented elsewhere. The given examples that are presented in the following are thus more exemplary and aimed at showing the readiness of the reconstruction software on the one hand, and the power of the component model on the other hand[3]. While Fig. 5 presents some results based on Monte Carlo simulated single track events, Sec. 3.1 focusses on the special aspect of electron track reconstruction and puts it into context with the component software module.

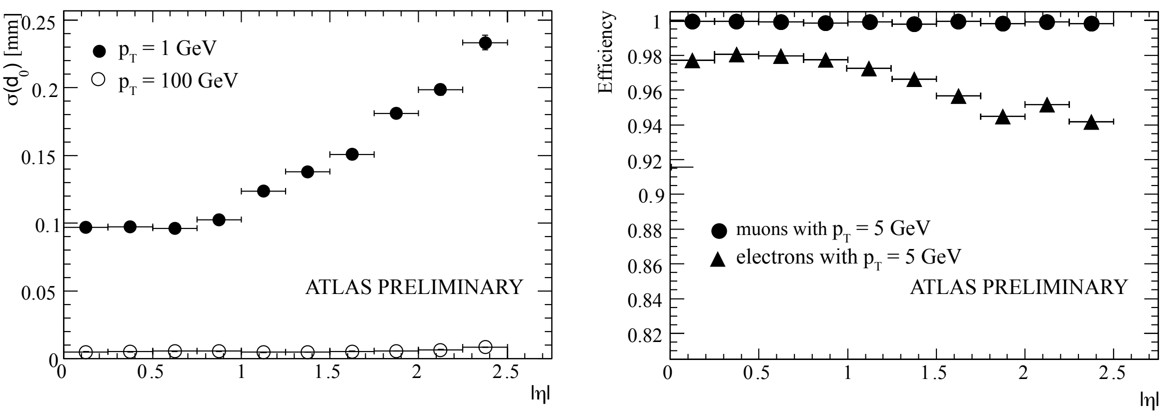

Figure 5. Single track reconstruction performance figures for the new track reconstruction in the Inner Detector; the left plot shows the transverse impact parameter resolution $\sigma(d_0)$ for muon tracks of low and high momentum versus the pseudorapidity, reaching a resolution of less than 10μm when multiple scattering can be neglected. The right plot shows the reconstruction efficiencies for muon and electron tracks with a transverse momentum of 5 GeV. The performance figures are shown for the inside-out sequence.

3.1. Electron fitting

A striking example for the power of the component software model is the fit of electron tracks. Electrons lose, in general, a significant amount of energy due to bremsstrahlung and thus the Gaussian assumption of the noise added to the track fit due to material interactions, which is inert to most track fitters, is far from being optimal. Dedicated fitting techniques, such as the *Gaussian Sum Filter* (GSF) [9] or a dynamic noise adjustment schema have been developed to enhance a better performance, either through a modified error description, or through a multi-component model as for the GSF. In ATLAS, both fitting types expand the common abstract fitter interface, such that each of the given track fitters can be exchanged at every stage of the event reconstruction, simply through a modified job configuration. Figure 6 shows as an example the improvement of the Z mass distribution for $Z \rightarrow e^+e^-$, when the electron tracks are (re-)fitted the GSF.

3.2. Timing Performance

The minimisation of the CPU time that is consumed by the reconstruction algorithm is a very important task in modern high energy physics experiments. Since large event samples have to

[3] A real evaluation of the track reconstruction performance has to be carried out anyway within the commissioning runs and first data runs. It is thus in the interest of the authors to explicitly indicate that these performance figures are obtained under the assumption of optimal conditions (i.e. perfect alignment of the detector, well known material budget, etc.), which may certainly not apply at the startup phase of the experiment in 2008.

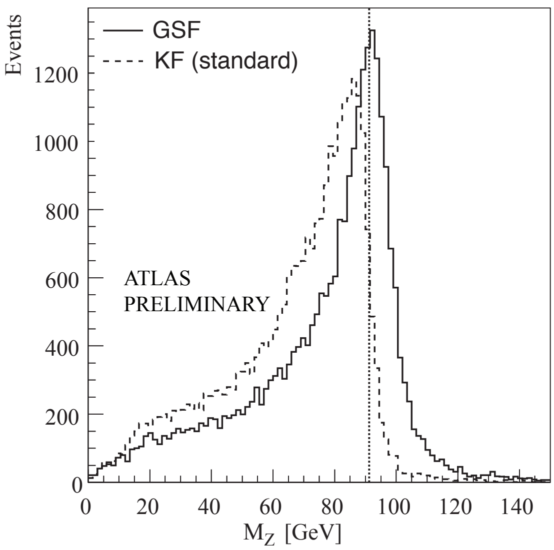

Figure 6. Improvement of the Z mass distribution for $Z \to e^+e^-$, when the electron tracks are (re-)fitted with the Gaussian sum filter instead of the standard ATLAS Kalman filter. In the standard Kalman filter, energy loss is applied only as mean ionisation loss and even approximated as being Gaussian distributed.

be reconstructed repeatedly, achieving a low execution time of the reconstruction applications is of particular interest, not only to fit with the computing budget of the experiments or associated institutes, but also to achieve a workable environment for clients and users of the reconstruction software. For the ID, an event reconstruction frequency of about 1 Hz has been targeted for relevant physics events, a number that could be achieved with the ID NEWT reconstruction. Table 1 gives an overview of the main contributions to the execution time per event for the ID NEWT reconstruction, based on the measurement of 100 $t\bar{t}$ events on an Intel® Xeon® CPU 5150 (2.66 GHz).

Table 1. Timing contributions of the main components executed in the NEWT ID sequence.

Module	**approx. time/event [ms]**
silicon space point seeded track finding	370
ambiguity solving on silicon track candidates	240
extension into the TRT	270
TRT segment finding	340

4. Conclusion

A new modular track reconstruction software has been established for the ATLAS experiment that includes a common EDM, an underlying reconstruction geometry and is based on well-defined interfaces in a component pattern design. A first complete algorithm chain has been deployed in the Inner Detector and is competitive in terms of performance and CPU time consumption to prior ATLAS reconstruction programs, while providing an open and interactive model for future modifications and adoptions. NEWT builds in addition the backbone of the ATLAS Inner Detector third level trigger by providing the seeded pattern recognition and fitting tools used in the trigger slices.

Moreover, all track reconstruction algorithms now provide the EDM objects as the output data, allowing common validation and analysis applications to work independently on the reconstruction chain used for track finding and fitting.

4.1. Conclusion and Outlook

We have presented a new track reconstruction chain that became the default track reconstruction strategy for the ATLAS Inner Detector. It is characterised by a flexible component software model that allows future extension and necessary adaptions during the long lifetime of the ATLAS experiment.

The full migration of the previous track reconstruction programs into the New Tracking schema is an ongoing effort of the ATLAS offline software project. NEWT has proven to allow both, the easy introduction of newly developed concepts as modules to the common tracking effort and the re-integration of the existing well-performing algorithms from past reconstruction packages. Latter is deliberately foreseen without wiping the identity (in both concepts and performance) of these well-tested algorithms.

A second big field of further development is the realisation of combined reconstruction and recovery and detection strategies of bremsstrahlung radiation using the ATLAS New Tracking tools. Clearly many parts of NEWT that have been described in this document can become parts of new algorithm chains that concentrate on higher stage pattern recognition, particle identification and event topology classifications.

Acknowledgments

The authors would like to thank the various contributors to both fields the conceptual design as well as the software implementation of the ATLAS New Tracking.

References

[1] Atlas Collaboration, *ATLAS Computing Technical Design Report*, ATLAS TDR, CERN-LHCC-2005-022, 2005.
[2] G. Barrand et al., *GAUDI - A software architecture and framework for building LHCb data processing applications*, Proc. of CHEP 2000, 2000.
[3] F. Akesson et al., *The ATLAS Tracking Event Data Model*, ATLAS Public Note, ATL-SOFT-PUB-2006-004, 2006.
[4] A. Salzburger (Editor) et al., *Concepts, Design and Implementation of the ATLAS New Tracking (NEWT)* ATL-PUB-SOFT-2007-007, ATLAS Public Note, 2007.
[5] W. Lavrijsen et al., *Physics-level Job Configuration*, Proceedings of CHEP 2006, Mumbai, India, 2006.
[6] R. Frühwirth et al., *Application of Kalman Filtering to Track and Vertex Fitting*, Nucl. Inst. Meth., A 262, 1987
[7] I. Gavrilenko, *Description of Global Pattern Recognition Program (XKALMAN)*, ATLAS Internal Note, ATL-INDET-97-165, 1997.
[8] A. Strandlie et al, *Adaptive multitrack fitting*, Comp. Phys. Comm 133, 2000.
[9] R. Frühwirth and A. Strandlie, *Track finding and fitting with the Gaussian-sum filter*, Proceedings of CHEP 1998, Chicago, Illinois, USA, 1998.