

DISSERTATION

CERN LIBRARIES, GENEVA

CERN LIBRARIES, GENEVA



CM-P00040552

Matthias Winkler

DISSERTATION

A comparative study of track reconstruction methods in the context of CMS physics

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines
Doktors der technischen Wissenschaften unter der Leitung von

Univ.Doz. Dipl.-Ing. Dr.techn. Rudolf Frühwirth (E107)
am Institut für Hochenergiephysik der Österreichischen Akademie der
Wissenschaften
und

Dr. Theodore Todorov (IN2P3 Strasbourg)
am CERN, European Organization for Nuclear Research

eingereicht an der Technischen Universität Wien
Fakultät für Technische Naturwissenschaften und Informatik

von

Dipl.-Ing. Matthias Winkler
Matr.-Nr.: 9326039
A-3730 Eggenburg, Engelsdorferweg 13

Wien, im Mai 2002

Kurzfassung

Diese Doktorarbeit behandelt das Problem der Spurrekonstruktion und speziell des Spurfits geladener Teilchen aus Messungen des CMS Trackers. Es ist die Aufgabe des Spurfits, die vorhandene Information (die Teilmenge der Messungen, welche zu einer Teilchenspur gehören) in einer statistisch optimalen Weise für die Schätzung der Spurparameter und der assoziierten Fehler zu verwenden. Wegen der hohen Zahl an Spuren bei LHC und dem großen Untergrund an Störmessungen haben kombinatorische Methoden wie der Kalman Filter Probleme bei der Zuordnung von Messungen zu Spuren, was zu einer Verfälschung des Resultats durch die falsche Zuordnung von Messungen zu einer Spur führen kann. Zuletzt entwickelte nicht-lineare adaptive Methoden wie der Deterministic Annealing Filter und der Multi Track Filter versuchen das Problem der Zuordnung während des Spurfits zu lösen, indem spezielle Zuordnungswahrscheinlichkeiten (“assignment probabilities”) einer Messung zu einer Spur bestimmt werden. Die Methoden wurden für den CMS Tracker verglichen anhand verschiedener Ereignistopologien.

Die Hauptteile der Doktorarbeit sind die folgenden:

- Die Einleitung in Kapitel 1 beginnt mit einem kurzen Überblick über den CMS Detektor und vor allem den CMS Tracker. Anschließend werden Grenzen der Spurrekonstruktion dargestellt, wie zum Beispiel der minimale und maximale transversale Impuls (p_T). Diese Grenzwerte wurden in einem halb-experimentellen Zugang ermittelt mit der “Fast Simulation” (Particle Gun) des CMS Trackers. In Kapitel 2 wird ein Überblick über das Objektmodell des CMS Trackers in ORCA, der objektorientierten Rekonstruktionssoftware von CMS, präsentiert, zusammen mit einer Einführung in die grundlegenden Klassen-Komponenten, welche in der Spurrekonstruktion verwendet werden.
- Das Design und die Implementierung des Kalman Filters und des Deterministic Annealing Filters in ORCA werden in Kapitel 3 beschrieben. Zur Verifizierung der Algorithmen wurde die Particle Gun in einer kontrollierten Simulationsumgebung verwendet. Durch das kontrollierte Hinzufügen von gestörten Messungen entlang der Spur wurde das Verhalten des Kalman Filters und des Deterministic Annealing Filters im Detail untersucht. Es wird gezeigt, daß der Deterministic Annealing Filter in solchen verrauschten Datensätzen zu qualitativ und quantitativ besseren Resultaten führt als der Kalman Filter. Auf analoge Weise wird das Design und die Implementierung des Multi Track Filters in Kapitel 4 erläutert und verifiziert.
- In den Kapiteln 5 und 6 werden diese Methoden auf ausgesuchte Topologien von Ereignissen angewandt und qualitativ und quantitativ verglichen. Die Ergebnisse zeigen, daß sich bei isolierten Spuren Kalman Filter und Deterministic Annealing Filter gleich verhalten und konsistente Resultate produzieren. Entscheidende Unterschiede in den Resultaten zeigen sich in Ereignissen mit

hoher Spurdichte wie zum Beispiel in kollimierten b -Jets mit hoher Transversalenergie. Hier erzeugt der Deterministic Annealing Filter qualitativ und quantitativ bessere Resultate als der Kalman Filter. Als Konsequenz führt der Deterministic Annealing Filter zu einer besseren Leistung beim Auffinden sekundärer Vertizes und beim b -tagging.

Die Leistung des Multi Track Filters wurde evaluiert anhand stark kollimierter 3-prong τ -Jets 500 GeV/ c^2 Higgs (SUSY). Die Resultate wurden mit jenen des Kalman Filters und des Deterministic Annealing Filters verglichen. Der Multi Track Filter erreicht dieselbe Auflösung wie der Deterministic Annealing Filter, aber die Fehlerschätzung ist um weitere 10% besser.

- Kapitel 7 behandelt das Problem des Ausrichtens von Spurdetektoren mit Spuren, indem zugleich ein neuer Zugang zur Lösung des Problems vorgestellt wird. Die Ausrichtung von Spurdetektoren mit Spuren wurde durch einen erweiterten Kalman Filter beschrieben und damit auf eine mathematisch-formale Grundlage gestellt. Da eine detaillierte Studie möglicher Strategien zur Ausrichtung im CMS Tracker mit all seinen Strukturen sehr kompliziert ist (und auch nicht das vorrangige Ziel dieser Arbeit darstellt), wurde dieser neue Zugang auf ein vereinfachtes simuliertes Teststrahl-Experiment angewandt.
- Kapitel 8 beinhaltet eine abschließende Zusammenfassung der Resultate der vorangegangenen Kapitel.

Die folgenden Teile der Rekonstruktionssoftware wurden vom Autor beigesteuert:

- Der Hauptteil der Arbeit bestand in einem umfassenden Design und der Implementierung des Deterministic Annealing Filters und des Multi Track Filters. Die Teile des Kalman Filters, welche den Spurfit betreffen (Smoother), wurden ebenfalls in ORCA implementiert.
- Die Particle Gun (Tracker Fast Simulation) wurde entwickelt zum Entdecken von Fehlern und zur Verifizierung der Spurrekonstruktion. Sie wurde zu einem regelmäßig verwendeten Werkzeug sowohl in der Verifizierung von Spurrekonstruktionsalgorithmen und Teilen davon, als auch zu Layout-Studien jeder Art. Sie wird ebenfalls zur Verifizierung des Vertexfits und für Teststrahl-Studien verwendet.
- Die Generierung von Startwerten von Spuren wurde entworfen und implementiert, unter Verwendung neuester Entwicklungen (Riemann Fit [1]) zur Schätzung von Kreisparametern.
- Die Spurrekonstruktion für Teststrahl-Experimente wurde entwickelt und in ORCA implementiert, basierend auf Klassen und Komponenten der regulären Rekonstruktion für den CMS Tracker.
- Die Methode des Ausrichtens von Spurdetektoren mit Spuren wurde für einen typischen Aufbau wie in einem Teststrahl-Experiment implementiert.

Abstract

This thesis deals mainly with the problem of reconstructing the tracks of charged particles from the measurements recorded by the CMS Tracker, in particular with track fitting. The task of the track fit is to use the available information (the subsets of hits belonging to the track of a particle found by a track finder) in a statistically optimal way for the estimation of the track parameters and the associated errors. Due to the high track multiplicity at LHC and the large background of noise hits, combinatorial methods such as the Kalman Filter may have problems in the association of hits to tracks, and their results will be biased in the case of wrong hits assigned to a track. Recently developed non-linear, adaptive methods such as the Deterministic Annealing Filter and the Multi Track Filter try to solve the assignment problem during the track fit, by determining an assignment probability of a hit to a track. The methods are studied in the CMS Tracker using various event topologies.

The main parts of the thesis are the following:

- The introduction in section 1 starts with a brief overview of the CMS Detector and the CMS Tracker in particular. Subsequently track reconstruction and limits of the CMS Tracker are explained, such as minimum and maximum transverse momentum (p_T). These limits have been determined using a semi-experimental approach with the Fast Simulation (Particle Gun) of the CMS Tracker. In section 2 an overview of the object model of the CMS Tracker in ORCA, the Object oriented Reconstruction for Cms Analysis, is presented together with an introduction to basic class components used in the track reconstruction.
- The design and the implementation of the Kalman Filter and of the Deterministic Annealing Filter in ORCA are described in section 3. The algorithmic verification of the implementation was done by using the Particle Gun in a fully controlled simulation environment. By adding random noise hits in a controlled way along the tracks, the behaviour of the Kalman Filter and Deterministic Annealing Filter was investigated in detail. It is shown that the Deterministic Annealing Filter leads to both quantitatively and qualitatively better results in a polluted environment. In a similar way the design and implementation of the Multi Track Filter is expounded in section 4, and results of the verification procedure are shown.
- In sections 5 and 6 these methods are applied to dedicated event topologies in a realistic environment and compared both qualitatively and quantitatively. The results show that for isolated tracks both Kalman Filter and Deterministic Annealing Filter behave equally well and produce consistent results. Significant differences in the results are observed in dense track environments such as collimated b -jets with high transverse energy. Here the Deterministic Annealing Filter produces qualitatively and quantitatively better results than the Kalman

Filter. As a consequence, the Deterministic Annealing Filter gives a better performance in terms of secondary vertex finding and b -tagging efficiency.

The performance of the Multi Track Filter was evaluated on highly collimated 3-prong τ -jets from 500 GeV/ c^2 Higgs (SUSY) and compared with the results achieved by the Kalman Filter and the Deterministic Annealing Filter. It is shown that the Deterministic Annealing Filter gives significantly better results than the Kalman Filter. The Multi Track Filter produces the same resolutions as the Deterministic Annealing Filter, but improves the error estimation by additional 10%.

- Section 7 deals with the problem of aligning a tracking detector with tracks. A new approach of solving this problem formally is presented, based on an Extended Kalman Filter. As a detailed study of possible alignment strategies in the CMS Tracker with all its sub- and super-structures is highly complicated (and was not the principal aim of this thesis), this new approach is demonstrated for a simplified simulated test-beam like setup.
- Section 8 gives a final summary of the results presented in the preceding chapters.

The following parts of the ORCA software have been contributed by the author:

- The main part of the work consisted of the complete design and implementation of the Deterministic Annealing Filter and the Multi Track Filter. The track fitting parts (smoothers) of the Kalman Filter were implemented in ORCA as well.
- The Particle Gun (Tracker Fast Simulation) was developped in order to debug and verify components used for track reconstruction. It has become a standard tool used in the verification process of the development and implementation of track reconstruction algorithms and parts of it, as well as for layout studies of any type. It is also used for the verification of the vertex fit and for test beam studies.
- The seed generation for tracks was designed and implemented, using latest developments in Riemann fits for the circle parameter estimation [1].
- Track reconstruction in test beams was developped and implemented in ORCA, based on classes and components from the standard track reconstruction for the CMS Tracker.
- Alignment with tracks was formulated within an extended Kalman Filter and therefore put on a formal mathematical base. The method was implemented for test-beam like setups and verified by a simulation experiment.

To Yasemin

Acknowledgments

I would like to thank my professor Rudi Frühwirth and my supervisor at CERN, Teddy Todorov. I have learned from Teddy a lot about object-oriented programming, and together we have written key parts of the ORCA track reconstruction framework for the CMS Tracker. Our work was not only restricted to finding the most beautiful software design, but continued during week-ends in cooking and baking competitions including wives and children. Rudi, on the other hand, tried to teach me with great patience statistics and statistical data analysis the way a mathematician would like a physicist to know about. It was a great pleasure and a great honour to work with him, and he appreciated very much the cooking competitions during his stays at CERN.

I would like to thank all my colleagues and friends for their support. Most of them are working in CMS Computing, in the CMS Tracker b/τ group and at the Institute for High Energy Physics in Vienna. Many of our best discussions we had during coffee breaks at CERN, which once again proves of the scientific importance of coffee breaks in general.

Last, not least, all my thanks to my family. All of them supported my work individually and in the best way they could. Unfortunately my grandfather and his sister have not lived to see the end of my thesis. On the other hand, my wife and my children reminded me every day that there is a sort of life besides science. They insisted that there are more important things to do, including washing the dishes, changing nappies and reading fairy tails, and they managed that I did not lose contact with Earth.

This work was made possible by the Austrian doctoral studentship programm at CERN, funded by the Austrian Ministry of Education, Science and Culture.

Preface

This thesis was written while the author was a doctoral student at CERN in Geneva. CERN is the European Organization for Nuclear Research, the world's largest particle physics centre. Founded in 1954, the laboratory was one of Europe's first joint ventures, and has become a shining example of international collaboration. From the original 12 signatories of the CERN convention, membership has grown to the present 20 Member States. CERN's aim is to explore the structure of matter and the forces which hold matter together. CERN is located in Geneva, at the border between France and Switzerland. It therefore symbolizes the transnational collaboration in fundamental physics research which is the reason for its success.

CERN employs some 2500 people, and about 6500 scientists, half of the world's particle physicists, come to CERN for their research. They represent 500 universities and over 80 nationalities.

CERN's tunnel for its biggest accelerator is 27 kilometres around and about 100 meters below the surface. Currently a new proton-proton accelerator, the Large Hadron Collider or LHC is installed in this tunnel. The LHC will bring protons into head-on collision at higher energies (14 TeV) than ever achieved before to allow scientists to penetrate still further into the structure of matter and recreate the conditions prevailing in the Universe just 10^{-12} seconds after the "Big Bang" when the temperature was 10^{16} degrees. Proton-proton bunches of the LHC will collide 40 million times per second, leading to an enormous amount of particles created in these collisions which puts new demands on radiation hard electronics and detectors as well as fast (25 nanoseconds) electronics to decide about the selection or rejection of events. The LHC is foreseen to start operation in 2006–2007.

Four experiments have been approved for the LHC, one of them is the Compact Muon Solenoid or CMS experiment. CMS is a collaboration of about 2000 scientists from 31 countries and 150 institutes. The CMS detector has a total weight of 12500 tons, an overall diameter of 15 meters and an overall length of 21.5 meters. The main components of the CMS detector are the muon chambers, the hadronic and the electromagnetic calorimeter, a superconducting magnet and a central tracking system for the reconstruction of particle tracks produced in proton-proton collisions by the LHC — the CMS Tracker. Among fundamental questions like the structure of matter or the nature of the fundamental forces which keep our universe together, CMS is built to discover the Higgs particle, the particle which is supposed to reveal why matter has mass and which has not yet been found. The CMS Tracker is supposed to play a key role in the discovery of the Higgs and other new particles.

This thesis gives a detailed overview of the track reconstruction performance with the CMS Tracker in ORCA (the Object oriented Reconstruction for Cms Analysis), using latest developments in adaptive track fitting methods and comparing these methods to the well-known Kalman Filter track fit.

Contents

Kurzfassung	I
Abstract	III
Acknowledgments	VI
Preface	VII
Table of Contents	VIII
List of Figures	XII
1 Introduction	1
1.1 The CMS Detector	1
1.2 The CMS Tracker	2
1.2.1 Physics objectives	2
1.2.2 CMS Tracker performance	3
1.2.3 CMS Tracker layout	4
1.3 CMS Tracker material budget	7
1.4 The track model	7
1.4.1 Track parameters in ORCA and coordinate frames	11
1.5 Track reconstruction limits	14
1.5.1 General limits	15
1.5.2 Minimum transverse momentum ($p_{T,\min}$)	15
1.5.3 Maximum transverse momentum ($p_{T,\max}$)	16
1.5.4 Number of simulated hits for single muons and pions	18
2 Tracker object model in ORCA	25
2.1 Tracker reconstruction geometry and sensitive volumes	26
2.2 Detector-module level simulation and reconstruction	29
2.3 Pattern recognition	32

2.4	Event level simulation and reconstruction	34
3	Track fitting methods in ORCA	37
3.1	Principles of track fitting	38
3.1.1	The Global Fit	39
3.1.2	The Kalman Filter	39
3.1.3	The Gaussian Sum Filter	41
3.1.4	The Elastic Arm Algorithm	42
3.1.5	The Deterministic Annealing Filter	42
3.1.6	The MultiRecHit	45
3.2	Methods implemented in ORCA	45
3.2.1	Design and implementation of the Kalman Filter in ORCA . .	45
3.2.2	Design and implementation of the Deterministic Annealing Filter in ORCA	51
3.3	Algorithm verification	55
3.3.1	Track parameter histograms	56
3.3.2	Track parameter overviews	57
3.3.3	Track parameters in presence of random noise hits	58
3.4	Conclusion	61
4	Multi track fitting methods in ORCA	65
4.1	Principle	65
4.2	Implementation in ORCA	67
4.3	Algorithm Verification	72
4.3.1	Track Parameters	73
4.3.2	Comparison of weight computation strategies	77
4.3.3	Effects of hit merging on p_T^{-1}	78
4.4	Conclusion	81
5	Performance studies of single track reconstruction	83
5.1	Track selection and parameter settings	83

5.2	Single muon track reconstruction performance	84
5.3	Track reconstruction efficiency of single pions	85
5.4	Impact parameters, efficiencies and fake rates in b -jets	93
5.4.1	Vertex selection	93
5.4.2	Impact parameter resolutions	94
5.4.3	Impact parameter pulls and χ^2 -probability	94
5.4.4	Track reconstruction efficiency	98
5.4.5	Track reconstruction fake rate	99
5.4.6	Secondary vertex finding efficiency	109
5.4.7	b -tagging efficiency by secondary vertex reconstruction in trigger level-2 jets	110
5.5	Mass reconstruction of $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$	112
5.5.1	Low luminosity scenario	112
5.5.2	High luminosity scenario	113
5.6	Conclusion and comparison of results with TDR [2]	115
6	Performance studies of multitrack reconstruction	121
6.1	Track selection and parameter settings	121
6.2	Simulation information	122
6.3	Efficiency of reconstructing 3 tracks in τ jets	123
6.4	Track parameter resolution in 3-prong τ jets	127
6.5	Track parameter pulls and χ^2 -probability in 3-prong τ jets	132
6.6	Conclusion	132
7	Alignment	139
7.1	Formalism	139
7.2	Design and implementation for a test-beam like setup	141
7.2.1	Overview	142
7.2.2	Tracker geometry	142
7.2.3	Alignment geometry	142

7.2.4	Tracker hits and detector readout	144
7.2.5	Track reconstructor with alignment	144
7.3	Results of a simulation experiment	145
7.3.1	The detector model	145
7.3.2	Global and local frame	145
7.3.3	Track simulation and track model	147
7.3.4	The misalignment model	147
7.3.5	The estimation procedure	149
7.3.6	Sensitivity and convergence of alignment parameters	150
7.3.7	Precision of the estimated alignment parameters	151
7.3.8	Track reconstruction after alignment	155
7.3.9	Resolution of the track parameters	155
7.3.10	Pulls of the track parameters	155
7.4	Summary and conclusion	156
7.5	Addendum to the formalism	158
8	Summary and outlook	160
	References	163
	Curriculum Vitae	164

List of Figures

1.1	Three dimensional view of the CMS Detector.	2
1.2	Three dimensional view of the CMS Tracker.	5
1.3	r - z view of the silicon part of the Tracker.	5
1.4	Three dimensional view of the pixel detectors.	6
1.5	Tracker material budget: radiation lengths of subdetectors.	8
1.6	Tracker material budget: radiation lengths of both sensitive and non-sensitive volumes.	9
1.7	Tracker material budget: absorption lengths.	10
1.8	Helix parameters in the x - y plane projection.	11
1.9	Helix parameters in the $R\Phi$ - z plane projection.	11
1.10	Definition of the curvilinear frame.	13
1.11	Definition of the local frame.	13
1.12	Definition of the measurement frame.	15
1.13	Definition of the minimum transverse momentum $p_{T,min}$	16
1.14	Minimum p_T as a function of η	17
1.15	Definition of the sagitta s	18
1.16	Average p_T of tracks.	19
1.17	Average number of simulated hits per simulated muon track.	20
1.18	Average number of simulated hits per simulated pion track.	21
1.19	Fraction of muon tracks with more than five or eight hits.	22
1.20	Fraction of pion tracks with more than five or eight hits.	23
1.21	Last positions of pions.	24
2.1	Definition of a class.	25
2.2	A simple example of inheritance.	26
2.3	Principle of polymorphism.	27
2.4	Class diagram of the Det.	28
2.5	Inheritance tree for the DetType class.	28
2.6	Inheritance tree for the Surface abstract base class.	30
2.7	The CmsTracker pure abstract base class.	30

2.8	The relationship between the DetUnit and the SimDet.	30
2.9	The SimHit class.	31
2.10	The RecHit class.	31
2.11	The TkHitAssociator class.	32
2.12	The TrajectoryStateOnSurface class.	32
2.13	Class diagram for the Propagator.	33
2.14	The MeasurementEstimator class.	33
2.15	The Updator pure abstract base class.	34
2.16	The TrajectoryMeasurement class.	34
2.17	The Track class.	35
2.18	The TrackAssociator class.	36
3.1	Class diagram of the ModularKFRConstructor.	47
3.2	The TrajectorySeed class.	47
3.3	The SeedGenerator class.	47
3.4	The Trajectory class.	48
3.5	The TrajectoryBuilder class.	49
3.6	The TrajectorySmoother class.	49
3.7	The TrajectoryCleaner class.	50
3.8	Sequence diagram of the ModularKFRConstructor.	51
3.9	Class diagram of the DAFReconstructor.	52
3.10	Sequence diagram of the DAFReconstructor.	53
3.11	Class diagram of the HitCollector.	53
3.12	Class diagram of the DAFTrajectorySmoother.	54
3.13	Sequence diagram of the DAFTrajectorySmoother.	54
3.14	Typical pull histograms and χ^2 -probability using the ParticleGun. . .	57
3.15	Typical pull histograms and χ^2 -probability with the Particle Gun and clusterized RecHits.	58
3.16	Overview of RMS of pulls.	59
3.17	Parameter resolution of the DAF with respect to the Kalman Filter. .	60
3.18	Pull distributions for the DAF with one additional random noise hit.	61

3.19	Overview of the pulls p_T , the transverse impact parameter and the mean χ^2 probability.	62
3.20	Overview of the relative resolution of the DAF with respect to the Kalman Filter.	63
3.21	Evolution of the RMS of pulls in presence of random noise hits. . . .	64
4.1	Class diagram of the MTFReconstructor.	68
4.2	Sequence diagram of the MTFReconstructor.	69
4.3	Class diagram of the MTFHitCollector.	69
4.4	Sequence Diagram of the MTFHitCollectorFromTrackFinder.	70
4.5	Implementation of the MTFSmotherBase.	71
4.6	Sequence diagram of the MFTTrajectorySmoother.	71
4.7	Class diagram of the MTFMultiRecHitUpdater.	72
4.8	Class inheritance diagram of the MTFWeightCalcStrategy.	72
4.9	Pulls of the Multitrack Filter with Gaussian smeared hits.	74
4.10	Pulls of the Multitrack Filter with clusterized hits.	75
4.11	Overview of RMS of pulls and mean χ^2 -probability for different types of track filters and smoothers.	76
4.12	Overview of the resolutions of the MTF with respect to the Kalman Filter.	77
4.13	Overview of pull RMS and mean χ^2 -probability values of the MTF smoother with different update strategies of the assignment weights. .	78
4.14	Overview of relative resolutions of the MTF smoother.	79
4.15	Average number of simulated hits and reconstructed hits per two close tracks.	79
4.16	Hit resolution and pull for pixel barrel hits for single tracks.	80
4.17	Hit resolution and pull for pixel barrel hits for very close pairs of tracks.	81
4.18	Bias of p_T^{-1} pulls in units of standard deviations.	82
5.1	Residuals of the DAF for $p_T = 10$ GeV/ c muon tracks.	86
5.2	Pulls of the DAF for $p_T = 10$ GeV/ c muon tracks.	87
5.3	Comparison of the track parameter resolutions of single muons. . . .	88
5.4	Comparison of the track parameter pulls of single muons.	89
5.5	Bias of residuals for single muons.	90

5.6	Track reconstruction efficiency of single muons.	91
5.7	Track reconstruction efficiency of single pions.	92
5.8	Transverse impact parameter resolution in $E_T = 200$ GeV, $0 < \eta < 0.7$ b -jets.	94
5.9	Longitudinal impact parameter resolution in $E_T = 200$ GeV, $0 < \eta < 0.7$ b -jets.	95
5.10	Impact parameter resolution in $E_T = 200$ GeV b -jets.	95
5.11	Transverse impact parameter pulls in $E_T = 200$ GeV, $0 < \eta < 0.7$ b -jets.	96
5.12	Longitudinal impact parameter pulls in $E_T = 200$ GeV, $0 < \eta < 0.7$ b -jets.	97
5.13	χ^2 -probability in $E_T = 200$ GeV, $0 < \eta < 0.7$ b -jets.	97
5.14	Impact parameter pulls in $E_T = 200$ GeV b -jets.	98
5.15	Track reconstruction efficiency in $E_T = 50$ GeV b -jets.	99
5.16	Track reconstruction efficiency in $E_T = 100$ GeV b -jets.	100
5.17	Track reconstruction efficiency in $E_T = 200$ GeV b -jets.	101
5.18	Overall track reconstruction efficiency in b -jets for DAF and KF.	102
5.19	Track reconstruction fake rate in $E_T = 50$ GeV b -jets.	103
5.20	Track reconstruction fake rate in $E_T = 100$ GeV b -jets.	104
5.21	Track reconstruction fake rate in $E_T = 200$ GeV b -jets.	105
5.22	Overall track reconstruction fake rate in b -jets.	106
5.23	Integral of fake tracks as a function of the number of effective hits for the DAF.	107
5.24	χ^2 -probability distributions of the DAF for fake tracks.	108
5.25	Secondary vertex finding efficiency in b -jets.	109
5.26	b -tagging efficiency in L2 jets.	110
5.27	b mistagging in L2 jets.	111
5.28	Simulation values of the decay $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$	113
5.29	Reconstructed mass and momentum of the $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$ for the low luminosity scenario.	114
5.30	Transverse and longitudinal impact parameter of the muons.	115
5.31	Bias in reconstructed mass as function of η and φ of the $B_{(d)s}^0$ mass.	116

5.32	econstructed mass and momentum of the $B_{(d)s}^0$ for high luminosity. . .	117
5.33	Transverse and longitudinal impact parameter of the muons.	118
5.34	TDR-like overview.	119
6.1	The τ radial distance, η , ΔR and p_T distributions.	124
6.2	The number of selected simulated tracks of τ s.	125
6.3	Fraction of reconstructed 3-prong τ s with three tracks.	126
6.4	Global track reconstruction efficiency of three tracks.	127
6.5	Algorithmic reconstruction efficiency of three tracks.	128
6.6	Transverse impact parameter resolution at the τ vertex.	129
6.7	Longitudinal impact parameter resolution at the τ vertex.	130
6.8	Transverse momentum resolution at the τ vertex.	131
6.9	Pull distributions of the transverse impact parameter.	134
6.10	Pull distributions of the longitudinal impact parameter.	135
6.11	Pull distributions of the inverse transverse momentum.	136
6.12	χ^2 -probability distributions for the KF, the DAF and the MTF. . . .	137
6.13	Average number of simulated and reconstructed hits per layer in a τ jet.	138
7.1	Object model of the Tracker geometry.	143
7.2	Object model of the alignment geometry.	143
7.3	Class collaboration diagram for the detector readout and the HitPop- ulatorFromGun.	144
7.4	Object model of the track reconstructor.	145
7.5	Layout of the simulated test beam tracker.	146
7.6	Definition of the local coordinate frame for a strip detector.	146
7.7	Illustration of shifting and rotating the planes for misalignment. . . .	148
7.8	Development of estimated alignment parameters.	152
7.9	Residual histograms of the three alignment parameters.	153
7.10	Standard deviation of the residuals.	153
7.11	Pull histograms of the three alignment parameters.	154
7.12	Standard deviation of the standardized residuals (pulls).	154

7.13 Histograms of the position resolution.	156
7.14 Histograms of the pulls.	156

1 Introduction

1.1 The CMS Detector

The CMS (Compact Muon Solenoid) Detector [3, 4, 5, 6] is one out of four particle detectors built for the new proton-proton collider LHC (the Large Hadron Collider) [7] at CERN [8], the European Organization for Nuclear Research. The LHC will produce proton-proton collisions far beyond the reach of present accelerators, at an energy of 14 TeV in the center of mass, with quarks and gluons interacting at energies up to several TeV. LHC and CMS are therefore designed to answer fundamental open questions in particle physics:

- What is the origin of mass? The Standard Model predicts the existence of the Higgs boson, but the latter has not been observed to date.
- Is there experimental evidence to support the hypothesis of Grand Unification of the three fundamental interactions — electromagnetic, weak, and strong?
- What is the origin of “dark matter” in the universe? Is there new physics beyond the Standard Model, supersymmetry, with new particles that could explain the existence of “dark matter”, the neutralinos?
- Why is there an asymmetry in the quantity of matter as compared to antimatter?
- Why are there only three families of quarks and leptons?
- Do elementary particles have a substructure?
- Is there a new form of matter, the so-called quark-gluon plasma, as it should have existed in the early universe?

The CMS Detector is designed to fully exploit the discovery potential offered by LHC both at low and at high luminosities. The basic design criteria are therefore the following:

1. a very good muon (μ^\pm) identification and momentum measurement,
2. measurement of the energies of photons (γ) and electrons (e^\pm) with best possible precision and consistent with (1),
3. a central tracking detector for the precise momentum and impact parameter measurements of charged particle tracks in order to achieve (1) and (2),
4. robustness and cost effectiveness.

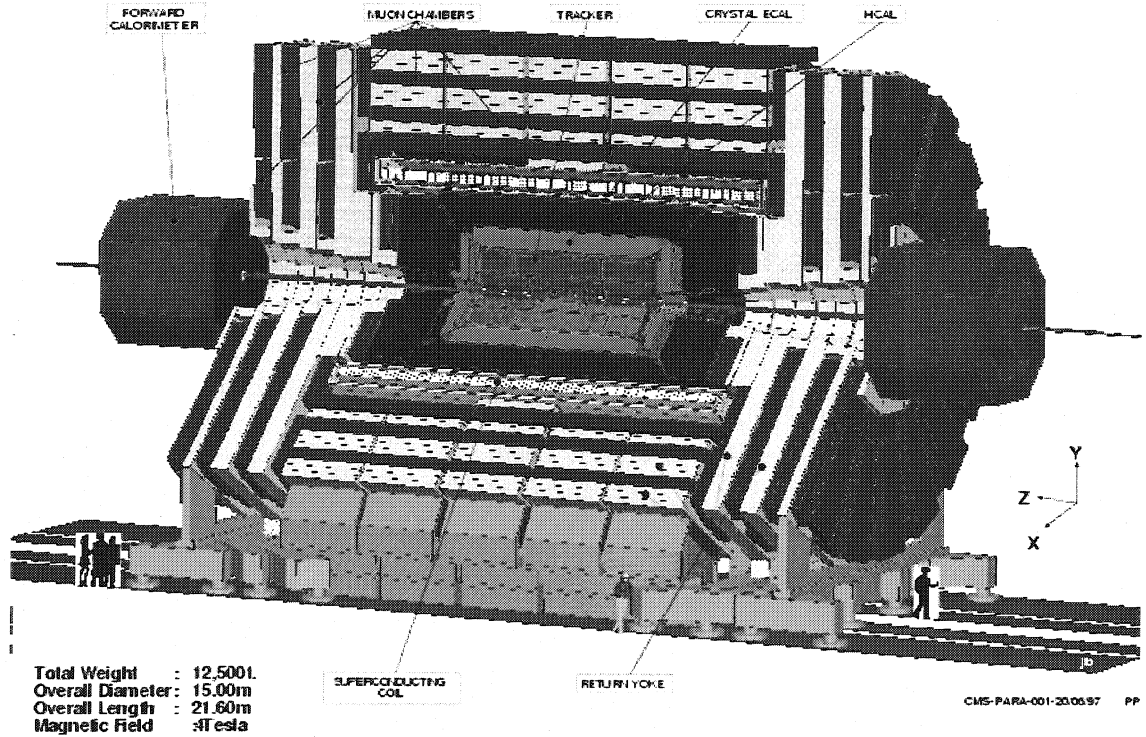


Figure 1.1: Three dimensional view of the CMS Detector. The detector is of a cylindrical shape, approximately symmetric in the azimuthal angle ϕ . Its overall dimensions are 15 m in diameter and 21 m in length. A superconducting solenoid generates a uniform magnetic field of 4 T inside the coil.

1.2 The CMS Tracker

The CMS Tracker [9, 10, 2, 11] constitutes the central part of the CMS Detector. As the CMS Tracker plays the key role in track reconstruction at CMS, a short overview is presented in the following.

1.2.1 Physics objectives

Robust track and vertex reconstruction play an essential role in exploiting the full range of physics accessible at the LHC. The detection of events involving Gauge bosons (W^\pm, Z^0) will be of primary importance. In particular, their leptonic decays provide especially clean experimental signatures. To fully exploit these signatures, the CMS Tracker has to provide an optimal momentum and position measurement for energetic leptons. Both resolution and efficiency are affected by the material within the tracker (energy loss, multiple scattering, bremsstrahlung), therefore it is desirable to reduce the amount of material within the tracker as much as possible. In addition to an efficient reconstruction of leptons and also photons, the CMS Tracker must establish that they are in fact isolated in order to suppress sufficiently backgrounds ($t\bar{t}, Zb\bar{b}$), for example to allow the observation of a Higgs boson decaying into four

leptons:

$$H \rightarrow ZZ^{(*)} \rightarrow 4l^{\pm}.$$

Effective isolation criteria rely on the efficient reconstruction of hadronic particles down to transverse momenta as low as $p_T = 1 \text{ GeV}/c$.

The ability of both tagging and reconstructing b - and τ - jets in detail is also of fundamental importance. They are not only of particular interest for studying signatures of a broad spectrum of new physics (SUSY), but they may also give insights into the question of CP violation. This requires the ability of identifying tracks coming from multiple vertices, to reconstruct the possible decay chains and to measure the b decay proper time. Both b tagging for high momenta jets coming from new particles and resulting in severe constraints on multiple track separation capability and efficient b tagging over the full CMS Tracker acceptance volume are required. Therefore it is mandatory to control tails in impact parameter measurements due to track fitting errors and material interactions.

Most of the interesting physics questions require the highest luminosity achievable at LHC, up to $10^{34} \text{ cm}^{-2} \text{ s}^{-1}$. This requires not only the development of radiation hard tracking detector and electronics, but puts also an additional burden on the task of correct pattern recognition, as typically 20 unrelated minimum bias events are superimposed at each bunch crossing.

LHC will also provide a unique opportunity to explore the physics of heavy-ion collisions at very high energy. In the case of lead-lead collisions this will result into an extremely large multiplicity of charged particles, up to 25000. Although the CMS Tracker was not designed for such physics, it can actually provide reconstruction of muon-pair invariant masses in conjunction with the CMS Muon system. This, in turn, is a powerful tool to study quark-gluon plasma physics which may be evidenced in heavy-ion collisions at LHC.

1.2.2 CMS Tracker performance

According to the CMS Tracker TDR [2] the performance of the CMS Tracker can be summarized as follows:

- The transverse momentum resolution of isolated tracks in the central tracker region $-1.6 < \eta < 1.6$ is better than $\delta p_T/p_T \approx (15 \cdot p_T \oplus 0.5)\%$, gradually degrading to $\delta p_T/p_T \approx (60 \cdot p_T \oplus 0.5)\%$ with η approaching 2.5 (p_T in TeV/c).
- In conjunction with the CMS Muon system, the momentum resolution of muon tracks with $p_T 100 \text{ GeV}/c$ can be parametrized as

$$\delta p_T/p_T \approx (4.5 \cdot \sqrt{p}) \%,$$

with p in TeV/c , for the rapidity range up to $|\eta| = 2$.

- The transverse impact parameter resolution in the plane perpendicular to the beam line is better than $35\mu\text{m}$ over the full rapidity range. The longitudinal impact parameter resolution is better than $75\mu\text{m}$. Degradation effects of misalignment are not taken into account in these figures.
- The reconstruction efficiency for muons is better than 98% over the full η range. Charged hadrons with a p_T above $10\text{GeV}/c$ are reconstructed with an efficiency close to 95%, hadrons with a p_T of the order of $1\text{GeV}/c$ are reconstructed with an efficiency of about 85%. High energy electrons are reconstructed with an efficiency above 90%.
- The tagging efficiency of b -jets is 50% in the central rapidity range and 40% in the forward rapidity range, for jets ranging from 50 GeV to 200 GeV of transverse energy and 1%–2% mistagging probability.

This pattern recognition performance and track reconstruction quality should be achieved at the highest luminosity foreseen at LHC.

1.2.3 CMS Tracker layout

Due to the high event rate and particle flux (luminosity) at LHC, the technological requirements are

- radiation hard detectors,
- fast detector response (25 ns), and
- a sufficient number of channels per detector to keep the occupancy at the level of a few percent.

The Tracker is designed to detect particles in the rapidity range $|\eta| \leq 2.5$. In the CMS-125 layout the Tracker has three pixel layers and ten silicon strip layers in the barrel, plus two pixel layers, three inner, and nine outer forward disks of silicon detectors in each end cap (see Fig. 1.2). Up-to-date information about the layout can be found at [12].

The momentum reconstruction in CMS makes use of a 4 T magnetic field provided by a super-conducting solenoidal magnet [13]. This high magnetic field affects event topologies, by confining low p_T tracks of charged particles to helical trajectories with small radius. Together with the steeply falling p_T spectrum of minimum bias events, this results in a track density rapidly decreasing with increasing radius.

The outer radius of the CMS Tracker extends up to 107–110 cm. The length of the CMS Tracker is approximately 270 cm on each side of the interaction point (see Fig. 1.3). Momentum reconstruction within the rapidity range of $-1.6 < \eta < 1.6$ benefits from the full momentum analysing power.

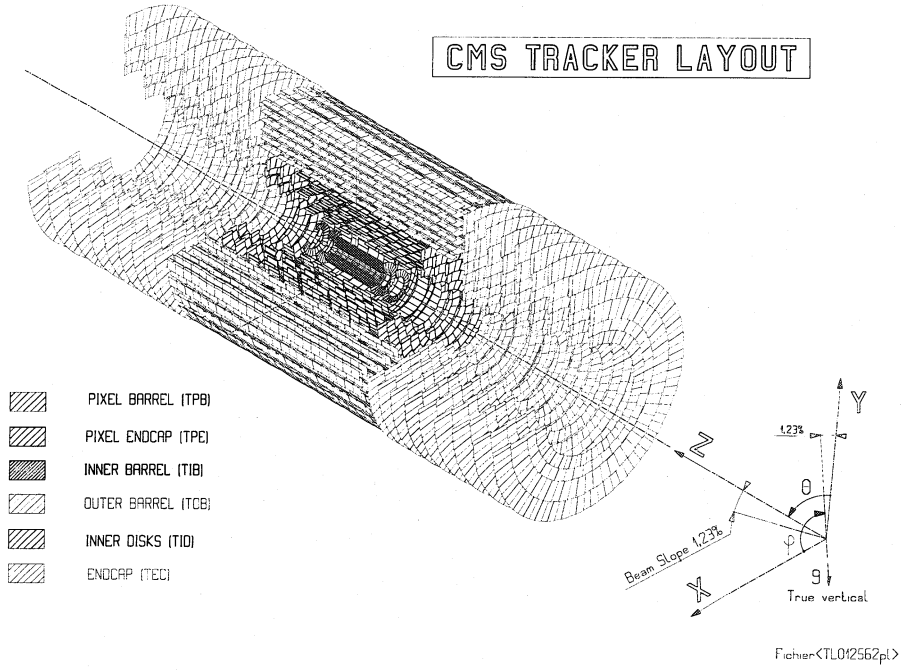


Figure 1.2: Three dimensional view of the CMS Tracker. The Tracker consists of three different parts, a cylindrical barrel and two disk-like end caps. Two different types of modules are used, pixel detectors for the very central part of the Tracker, and silicon strip modules for the inner and outer parts of the detector.

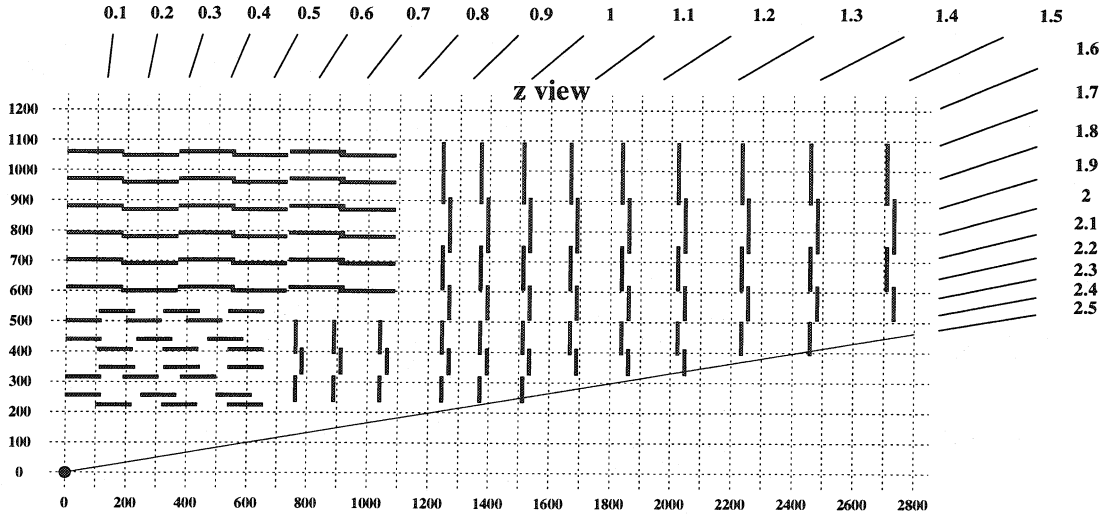


Figure 1.3: r - z view of a quadrant of the silicon part of the Tracker. Positions in r (vertical axis) and z (horizontal axis) are in mm. Red lines represent single-sided modules, blue lines double-sided modules. The inner barrel has four layers, the outer barrel has six layers. The inner end cap is made of three small disks on each side which close the inner barrel. The outer endcap is made of nine big disks on both sides of the Tracker [12].

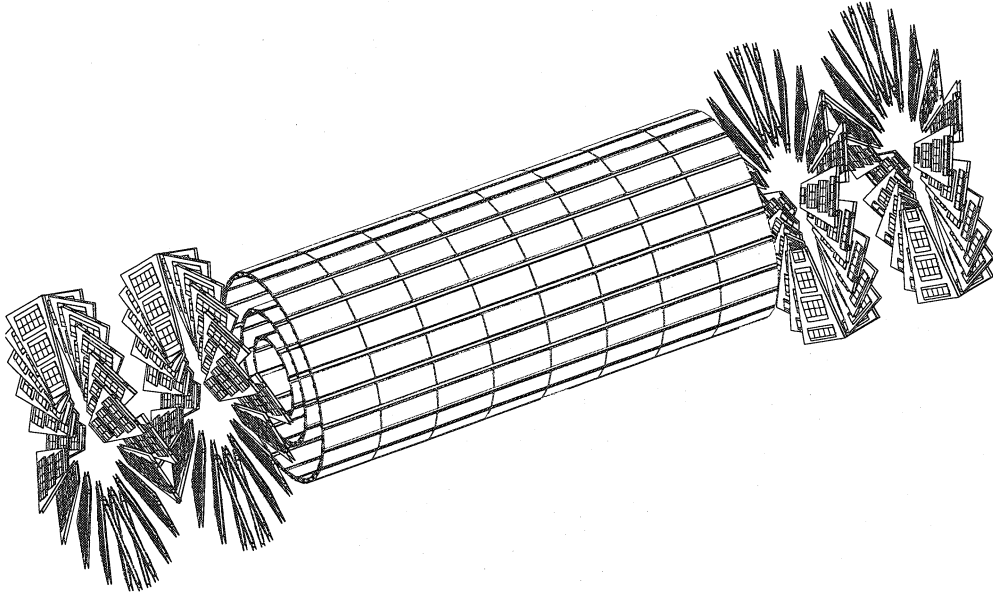


Figure 1.4: Three dimensional view of the pixel detectors (CMS-123 layout). The three barrel pixel layers are approximately at the distance of $r = 4, 7$ and 11 cm from the center of the detector. The two pixel endcaps on each side are approximately at $z = \pm 35$ and ± 47 cm from the center.

The CMS Tracker is built of 15148 silicon strip modules in total. A list with silicon sensor pitch and hit resolution information for CMS-125 is shown in the following table:

part	detectors	thickness μm	pitch μm	resolution μm
inner barrel	2724	320	81/118	21/31
outer barrel	5208	500	123/183	31/50
inner disks	816	300	97/128/143	26/36/40
full disks	2512	300	96/126/128/143	26/35/36/40
	3888	500	143/158/183	40/44/51

Figures for pitch and resolution are approximate only.

Pixel detectors measure two coordinates simultaneously with a high precision (about $10 \mu\text{m}$). In the CMS-125 layout the pixel detector consists of three barrel layers and two end cap layers on each side (see Fig. 1.4), but other layout configurations are being studied. Due to its high position resolution the pixel detector determines the impact parameter of charged particles. It allows vertex reconstruction in three dimensions and permits the identification of b - and τ -jets.

1.3 CMS Tracker material budget

The material within the CMS Tracker volume is composed of sensitive detector volumes and non-sensitive material. Due to the fact that the entire Tracker has to be cooled down to -10 degrees, a large fraction of the non-sensitive material consists of cooling services. Other non-sensitive material parts are cables, support structures, electronics, the beam-pipe and the thermal screen at the outside of the Tracker.

The latest analysis of the Tracker material budget for the CMS-125 layout has shown a significant increase in radiation lengths compared to the one of the Tracker TDR. This effect comes for two reasons, first the replacement of the microstrip gas chamber (MSGC) detectors in the outer part of the Tracker by silicon modules, and second a better understanding of non-sensitive materials like cooling and support structures.

The decomposition of the Tracker material in units of radiation lengths in terms of subdetectors is shown in Fig. 1.5, the decomposition in terms of sensitive and non-sensitive volumes can be seen in Fig. 1.6 [14].

Finally, the material in terms of absorption length is shown in Fig. 1.7 [14].

1.4 The track model

The equation of motion of a charged particle in a static magnetic field $\mathbf{B}(\mathbf{x})$ is described by the *Lorentz force* derived from the Maxwell equations [15, 16]:

$$\mathbf{F} = \frac{d}{dt} \left(m \frac{d\mathbf{x}}{dt} \right) = q\mathbf{v} \times \mathbf{B}(\mathbf{x}).$$

\mathbf{F} is the *Lorentz force*, q is the signed charge of the particle, \mathbf{v} is the velocity and $\mathbf{B}(\mathbf{x})$ is the static magnetic field. In the notation where $s(t)$ is the distance along the trajectory (path length) and $v = ds/dt$ is the absolute value of the velocity \mathbf{v} ,

$$\frac{d\mathbf{x}}{dt} = \frac{d\mathbf{x}}{ds} \frac{ds}{dt} = \frac{d\mathbf{x}}{ds} v,$$

$$\frac{d^2\mathbf{x}}{dt^2} = \frac{d^2\mathbf{x}}{ds^2} v^2,$$

the above equation can be rewritten in terms of geometrical quantities only:

$$\frac{d^2\mathbf{x}}{ds^2} = \frac{q}{mv} \frac{d\mathbf{x}}{ds} \times \mathbf{B}(\mathbf{x}(s)).$$

From this equation follows that there are six integration constants. However, with the identity

$$\left(\frac{dx}{ds} \right)^2 + \left(\frac{dy}{ds} \right)^2 + \left(\frac{dz}{ds} \right)^2 = 1$$

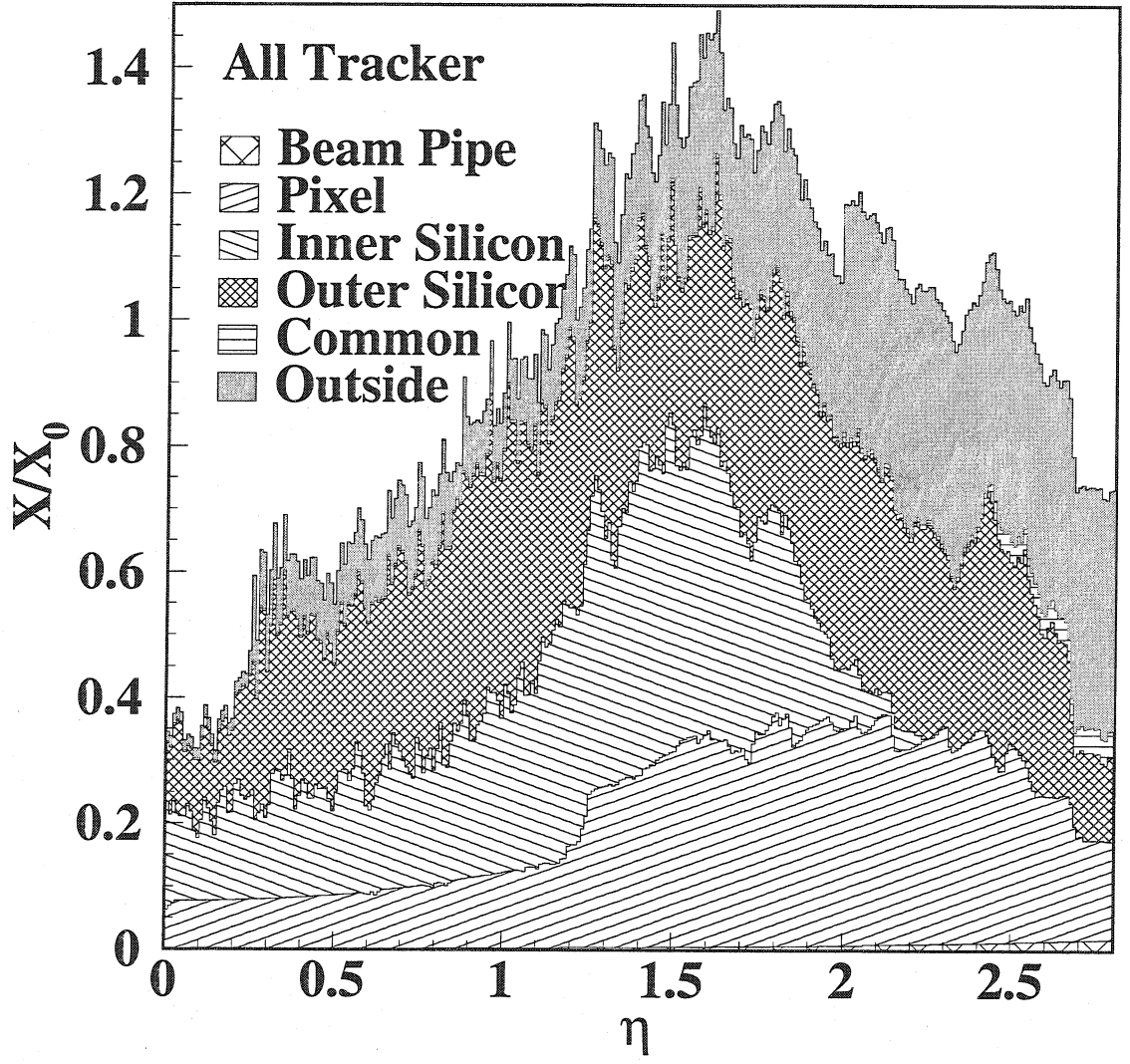


Figure 1.5: Tracker material in the CMS-125 layout in units of radiation lengths, as a function of η for the different sub-detectors [14].

and an arbitrary choice of one coordinate (the reference surface), there are only five free parameters defining the track with known mass of the particle: two parameters for the impact point at a reference surface at $z = \text{const.}$, two for the direction of the track at that point, and one for the inverse momentum.

In the special case of a homogeneous magnetic field parallel to the z -axis, $\mathbf{B} = B\mathbf{e}_z$,

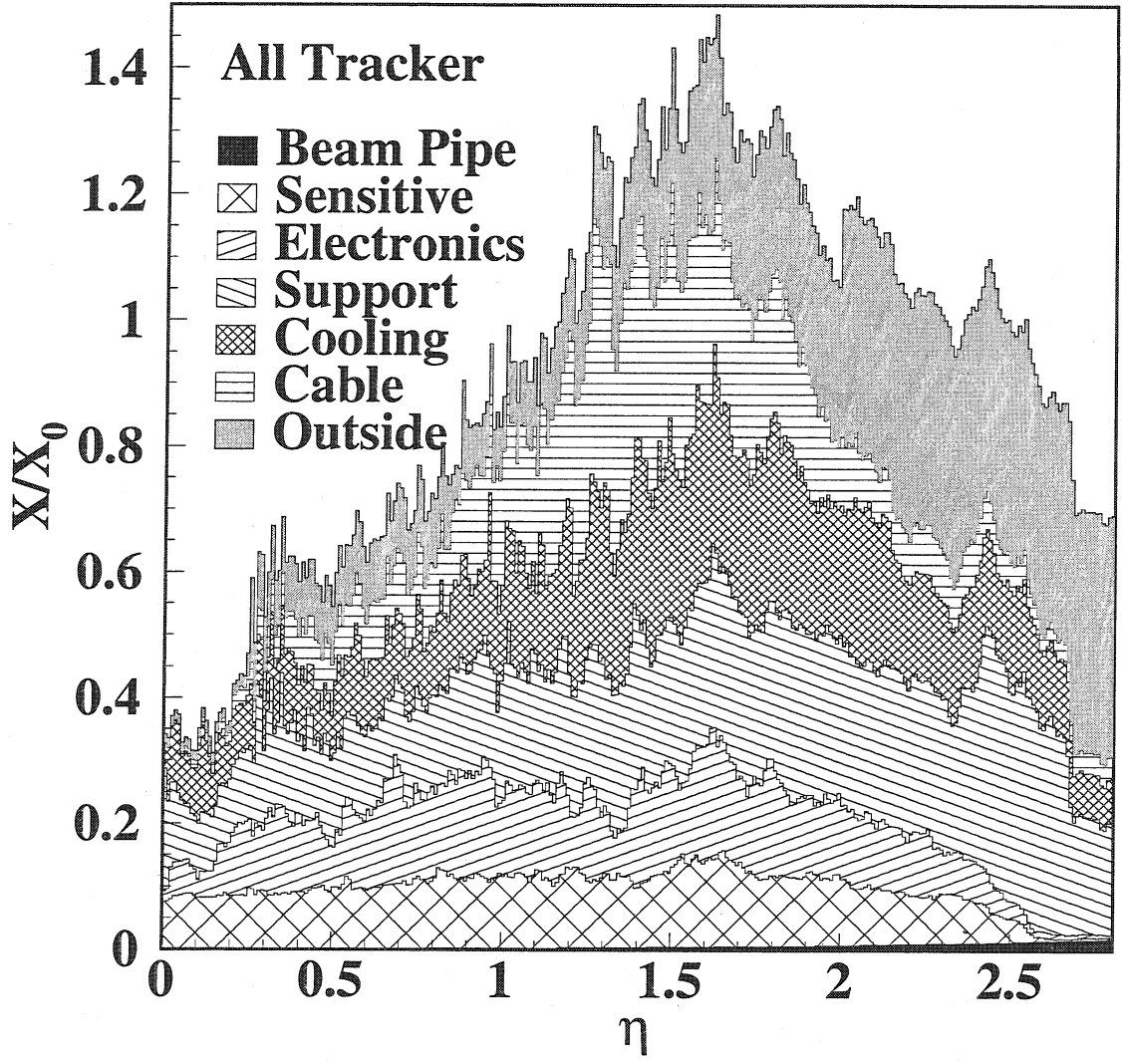


Figure 1.6: Tracker material in the CMS-125 layout in units of radiation lengths, as a function of η for sensitive and non-sensitive volumes [14].

$\mathbf{e}_z = (0, 0, 1)^T$, the above equation takes the form

$$\begin{aligned}\frac{d^2x}{ds^2} &= \frac{q}{mv} \frac{dy}{ds} B, \\ \frac{d^2y}{ds^2} &= -\frac{q}{mv} \frac{dx}{ds} B, \\ \frac{d^2z}{ds^2} &= 0.\end{aligned}$$

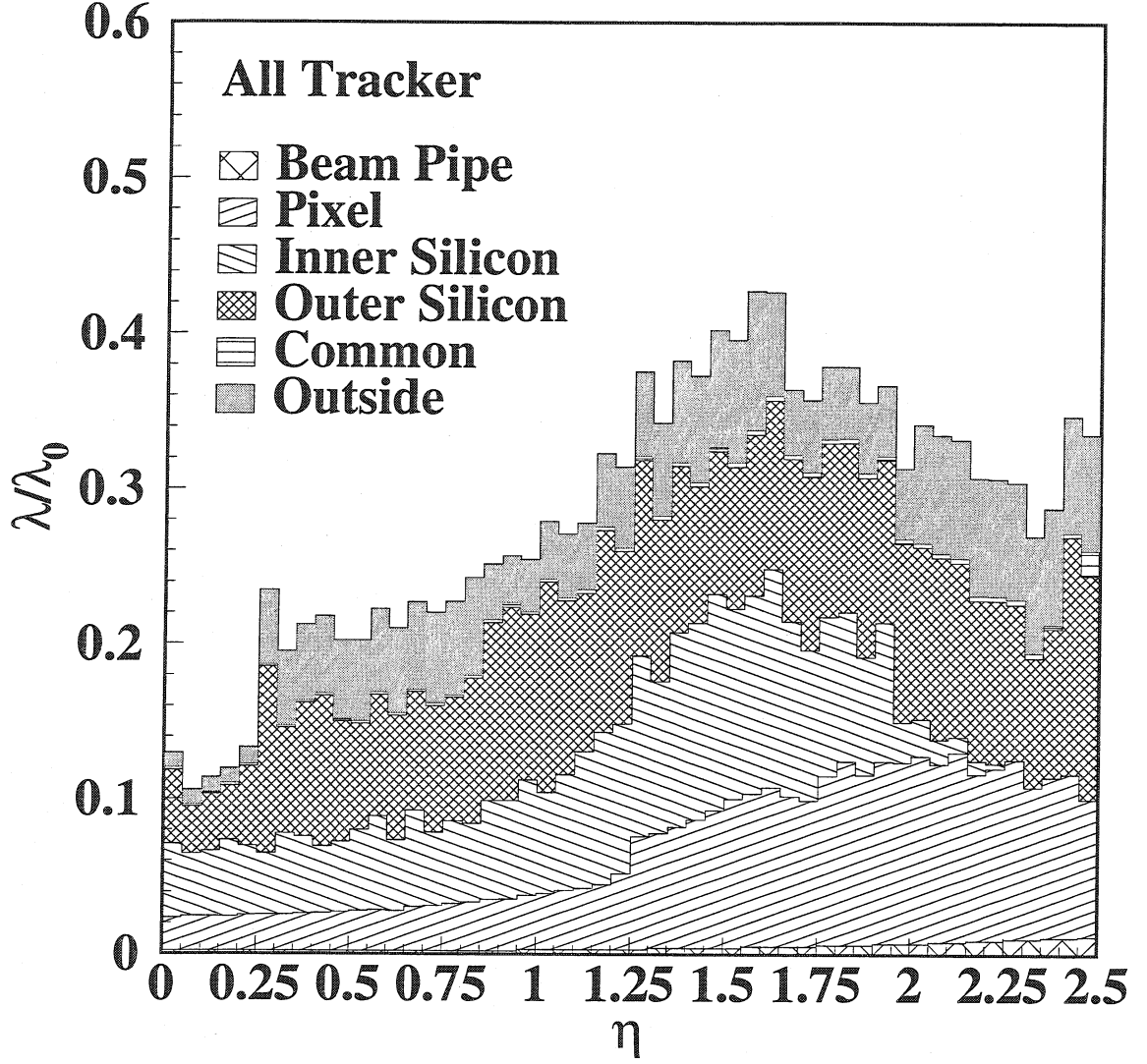


Figure 1.7: Tracker material in the CMS-125 layout in units of absorption lengths, as a function of η for the different sub-detectors [14].

The solution is a helix with its axis parallel to z :

$$\begin{aligned}
 x(s) &= x_0 + R_H \left(\cos \left(\Phi_0 + \frac{hs}{R_H} \cos \lambda \right) - \cos \Phi_0 \right), \\
 y(s) &= y_0 + R_H \left(\sin \left(\Phi_0 + \frac{hs}{R_H} \cos \lambda \right) - \sin \Phi_0 \right), \\
 z(s) &= z_0 + s \sin \lambda.
 \end{aligned}$$

s is the path length along the helix, $(x_0, y_0, z_0)^T$ is the starting point at $s = 0$, $\lambda = \arcsin(dz/ds)$ is the *dip angle* ($-\pi/2 \leq \lambda \leq \pi/2$), and R_H is the projected radius of the helix (see Fig. 1.8 and Fig. 1.9). The sense of rotation of the projected

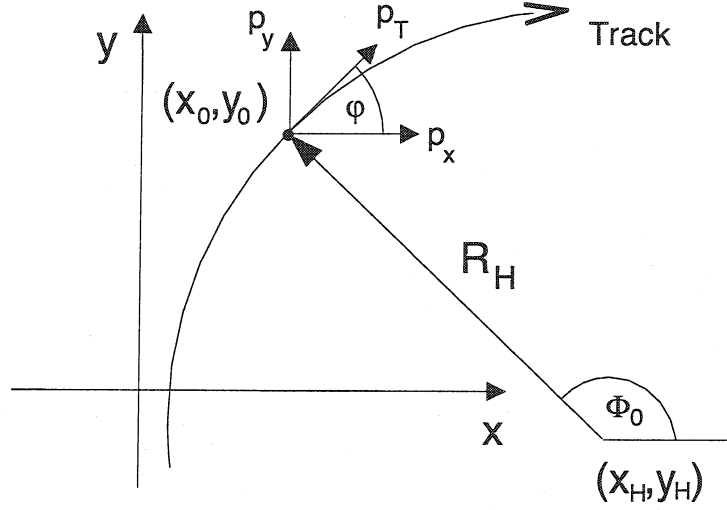


Figure 1.8: Helix parameters in the x - y plane projection.

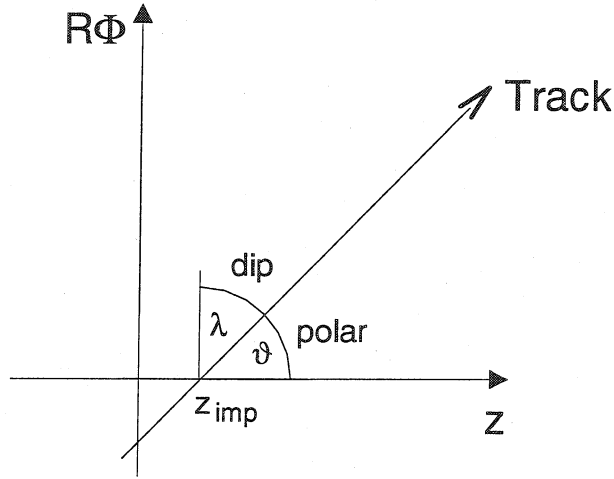


Figure 1.9: Helix parameters in the $R\Phi$ - z plane projection.

helix in the $x - y$ plane is given by $h = -\text{sign}(qB_z) = \pm 1$. Φ_0 is the azimuthal angle of the starting point.

1.4.1 Track parameters in ORCA and coordinate frames

The choice of parameters in ORCA follows the GEANE definition of track parameters [17]. The track parameters can be defined in different coordinate frames, the *global frame* in global space, the *local frame* on a detector surface, and the *measurement frame* in units of strips or pixels on a detector surface.

- **The Global Frame.** The *global frame* defines the position and direction (momentum) of a particle in global coordinates. Currently there are two ways to define the parameters of a track in global space, the *cartesian parameters* and the *curvilinear parameters*.

- The *cartesian parameters* define a particle by a vector of six coordinates, its position

$$(x, y, z)^T$$

and its momentum (direction)

$$(p_x, p_y, p_z)^T$$

in global coordinates. The associated covariance matrix has size 6×6 and is degenerate.

- The *curvilinear parameters* are defined by a vector of five coordinates:

$$\left(\frac{q}{p}, \lambda, \varphi, x_t, y_t \right)$$

q/p is the *signed inverse momentum*, λ is the dip-angle, defined by:

$$\tan \lambda = \frac{p_z}{p_T},$$

with $p_T = \sqrt{p_x^2 + p_y^2}$, and φ is defined by:

$$\tan \varphi = \frac{p_y}{p_x}.$$

Therefore the relation of momentum and angular coordinates can be written as:

$$\begin{aligned} p_x &= p \cos \lambda \cos \varphi, \\ p_y &= p \cos \lambda \sin \varphi, \\ p_z &= p \sin \lambda. \end{aligned}$$

The two coordinates for the spatial location, x_t, y_t , are defined in the following way: x_t and y_t are the coordinates of the trajectory in a local orthonormal reference frame with the z_t axis along the particle direction and the x_t coordinate being parallel to the global $x - y$ plane and perpendicular to the global z -axis (see Fig. 1.10). The origin of this frame ($x_t = y_t = 0$) is the crossing point of the trajectory with the surface.

- **The Local Frame.** The *local frame* is defined by a particular detector surface in space. The five coordinates are:

$$\left(\frac{q}{p}, \frac{dx}{dz}, \frac{dy}{dz}, x_l, y_l \right).$$

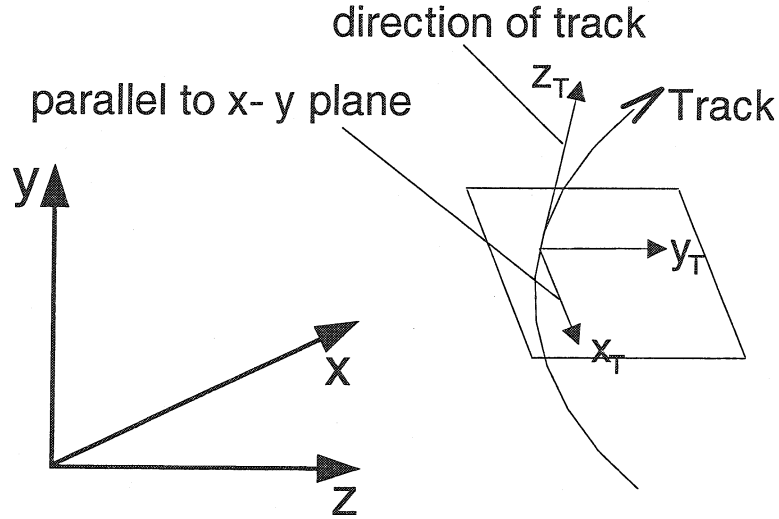


Figure 1.10: Definition of the curvilinear frame: x_t and y_t are the coordinates of the trajectory in a local orthonormal reference frame with the z_t axis along the particle direction and the x_t coordinate being parallel to the global x - y plane and perpendicular to the global z -axis.

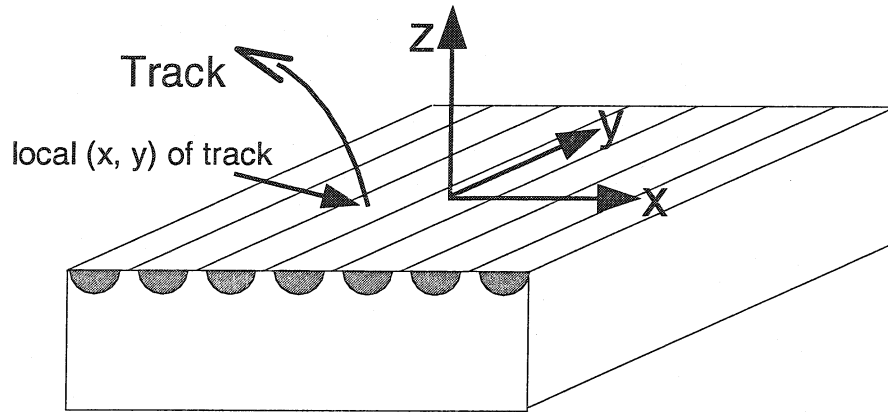


Figure 1.11: Definition of the local frame: The local z -axis z_l is perpendicular to the detector surface and points to the side of the strips. Local x is perpendicular to the strips, local y parallel to the strips.

The local z -axis z_l is perpendicular to the detector surface and points to the side of the strips. The orientation of the coordinate axes in the detector plane is chosen such that x_l is the more accurate measurement, perpendicular to the strips in the case of a strip detector. However, this relationship is only approximate for strip layouts with non-parallel strips, such as those used in the endcap part of the CMS Tracker. An illustration can be seen in Fig. 1.11.

- **The Measurement Frame.** In the *measurement frame* the coordinates are defined in units of strips or pixel. The transformation from the local frame to

the measurement frame (see also Fig. 1.12) can be described in the following way:

$$\begin{pmatrix} x_m \\ y_m \end{pmatrix} = \mathbf{F} \cdot \mathbf{R}(\phi) \cdot \begin{pmatrix} x_l - x_0 \\ y_l - y_0 \end{pmatrix}.$$

\mathbf{F} is a scaling matrix which scales the local x - and y - coordinates into measurement coordinates. In a strip detector it is equal to

$$\mathbf{F} = \begin{pmatrix} \frac{1}{\text{pitch}} & 0 \\ 0 & \frac{1}{\text{striplength}} \end{pmatrix}.$$

\mathbf{R} is a rotation matrix which takes into account a possible rotation between the orientation of the strips and the local detector surface (e.g. in trapezoidal detectors in the forward region of the CMS Tracker). It is described by a simple rotation in two dimensions:

$$\mathbf{R} = \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix}.$$

x_0 and y_0 are the center of the measurement frame in local coordinates. The transformation of the measurement frame to the local frame is therefore a simple inverse transformation of the equation above:

$$\begin{pmatrix} x_l \\ y_l \end{pmatrix} = \mathbf{R}(\phi)^{-1} \cdot \mathbf{F}^{-1} \cdot \begin{pmatrix} x_m \\ y_m \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \end{pmatrix},$$

with \mathbf{F}^{-1} being:

$$\mathbf{F}^{-1} = \begin{pmatrix} \text{pitch} & 0 \\ 0 & \text{striplength} \end{pmatrix},$$

and \mathbf{R}^{-1} the inverse rotation:

$$\mathbf{R}^{-1} = \mathbf{R}^T.$$

1.5 Track reconstruction limits

A track is characterized by its parameters which depend on the underlying track model. In the absence of a magnetic field, the track model of charged and neutral particles is a straight line in space, which is defined by position and direction in three dimensional space. Inside a constant magnetic field, the trajectories of charged particles are helices.

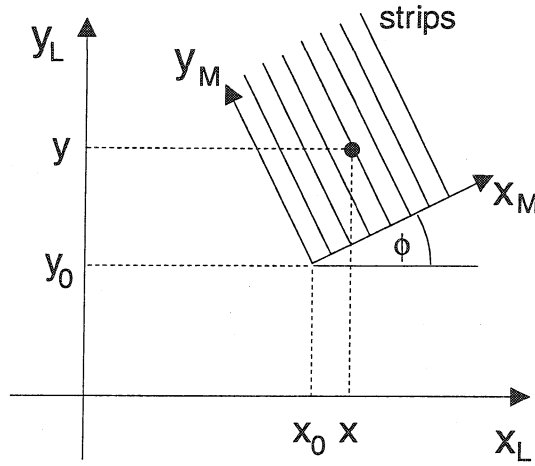


Figure 1.12: Definition of the measurement frame. The transformation from local to measurement frame can be done via simple algebraic operations.

1.5.1 General limits

The minimum number of measurements required for estimating the track parameters depends on the number of estimated parameters:

$$\text{Number of degrees of freedom} = \text{number of measurements} - \text{number of estimated parameters} \geq 0$$

In the case of the motion of a charged particle in a magnetic field the number of estimated parameters is five.

1.5.2 Minimum transverse momentum ($p_{T,\min}$)

The transverse momentum p_T is related to the radius R_c of curvature in the plane transverse to the magnetic field via

$$p_T = 0.29979 \cdot R_c \cdot B_z,$$

with the units of p_T in GeV/c, R in m and B_z the magnetic field in direction of global z in Tesla. In principle p_T can be determined by fitting a circle to (x, y) measurements of a set of points along the track in the plane transverse to the magnetic field. Thus the minimum p_T of particle is defined by the radius of a full circle linking the origin and the outermost measurement at a given distance d to the origin (see Fig. 1.13):

$$p_{T,\min} = 0.29979 \cdot \frac{d}{2} \cdot B_z.$$

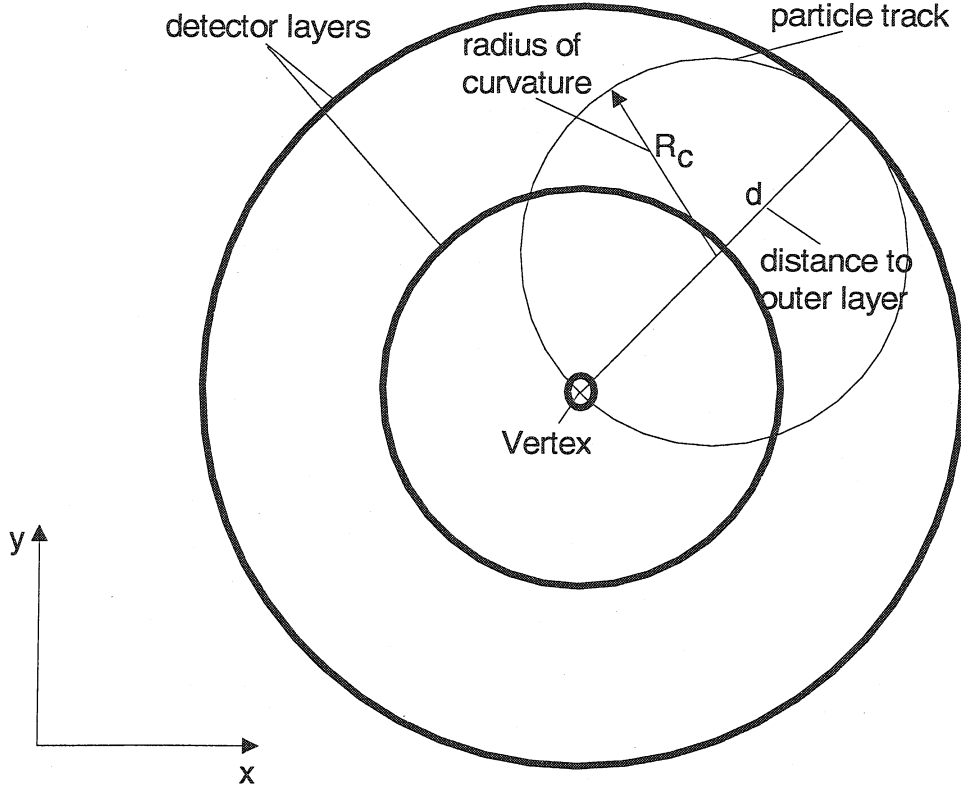


Figure 1.13: Definition of the minimum transverse momentum $p_{T,\min}$. $p_{T,\min}$ is defined by the radial distance of the outermost measurement from the vertex. The radius of curvature of a track originating in the vertex and reaching the layer of the outermost measurement is given by half the distance from the vertex to the measurement.

For instance, a particle with a measurement at $d = 1$ m from the origin (about the outermost layer of the tracker) has a minimum p_T (at magnetic field of 4 T) of about 0.6 GeV. Particles with a transverse momentum less than 0.6 GeV will never reach the outermost layer of the CMS Tracker. A more precise calculation of the minimum p_T as a function of η is shown in Fig. 1.14.

1.5.3 Maximum transverse momentum ($p_{T,\max}$)

The maximum p_T is defined by the minimum sagitta s which allows the distinction of the arc from the straight line of the chord (see Fig. 1.15).

The following relation holds:

$$R_c = \frac{s^2 + \left(\frac{d}{2}\right)^2}{2s}.$$

With the approximate distance $d = 1$ m from the innermost pixel to the outermost silicon measurement and assuming that the sagitta is equal to the pitch in the middle

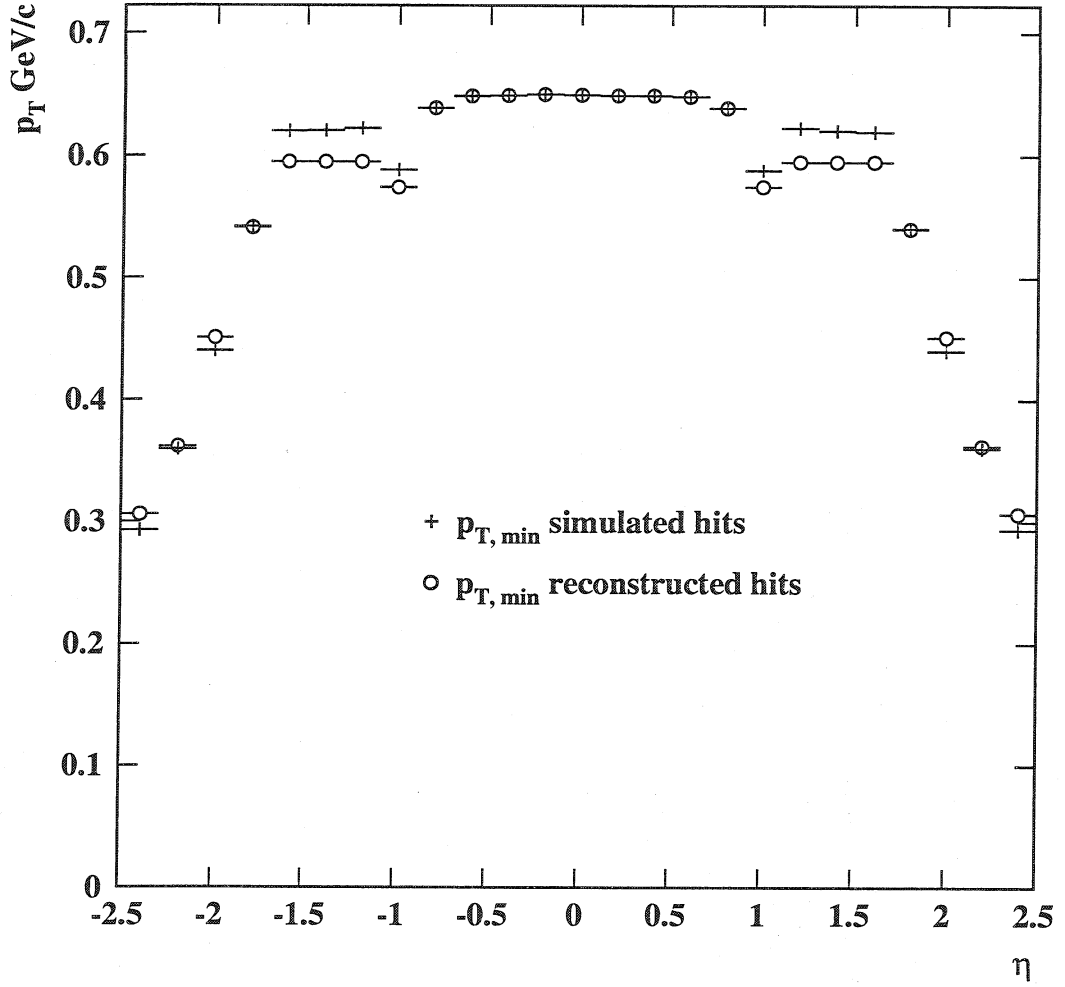


Figure 1.14: Minimum p_T as a function of η for simulated and reconstructed hits of the outermost surface of the CMS Tracker. In the interval $-1.6 < \eta < 1.6$ the limit is of the order of 0.65 GeV/c, while it drops to 0.25 GeV/c in the very forward region.

layer ($s = 120 \mu\text{m}$), the maximum reconstructable p_T of a particle is of the order of 1 TeV. More precisely, the sign of the transverse momentum and therefore the charge of a particle cannot be reliably estimated if the error on the reconstructed p_T^{-1} is of the order of p_T^{-1} :

$$p_T^{-1} \approx \sigma(p_T^{-1}).$$

A more precise estimate based on reconstructed tracks is shown in Fig. 1.16. It shows that tracks with a p_T up to 200 GeV/c can be reconstructed without problems everywhere, and up to 1 TeV/c in the region $-1.6 < \eta < 1.6$.

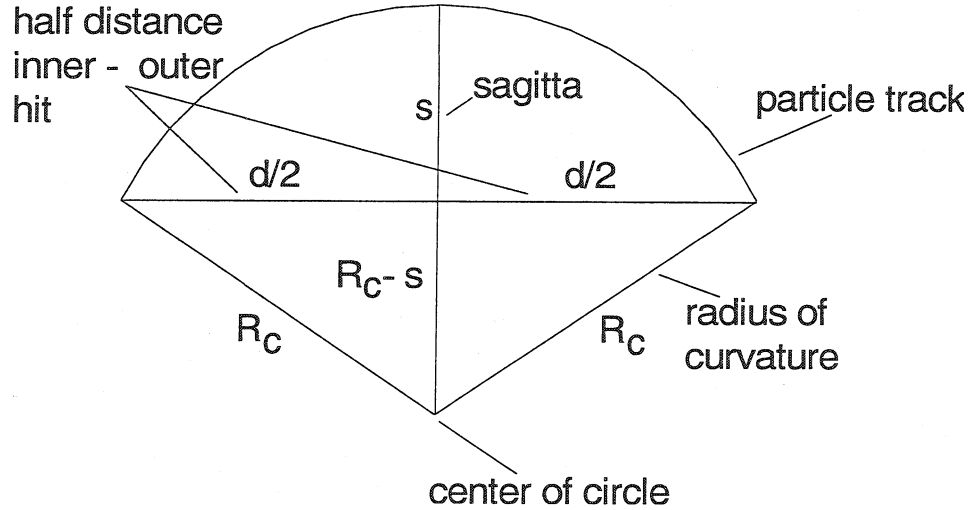


Figure 1.15: Definition of the sagitta s . The sagitta s is the distance of the midpoint of the arc to the chord defined by two points of the circle. With increasing transverse momentum of a track, the sagitta s becomes smaller. If the sagitta s is smaller than the measurement error, the transverse momentum is no more reconstructable and the track is compatible with a straight line.

1.5.4 Number of simulated hits for single muons and pions

The knowledge (and thus the quality) of a reconstructed track becomes better with the number of measurements (or reconstructed hits) used in the track fit. Usually one has ask for a minimum number of measurements in a track, reflecting a compromise between efficiency and ghost rate. The more hits are required in a track fit, the lower becomes the probability of reconstructing a track with random consecutive hits which has no original corresponding true track (a so-called “ghost track”). On the other hand, not all of the tracks leave more than a certain amount of hits within the Tracker, because of material interactions, geometrical inefficiencies, detector inefficiencies, etc. Usually one requires a reconstructed track to have between five and eight hits minimum.

A general overview of the average number of generated simulated hits (SimHits) per simulated muon track is shown in Fig. 1.17; for pions it is illustrated in Fig. 1.18. The average number of generated SimHits in the region $-1.5 < \eta < 1.5$ is between 16 and 18. The number of SimHits increases then up to 24 in the η -bin around 1.8, which corresponds to the outermost edge of the CMS Tracker where the particle tracks have their longest path within the Tracker volume. For pions these values are slightly lower.

A precise overview of the fraction of tracks with more than five or more than eight hits can be seen in Fig. 1.19 for muons, and in Fig. 1.20 for pions. It shows that the number of created SimHits per muon depends only on the η -region of interest (and therefore on geometrical constraints). For $|\eta|$ -values less than 2.3 virtually all muon

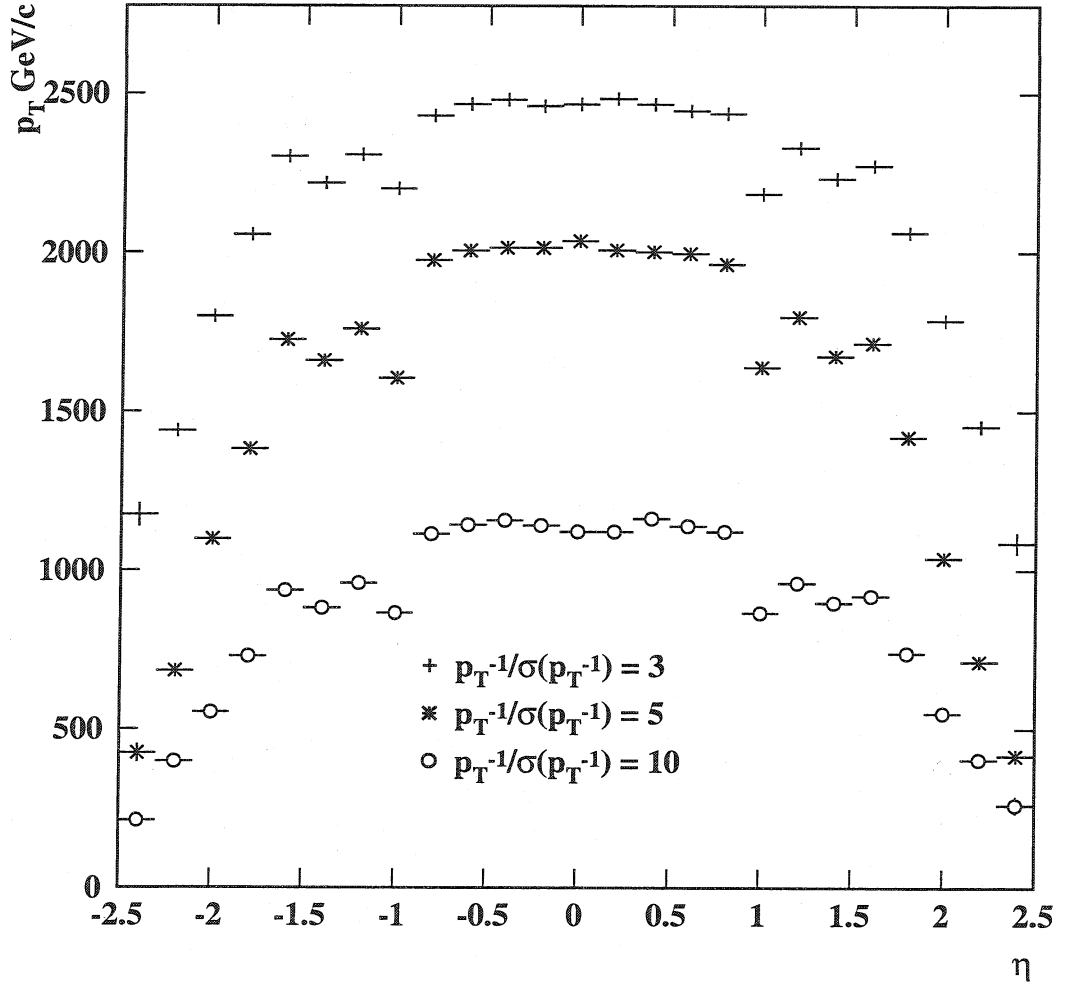


Figure 1.16: Average p_T of tracks with different values of $p_T^{-1}/\sigma(p_T^{-1})$. Tracks with a $p_T^{-1}/\sigma(p_T^{-1})$ -value of about 10 have a transverse momentum of about 1 TeV/c in the interval $-1.6 < \eta < 1.6$. This drops to 200 GeV/c in the very forward region due to the shorter leverarm.

tracks have more than eight hits.

In contrast to muons, the fraction of pion tracks with more than five or eight hits is significantly less than 1 everywhere. For pions with a p_T of 1 GeV/c, about 85% of the tracks have more than five hits in the forward region of the tracker, and between 90%–95% of the tracks have more than five hits in the central region of the barrel. Asking for at least eight hits, only 80% of the 1 GeV/c pion tracks in the forward, and 85%–90% in the central region fulfill this criterion. For pions with a transverse momentum of 10 GeV/c and 100 GeV/c these values are about 2%–5% higher.

A visual r - z plot of the last positions in a sensitive volume for pions of $p_T = 1$ GeV/c before they leave the tracker or decay within the tracker volume is shown in Fig. 1.21.

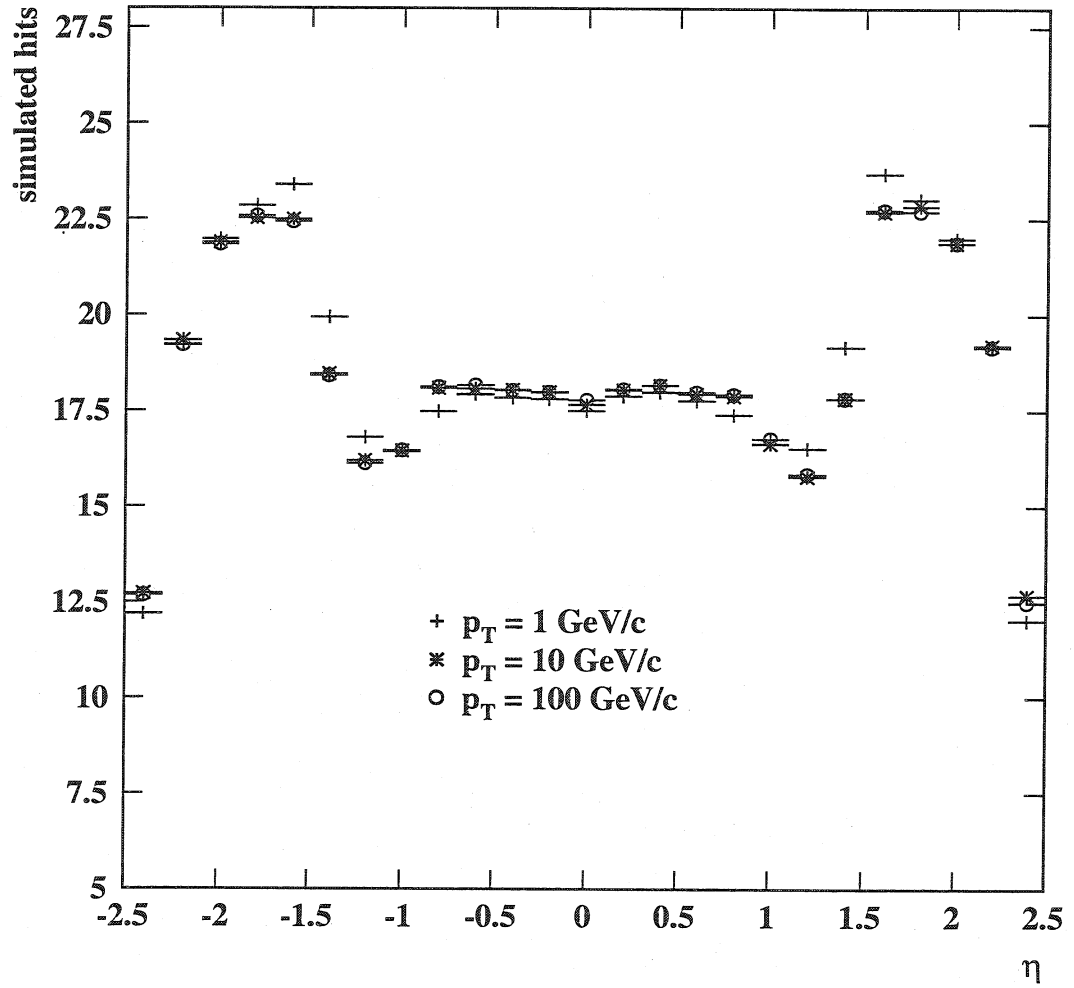


Figure 1.17: Average number of simulated hits per simulated muon track as a function of η for muons of three different p_T values.

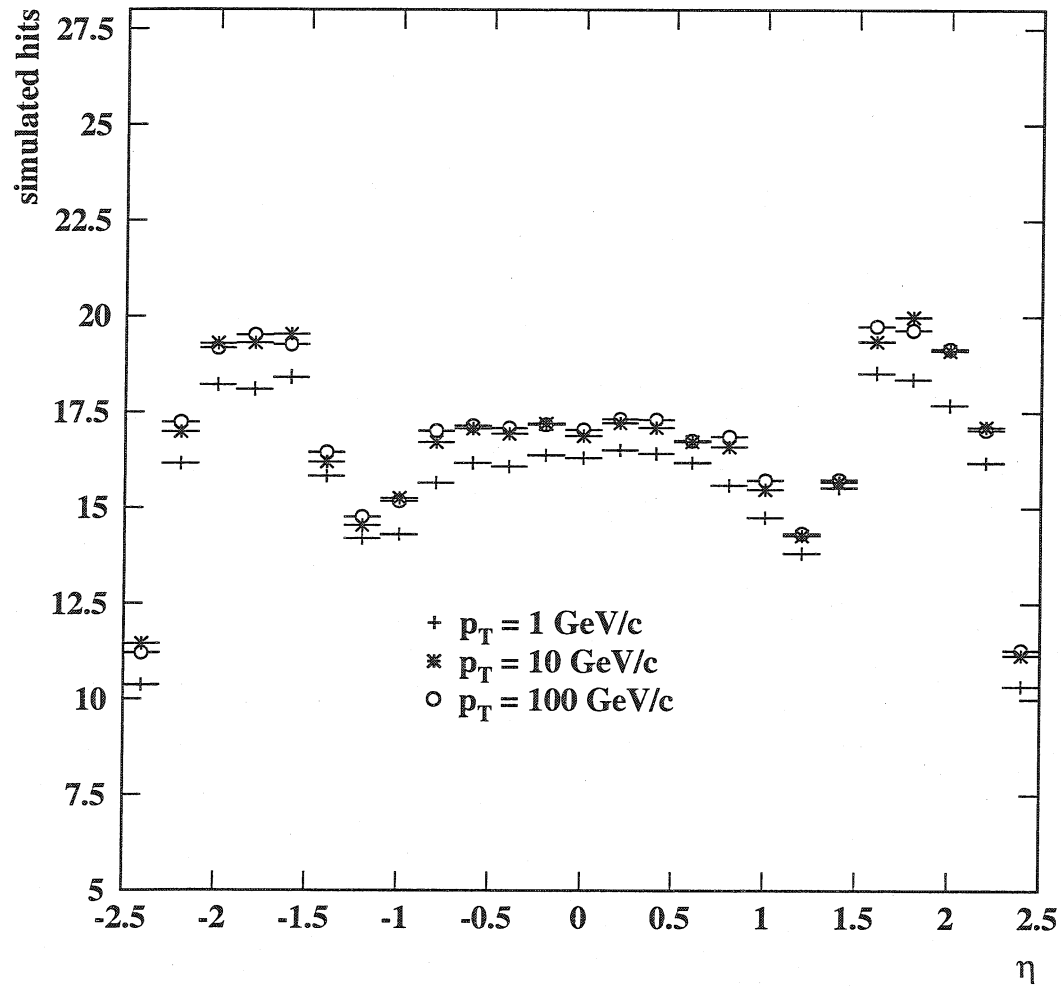


Figure 1.18: Average number of simulated hits per simulated pion track as a function of η for pions of three different p_T values.

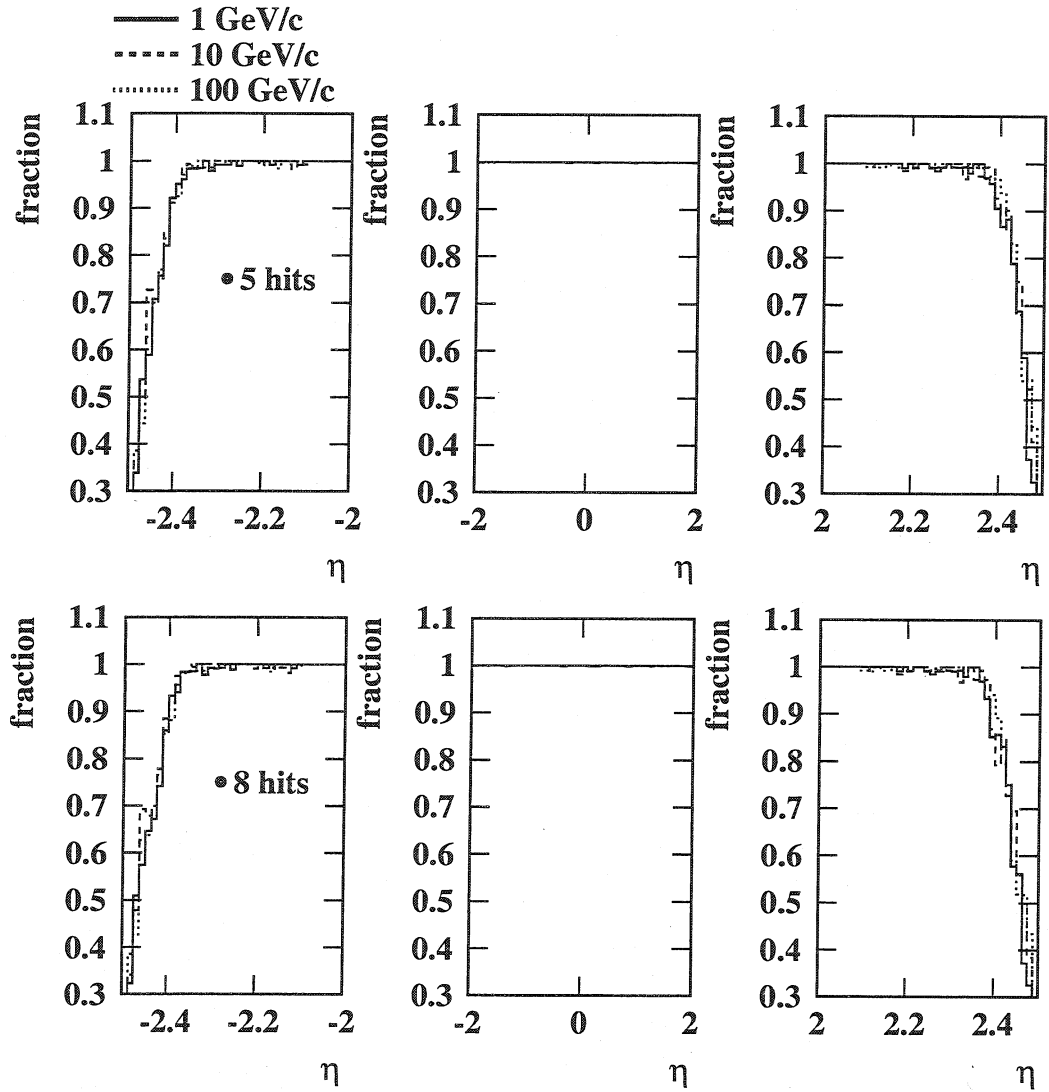


Figure 1.19: Fraction of muon tracks with more than five hits (top) or more than eight hits (bottom), for three different values of p_T . The fraction of tracks with more than eight hits is below 1 only in the very forward region above $|\eta| \geq 2.3$. Below $|\eta| < 2.3$ virtually all muon tracks have more than eight hits.

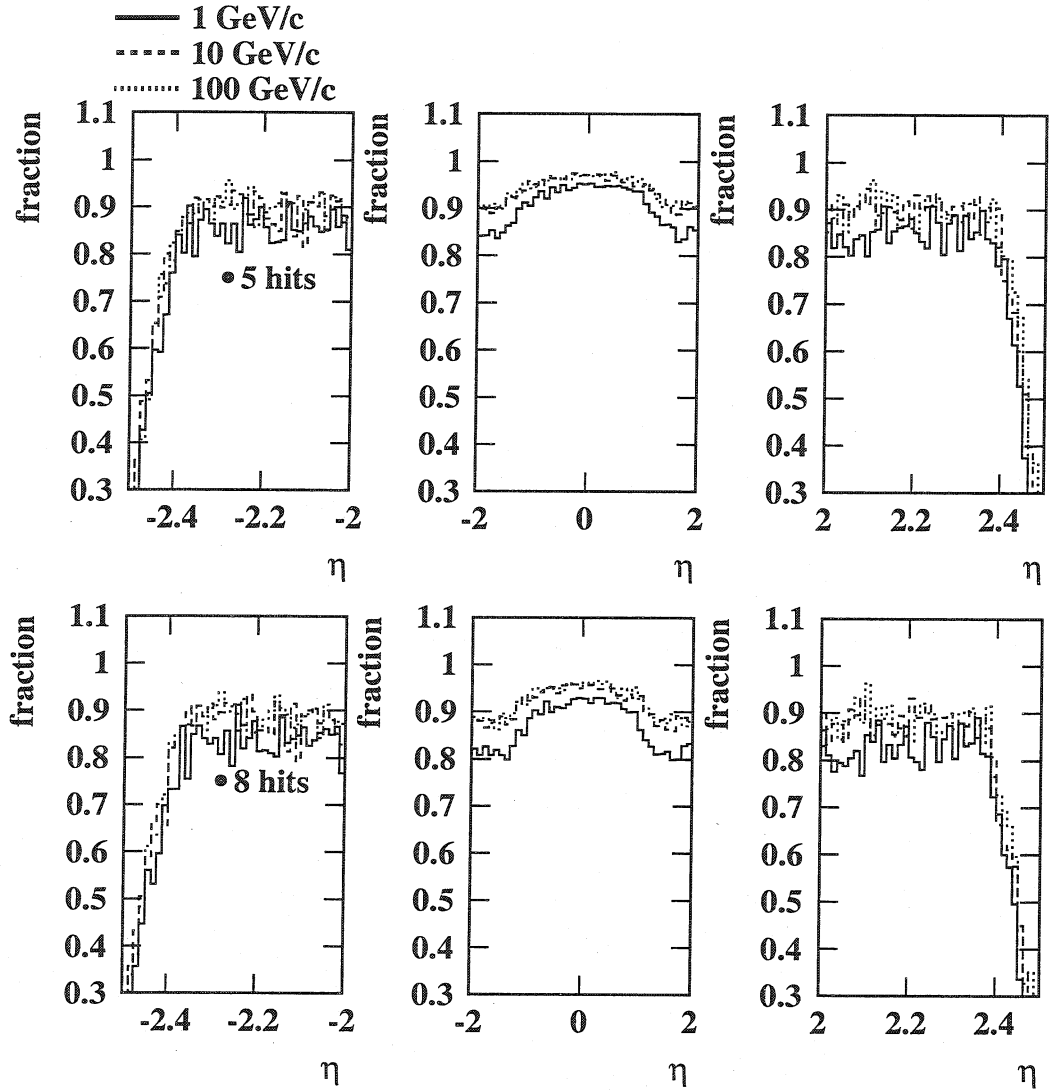


Figure 1.20: Fraction of pion tracks with more than five hits (top) or more than eight hits (bottom), for three different values of p_T . The fraction of tracks with more than eight hits is lower than 1 everywhere, between 80%–90% forward region, and 85%–95% in the central region of the tracker. On average, there are 5% less pion tracks of 1 GeV/c which fulfill the criterion of having more than 5 or eight hits than pion tracks with higher momenta.

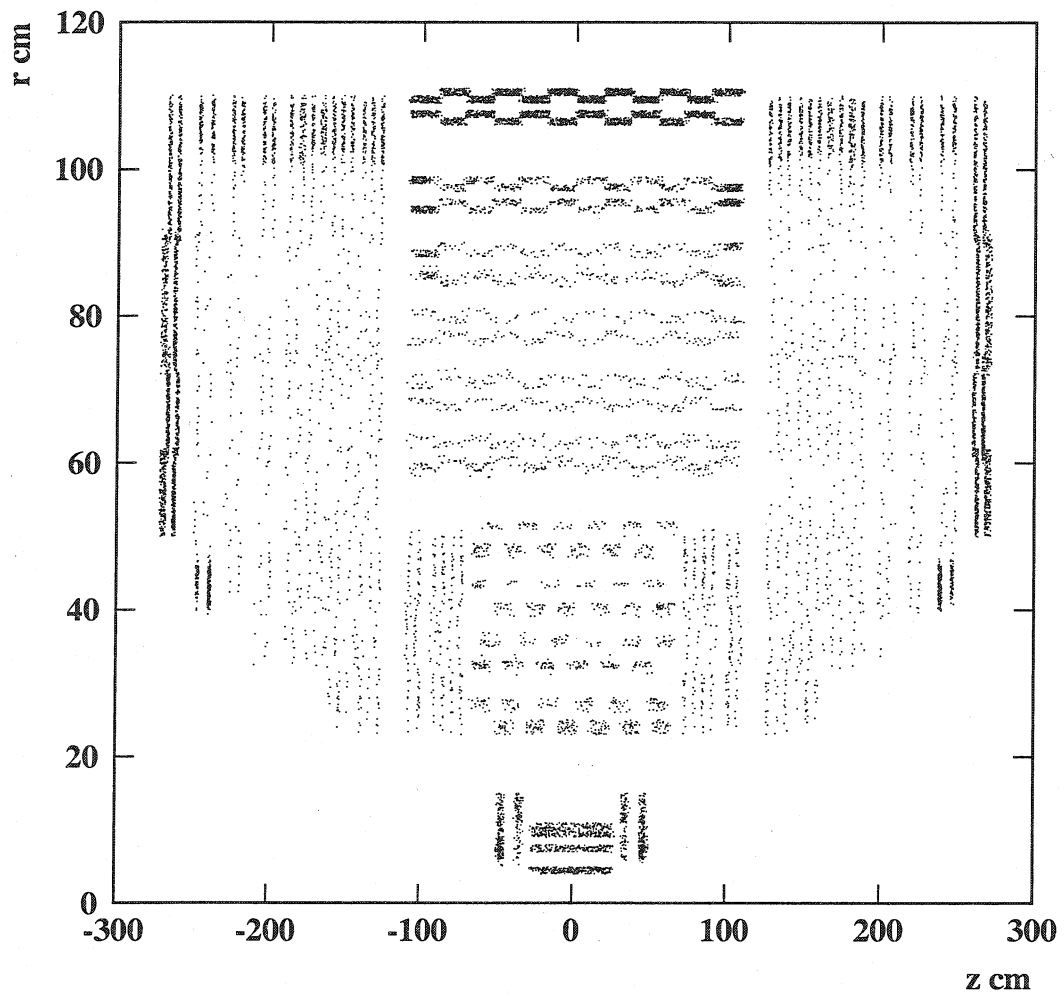


Figure 1.21: Last positions of pions of $p_T = 1 \text{ GeV}/c$ in sensitive volume before they leave the tracker or decay within the tracker volume.

2 Tracker object model in ORCA

Due to the complexity of the CMS Tracker, with its thousands of detector modules, millions of channels, and tens of thousand hits per bunch crossing, it is not obvious that there is a single optimal track reconstruction algorithm in all circumstances. It is more likely that there should be several specific algorithms, each of them optimized for a specific task within the complex physics at CMS. Therefore a flexible framework for developing and evaluating track reconstruction algorithms is highly desirable.

In addition, the mathematical complexity of track reconstruction limits the number of developers. Usually the algebra involved is localized in a few places and largely independent of the various types of track reconstruction algorithms. Therefore the development of track reconstruction algorithms can be made easier for developers if the algebra can be separated from the logic of the algorithm.

These demands can be met if one adopts an object-oriented approach to the design and the implementation of the track reconstruction program [18]. Three concepts of object-oriented programming are particularly important in attaining the goal of modular, flexible and easy maintainable software: encapsulation, inheritance, and polymorphism.

- In object-oriented programming, a *class* is fully described by
 - its name,
 - its state and
 - its behaviour.

Objects are then instances of classes. The summary of name, state and behaviour of an object is called *encapsulation*. Every object contains all necessary information and is therefore fully described by these three features. This is the main distinction of object-oriented programming with respect to procedural programming styles, where data and functions are separated. Encapsulation is an important principle which allows object-oriented software to have a modular structure. Fig. 2.1 shows how a class is represented in UML (Unified Modeling Language) [19].

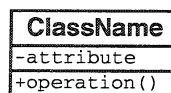


Figure 2.1: A class is defined by its name, its state and its behaviour. The *attributes* of a class are its *data members* defining the state, *operations* on these data members via *methods* define the behaviour of a class.

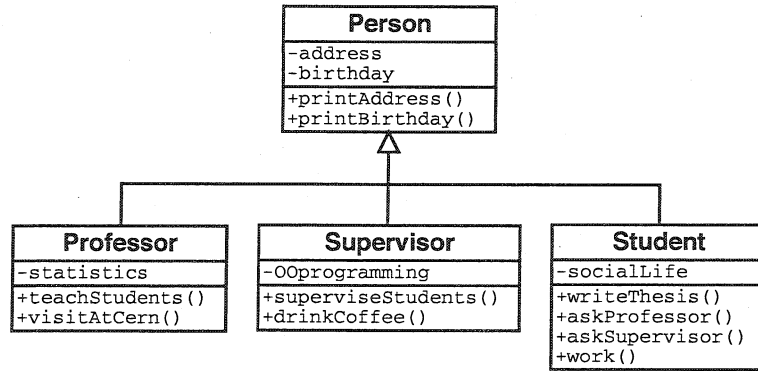


Figure 2.2: UML diagram of a simple inheritance example. The classes Professor, Supervisor and Student inherit from the same base class (Person), and extend the existing interface by additional methods.

- Another important concept of object-oriented programming besides class building is *inheritance*. Existing classes can be extended with specific features, attributes or methods, which specialize the existing class. The inheritance relations between classes can therefore be described as *generalization / specialization*. Fig. 2.2 shows how inheritance is described in UML.
- *Polymorphism* is the concept which allows objects to react differently to the same message. It is inseparably linked to so-called *late binding*. This means that it will be decided at runtime of an object-oriented program which method is called by a message. This is achieved by redefining in the subclass methods which have been declared *virtual* in the base class. Both the method in the base class and the one in the subclass must have the same name, argument list and return type. If the method in the base class is declared *pure virtual*, it has to be implemented in the derived subclass. In this case the base class is called an *abstract base class* (ABC), as it defines a common interface for all classes of same type. If in addition the abstract base class has no implementation at all and defines only a set of pure virtual methods, it is called a *pure abstract base class* (pABC). Fig. 2.3 shows an example of how abstract base classes and methods are described in UML.

2.1 Tracker reconstruction geometry and sensitive volumes

A track reconstruction geometry is built out of detector layers, and detector layers are built out of individual sensitive (physical) volumes. However, detector layers need not be uniform — in the CMS Tracker a barrel detector layer is different from a forward disk, and pixel modules are different from silicon modules. The layer structure of the tracker geometry is used for the track reconstruction, whereas the sensitive volumes are connected to read-out units and measure the passage of a charged particle.

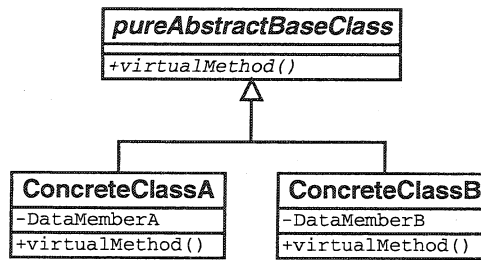


Figure 2.3: Principle of polymorphism. A virtual method defined in the base class can be reimplemented in the derived subclass. If the virtual method in the base class has no implementation, it is declared to be pure virtual, and all derived subclasses have to implement it. Such a base class is called an *abstract base class* (ABC). If, in addition, an abstract base class has no implementation at all, even no data members, it is called a *pure virtual base class* (pABC). Such a pABC defines a common interface for all derived classes. In UML, abstract classes and methods have their names written in italic.

In the following it will be explained

- how the conceptually different functionalities of detector layers and sensitive volumes work together, and
- how the software implementation of the detector layers and detector units follow the design of the layout, the detector hardware and the electronics.

Fig. 2.4 shows a class diagram of the most important classes involved.

The Det class is an abstract base class for any type of detector (silicon strip detectors, pixel detector, etc). Its main responsibility is to provide access to its reconstructed hits (measurements of a detector). It also provides measurements compatible with a trajectory state on demand, in an optimal way. Therefore the interface of the Det class is basically defined by two methods — the `recHits()` and the `measurements()` methods.

The Det interface is extended by the derived abstract base class DetUnit. The DetUnit provides a common interface for detector specific information (e.g. the DetReadout or the DetType) and thus represents the physical volume of a detector itself. Further subclasses are for example the StripDet, an abstract base class common to all strip detectors, and the SiStripDet, which is finally one of the concrete classes to implement the interface.

A DetUnit has properties which are identical to hundreds or thousands of other DetUnits. Each DetUnit needs to know its own topology, but there are only a few tens of different (strip or pixel) topologies for several thousands of DetUnits. The DetType is a class which keeps the DetUnit relevant information which is identical for many DetUnits. Each DetUnit has a method `type()` which returns the corresponding DetType. The inheritance tree of the DetType is shown in Fig. 2.5. The methods

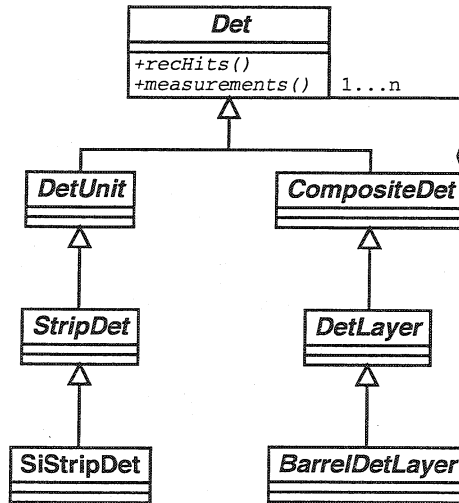


Figure 2.4: Class diagram of the **Det**. The left branch shows the inheritance tree of the sensitive volumes. The corresponding interface for sensitive detector volumes is defined by the **DetUnit** class. The right branch illustrates the inheritance tree of the classes related to the geometry of the Tracker. The main geometrical component of the Tracker is the **DetLayer**, which is an abstract base class and can either be implemented by a **BarrelDetLayer** or a **ForwardDetLayer** (not shown).

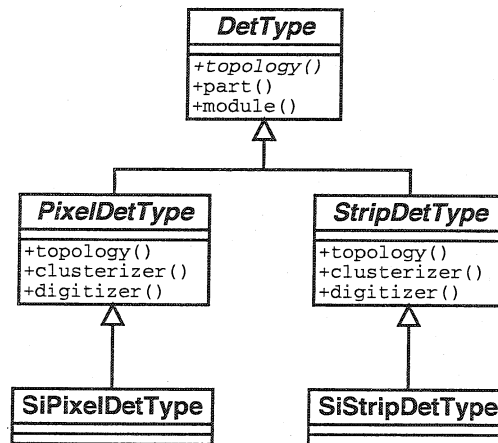


Figure 2.5: Inheritance tree for the **DetType** class.

`part()` and `module()` are enumerators. `part()` returns the values `barrel` or `forward`, `module()` returns `pixel`, `silicon`, `msgc` or `muon`.

The software model of the CMS Tracker provides a common behaviour for all types of Trackers with layer oriented design from the track reconstruction point of view, hence also for test-beam like structures. The software representation of a detector layer is the so-called **DetLayer**, which provides two essential functions:

- efficient access to reconstructed hits compatible with a trajectory state, using a user-defined compatibility criterion (e.g. χ^2), and
- providing links to other layers which might contain hits of a given track (navigation).

A concrete Tracker is fully described by its layers.

A `DetLayer` is a composition of detectors (a `CompositeDet`) and thus represents a geometrical superstructure of the CMS Tracker. As the basic functionality is the same as the one of the `DetUnit` — providing access to reconstructed hits — both the inheritance branch implementing the sensitive volume part and the inheritance branch implementing the geometrical structure inherit from the same abstract base class, the `Det`.

Two different types of `DetLayers` exist, the `BarrelDetLayer` and the `ForwardDetLayer`. Both layers differ by their associated surface: the `BarrelDetLayer` has an associated `BoundCylinder` surface, the `ForwardDetLayer` has an associated `BoundDisk` surface. Also `DetUnits` have an associated surface called `BoundPlane`. All types of surfaces inherit from an abstract base class `Surface`. The inheritance tree is shown in Fig. 2.6.

The class diagram of the `CmsTracker` is shown in Fig. 2.7. It is a pure abstract base class which defines a common interface for all types of `CmsTrackers` inheriting from `CmsTracker`. It defines access to its `DetTypes`, its `DetUnits` and its `DetLayers`.

2.2 Detector-module level simulation and reconstruction

All information associated with a detector module like

- simulated hits (`SimHits`),
- Digis (amplitude per strip or pixel) and
- reconstructed hits (`RecHits` = clusters of digis)
- ...

is accessible via the corresponding `DetUnit` object. In order not to introduce an overhead in real data processing, the `SimHits` do not reside inside the `DetUnit`, but in a separate class, the `SimDet`. `DetUnit` and `SimDet` are related via a has-a-relationship (see Fig. 2.8).

Other simulation-related information is stored as well in the `SimDet`. In the case of real data, the `RecHits` will not be produced using the `SimHit` information, and therefore the `SimDet` will not exist. The basic functionality of the `SimHit` is shown in Fig. 2.9.

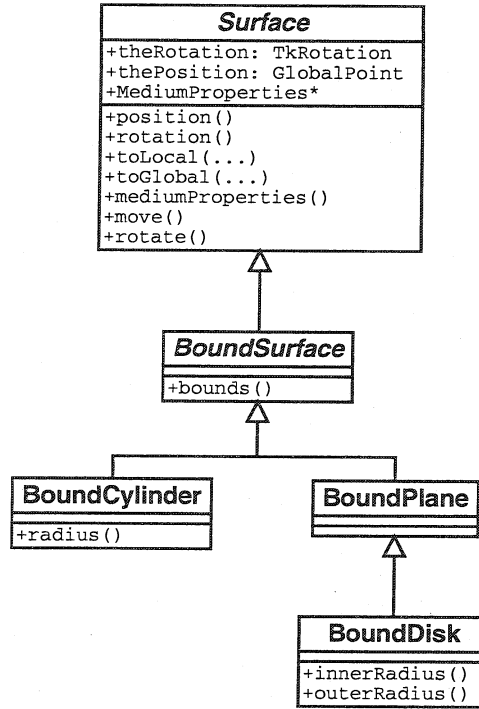


Figure 2.6: Inheritance tree for the Surface abstract base class. The Surface performs the transformation from local to global coordinates and vice versa. For alignment purposes it can be moved and rotated. The Surface has a position and rotation which defines its orientation in global space.

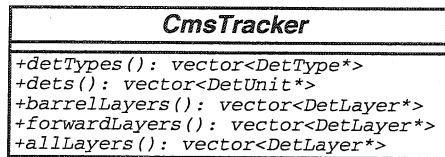


Figure 2.7: The CmsTracker pure abstract base class. The CmsTracker defines a common interface for all types of trackers inheriting from it. It prescribes access to its DetTypes, its DetUnits and its DetLayers. It is possible to ask separately for the barrel detector layers and the forward detector layers.

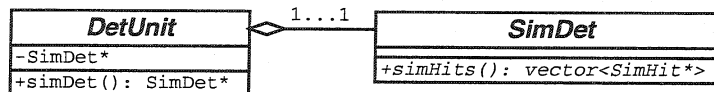


Figure 2.8: The relationship between the DetUnit and the SimDet. In order not to introduce an overhead for real data processing, all simulation relevant information for local reconstruction is encapsulated in a separate class from the DetUnit.

SimHit
-const PSimHit*
-timeOffset: float
-EncodedSimTrackId
+track(): const SimTrack*
+pabs()
+energyLoss()
+particleType()
+...()

Figure 2.9: The SimHit class contains all necessary information needed to create Digis. In addition, it also knows its corresponding SimTrack and its particle type.

RecHit
+localPosition()
+localPositionError()
+measurementPosition()
+measurementError()
+det()
+layer()

Figure 2.10: The RecHit class. The position, direction and error of a RecHit can be defined within different frames. The change of different parametrizations is done internally by the RecHit, using methods from the Det class.

The RecHit class represents the fundamental measurements of the tracking detectors. It contains all the necessary information of a hit on a detector, such as

- the position (global and local) of the hit,
- the direction (global and local) if any,
- the corresponding covariance matrices and errors,
- the detector.

The hit can be parametrized in different coordinate frames: the measurement frame in terms of strips or pixels, the local coordinate frame of the corresponding surface, or the global frame. Conversions between different frames are done internally by forwarding the necessary methods from the underlying detector. Some of the methods of the RecHit class can be seen in Fig. 2.10.

In the analysis phase there is an interest to have an object which does the association of SimHits to RecHits or vice versa. Such a functionality is provided by the TkHitAssociator (see Fig. 2.11). The TkHitAssociator has a method to return SimHits associated to a particular RecHit, another method to return RecHits associated to a SimHit on a particular DetUnit, and a method to determine whether a SimHit and a RecHit are associated.

TkHitAssociator
<pre> +(RecHit): vector<const SimHit*> +(SimHit*, vector<RecHit>): vector<RecHit> +associated(SimHit, RecHit): bool </pre>

Figure 2.11: The TkHitAssociator allows association of RecHits to SimHits and vice versa via three methods.

TrajectoryStateOnSurface
<pre> +globalParameters() +localParameters() +curvature() +cartesianError() +curvilinearError() +localError() +surface() </pre>

Figure 2.12: The TrajectoryStateOnSurface (TSOS) contains all necessary informations of a track in different parameterizations. The change from one parameterization to another one (state vectors plus errors) is done internally and not exposed to the client.

2.3 Pattern recognition

The TrajectoryStateOnSurface (TSOS) is a basic object in track reconstruction. It contains all the necessary information of a track locally on a surface, such as

- the position (global and local) of the track,
- the direction (global and local),
- its curvature and
- the corresponding covariance matrices.

The TSOS is providing all useful parameterizations of track states. It internally converts one parameterization into another one on demand. Jacobians are hidden internally and not accessible from outside the class. The names of the most important methods of the TSOS are shown in Fig. 2.12.

Propagation (update of a track state in time) is done by a separate object, the so-called **Propagator**. It propagates any track state to any surface (if this surface is reachable) and returns a new TSOS. Usually the propagation is done in two steps: the first one is the geometrical extrapolation following the equation of motion of a particle (depends on the underlying track model), the second one takes into account material effects in the track parameters and errors, such as multiple scattering and energy loss.

The **Propagator** is an abstract base class (ABC), defining a common interface for several concrete implementations of propagators, which can be used interchangeably.

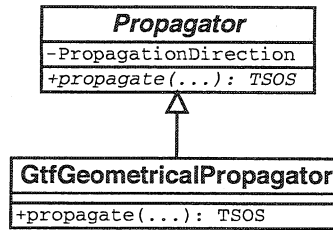


Figure 2.13: Class diagram for the Propagator. The Propagator abstract base class defines a common interface for all implementations of propagators. The GtfGeometricalPropagator does the extrapolation of charged particles within a magnetic field.

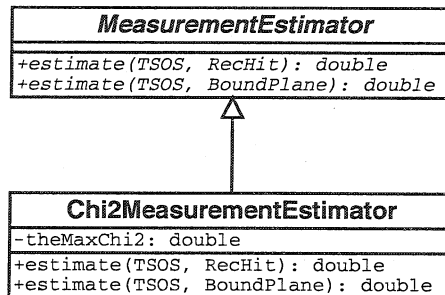


Figure 2.14: The MeasurementEstimator defines a common interface for compatibility estimators. Two types of compatibility of a trajectory state are needed: one compatibility criterion for a state compatible with a RecHit, the second criterion for a state geometrically compatible with a Surface.

The Propagator completely encapsulates the algebra of the track propagation and the associated Jacobians, which are not exposed to clients. The inheritance class diagram for the Propagator is shown in Fig. 2.13.

The MeasurementEstimator is an object used to define the compatibility of a trajectory state (TSOS)

- with a RecHit, or
- the geometrical compatibility with a Surface.

Fig. 2.14 shows the inheritance diagram of the MeasurementEstimator. It prescribes two methods, one for the RecHit compatibility and one for the geometrical compatibility. A concrete implementation of the MeasurementEstimator interface is the Chi2MeasurementEstimator.

The Updator updates a TSOS with a measurement (RecHit) from the same surface (local update of a track state) and produces a new “updated” or “filtered” state. It operates in the local frame of the detector surface.

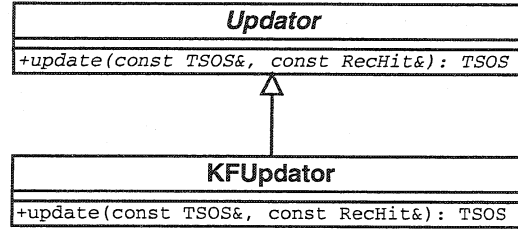


Figure 2.15: The interface of the pABC Updator is implemented for example by the KFUpdator. The KFUpdator updates the TSOS with a RecHit and returns a new TSOS. The mathematics of the update mechanism (Kalman Gain Formalism [20]) of the KFUpdator is encapsulated in the method `update(...)`.

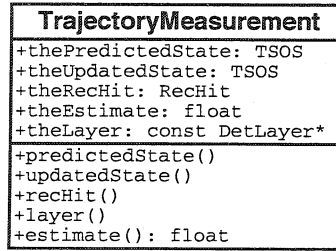


Figure 2.16: The TrajectoryMeasurement class keeps all the information after a propagation and update step. It also contains a measure of the compatibility of the RecHit with the predicted state (usually the χ^2).

The Updator is a pure abstract base class (pABC), defining a common interface for several concrete implementations of updators. The mathematics of error minimization is totally encapsulated within this object. The class diagram of the Updator is shown in Fig.2.15.

The TrajectoryMeasurement is a data object which keeps all the information after a particular propagation and update step in track reconstruction. It contains the predicted state after the propagation step on the surface of the detector of the RecHit and the updated state after the update of the predicted state with the RecHit. It also has a measure of the compatibility of the RecHit with the predicted state, usually the χ^2 . The DetLayer information is kept as well. In principle a track is defined by a vector of TrajectoryMeasurements, as these TrajectoryMeasurements contain all the information of a track on a particular surface in space. The content and the methods of the TrajectoryMeasurement class are shown in Fig. 2.16.

2.4 Event level simulation and reconstruction

All track relevant information associated with a particular event like

- simulated tracks and

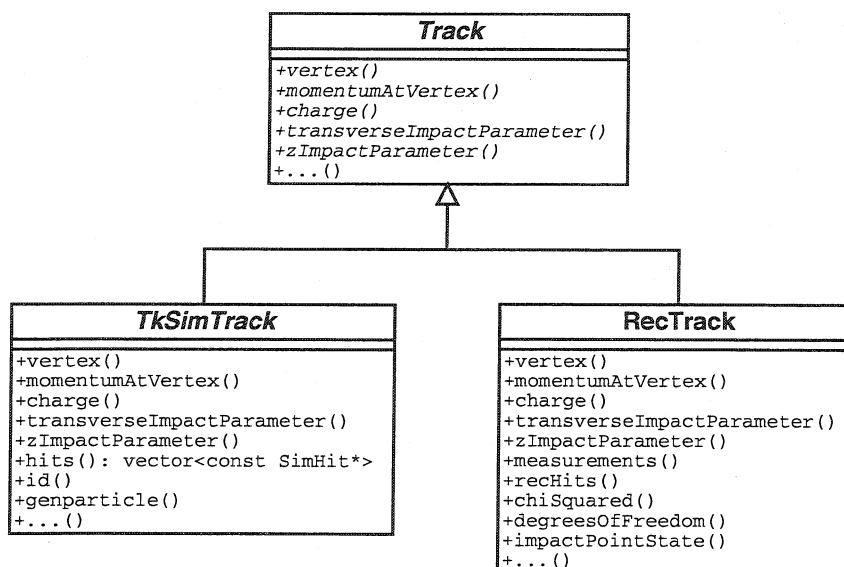


Figure 2.17: The Track defines an abstract interface for particle tracks. The RecTrack implements and extends the Track interface, by adding information about χ^2 , degrees of freedom, RecHits, etc.

- reconstructed tracks

has to be available for proper analysis. TrackFinders like the ModularKFFReconstructor reconstruct tracks within the CMS Tracker volume. The class RecTrack summarizes the final result of track finding and track fitting of a particle track. The RecTrack inherits from a general abstract Track class and implements the Track interface (see Fig. 2.17). In addition, the RecTrack gives access to information such as position, direction, momentum, impact parameters, total χ^2 , and the number of degrees of freedom.

As can be seen from Fig. 2.17, the TkSimTrack also inherits from the Track interface. The TkSimTrack is the simulation analogon to the RecTrack and keeps the necessary true values from the simulation. Its most important methods are hits(), which returns a vector of pointers to SimHits, and id(), which defines the position of the track in the sequence of all simulated tracks.

For the association of RecTracks to SimTracks there exists a class TrackAssociator which defines a common interface for track associator classes. Track association can be done on the base of hits or by pulls. Fig. 2.18 shows the class inheritance diagram of the TrackAssociator and the TrackAssociatorByHits.

In a similar way there exists classes for Vertex (pure abstract base class), SimVertex (simulation information) and RecVertex (reconstruction results), as well for analysis purposes the association of reconstructed vertices to simulated ones and vice versa.

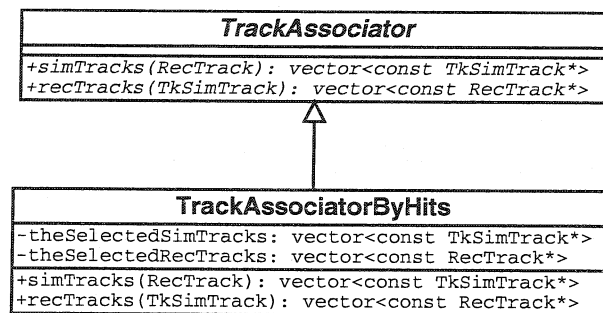


Figure 2.18: The `TrackAssociator` defines an abstract interface for the association of `TKSimTracks` to `RecTracks`. The interface is implemented by the `TrackAssociatorByHits`.

3 Track fitting methods in ORCA

A detailed overview of the history of track fitting and an introduction into the subject can be found in [16]. The experimental scenario of today's and tomorrow's high-energy physics experiments such as CMS can be summarized in the following points:

- high track multiplicity due to the high collision energy and luminosity obtained by LHC
- particle momenta ranging from a few hundred MeV/c up to several hundred GeV/c
- complex tracking detectors combining different measurement techniques (pixels and silicon strip detectors in the Tracker, muon gas chambers)
- multiple scattering in non-sensitive material layers such as support and cooling structures, cables etc.
- secondary interactions such as generation of high energy delta electrons by muons, hadronic interactions of pions, bremsstrahlung from electrons, gamma conversions

From the physics point of view the best precision and well-estimated errors should be obtained from optimal track fitting methods, for instance

- invariant mass resolution, e.g. for muons originating from a Z^0 decay, or
- track bundling for secondary vertex reconstruction.

This in turn requires realistic hit errors, adequate knowledge of the magnetic field, and precise treatment of multiple scattering and energy loss.

For the purpose of comparing various track fitting methods it is assumed that the association of hits to potential track candidates has already been solved to a large extent by a previous track finding (pattern recognition).

The methods for fitting a track (parameter estimation by error minimization) can be divided into two groups:

- Track fits with hard assignment of hits to tracks, where a hit either does or does not contribute to a track: The Global Fit [16], the Kalman Filter [20], and the Gaussian Sum Filter [21].
- Track fits with soft hit assignment, with hits contributing to a track according to their assigned weights: The Elastic Arm Algorithm [22] and the Deterministic Annealing Filter [23].

Currently the Kalman Filter and the Deterministic Annealing Filter are implemented in ORCA.

3.1 Principles of track fitting

In general, a track fit requires the knowledge of

- the detector layout and the detector resolution, and
- a track model which describes the trajectory of a particle.

The ingredients of such a track fit are therefore

- the measurements (RecHits) of a track and their associated errors,
- a stochastic model of material effects (multiple scattering and energy loss),
- a track model, depending on the magnetic field.

A track can be described at any point by a state vector \mathbf{x} of track parameters. In general, this state vector cannot be observed directly. The measurements \mathbf{m} of a detector (the hits) are functions of the state vector, corrupted by a measurement noise ϵ . This is described by the *measurement equation*:

$$\mathbf{m} = \mathbf{f}(\mathbf{x}) + \epsilon,$$

ϵ being the vector of measurement errors. In addition, it is assumed that the covariance matrix \mathbf{V} of the measurements is known:

$$\text{cov}(\epsilon) = \mathbf{V},$$

with the weight matrix \mathbf{G} being

$$\mathbf{G} = \mathbf{V}^{-1}.$$

It is now the task of the track fit to find a suitable function \mathbf{h} which maps \mathbf{m} on \mathbf{x} without bias and with minimum variance of the fitted parameters $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{x}} = \mathbf{h}(\mathbf{m}).$$

The expectation value of the fitted vector $\tilde{\mathbf{x}}$ is supposed to be the true value \mathbf{x}_{true}

$$\mathbf{E}(\tilde{\mathbf{x}}) = \mathbf{x}_{\text{true}},$$

and the variance of $\tilde{\mathbf{x}}$ is by definition

$$\sigma^2(\tilde{\mathbf{x}}) \equiv \mathbf{E}((\tilde{\mathbf{x}} - \mathbf{x}_{\text{true}})^2).$$

In the following there is a summary of the various track fitting methods, with a short explanation of the differences among them.

3.1.1 The Global Fit

The Global Track Fit based on the Least Squares Method (LSM) is the simplest among track fitting methods. The LS-estimate $\tilde{\mathbf{x}}$ of \mathbf{x} is the value for which the objective function

$$M(\mathbf{x}) = (\mathbf{m} - \mathbf{f}(\mathbf{x}))^T \mathbf{G} (\mathbf{m} - \mathbf{f}(\mathbf{x}))$$

attains its minimum. Note that in the global fit \mathbf{G} contains also material effects (multiple scattering), taking into account correlations between measurements.

In the special case that the track model can be sufficiently well approximated by a linear model in the neighbourhood of an expansion point \mathbf{x}_0 , \mathbf{f} can be written as:

$$\mathbf{f}(\mathbf{x}) = \mathbf{c} + \mathbf{H}\mathbf{x},$$

with

$$\mathbf{H} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x} = \mathbf{x}_0).$$

The LSM estimate is then given by

$$\tilde{\mathbf{x}} = (\mathbf{H}^T \mathbf{G} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{G} (\mathbf{m} - \mathbf{c}),$$

and its covariance matrix \mathbf{C} by

$$\mathbf{C}(\tilde{\mathbf{x}}) = (\mathbf{H}^T \mathbf{G} \mathbf{H})^{-1}.$$

It can be shown that in the case of an exact linear model and Gaussian errors the LSM is unbiased and has minimum variance among the class of unbiased estimators:

$$\mathbf{E}(\tilde{\mathbf{x}} - \mathbf{x}_{\text{true}}) = 0.$$

In addition, the LSM is consistent in the linear model:

$$\lim_{n \rightarrow \infty} \tilde{\mathbf{x}}_n = \mathbf{x}_{\text{true}}.$$

In the case of non-Gaussian errors and a non-linear model the LSM is asymptotically unbiased and consistent.

3.1.2 The Kalman Filter

The application of the Kalman Filter to a track fit was described in [20]. The Kalman Filter is a recursive LSM fit and is used for estimating states of a stochastic model evolving in time (dynamic system). In the case of n measurements of dimension m the LSM requires the inversion of a $nm \times nm$ covariance matrix, which is computationally expensive. In order to apply the Kalman Filter method to a track fit, the track has to be modeled as a dynamic system:

- If the track is described in each intersection point with a measurement surface by a state vector \mathbf{x}_k , its evolution is in fact a discrete dynamic system. The evolution of such a discrete dynamic system can be described by a *system equation*:

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1},$$

with the process noise \mathbf{w}_{k-1} , which takes into account random disturbance of the track between \mathbf{x}_{k-1} and \mathbf{x}_k (mostly multiple scattering). The function \mathbf{f}_{k-1} propagates the track state from detector $k - 1$ to detector k .

- Usually the state vector cannot be observed directly. The relation of the measurement \mathbf{m}_k of detector k and the track state vector \mathbf{x}_k is described by a measurement equation in a most general way:

$$\mathbf{m}_k = \mathbf{h}_k(\mathbf{x}_k) + \boldsymbol{\epsilon}_k,$$

with the measurement noise $\boldsymbol{\epsilon}_k$. In other words, the measured quantities \mathbf{m}_k are functions of the track state vector, corrupted by a measurement noise $\boldsymbol{\epsilon}_k$.

In the simplest case \mathbf{f} and \mathbf{h} are linear functions. In this case the system equation and the measurement equation can be written as:

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1},$$

and

$$\mathbf{m}_k = \mathbf{H}_k\mathbf{x}_k + \boldsymbol{\epsilon}_k.$$

If \mathbf{f} and \mathbf{h} are non-linear they can be approximated by their first-order Taylor expansion. This is called an Extended Kalman Filter. In this case the Jacobians \mathbf{F} and \mathbf{H} have to be computed:

$$\mathbf{F}_{k-1} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{x} = \mathbf{x}_{k-1})$$

$$\mathbf{H}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}(\mathbf{x} = \mathbf{x}_k).$$

Track fitting with the Kalman Filter is performed by three types of operations:

- *Filtering* is the update (estimation) of the state vector with a local measurement (local update of \mathbf{x}).
- *Prediction* is the estimation of the state vector in future time (update of \mathbf{x} in time).
- *Smoothing* is the estimation of the state vector in the past, using all measurements collected up to the present time.

If the system is strictly linear and \mathbf{w}_k and $\boldsymbol{\epsilon}_k$ are Gaussian, the Kalman Filter is the optimal filter. In the case of non-Gaussian errors it is the best linear filter.

Extrapolation in a magnetic field is usually non-linear and need not be done in the linear extrapolation:

$$\tilde{\mathbf{x}}_k = \mathbf{f}_{k-1}(\tilde{\mathbf{x}}_{k-1}).$$

The extrapolation of the covariance matrix \mathbf{C} of $\tilde{\mathbf{x}}$ is done by linear error propagation:

$$\mathbf{C}_k = \mathbf{F}_{k-1} \mathbf{C} \mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1},$$

where \mathbf{Q} is the covariance matrix of \mathbf{w}_k and is assumed to be known. \mathbf{F} is defined by the underlying track model.

The local update of a prediction of $\tilde{\mathbf{x}}$ with a measurement \mathbf{m} on a detector surface can be written in the following:

$$\tilde{\mathbf{x}}_{k,\text{upd}} = \tilde{\mathbf{x}}_{k,\text{pred}} + \mathbf{K}_k (\mathbf{m}_k - \mathbf{H}_k \tilde{\mathbf{x}}_{k,\text{pred}}).$$

\mathbf{K} is the Kalman gain matrix:

$$\mathbf{K}_k = (\mathbf{C}_{k,\text{pred}}^{-1} + \mathbf{H}_k^T \mathbf{V}_k^{-1} \mathbf{H}_k)^{-1} \mathbf{H}_k^T \mathbf{V}_k^{-1}.$$

The updated covariance matrix $\mathbf{C}_{k,\text{upd}}$ is:

$$\mathbf{C}_{k,\text{upd}} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{C}_{k,\text{pred}}.$$

Smoothing of the estimated state vector $\tilde{\mathbf{x}}$ can be done by running two filters in opposite direction and combining both predictions on a detector surface with the measurement. Thus the available information is used in a statistical optimal way.

The advantages of the Kalman Filter with respect to the Global Fit are:

- it is suitable for combined track finding and track fitting,
- no large matrices have to be inverted,
- the estimated track parameters closely follow the true track,
- the linear approximation of the track model needs to be valid only over a short range.

3.1.3 The Gaussian Sum Filter

Least Square Methods have optimal properties if two conditions are met:

- the track model can be approximated by a linear function in a sufficiently large neighbourhood of the track and

- the measurement errors and the process noise are Gaussian distributed.

Especially in the case of multiple scattering and energy loss the second condition might not be fulfilled. If the non-Gaussian error distributions can be described by mixtures of Gaussians, the Kalman Filter can be generalized in order to deal with non-Gaussian noise (see [21]). In this case the resulting algorithm (Gaussian Sum Filter) consists of several Kalman Filters running in parallel but following one and the same physical track.

The Gaussian Sum Filter is a potential candidate for electron filtering, as energy loss of electrons is dominated by bremsstrahlung which has a highly non-Gaussian distribution. One major draw-back of the Gaussian Sum Filter is its computational complexity, another shortcoming is its missing protection against the assignment of a wrong hit when there is no competing “good” one from the actual track.

3.1.4 The Elastic Arm Algorithm

The Elastic Arm Algorithm [22] works with deformable track templates which are attracted to the hits, thus performing track finding and track fitting at the same time. The association between hits and tracks is governed by binary switches, and the attraction depends on the distance between the track and the hits. The problem can be coded in a single energy function with both continuous and discrete variables. In order to reach the global optimum one introduces annealing in which case the binary associators take values between 0 and 1 and can be interpreted as the assignment probabilities between hits and tracks. Thus several (physically different) tracks may compete for a hit. The contribution of a hit to a track is downweighted by the assignment probability. If the final point of the annealing schedule is at zero temperature, the soft assignment becomes a hard assignment.

Another approach separates the estimation of the track parameters from the estimation of the assignment weights by implementing the Elastic Arm Algorithm via an iterated Kalman Filter, which alternatively does a track fit, keeping the assignment weights fixed, and estimates the assignment weights, keeping the track parameters fixed. This is effectively the EM algorithm [24].

3.1.5 The Deterministic Annealing Filter

A way to protect against assignment of wrong hits and at the same time to reduce combinatorics in track reconstruction is to replace hard hit-to-track assignment, as it is done by the Kalman Filter, by soft hit assignment. The assignment of a hit to a track is then no longer binary (yes/no) but expressed via an associated assignment probability, which can take any value between zero and one. Thus hits on a detector surface, which are close enough to a particular track, can compete for a contribution to the track. This approach is very similar to the Elastic Arm Algorithm, but replaces

competition between tracks by competition between hits.

The Deterministic Annealing Filter (DAF, see [23]) allows such a competition between several hits for contribution to a track. The computation of the assignment probability (the weight) of a hit is based on the residuals (predicted or smoothed) of the hit. The problem of insufficient information at the beginning of a fit can be overcome by an iterated procedure which is again effectively the EM algorithm.

After a first complete track fit and smoothing with approximate starting weights, the track state can be predicted in every detector surface, using the information from all other detector surfaces. Based on these predictions, the assignment probabilities can be computed in every surface. If the probability of a hit drops below a certain threshold (e.g. a χ^2 -cut), the hit is suppressed in the next iteration. However, it might reappear after the re-calculation of the assignment probabilities after the next iteration.

The filter itself is an iterated Kalman Filter with annealing. The propagation is identical to the standard Kalman Filter. The update of the estimated state vector $\tilde{\mathbf{x}}$ on a detector surface with n competing hits $\mathbf{m}_1, \dots, \mathbf{m}_n$ is similar to the one of the Kalman Filter:

$$\tilde{\mathbf{x}}_{\text{upd}} = \tilde{\mathbf{x}}_{\text{pred}} + \mathbf{K} \sum_{i=1}^n p_i (\mathbf{m}_i - \mathbf{H}\tilde{\mathbf{x}}_{\text{pred}}).$$

The Kalman gain matrix \mathbf{K} is in this case:

$$\mathbf{K} = (\mathbf{C}_{\text{pred}}^{-1} + p \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{V}^{-1},$$

where p is equal to the sum of all p_i . Finally the updated covariance matrix is given by:

$$\mathbf{C}_{\text{upd}} = (\mathbf{C}_{\text{pred}}^{-1} + p \mathbf{H}^T \mathbf{V}^{-1} \mathbf{H})^{-1}.$$

The assignment weights are proportional to the Gaussian probability density function, using the measurement \mathbf{m}_i and the smoothed state $\tilde{\mathbf{x}}_{\text{smoo}}$ on a particular surface:

$$\phi_i = \frac{1}{(2\pi)^{n/2} \sqrt{\det \mathbf{V}_i}} \exp \left[\frac{1}{2} (\mathbf{m}_i - \mathbf{H}\tilde{\mathbf{x}}_{\text{smoo}})^T \mathbf{V}_i^{-1} (\mathbf{m}_i - \mathbf{H}\tilde{\mathbf{x}}_{\text{smoo}}) \right].$$

n is the dimension of the measurements \mathbf{m} . Finally the assignment weights are normalized:

$$p_i = \frac{\phi_i}{\sum_{j=1}^n \phi_j + \sum_{j=1}^n \phi_{j,\text{cut}}}.$$

$\phi_{j,\text{cut}}$ is a cut-off value which forces the assignment probability to be close to zero if the probability of a hit falls below a certain value. For example, if a 2-dimensional hit with a probability less than 0.001 should be suppressed, $\phi_{j,\text{cut}}$ is:

$$\phi_{j,\text{cut}} = \frac{1}{(2\pi) \sqrt{\det \mathbf{V}_i}} \exp \left[\frac{1}{2} \frac{13.81551}{\beta} \right]$$

The χ^2 -cut value of 13.81551 (probability less than 0.001) can be easily replaced with any other (e.g. 9.21 for a probability less than 0.01).

The variable β is the so-called *annealing factor*. It is not excluded that the final values of the assignment probabilities depend on the initial values at the beginning of the iteration. In order to avoid this annealing can be introduced, which means that the variance \mathbf{V} of a hit depends on the iteration cycle I :

$$\mathbf{V}_I = \beta_I \mathbf{V}$$

A suitable annealing schema can look like $\beta = \{81, 9, 4, 1, 1, 1\}$. In this case the initial value of \mathbf{V}_I at the beginning of the iteration is $81 \cdot \mathbf{V}$. At the end three iterations are performed at the nominal value of \mathbf{V} . However, if one is sure that the starting value of the iteration is close to the true track, annealing is not necessary and a few iterations at nominal variances of the hits are sufficient.

The χ^2 of a track with the competing hits on a surface j is the sum of the individual chi-square:

$$\chi_j^2 = \sum_i \chi_{i,j}^2$$

with

$$\chi_{i,j}^2 = p_i (\mathbf{m}_i - \mathbf{H}\mathbf{x}_{j,\text{pred}})^T (\mathbf{V}_I + \mathbf{H}\mathbf{C}_{j,\text{pred}}\mathbf{H}^T)^{-1} (\mathbf{m}_i - \mathbf{H}\mathbf{x}_{j,\text{pred}}).$$

The total χ^2 of the track is then the sum over all χ_j^2 :

$$\chi^2 = \sum_j \chi_j^2.$$

For the number of degrees of freedom (n_{df}), the number of measurements (n_{m}) has also be multiplied with the assignment probability. For a particular surface j with competing hits of dimension dim_j the number of measurements is then:

$$n_{\text{m},j} = \sum_i p_{i,j} \text{dim}_j = p_j \text{dim}_j$$

The total number of degrees of freedom is therefore:

$$n_{\text{df}} = \sum_j n_{\text{m},j} - n_{\text{p}},$$

where n_{p} is the number of estimated parameters, usually 5 for a track in a magnetic field. Note that the number of measurements n_{m} is not integer, and therefore also the total number of degrees of freedom is not integer.

3.1.6 The MultiRecHit

The DAF-update of a track state with several competing hits can be re-written in the following way:

$$\tilde{\mathbf{x}}_{\text{upd}} = \tilde{\mathbf{x}}_{\text{pred}} + \mathbf{K}'(\mathbf{m}' - \mathbf{H}\tilde{\mathbf{x}}_{\text{pred}}).$$

\mathbf{m}' is a virtual measurement defined as the weighted mean of all competing measurements \mathbf{m}_i :

$$\mathbf{m}' = \left(\sum_i p_i \mathbf{G}_i \right)^{-1} \sum_i p_i \mathbf{G}_i \mathbf{m}_i,$$

with $\mathbf{G}_i = \mathbf{V}_i^{-1}$. The covariance matrix \mathbf{V}' associated to \mathbf{m}' is:

$$\mathbf{V}' = \left(\sum_i p_i \mathbf{G}_i \right)^{-1}.$$

The Kalman Gain matrix and the update of the covariance matrix of the predicted state can then be written in the usual way:

$$\begin{aligned} \mathbf{K} &= (\mathbf{C}_{\text{pred}}^{-1} + \mathbf{H}^T \mathbf{V}'^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{V}'^{-1}, \\ \mathbf{C}_{\text{upd}} &= (\mathbf{C}_{\text{pred}}^{-1} + \mathbf{H}^T \mathbf{V}'^{-1} \mathbf{H})^{-1}. \end{aligned}$$

Thus it is possible to use identical formulas in the update of a track state with several competing hits, by creating a hit with virtual position and virtual covariance matrix out of the individual hits and their assignment weights.

3.2 Methods implemented in ORCA

Currently the Kalman Filter and the Deterministic Annealing Filter are implemented in ORCA. The implementation of other methods should pose no major problems, however, this being foreseen already in the design of ORCA in general and of the track reconstruction code in particular.

3.2.1 Design and implementation of the Kalman Filter in ORCA

After a combinatorial trajectory builder the track parameters are known with full precision at the last point in the sequence of surfaces with measurements, as the entire information of the previous measurements enters the final update. In order to know the track parameters with full precision in all detector surfaces, an independent backward filter has to be run. The standard smoother implemented in ORCA runs an independent Kalman filter in the opposite direction (backward filter). The

information from the backward filter is then combined with the information from the forward filter by a weighted mean, thus providing optimal information of the track in all detector surfaces.

Track reconstruction in ORCA can be decomposed in four parts:

- generation of seeds (starting values) of trajectories,
- building trajectories starting from seeds,
- smoothing of a trajectory, and
- resolving of ambiguities among multiply reconstructed trajectories.

The corresponding object which carries out the task of track reconstruction is the `ModularKFReconstructor`. The class diagram of the `ModularKFReconstructor` is shown in Fig. 3.1. It uses

- a `SeedGenerator`,
- a `TrajectoryBuilder`,
- a `TrajectorySmoother` and
- a `TrajectoryCleaner`

which will be explained in the following.

SeedGeneration

Starting values of potential track candidates are kept within the `TrajectorySeed` object (see Fig. 3.2). It gives access to the `RecHits` used in the seed and the starting values of the track. In addition, the `DetLayers` of the `RecHits` of the seed can be asked. In the case of consecutive hits, which are at the beginning of a track, it can also be asked for the first two `TrajectoryMeasurements`.

The `SeedGenerator` creates `TrajectorySeeds` of potential track candidates. `SeedGenerator` is again a pure abstract base class and only defines a common interface for all specific implementations of possible seed generators. One concrete implementation of the `SeedGenerator` interface is the `CombinatorialSeedGeneratorFromPixel`, as shown in Fig. 3.3

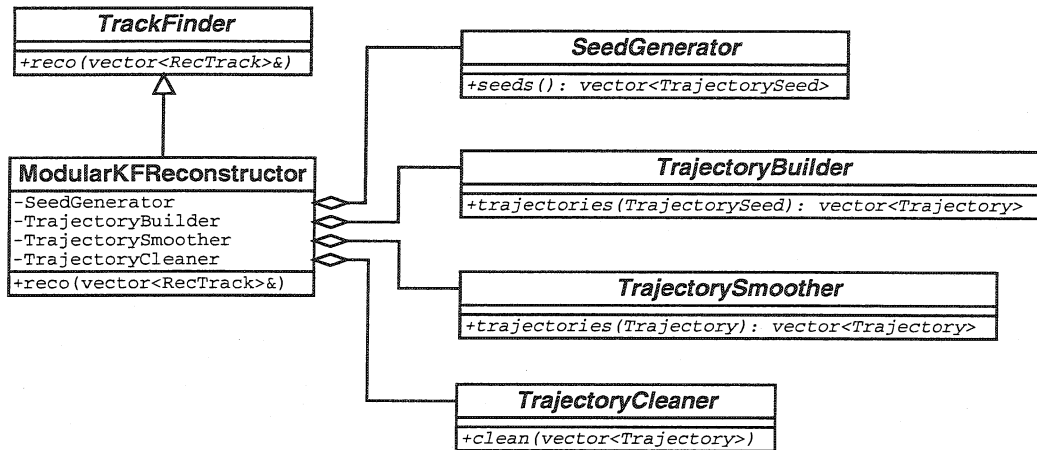


Figure 3.1: Class diagram of the ModularKFReconstructor. The ModularKFReconstructor implements the TrackFinder interface. In order to reconstruct a track, it uses a SeedGenerator and a TrajectoryBuilder for the track finding, a TrajectorySmoother for the track fitting (smoothing) and a TrajectoryCleaner in order to clean the result from multiply reconstructed tracks.

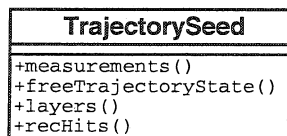


Figure 3.2: The TrajectorySeed is the object which contains starting values of potential track candidates. It can be asked for its RecHits, its DetLayers and its starting position and direction. In the case of two consecutive hits, it can also be asked for the Trajectory-Measurements.

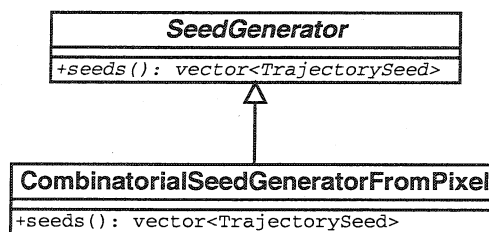


Figure 3.3: The SeedGenerator defines a common interface for all types of seed generators via the virtual method seeds(). One concrete implementation of the SeedGenerator interface is the CombinatorialSeedGeneratorFromPixel.

Trajectory
-vector<TrajectoryMeasurement> -theChi2: double -TrajectorySeed
+measurements() +recHits() +seed() +chiSquared() +push(TrajectoryMeasurement) +pop()

Figure 3.4: The Trajectory is a ordered sequence of TrajectoryMeasurements and a TrajectorySeed. It gives access to its TrajectoryMeasurements and its TrajectorySeed. It can be grown by adding individual TrajectoryMeasurements via the method push(...) and it can be shrunk be the method pop().

Trajectory building

An object of the class Trajectory is an ordered sequence of TrajectoryMeasurements plus the TrajectorySeed it is originating from. Measurements can be added and/or removed from a Trajectory. The Trajectory is a intermediate object during track reconstruction. Objects of class Trajectory are built by the TrajectoryBuilder. Some of the most important methods defined in the class Trajectory are shown in Fig. 3.4.

Trajectory building involves the task of creating one or more candidate trajectories from one input seed. A pure abstract base class TrajectoryBuilder provides a common interface for all types of trajectory builders, for instance the concrete implementation CombinatorialTrajectoryBuilder. The CombinatorialTrajectoryBuilder uses the TrajectorySeeds from a SeedGenerator and finds a possible continuation of the track on the following layers. To achieve this the CombinatorialTrajectoryBuilder uses repeatedly two functions of the DetLayer: the ability to return a list of next layers compatible with a track candidate, and the ability to return the RecHits compatible with a track candidate.

The first function (navigation) is a property of the whole Tracker, and the navigation paths are computed for all layers at initialization time; the resulting layer-to-layer links are stored in the DetLayers for faster access.

The second function (finding the compatible RecHits on a layer) is the most computationally expensive part of the pattern recognition. It is optimised for each specific type of layer to minimize the total number of RecHits that have to be reconstructed and checked for compatibility. The propagation of the candidate track parameters to the measurement surfaces of the layer is performed internally by the DetLayer, since the propagated states are needed to check hit compatibility. To preserve this information the DetLayer returns a collection of compatible TrajectoryMeasurements (see section 2), not just RecHits.

The TrajectoryBuilder uses the Updater (see section 2) to include a next compatible hit in the current Trajectory. In case of more than one compatible hit the Trajectory-

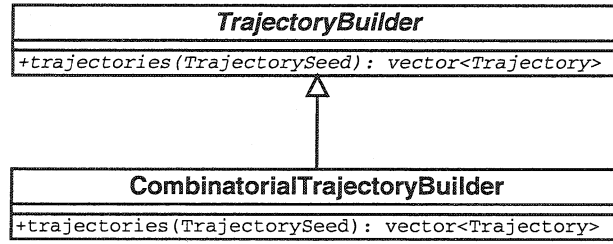


Figure 3.5: The TrajectoryBuilder defines an abstract interface for creating trajectories from input seeds. A concrete implementation is the CombinatorialTrajectoryBuilder.

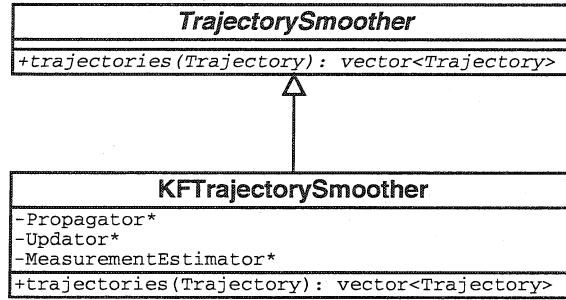


Figure 3.6: The TrajectorySmoother defines an abstract interface for the smoothing of trajectories. A concrete implementation is the KFTrajectorySmoother, which does Kalman Filter smoothing according to [20].

Builder creates one candidate Trajectory for each compatible hit. This procedure continues until the end of the Tracker (no compatible layers), the absence of compatible hits (typical for fake candidates), or some additional stopping condition. If the combinatorial growth of the number of candidate trajectories is not limited, the TrajectoryBuilder can take arbitrarily large computing time and memory for some complex events.

The CombinatorialTrajectoryBuilder uses various strategies to limit the number of candidate trajectories. The most effective is, for a given TrajectorySeed, to truncate the total number of candidate trajectories at each layer to a fixed number, keeping those with the smallest χ^2 values.

Smoothing

The interface for the backward filter with smoothing is provided by the pure abstract base class TrajectorySmoother. A concrete implementation of the TrajectorySmoother is the KFTrajectorySmoother (KF stands for Kalman Filter [20]), see Fig. 3.6. The Kalman Filter smoother combines the results from two track fits in opposite directions and thus uses the available information in a statistically optimal way.

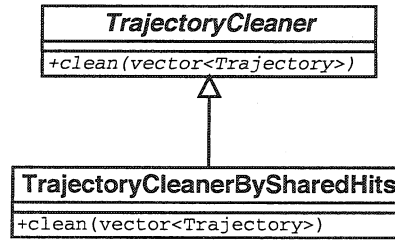


Figure 3.7: The `TrajectoryCleaner` class. It defines an interface for the resolution of ambiguities among multiple reconstructed tracks. A concrete implementation is the `TrajectoryCleanerBySharedHits`, which does ambiguity resolution based on the number of shared ReHits of trajectories.

Trajectory cleaning

A common abstract interface to resolves ambiguities among multiply reconstructed trajectories is provided by the `TrajectoryCleaner`. Ambiguity resolution based on the number of shared hits is done by the `TrajectoryCleanerBySharedHits`. The class diagram is shown in Fig. 3.7.

Fig. 3.8 illustrates the different sequential steps of the process of track reconstruction in ORCA, using these components:

- In a new event, the `ModularKFReconstructor` is asked to reconstruct tracks via the method `reco(...)`.
- The `ModularKFReconstructor` then asks the `SeedGenerator` for seeds. The seeds can be internal to the tracker (e.g. using the Pixel hits to produce initial values of potential track candidates from two hits and a vertex assumption), or external, e.g. a track candidate from the muon chambers or from the calorimeters
- For each seed, the `TrajectoryBuilder` tries to find compatible hits in other layers and to construct a valid `Trajectory`.
- As there is usually more than one seed per track and more than one candidate trajectory per seed, some tracks end up reconstructed several times, leading to ambiguities. The `TrajectoryCleaner` solves these ambiguities and keeps only the track with the best quality.
- Finally a `TrajectorySmoother` combines the results from the track fits in both directions and produces the final result with optimal knowledge of the track in all layers.

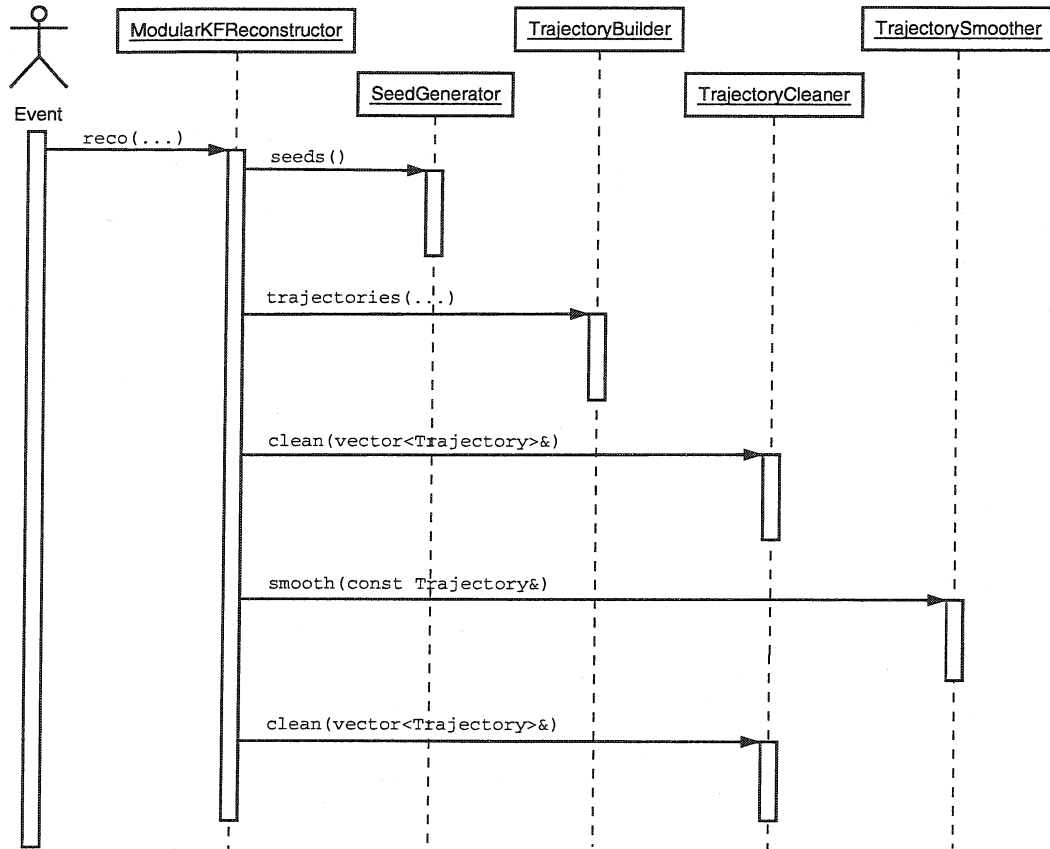


Figure 3.8: Sequence diagram of the ModularKFReconstructor. A SeedGenerator generates seeds of potential track candidates, a TrajectoryBuilder tries to find a track to each seed. A TrajectoryCleaner solves ambiguities between multiple reconstructed tracks of the same physical particle. Finally a TrajectorySmoother produces the final result with optimal knowledge of the track in all measurement surfaces.

3.2.2 Design and implementation of the Deterministic Annealing Filter in ORCA

The class DAFReconstructor uses the method of deterministic annealing for smoothing. The DAFReconstructor inherits from the abstract base class TrackFinder and is composed of the following objects:

- a HitCollector which creates HitCollections,
- an AdaptiveSmootherBase which smoothes a HitCollection, and
- a RecTrackFactory which converts the result of the smoothing into a RecTrack.

The class diagram of the DAFReconstructor is shown in Fig. 3.9.

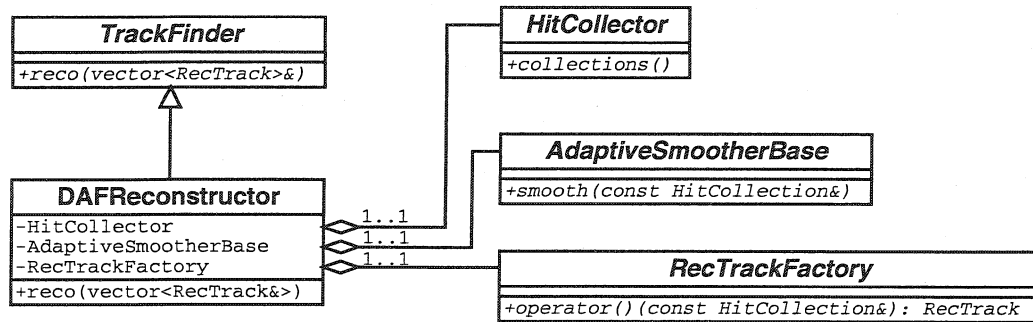


Figure 3.9: Class diagram of the DAFReconstructor. The DAFReconstructor inherits from the abstract base class TrackFinder and has the following objects: a HitCollector, an AdaptiveSmootherBase and a RecTrackFactory.

The chronological order in which the DAFReconstructor uses the objects it is composed of is illustrated in Fig. 3.10. Track reconstruction is done in three steps by the DAFReconstructor:

- The assignment of hits to tracks (track finding) is done by a HitCollector. The HitCollector creates HitCollections which are the base for the later smoothing.
- Adaptive smoothing with annealing is carried out by an AdaptiveSmootherBase. The interface is implemented by the DAFTrajectorySmoother, which uses a configurable annealing schema for the iterations. The result is a smoothed HitCollection.
- Finally the result of the smoothing has to be converted to a RecTrack. This final step is done by a RecTrackFactory, which takes a HitCollection and returns a RecTrack.

The class inheritance diagram of the HitCollector is shown in Fig. 3.11. The interface of the pure abstract base class HitCollector is implemented by two concrete classes, the HitCollectorFromSim and the HitCollectorFromTrackFinder. The HitCollectorFromSim takes preselected SimTracks of a particular event and transforms each of them into a HitCollection which is passed to an AdaptiveSmootherBase afterwards. The HitCollectorFromTrackFinder uses a TrackFinder to find HitCollections, which are again subject to an AdaptiveSmootherBase. The initial TrackFinder can be a standard Kalman Filter.

Fig. 3.12 shows the DAFTrajectorySmoother class diagram. The DAFTrajectorySmoother implements the AdaptiveSmootherBase interface and does smoothing and annealing. In order to do the smoothing, it uses a DAFFitter and a DAFSmoother. A MultiRecHitUpdater is used to recalculate the individual assignment weights after a complete smoothing step (forward fit and backward fit with smoothing).

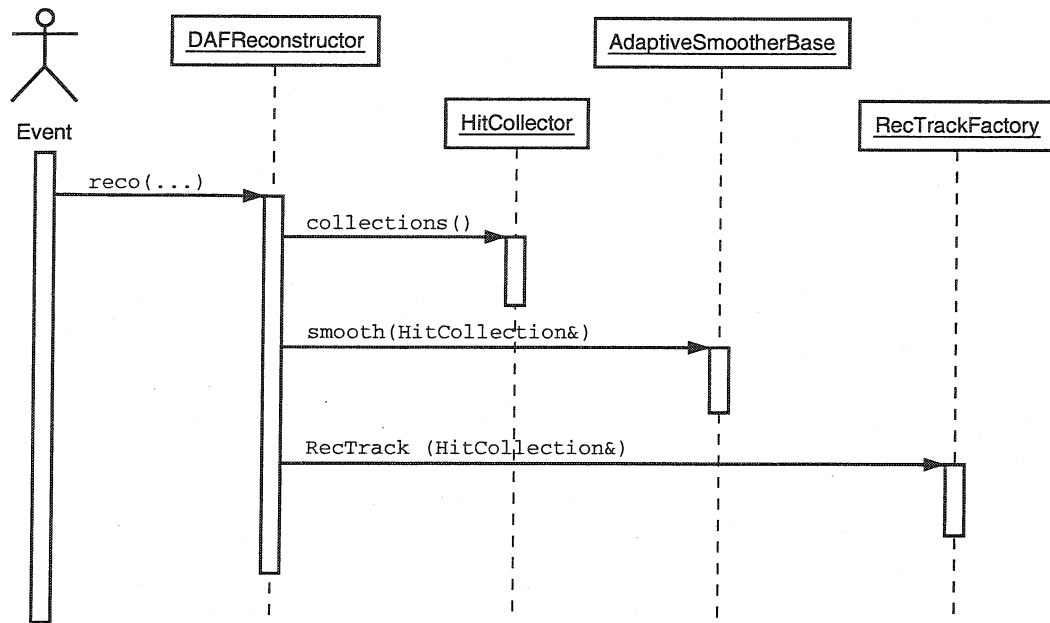


Figure 3.10: Sequence diagram of the DAFReconstructor. Track reconstruction by the DAFReconstructor is done in three consecutive steps: Track finding (assignment of hits to tracks) is done by a HitCollector. Track fitting (track parameter estimation) is done by an AdaptiveSmootherBase. The final result has to be converted to a RecTrack, which can be used for further analysis or vertex finding.

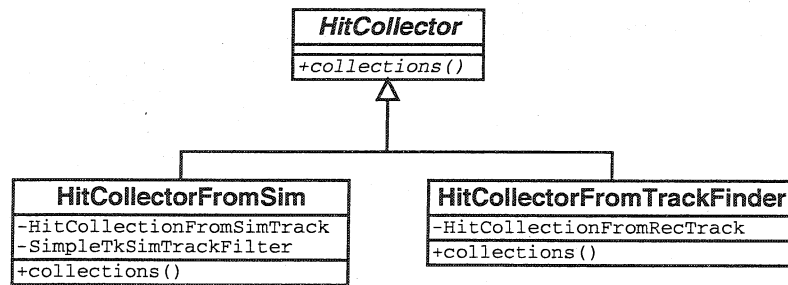


Figure 3.11: The HitCollector class diagram. The interface of the HitCollector is implemented by two concrete classes, the HitCollectorFromSim and the HitCollectorFromTrackFinder.

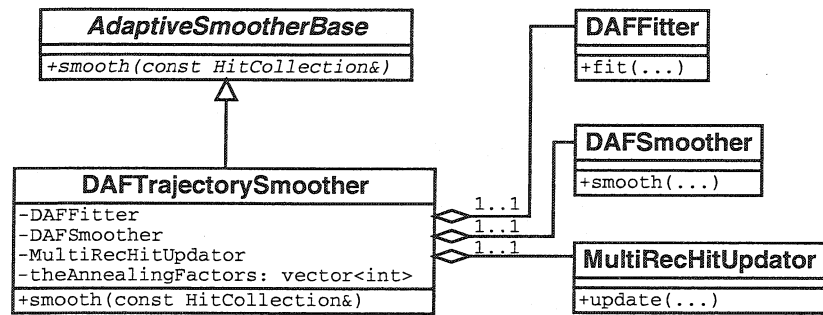


Figure 3.12: Class diagram of the **DAFTrajectorySmoother**. In order to smooth a Hit-Collection, it uses a **DAFFitter** and a **DAFSmoother**. Afterwards the update on the assignment probabilities, using the **MultiRecHitUpdater**.

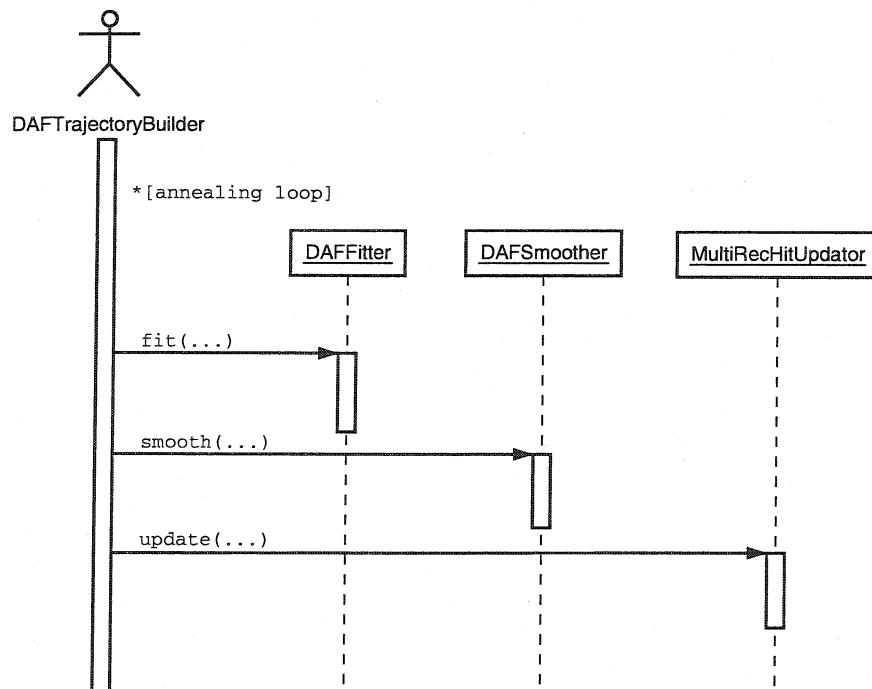


Figure 3.13: Sequence diagram of the **DAFTrajectorySmoother**.

Fig. 3.13 shows the sequential order in which annealing and smoothing is done by the **DAFTrajectorySmoother**. During one iteration with a specific annealing factor, first the **DAFFitter** and then the **DAFSmoother** are called. In the third step, the assignment probabilities are updated with the knowledge of the preceding fit. In a new iteration cycle the fitting and the smoothing are done in the same way as in the previous cycle, and afterwards the assignment probabilities are updated with a lower annealing factor. The final result is returned after the annealing schema has been worked off.

The explicit separation of track finding and track smoothing within the DAF has several advantages:

- the implementations of the `HitCollector` and the `AdaptiveSmootherBase` can be changed and exchanged independently, without affecting each other or other classes,
- as for the smoothing with the DAF the trajectory parameters need not to be known to the ultimate precision at the beginning, the `HitCollector` can be implemented by any other method which does a rough assignment of hits to tracks, taking into account compatible hits at one detector surface,
- the use of a combinatorial Kalman Filter for a `HitCollector` is computationally certainly the most expensive way to find and create `HitCollections` but also the most reliable one. However, any method of track finding and track fitting other than the Kalman Filter has to be compared to it in order to quantify a possible gain in performance and precision. An obvious way to quantify and qualify a gain in precision in a direct way is done by smoothing tracks from the Kalman Filter.

3.3 Algorithm verification

In order to verify the proper behaviour and functionality of the implemented track reconstruction algorithms, a number of tests have to be passed. This is done by testing the goodness of the fit, i.e. the pull quantities and the χ^2 .

- In the first test it is verified that the reconstructed track parameters and track errors are consistent. For this, three conditions must be fulfilled:
 - the track model must be correct,
 - the error model must be correct,
 - the reconstruction program must work properly.

In the case of Gaussian errors, the reconstructed errors are also Gaussian. For a reconstructed track parameter x_i and its corresponding diagonal element of the covariance matrix C_{ii} its pull quantity is defined as

$$P_i = \frac{x_i - x_{\text{true},i}}{\sqrt{C_{ii}}}.$$

In the case of an unbiased estimator and Gaussian errors, P_i is distributed according to a standard normal distribution. In the case of non-Gaussian errors this distribution has mean 0 and variance 1.

- Assuming a correct track and error model with Gaussian errors and a linear model, the quantity

$$\chi^2 = (\mathbf{m} - \mathbf{f}(\mathbf{x}))^T \mathbf{G}(\mathbf{m} - \mathbf{f}(\mathbf{x}))$$

is exactly χ^2 -distributed and has $n - r = \dim(\mathbf{m}) - \dim(\mathbf{x})$ degrees of freedom. In the case of the DAF the number of degrees of freedom is non-integer, as the measurements contribute to the track fit according to their assignment weights. In this case the number of degrees of freedom for 2-dimensional measurements is

$$n_{\text{df}} = 2 \sum p_i - \dim(\mathbf{x}).$$

A χ^2 histogram has mean value $n - r$ and variance $2(n - r)$. Alternatively the χ^2 probability can be histogrammed, which has a uniform distribution in the interval $[0, 1]$.

In order to perform a proper algorithm verification at high level, track reconstruction is done in a fully controlled simulation environment. In this fully controlled simulation environment — the Tracker Fast Simulation — the same track model is used for both simulation and reconstruction of tracks. No material effects are simulated, and the RecHits in the reconstruction can be chosen to be Gaussian smeared SimHits or standard clusterized RecHits with non-Gaussian errors. Finally the qualitative and quantitative performance of the track reconstruction algorithms in presence of random noise hits close to the true track is investigated.

3.3.1 Track parameter histograms

Fig. 3.14 shows some typical pull histograms for tracks of the same type smoothed with the DAF. It shows the pull distributions of the five track parameters (the transverse impact parameter x , the longitudinal impact parameter z , the direction angles φ and λ , and the inverse transverse momentum p_T^{-1}) with Gaussian smeared hits, as well as the χ^2 -probability distribution. Tracks with transverse momentum $p_T = 10 \text{ GeV}/c$ in the η -range $[-2.5, 2.5]$ have been simulated, reconstructed and analysed. The pulls have mean value 0 and a RMS close to 1 in all cases, the χ^2 -probability has a mean value of 0.5.

Fig. 3.15 shows the pull distributions of the five track coordinates with standard clusterized RecHits and non-Gaussian errors for the same type of tracks and the DAF smoother. The pulls have mean 0 and their RMS is reasonably close to 1, within a band width of $\pm 4\%$. The distribution of the χ^2 -probability is biased towards larger values (smaller χ^2), with an approximate mean value of 0.57.

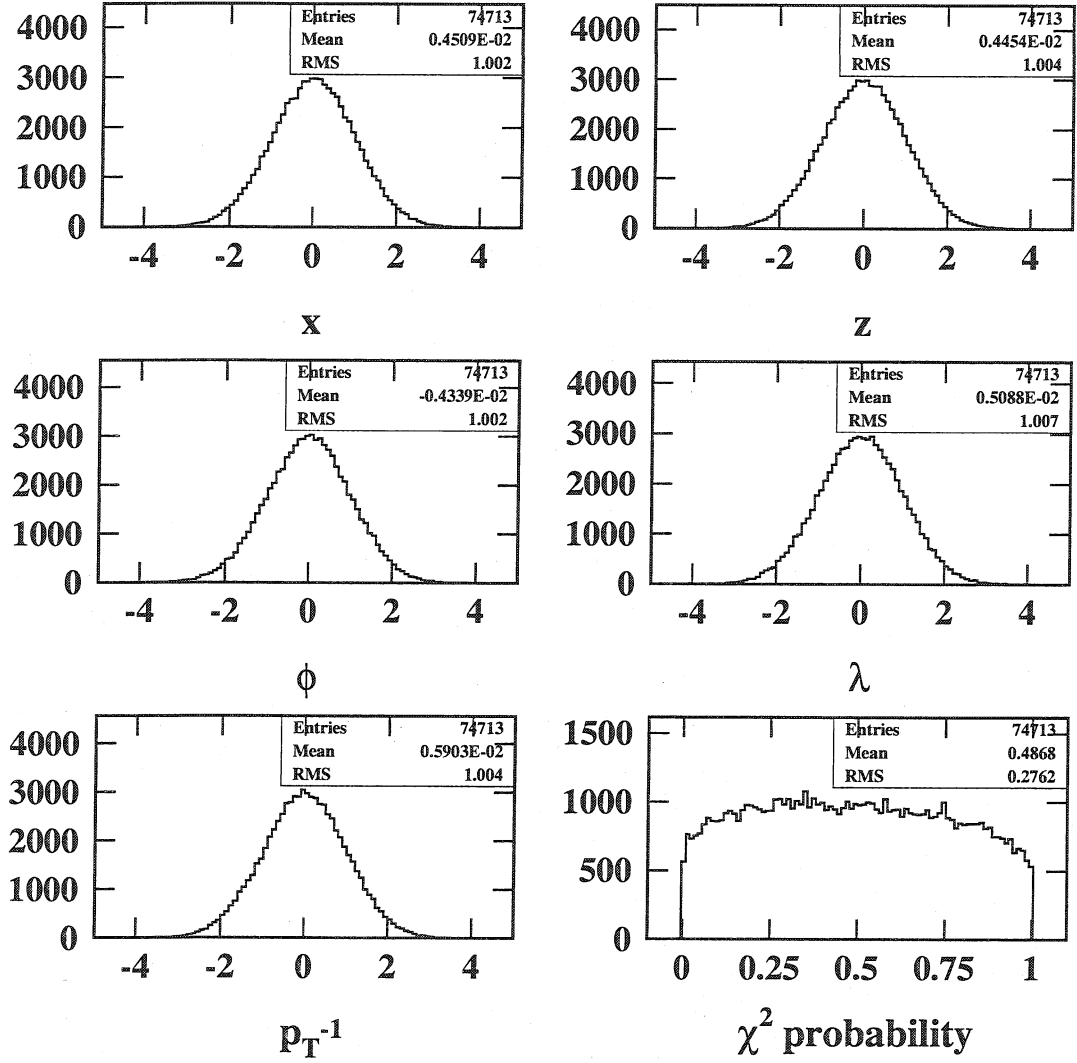


Figure 3.14: Typical pull histograms and χ^2 -probability of reconstructed tracks using the Fast Simulation (Particle Gun) of the CMS Tracker with Gaussian smeared hits. The pulls are unbiased and have an RMS close to one. The χ^2 -probability distribution of the reconstructed tracks has a mean-value close to 0.5 and has a maximum around 0.5.

3.3.2 Track parameter overviews

The RMS of the pulls of p_T^{-1} , the transverse impact parameter (ip) and the mean-value of the χ^2 probability as a function of η for three different p_T -values of the simulated tracks ($p_T = 1, 10$ and 100 GeV/c) are shown in Fig. 3.16. The different points correspond to the DAF with Gaussian smeared hits, DAF with clusterized hits and the Kalman Filter with standard clusterized hits.

In order to have a measure of the actual parameter resolution using clusterized hits, the p_T^{-1} and transverse impact parameter resolution of the DAF is compared with respect to the one of the Kalman Filter (see Fig. 3.17). The impact parameter

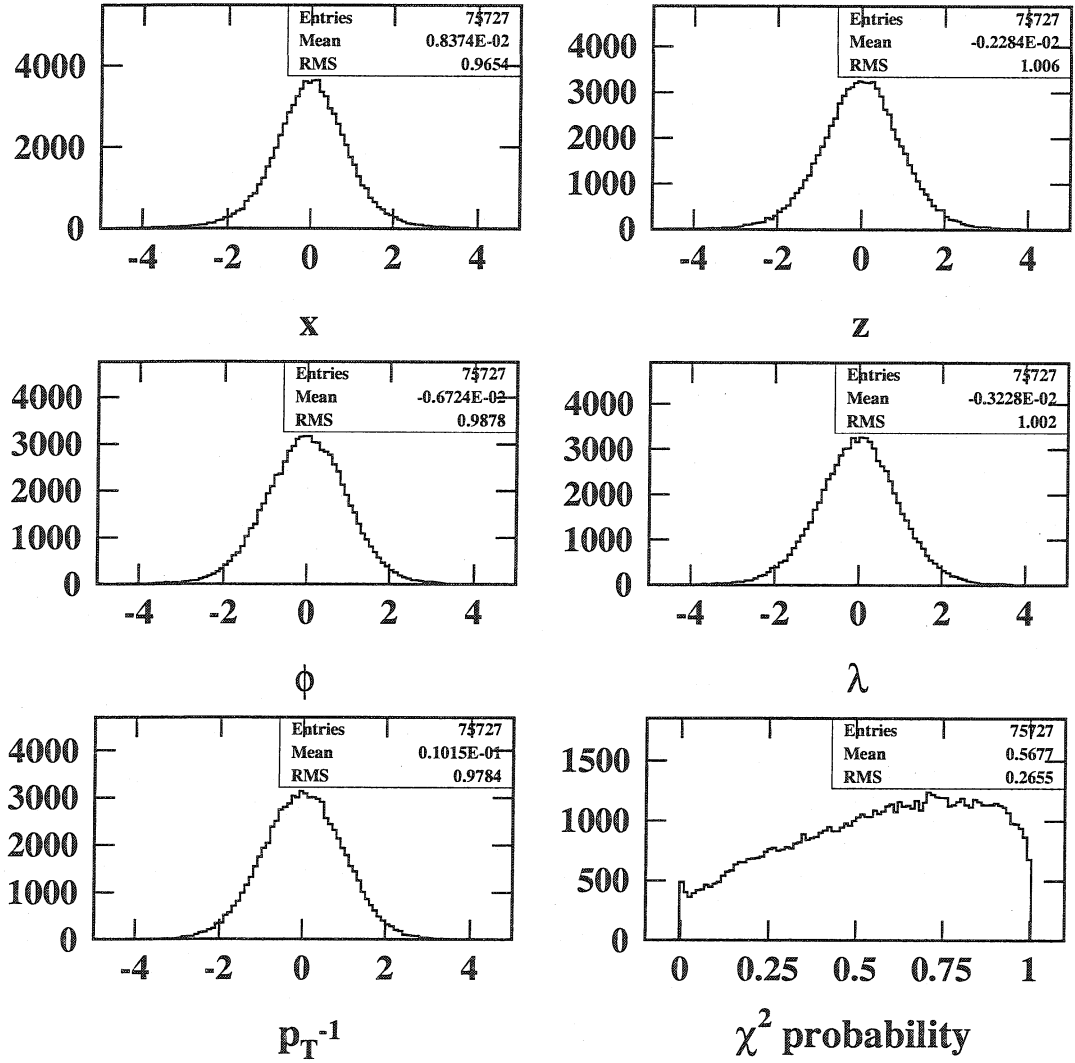


Figure 3.15: Typical pull histograms and χ^2 -probability with the Particle Gun and clustered RecHits (non-Gaussian errors). The pulls are unbiased and have an RMS reasonably close to one.

resolution of the DAF is at most 5% better than the one of the Kalman Filter. The p_T resolution is the same for 1 GeV/c tracks, and about 5% better for the others with the DAF.

3.3.3 Track parameters in presence of random noise hits

In order to investigate the effect of random noise hits along the track, an additional random noise hit was generated within a distance of 0.1 cm of each simulated hit. Using the standard clusterizer for generating RecHits, it is possible that these two simulated hits are merged during digitization and produce only one big cluster if the random noise hit is too close to the true one. Fig. 3.18 shows the effect on the pull

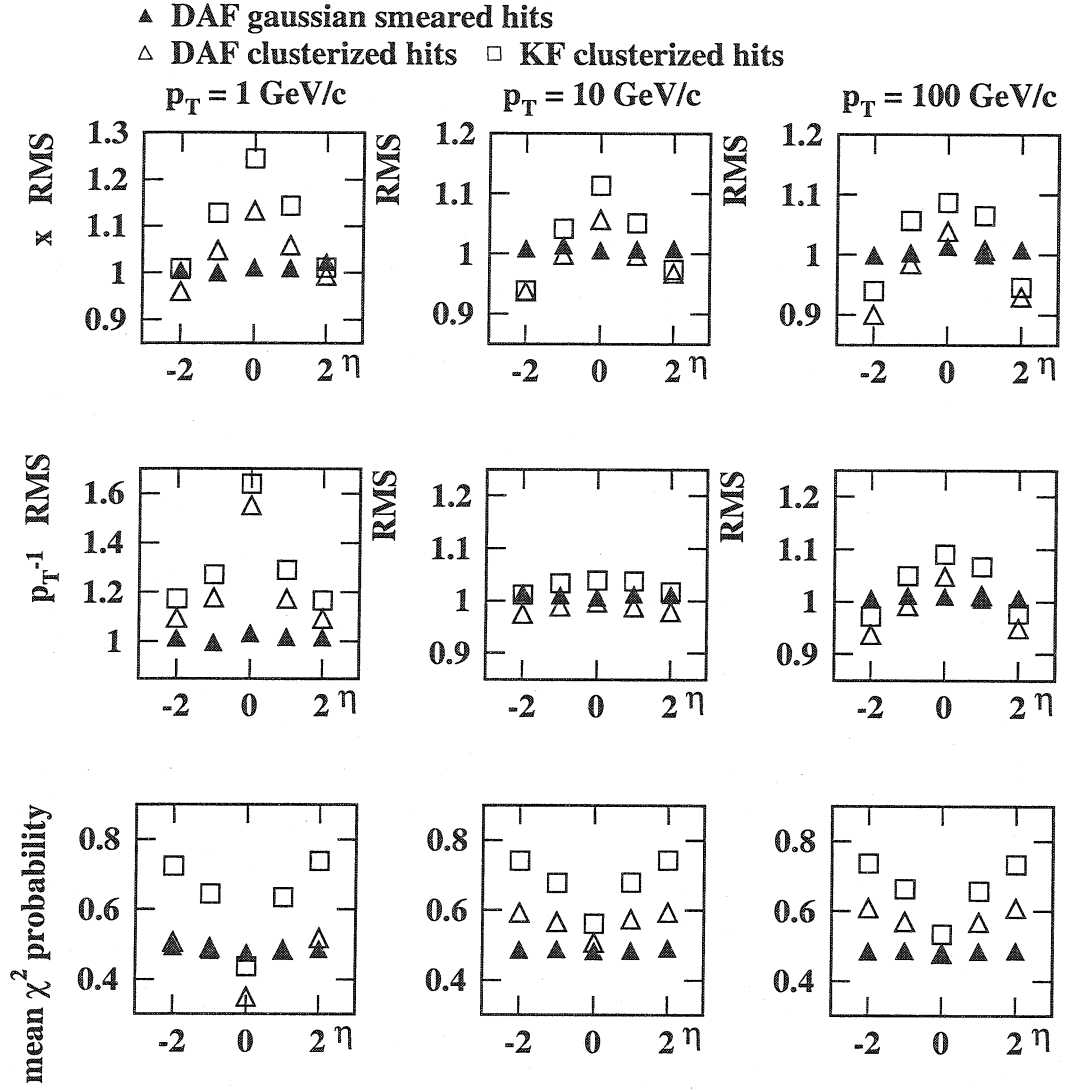


Figure 3.16: Overview of RMS of pulls of the inverse transverse momentum p_T^{-1} , the transverse impact parameter and the mean χ^2 probability. The results have been produced for three different p_T -values of the simulated tracks within the full η -range of the CMS Tracker.

histograms of the DAF. The RMS of the pull distributions is between 2 and 3, the mean value of the χ^2 -distribution is too small.

Fig. 3.19 gives an overview of the pulls for p_T^{-1} , transverse ip and the mean χ^2 probability. Although the RMS of the pulls is too big for both the DAF and the Kalman Filter, the DAF has systemically better reconstruction quality than the Kalman Filter.

Fig. 3.20 gives an overview of the precision for p_T^{-1} and transverse impact parameter of the DAF with respect to the Kalman Filter. The smoothing of the DAF leads to a 10% to 20% better resolution compared to the Kalman Filter.

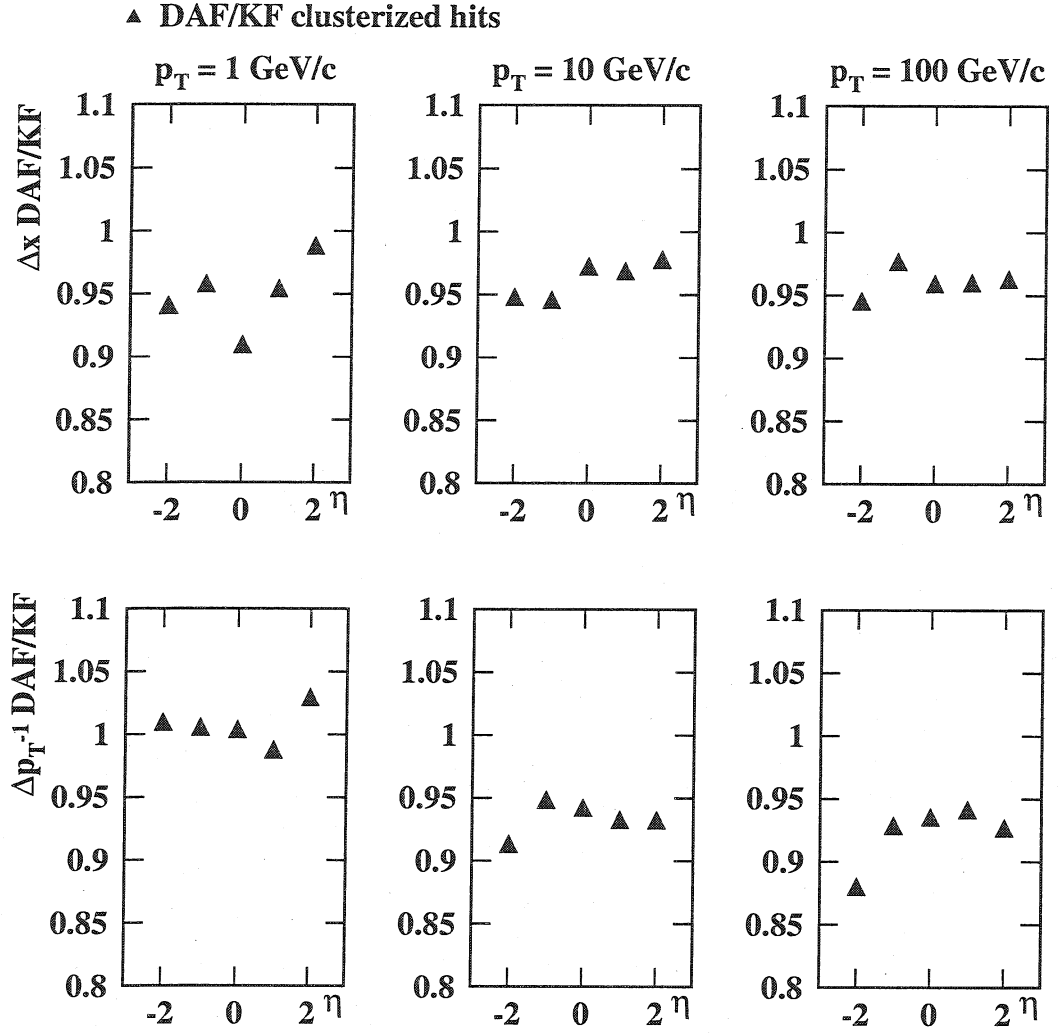


Figure 3.17: Parameter resolution of the DAF with respect to the Kalman Filter. The impact parameter resolution of the DAF is at most 5% better than the one of the Kalman Filter. The p_T resolution is the same for 1 GeV/c tracks, and about 5% better for the others with the DAF.

One reason for worse pulls and residuals in presence of random noise hits is the contamination of the “good” RecHits with charge from hits of other tracks. This can lead to a bias in the position coordinate and error estimation of the hit.

Fig. 3.21 shows the evolution of the RMS of the transverse impact parameter and p_T^{-1} pulls as a function of the fraction of contaminated hits in the track, normalized with respect to the pulls without the presence of random noise hits. The points follow a linear behaviour as the amount of contaminated hits increases.

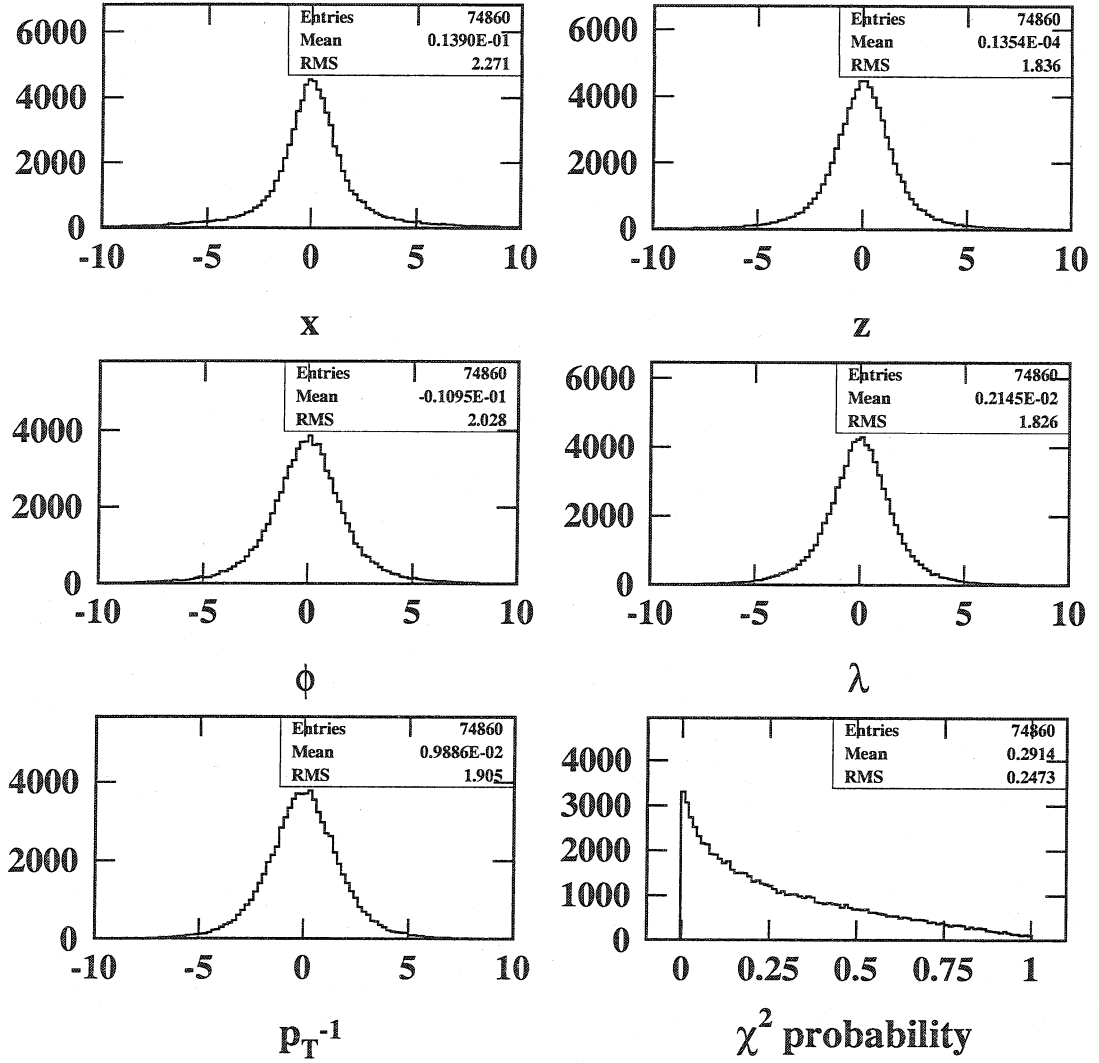


Figure 3.18: Pull distributions for the DAF with one additional random noise hit within 0.1 cm distance to the true track in each layer. The RMS of the pulls is between 2 and 3 and the mean value of the χ^2 distribution is too small.

3.4 Conclusion

The DAF smoother produces consistent results within the limit of a perfectly known track model and Gaussian errors of the measurements.

In the presence of non-Gaussian errors the pulls of the DAF and the Kalman Filter are too large for tracks of $p_T = 1 \text{ GeV}/c$. Their RMS is reasonably close to 1 for tracks above a p_T of $10 \text{ GeV}/c$. The results are worse in the presence of random noise hits, with the DAF producing systematically better results than the Kalman Filter.

The DAF smoothing leads to systematically better resolutions in track parameters than the Kalman Filter. The magnitude is of the order of 10% to 20%, depending

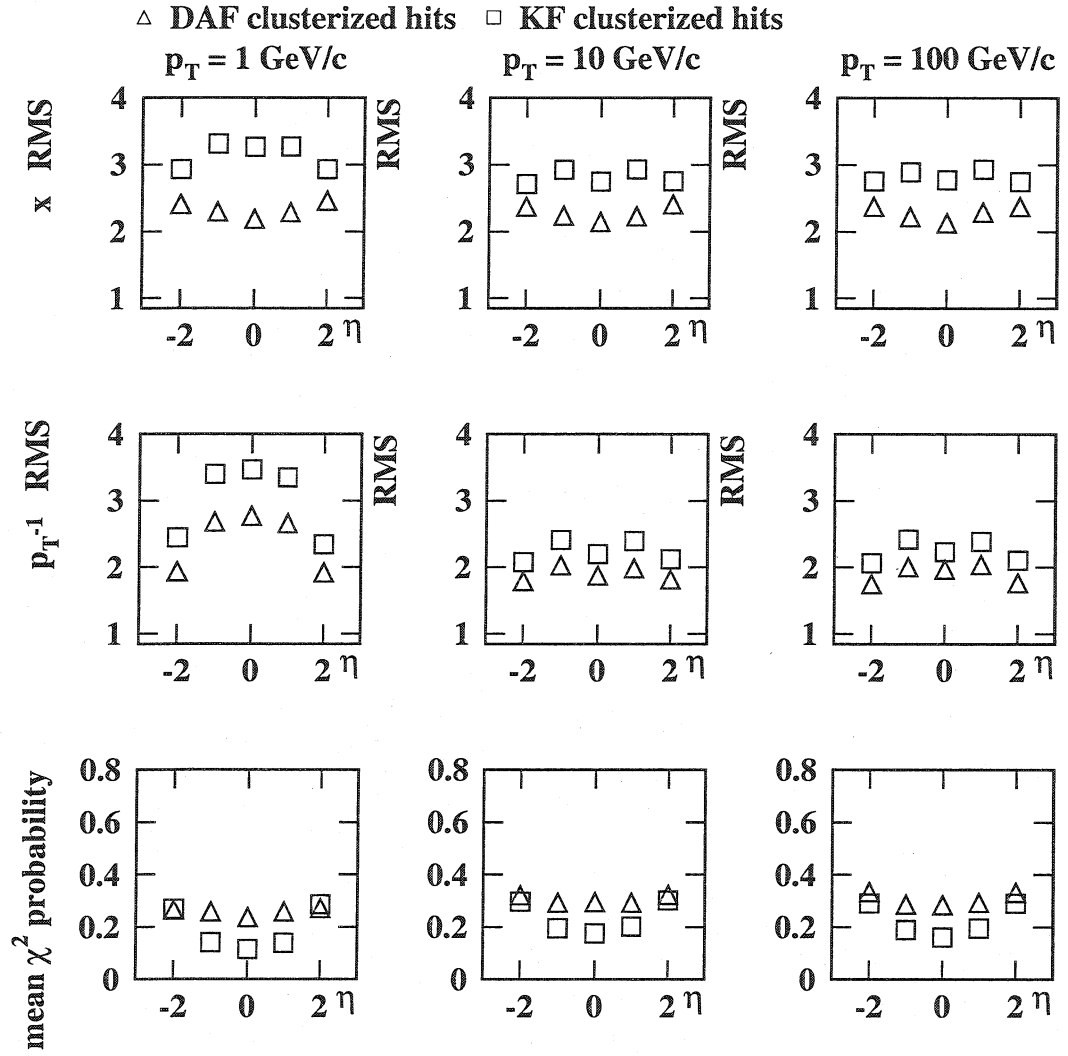


Figure 3.19: Overview of the pulls p_T , the transverse impact parameter and the mean χ^2 probability. The RMS of the pulls is too large both for the Kalman Filter and the DAF, the ones of the DAF being systematically smaller than the Kalman Filter pulls.

on the p_T and the η of the track.

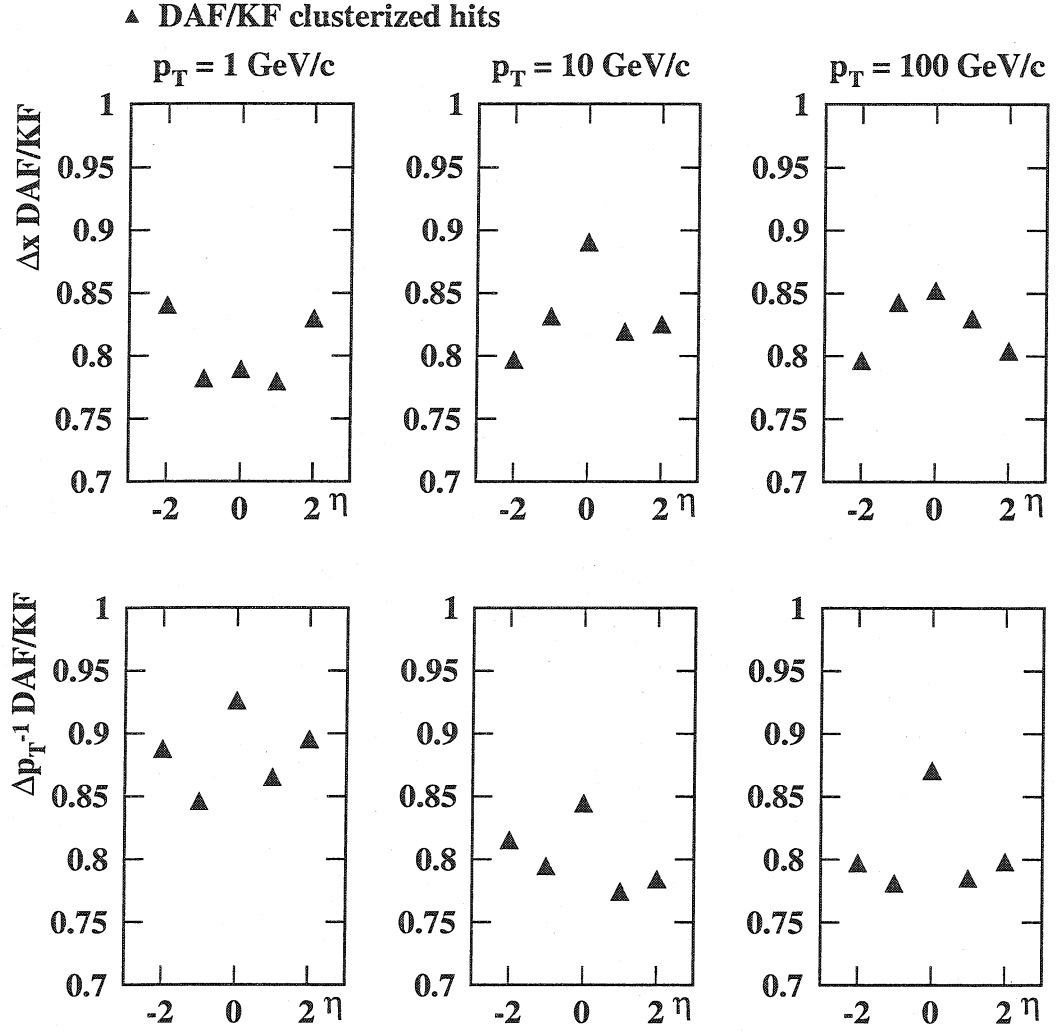


Figure 3.20: Overview of the relative resolution of the DAF with respect to the Kalman Filter in the presence of additional random noise hits. The resolutions obtained by the DAF are systematically better than the one from the Kalman Filter.

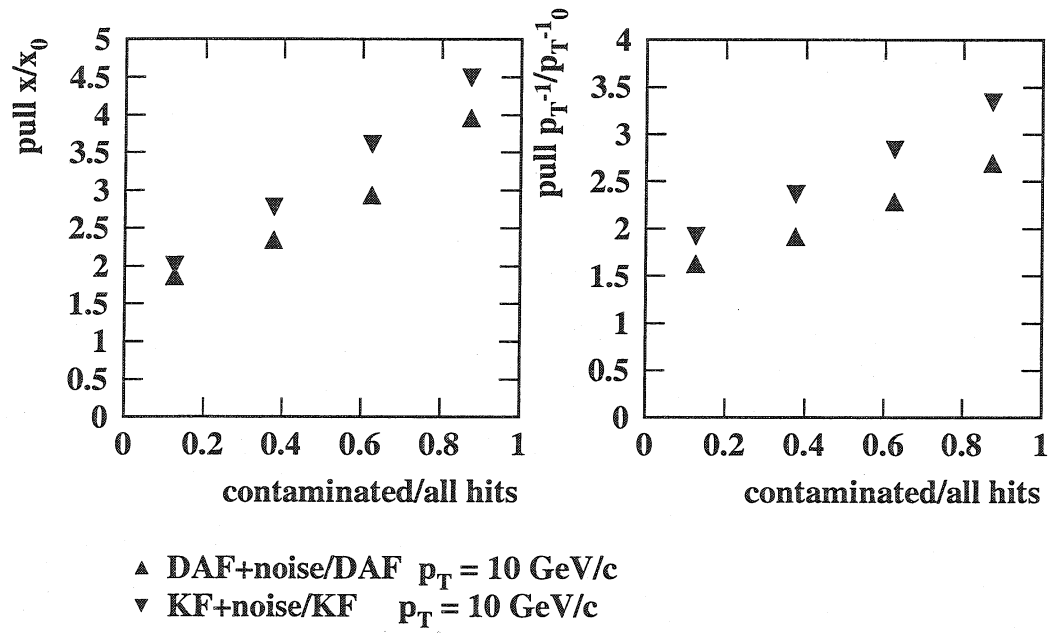


Figure 3.21: Evolution of the RMS of pulls in presence of random noise hits with respect to pulls without random noise hits. The points virtually follow a linear behaviour as the fraction of contaminated hits increases. The slope is smaller for the DAF than for the Kalman Filter.

4 Multi track fitting methods in ORCA

Even the reconstruction of isolated tracks is a difficult task in CMS because of the large amount of noise hits in the tracker (electronic noise, low momentum tracks, delta electrons etc). In the previous section it was shown that for isolated tracks the DAF [23] is a powerful tool for solving the hit-to-track assignment problem in a noisy environment. The problem becomes worse in dense jets of tracks where tracks are so close to each other that most hits could be associated to either of the tracks. It might even happen that hits of two tracks are merged in one cluster, thus leading to an overlap of the tracks. The DAF is not designed to deal with situations where two or more tracks are so close to each other that most hits could be associated to either of them.

Another well-known method for solving the problem of hit assignment and estimating track parameters at the same time is the Elastic Arm Algorithm (EAA) [22]. In the logic of the EAA, tracks are competing for the association to hits. In this respect it can be seen as the opposite to the DAF, where hits compete for the contribution to a track. The EAA is a global method in the sense that all tracks in question are treated in parallel.

A Multitrack Filter (MTF) as proposed in [25] tries to solve the problem of assigning hits and estimating parameters within a global approach, by allowing true competition among hits and tracks at the same time. Such an approach should perform better than pure competition between hits (DAF) or pure competition between tracks (EAA).

For the evaluation of the performance of the MTF and for basic algorithm verification purposes, events with pairs of close tracks have been generated with the Tracker fast simulation (particle gun). The results of the MTF will be compared to the ones of a standard Kalman Filter [20], which solves the assignment problem of hits to tracks by hard assignment, so that a hit either does or does not contribute to a track.

4.1 Principle

The MTF, like the DAF, is implemented as an iterated Kalman filter with the possibility of annealing. In contrast to the DAF, several tracks are fitted concurrently. The MTF starts with a collection of track candidates and a collection of hits which are compatible with at least one of the candidates. Finding the correct number of track candidates and reasonable starting values for the track parameters is a non-trivial pattern recognition problem. Here it is assumed that the pattern recognition has been done in a previous step, for instance by using simulation information or using tracks from the combinatorial Kalman Filter as a starting base.

From the initial track parameters the initial assignment probabilities of all hits with respect to all tracks are computed. These assignment probabilities are used as weights

in the following estimation of track parameters (track fit). A hit with a small assignment probability with respect to a certain track contributes very little to the estimation of the parameters of this track. The results of the track fits are then used in turn for recalculating the assignment probabilities, and so on. This iterated procedure is repeated until the convergence of the assignment probabilities.

Now suppose that on a detector surface of a certain detector layer there are m track candidates and n hits compatible with at least one of the track candidates. The state of track i can be described by a state vector \mathbf{x}_i , and the observed values of hit j by a measurement vector \mathbf{m}_j . The assignment probabilities p_{ij} of the hits with respect to the tracks on a detector surface can be arranged in a matrix which has one row for each track and one column for each hit:

	hits _{j} →
states _{i} ↓	p_{ij}

The calculation of the assignment probabilities is based on the normal probability density ϕ_{ij} :

$$\phi_{ij} = \phi(\mathbf{m}_j; \mathbf{H}\mathbf{x}_i, \mathbf{V}_j).$$

\mathbf{H} is the projection matrix (Jacobian) which maps the track state \mathbf{x} on the measurement \mathbf{m} , described by the measurement equation:

$$\mathbf{m} = \mathbf{H}\mathbf{x} + \epsilon.$$

ϵ is the error of the observation \mathbf{m} , which is assumed to have mean zero and the covariance matrix \mathbf{V} . \mathbf{V} depends on the annealing schedule: it is larger than the nominal measurement error at the beginning of the iterative track fit and decreases after each step (annealing).

Three methods can be chosen for computing the assignment probabilities:

- Competition between hits.

There is competition between all hits for each track, but there is no competition between the tracks. This procedure is equivalent to the DAF. The assignment probabilities are computed by normalizing the each row of ϕ_{ij} by the sum of the row plus a constant (normalization by rows):

$$p_{ij} = \frac{\phi_{ij}}{\sum_k \phi_{ik} + c}.$$

- Competition between tracks.

There is competition between all tracks for each hit, but there is no competition between the hits. This procedure is equivalent to the EAA. The assignment probabilities are computed by normalizing the each column of ϕ_{ij} by the sum of the column plus a constant (normalization by columns):

$$p_{ij} = \frac{\phi_{ij}}{\sum_l \phi_{lj} + c}.$$

- Competition between hits and tracks (global competition).

This procedure introduces true competition between hits and tracks which are incompatible, i.e. whenever they come from the same row or the same column. The assignment probabilities are computed by dividing ϕ_{ij} by the sum of all elements in the same row and column plus a constant (normalization by columns and rows):

$$p_{ij} = \frac{\phi_{ij}}{\sum_k \phi_{ik} + \sum_l \phi_{lj} - \phi_{ij} + c}.$$

The constant c defines a cut beyond which the assignment probability quickly drops to zero. All measurements \mathbf{m}_j in one row are combined by a weighted mean to a single virtual measurement. These combined measurements (MultiRecHits, see section 3) are then used in the track parameter estimation in the usual way.

As one can imagine, the initialization of the weights at the beginning of a multitrack fit is difficult, as the success of the fit depends on the quality of the initial guess. One possible way to get satisfying starting values is to use tracks from the Kalman Filter, which originate from the same vertex and are sufficiently close in momentum space (eventually share hits). For the Multitrack Fit these hits are merged in a collection of hits, and the results from the Kalman Filter track fit are used as starting values for the Multitrack Fit. This method can be compared to the initialization with knowledge of the parameters of the true tracks from the simulation.

4.2 Implementation in ORCA

The implementation of the MTF closely follows the implementation of the DAF. Analogous to the class `DAFReconstructor`, a class `MTFReconstructor` has been created which makes use of the multitrack fitting method. The `MTFReconstructor` inherits from the abstract base class `TrackFinder` and is composed of the following objects:

- a `MTFHitCollector` which creates `MTFHitCollections`,
- a `MTFSmoothingBase` which smoothes a `MTFHitCollection` and

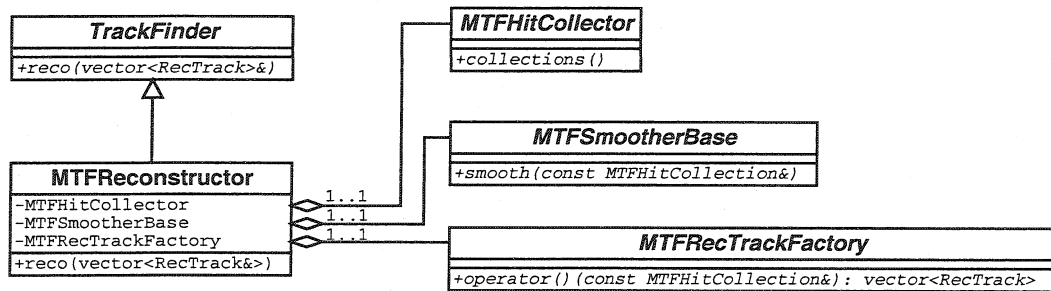


Figure 4.1: Class diagram of the MTFReconstructor. The MTFReconstructor inherits from the abstract base class TrackFinder and has the following objects: a MTFHitCollector, an MTFSmoothingBase and a MTFRecTrackFactory.

- a MTFRecTrackFactory which converts the result of the smoothing into the final result, the RecTracks.

Fig. 4.1 shows the class diagram of the MTFReconstructor.

Fig. 4.2 illustrates the chronological order in which the MTFReconstructor uses the its objects. Track reconstruction is done in three steps by the MTFReconstructor:

- The assignment of hits to tracks (track finding) is done by a MTFHitCollector. The MTFHitCollector creates MTFHitCollections which are the base for the later smoothing.
- The multi track smoothing with annealing is done out by a MTFSmoothingBase. The interface is implemented by the DAFTrajectorySmoother, which uses a configurable annealing schema for the iterations. The result is a smoothed MTFHitCollection.
- Finally the result of the smoothing has to be converted to reconstructed tracks (RecTracks). This final step is done by a MTFRecTrackFactory, which takes a MTFHitCollection and returns a vector of RecTracks.

Fig. 4.3 shows the class inheritance diagram of the MTFHitCollector. The interface of the pure abstract base class MTFHitCollector is implemented by two concrete classes, the MTFHitCollectorFromSim and the MTFHitCollectorFromTrackFinder. The MTFHitCollectorFromSim takes preselected SimTracks of a particular event and transforms them into a MTFHitCollection which is passed to an MTFSmoothingBase afterwards. The MTFHitCollectorFromTrackFinder uses a TrackFinder first and converts the found RecTracks to MTFHitCollections, which are again subject to an MTFSmoothingBase. The initial TrackFinder can be a standard Kalman Filter, and the MTFSmoothingBase is used to improve the result in order to get maximum precision.

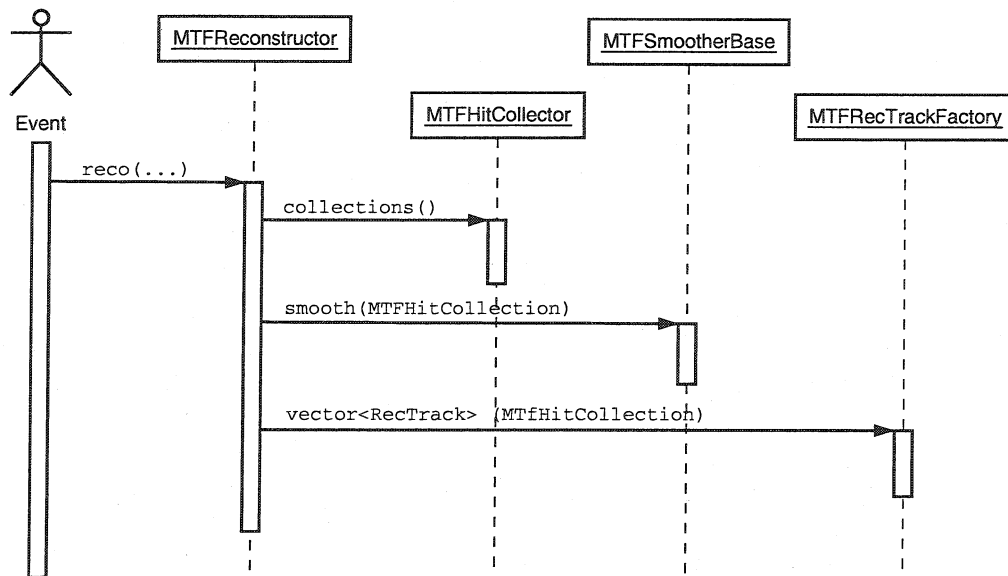


Figure 4.2: Sequence diagram of the MTFReconstructor. Track reconstruction by the MTF-Reconstructor is done in three consecutive steps: Track finding (assignment of hits to tracks) is done by a MTFHitCollector. Track fitting (track parameter estimation) is done by an MTFSmotherBase. The final result has to be converted to RecTracks, which can be used for further analysis or vertex finding.

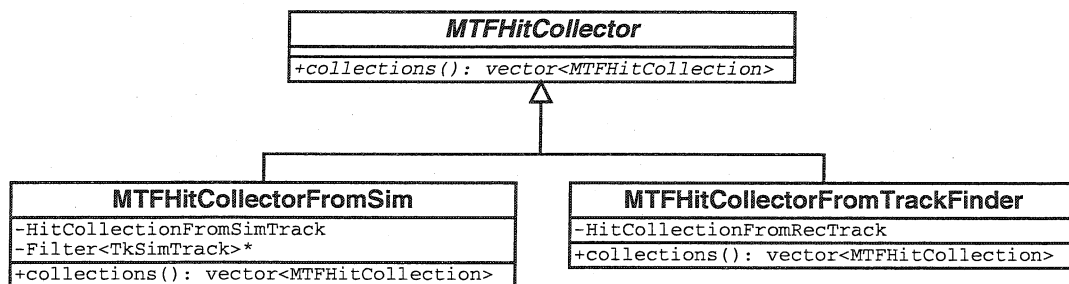


Figure 4.3: Class diagram of the MTFHitCollector. Actually the interface of the MTF-HitCollector is implemented by two concrete classes, the MTFHitCollectorFromSim and the MTFHitCollectorFromTrackFinder.

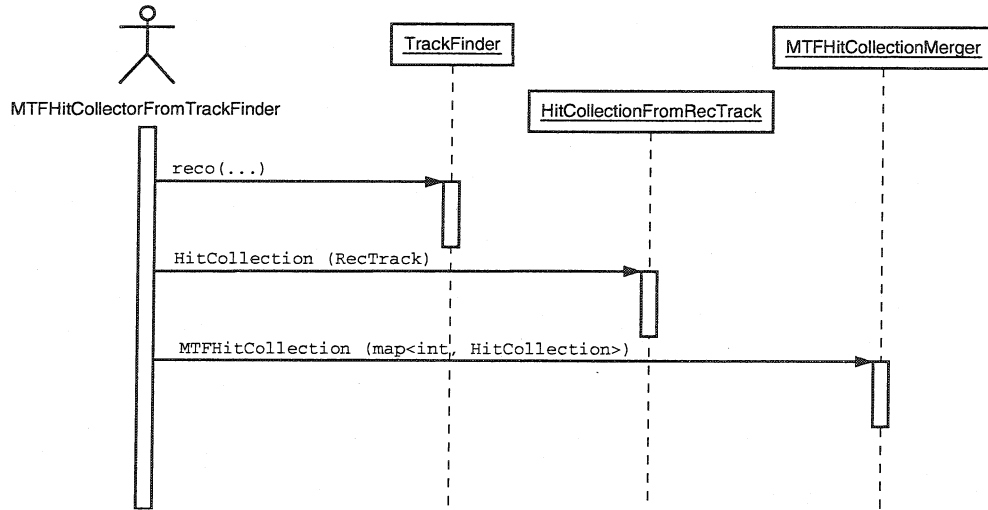


Figure 4.4: Sequence Diagram of the MTFHitCollectorFromTrackFinder. A TrackFinder is used to do the initial pattern recognition. Each RecTrack is then transformed into a HitCollection. Hit collections of tracks which are close enough in space are put into a map. This map of HitCollections is then transformed into a MTFHitCollection by the MTFHitCollectionMerger.

Fig. 4.4 shows the sequential order in which the MTFHitCollections are created by the MTFHitCollectorFromTrackFinder. First a TrackFinder is launched to reconstruct tracks. Each RecTrack is transformed into a HitCollection. Then HitCollections of tracks which are close enough in space are put into a map. This map of HitCollections is then transformed into a MTFHitCollection by the MTFHitCollectionMerger.

Fig. 4.5 shows the inheritance diagram for the MTFSmoothingBase. The interface is implemented by the two different classes. The CombinedMTFTrajectorySmoother re-implements the interface by looping over a sequence of MTFSmoothingBases, of which the CombinedMTFTrajectorySmoother consists. This allows to run multitrack smoothers with different weight-update schemas in a sequential order, e.g. first a Multitrack Fit with Elastic Arm update mode, then a Multitrack Fit with global competition. The MTFTrajectorySmoother uses a class MTFFitter, MTFSmoothing, MTFMultiRecHitUpdater and MTFMultiHitEstimator in order to implement the interface. The algorithm for updating the weights of the hits and tracks is encapsulated within the class MTFMultiRecHitUpdater.

Fig. 4.6 shows the sequential order in which annealing and smoothing is done by the MTFTrajectorySmoother. During one iteration with a specific annealing factor, first the MTFFitter and then the MTFSmoothing are used. In the third step, the assignment probabilities are updated with the knowledge of the preceding fit by the MTFMultiRecHitUpdater. The χ^2 is calculated by the MTFMultiHitEstimator in the final step. In a new iteration cycle the fitting and the smoothing are done in the same way as in the previous cycle, and afterwards the assignment probabilities are

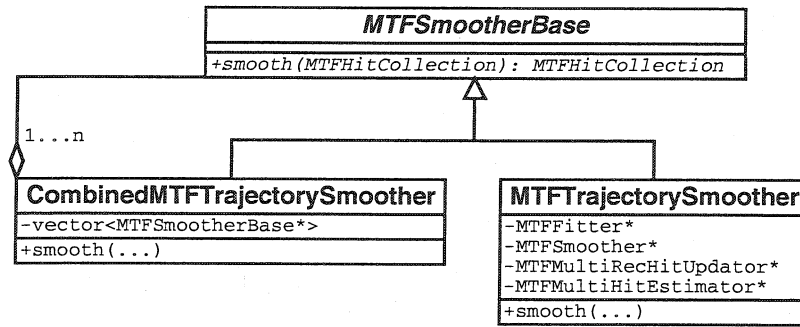


Figure 4.5: Implementation of the MTFSmootherBase. The interface is implemented by two classes, the CombinedMTFTrajectorySmoother and the MTFTrajectorySmoother.

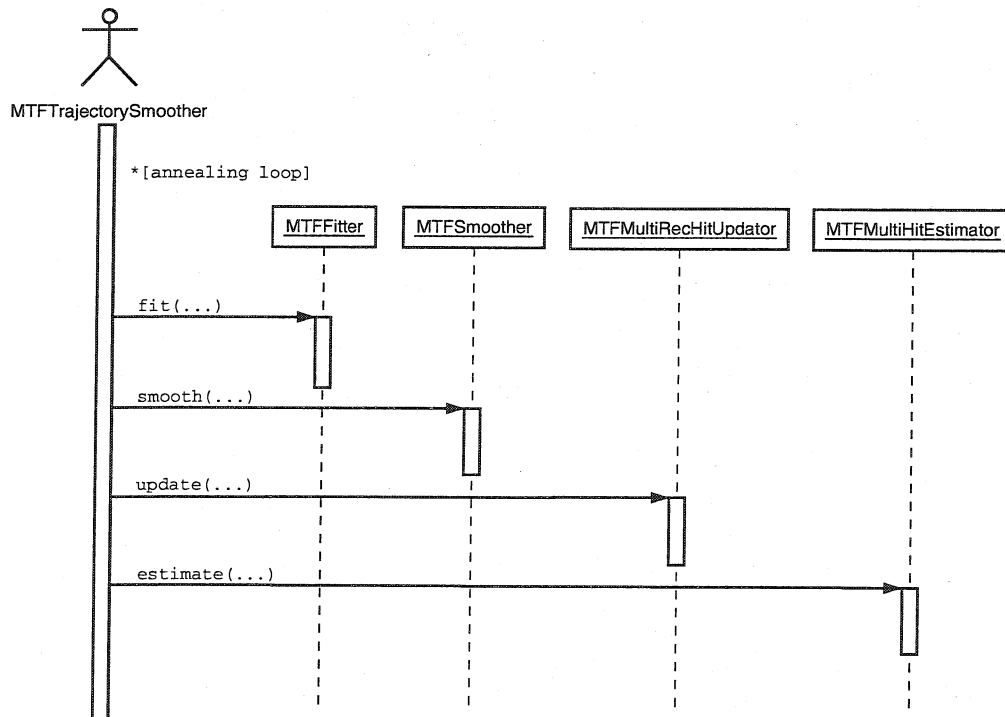


Figure 4.6: Sequence diagram of the MTFTrajectorySmoother.

updated now with a lower annealing factor. The final result is returned after the annealing schema has been worked off.

Fig. 4.7 shows how the MTFMultiRecHitUpdater uses the MTFWeightCalcStrategy in order to do the calculation of the assignment probabilities.

The pure abstract base class MTFWeightCalcStrategy is implemented by three different classes, as can also be seen from Fig. 4.8: the DAFWeightCalculator, the EA-WeightCalculator and the MTFWeightCalculator. These three classes correspond to the different ways of update mechanisms described above.

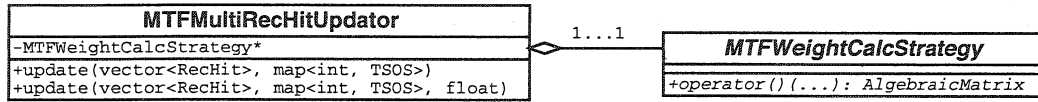


Figure 4.7: Class diagram of the MTFMultiRecHitUpdater. The MTFMultiRecHitUpdater uses a MTFWeightCalcStrategy in order to calculate the assignment probabilities.

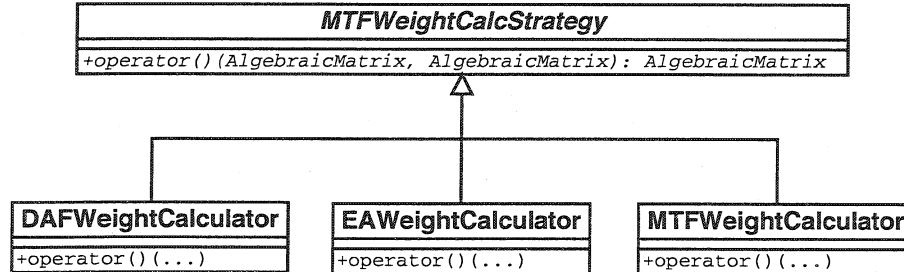


Figure 4.8: Inheritance diagram of the MTFWeightCalcStrategy. It is implemented by three different classes, the DAFWeightCalculator, the EAWeightCalculator, and the MTFWeightCalculator. They correspond to the different update mechanisms described above.

The design of the Multitrack Fit is flexible enough to allow also for other track finding methods (pattern recognition) than the Kalman Filter to be easily implemented. Once a MTFHitCollection has been created, the MTFTrajectorySmoother can be used. By choosing different MTFMultiHitUpdaters which differ by their MTFWeightCalcStrategy and thus creating MTFSmootherBases which are different by the way the assignment probabilities are calculated, it is possible to have consecutive smoothing with different update schemas.

4.3 Algorithm Verification

For the algorithm verification we have used a modified version of the Particle Gun which generates pairs of tracks in a fully controlled simulation environment. The tracks are generated according to the following settings:

- particles: pairs of muons μ^\pm
- p_T -range: 30 GeV/c – 300 GeV/c
- η -range: $-0.9 < \eta < 0.9$
- δp_T : $\pm 0.2\%$
- $\delta\varphi$: ± 0.002 rad
- $\delta\lambda$: ± 0.002 rad

The following tests were carried out in order to demonstrate the correctness of the implementation and to quantify and qualify the difference with respect to the standard Kalman Filter:

- In the first test Gaussian smeared hits were used. Proper initialization of the weights was forced by using the simulation information of the tracks. Pulls should then have a RMS of 1 and the χ^2 -probability must have a mean value of 0.5.
- In the second test the effect of non-Gaussian hit errors on the result were investigated, using again prior knowledge from the simulation.
- Finally it was investigated whether the Multitrack Fit can improve the result of the standard Kalman Filter, i.e. whether the Kalman Filter is suitable for initializing the Multitrack Fit.

4.3.1 Track Parameters

For the fit with the Multitrack Filter the following settings for the annealing schedule and the assignment probability χ^2 -cut were applied:

- the annealing schedule for the update of the assignment probabilities was set to (9, 4, 1, 1, 1);
- in the case of combined DAF/MTF or EAA/MTF the schedule was (9, 4, 1, 1) for EAA or DAF and (4, 1, 1, 1) for the following MTF;
- the χ^2 -cut in the calculation of the assignment probabilities was set to 9.21, which corresponds to a 1% probability cut for 2-dimensional hits.

Fig. 4.9 shows example histograms of the pulls of the five track parameters for the Multitrack Filter with Gaussian smeared hits. The pulls are unbiased, their RMS is reasonably close to 1, and the χ^2 -probability has mean value close to 0.5.

Fig. 4.10 shows the histograms of the pulls for the same Multitrack Filter, but with clusterized hits. The RMS of all pulls are significantly different from 1, and the pull of p_T^{-1} is biased. The χ^2 -probability has a mean value of 0.55.

Fig. 4.11 shows an overview of the RMS of the pulls of the transverse momentum p_T and the transverse impact parameter (ip) as well as the mean χ^2 -probability as a function of η . Four cases of fitting are compared to each other:

- the MTF with Gaussian smeared hits and initialization with knowledge of the true tracks (open upper triangles),
- the MTF with clusterized hits and initialization with knowledge of the true tracks (open squares),

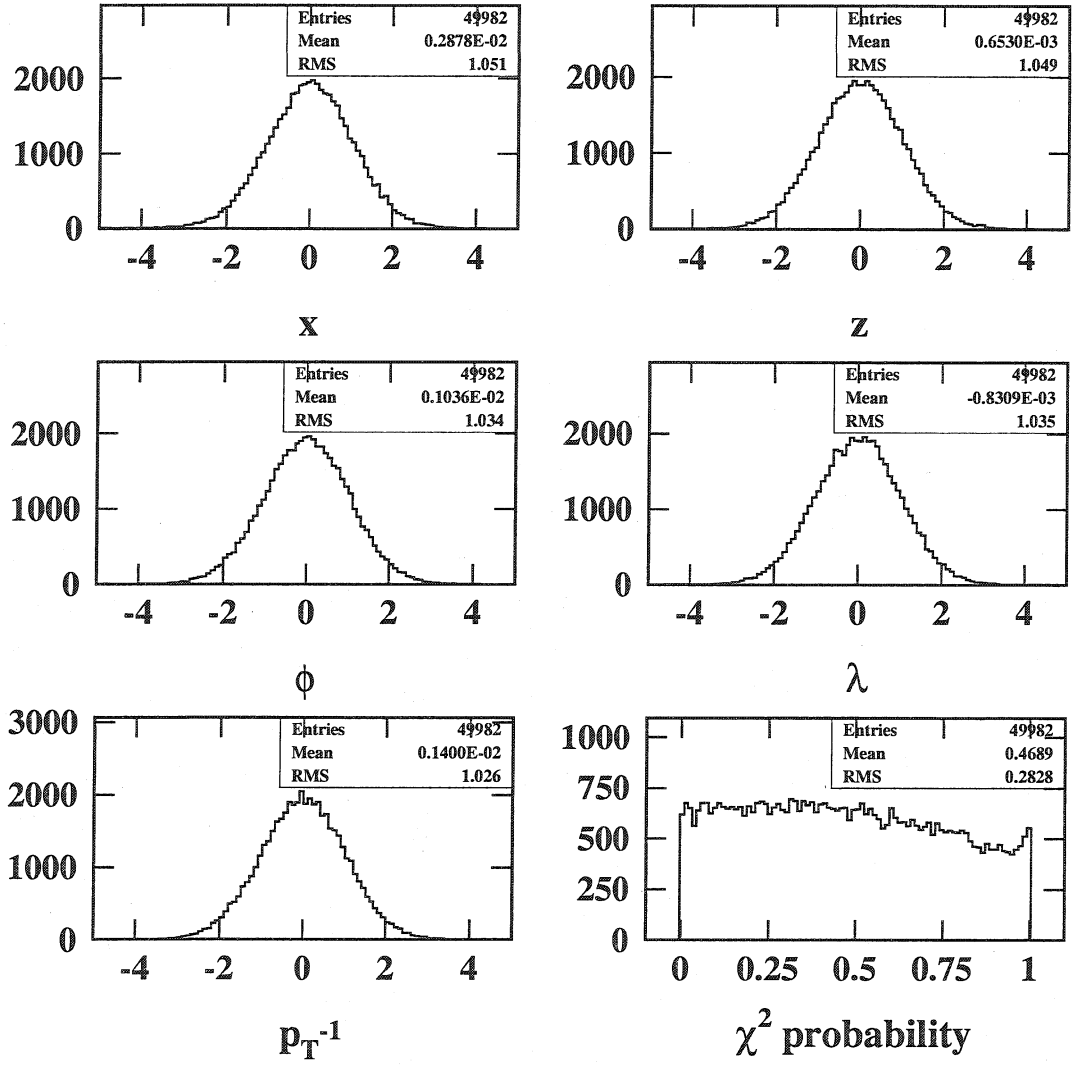


Figure 4.9: Pulls and χ^2 -probability of the Multitrack Filter with Gaussian smeared hits. The pulls are unbiased and their RMS is reasonably close to 1 for all five track parameters. The χ^2 -probability is flat and has a mean value close to 0.5.

- the Kalman Filter with clusterized hits (filled upper triangles), and
- the MTF with clusterized hits and initialization with the result from a preceding Kalman Filter (filled upside-down triangles).

The RMS of the pulls is 1 for the Multitrack Filter with Gaussian smeared hits, and its mean χ^2 -probability is close to 0.5 over the entire analysed η -range. For clusterized hits the pulls are too large in all cases. The Kalman Filter has pulls with highest RMS and lowest mean χ^2 -probability. The MTF run after the Kalman Filter can significantly improve the pulls, about 30% for the pull on the transverse impact parameter and about 10% for the pull on the inverse transverse momentum. The MTF initialized with the knowledge of the true track gives an idea what can be

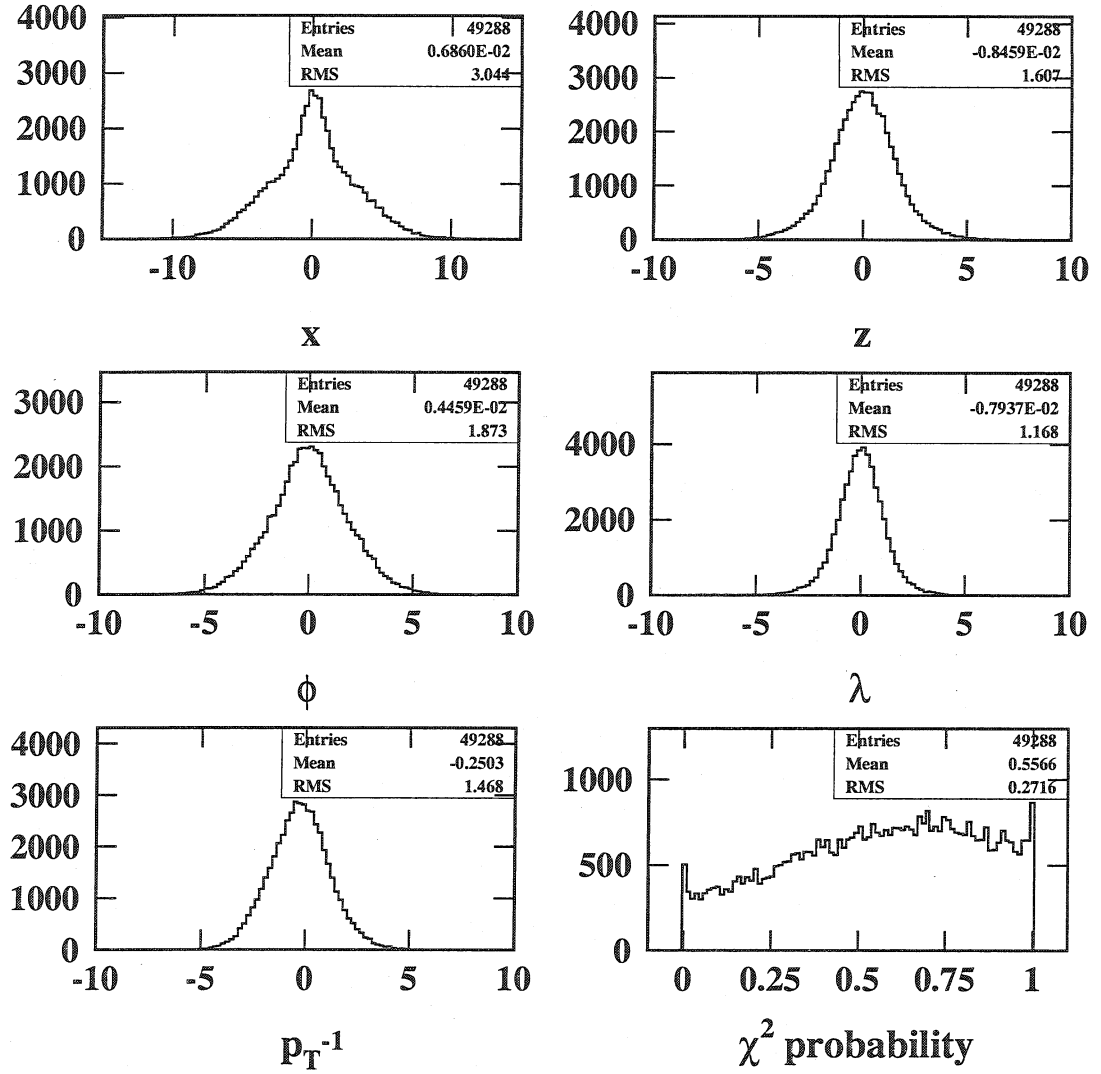


Figure 4.10: Pulls and χ^2 -probability of the Multitrack Filter with clusterized hits. The RMS is bigger than 1 for all five track parameters. The pull of p_T^{-1} is biased. The χ^2 -probability has a mean value of 0.55.

obtained in the case of an optimal pattern recognition. The RMS of the pulls are significantly larger than the ones with Gaussian smeared hits, but lower than in the other two cases.

Fig. 4.12 gives an overview on the relative resolution of the transverse impact parameter and the inverse transverse momentum of the Multitrack Filter with respect to the Kalman Filter, using clusterized hits. Two methods are compared:

- in the first case the result from the MTF with prior knowledge of the true track is compared to the result of the Kalman Filter (filled upper triangles),
- in the second case the result from the MTF run after the Kalman Filter is

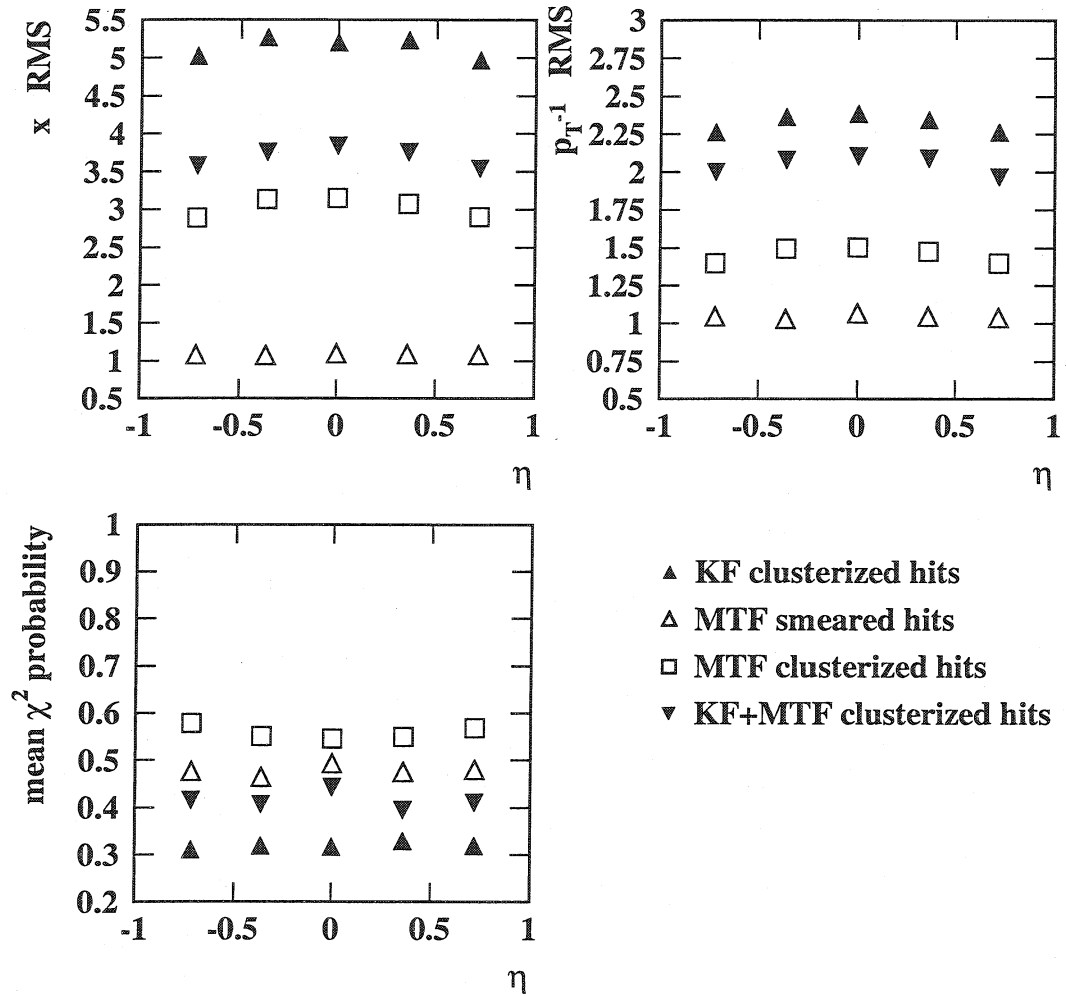


Figure 4.11: Overview of RMS of pulls and mean χ^2 -probability for different types of filters and smoothers. The Multitrack Filter has pulls with RMS of 1 in the case of Gaussian smeared hits, and pulls which are significantly larger than 1 in the case of standard clusterized hits. The Kalman Filter has pulls with largest RMS, the Multitrack smoothing improves the result of the Kalman Filter.

compared to the Kalman Filter (filled upside-down triangles).

The effects are most significant for the transverse impact parameter resolutions. The MTF run after the Kalman Filter leads to a resolution which is better by about 10%; if the MTF is initialized with the knowledge from the true track, the improvement is of the order of 20% to 25%.

The p_T^{-1} resolution cannot be improved by the MTF run after the Kalman Filter. If the MTF is initialized with the correct weights, the p_T^{-1} resolution is about 10% better than the one of the Kalman Filter.

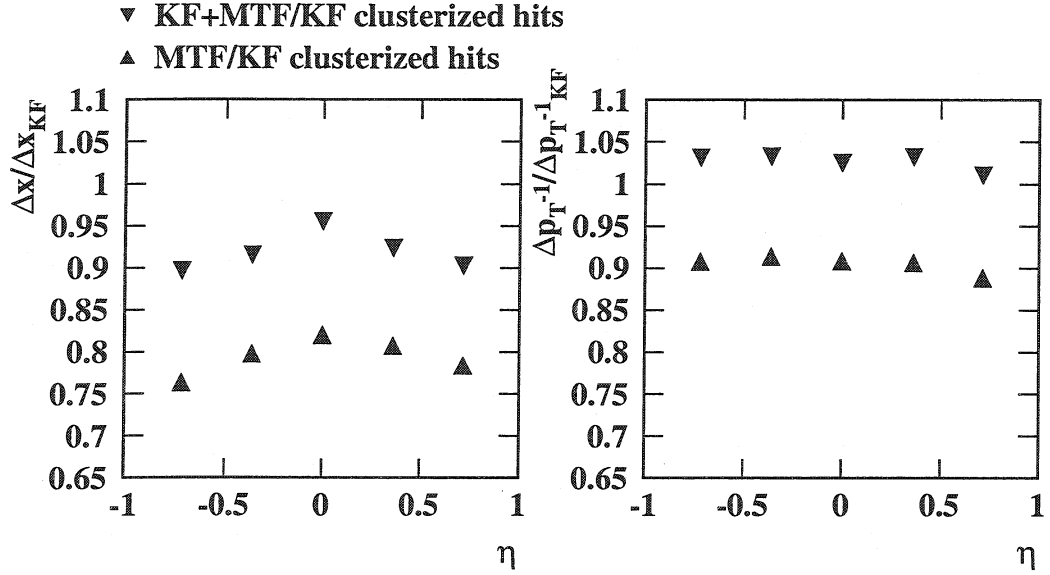


Figure 4.12: Overview of the resolutions of the MTF with respect to the Kalman Filter. If the MTF uses the result from the Kalman Filter for the initialization, it has about 10% better resolution for the transverse impact parameter. On the other hand, the resolution of p_T^{-1} cannot be improved by the MTF in that way. If the MTF is initialized with the knowledge from the simulation, the transverse impact parameter resolution is 20% to 25% better, and the p_T^{-1} resolution is 10% better than the one of the Kalman Filter.

4.3.2 Comparison of weight computation strategies

The logic for updating the assignment probabilities between hits and tracks allows for three different strategies, as explained above. The first one allows only competition between hits (DAF), the second one has competition only between tracks (Elastic Arm Algorithm), and the third method introduces global competition between hits and tracks. It is also possible to combine either of the methods, e.g. run the MTF smoother first with DAF-update and then with global update which could lead to different results if run after a Kalman Filter.

Fig. 4.13 shows an overview of the RMS of pulls and the mean values of the χ^2 -probability for the different methods if run after the Kalman Filter. It can be seen that the MTF with competition only between hits (DAF) gives the “worst” results. There is little difference between the other update strategies.

Fig. 4.14 shows an overview of the relative resolution for the transverse impact parameter and p_T^{-1} of the MTF with respect to the Kalman Filter. The transverse impact parameter can be improved by all methods of the MTF, whereas the p_T^{-1} resolution can not be improved.

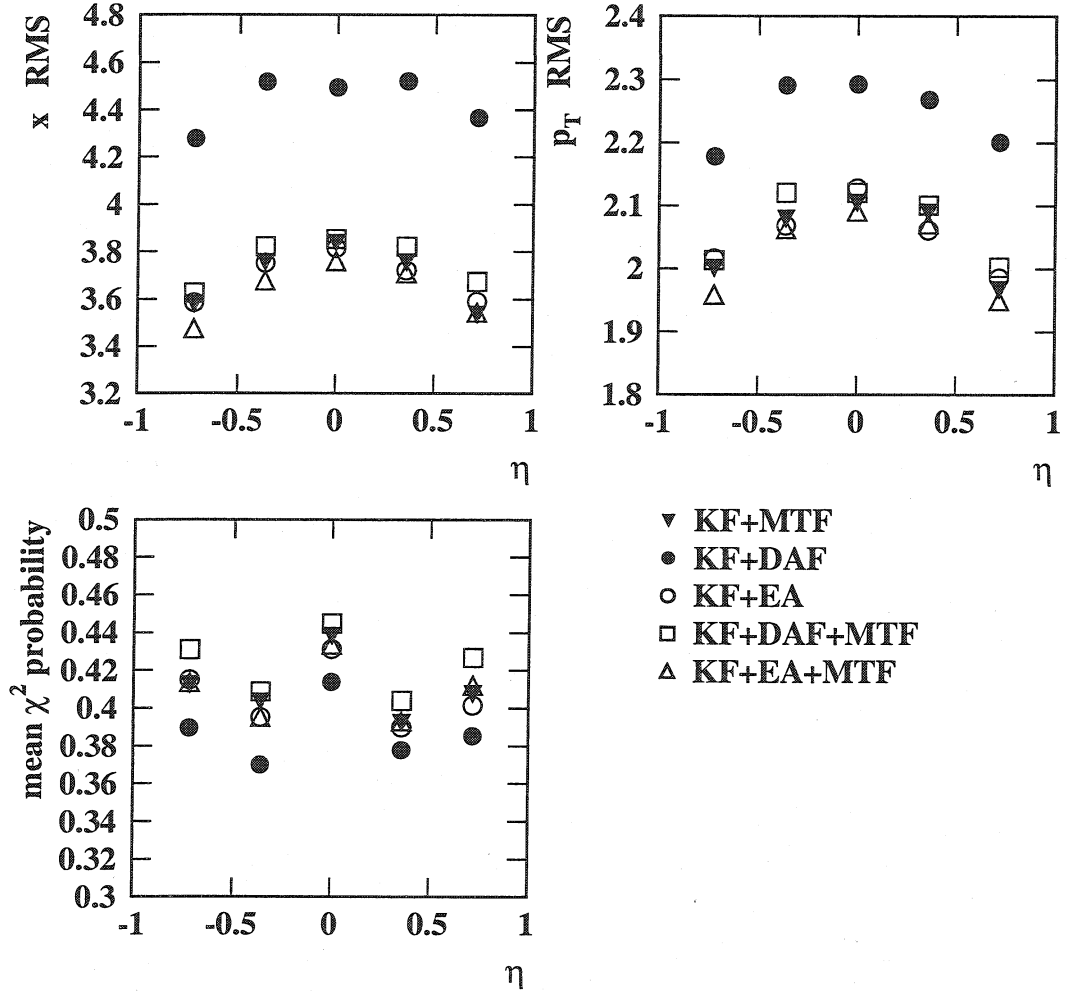


Figure 4.13: Overview of pull RMS and mean χ^2 -probability values of the MTF smoother with different update strategies of the assignment weights, always using the Kalman Filter tracks for initialization. In all cases the DAF gives the “worst” results. There is only little difference in the results between the other four ways of updating the assignment probabilities.

4.3.3 Effects of hit merging on p_T^{-1}

In order to understand better the effect of hit merging it is important to know in which layers merging of hits occurs.

Fig. 4.15 shows the average number of simulated hits (SimHits) and reconstructed hits (RecHits) per barrel layer for two tracks as a function of the radius of the barrel layer. It can be seen that the first three barrel layers, which are the pixels, have only 1–1.5 reconstructed hits per pair of tracks on average. The other layers have about two hits, except the layers with double sided detectors which have four hits. Therefore merging of hits occurs in the pixel detectors for close tracks.

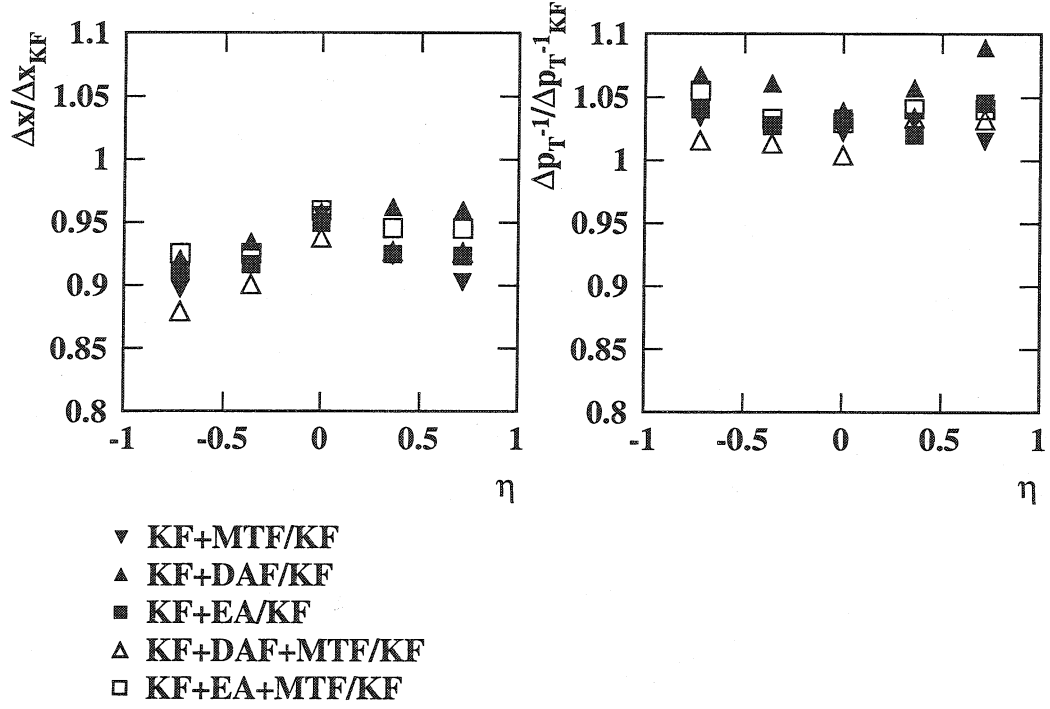


Figure 4.14: Overview of relative resolutions of the MTF smoother with respect to the Kalman Filter, with different update strategies of the assignment weights. The MTF can improve the resolution of the impact parameter. The resolution on p_T^{-1} is not improved by the MTF.

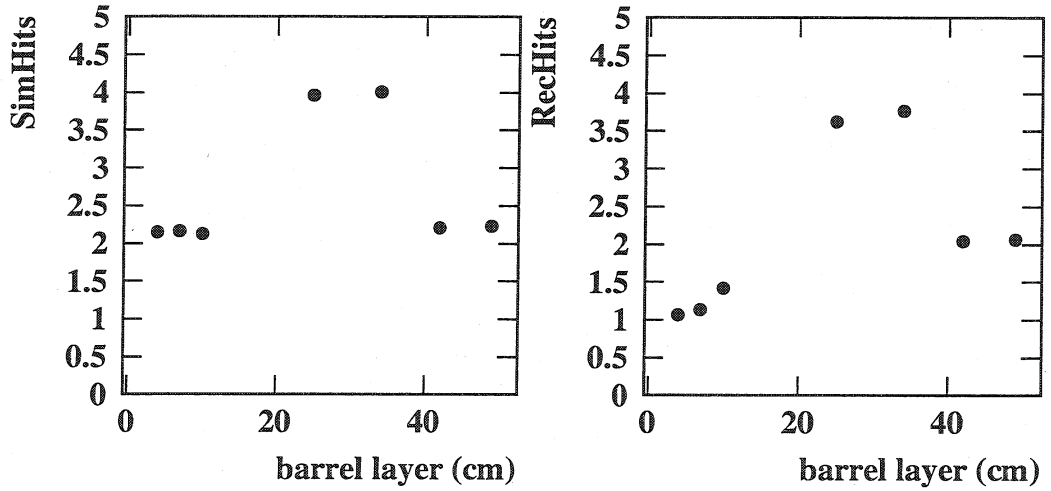


Figure 4.15: Average number of simulated hits (left) and reconstructed hits (right) per two close tracks as a function of the radius of the first barrel layers. The three innermost layers (Pixels) have about 1–1.5 RecHit per two tracks on average. In the outer layers reconstructed hits are separated. The layers with double sided detectors have four hits on average.

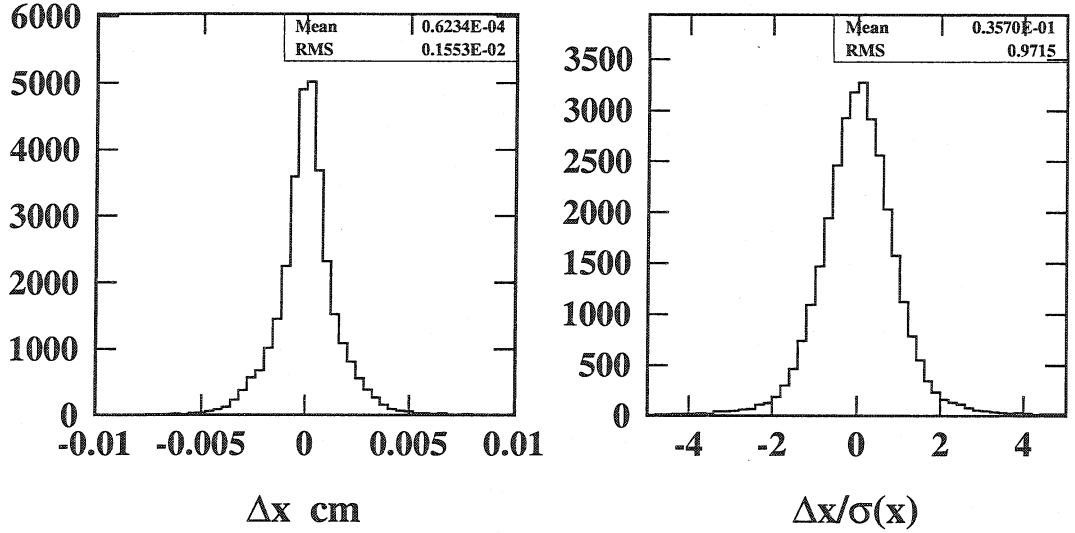


Figure 4.16: Hit resolution and pull for pixel barrel hits for single tracks. The hit resolution is about $16\mu\text{m}$ and the pull has a RMS of close to 1.

The merging of two clusters has an effect both on the position and on the error estimation of the pixel hits. Fig. 4.16 shows the hit residuals and hit pulls of the local x coordinate of the pixel barrel hits for single tracks within a clean environment. The pixel hits have a resolution of about $16\mu\text{m}$ and the RMS of the pull is close to 1.

Very close pairs of tracks, as have been used for the above analysis, produce clusters in the pixels which cannot be separated by the clusterizer. The effect of merging clusters into a single cluster and therefore also into a single resulting hit is shown in Fig. 4.17. The hit resolution is of the order of $56\mu\text{m}$, or about four times worse than the one of single clean hits. The RMS of the pull is about 4.4.

Fig. 4.18 shows the effect on p_T^{-1} of merging of two hits in one cluster. The y -axis shows the number of standard deviations of the shift of the mean value of the pull distributions, as a function of the absolute distance d in momentum space of the two tracks. Only in the case of Gaussian smeared hits the p_T^{-1} pulls have mean zero for the Multitrack Filter. The same Multitrack Filter but with clusterized hits has already a bias in its pull distribution, which is less than one standard deviation and converges to zero for increasing distance of the two tracks. The Kalman Filter has the biggest bias in its results and convergence is slow with increasing distance in momentum space. About half way between the points of the Kalman Filter and the ones of the Multitrack Filter with optimal initialization one can find the results of a combined Kalman Filter - Multitrack Fit, and the bias seems to converge to zero for increasing d .

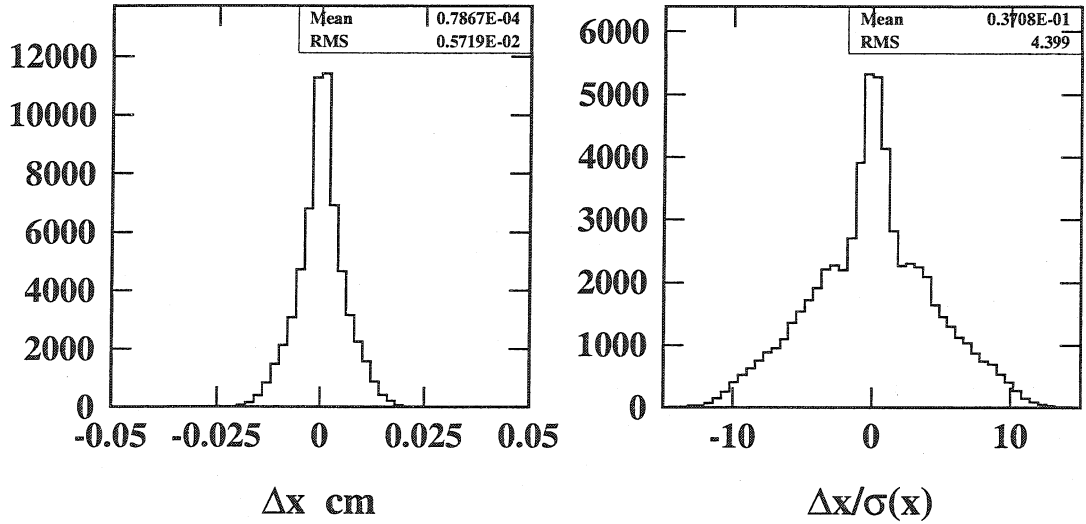


Figure 4.17: Hit resolution and pull for pixel barrel hits for very close pairs of tracks, as they have been used for the above track reconstruction. The hit resolution is of the order of $56\mu\text{m}$ and the pull has a RMS is about 4.4.

4.4 Conclusion

The MTF has been implemented in ORCA with the three update strategies according to [25]. The implementation has been verified using hits with Gaussian errors pairs of tracks within a fully controlled environment.

The results can be summarized as follows:

- The Multitrack Filter produces consistent results using Gaussian smeared hits. The pulls of all five track parameters have a RMS of 1 and the χ^2 -probability has mean 0.5. In this fully controlled environment the implemented algorithm is working correctly.
- Using clusterized hits (non-Gaussian errors), the RMS of the pulls is larger than 1 and the χ^2 -probability has a mean different from 0.5. In addition, the pulls of p_T^{-1} are biased.
- Using the tracks from the Kalman Filter as starting base for the MTF, the RMS of the pulls is smaller for all five track parameters. The shift of the p_T^{-1} -pull is smaller, too. The mean value of the χ^2 -probability distribution is higher than the one to the Kalman Filter.
- Using the tracks from the Kalman Filter as starting base for the MTF, the residuals of the transverse impact parameter can be improved by about 10%. There is no improvement for the residuals of p_T^{-1} with respect of the Kalman Filter.

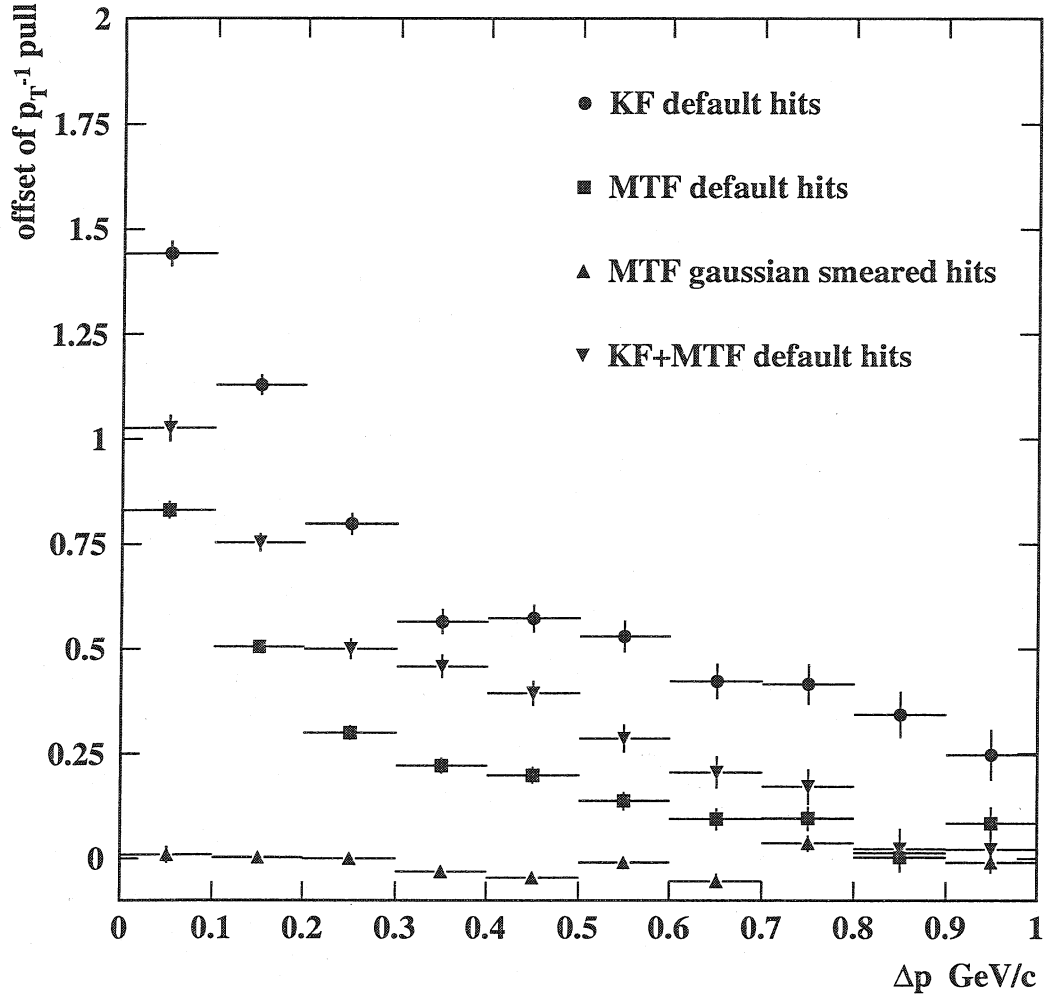


Figure 4.18: Bias of p_T^{-1} pulls in units of standard deviations as a function of the distance in momentum space of the two tracks. Only in the case of Gaussian smeared hits and optimal initialization the pulls are not biased. The shift is worst for the Kalman Filter-only tracks; this result can be improved by a combined Kalman Filter and Multitrack Fit.

- A comparison of the different update strategies shows that the DAF is worse than the MTF. However, the DAF is still better than the Kalman Filter. The difference between EAA and MTF is small. By running a combined DAF/MTF or EAA/MTF the results can not be improved substantially with respect to running the MTF only.
- Clusters of tracks merge mostly in the pixel layers, and to some extent also in the first silicon layer. Due to the increased cluster size the position and error estimation of the hits is suboptimal and most probably causes the bias in the estimation of p_T^{-1} and in the error estimation of the track parameters.

5 Performance studies of single track reconstruction

In order to quantify and qualify the performance of the DAF with respect to the KF, realistic event simulations, including material interactions, of different topologies have been chosen:

- Isolated muons with a transverse momentum of $p_T = 1, 10, 100 \text{ GeV}/c$ were used to determine quantitatively (track parameter resolutions, efficiency) and qualitatively (track parameter pulls, χ^2 -probability) the basic performance of the DAF and the KF. For the simulation of the muons the CMS 122 version was used, and ORCA_5.4.1 was used for the reconstruction of the muon tracks.
- The track reconstruction efficiency of isolated pions was determined for $p_T = 1, 10, 100 \text{ GeV}/c$. The pions were simulated in a dedicated production, using the CMS 125 layout of the CMS Tracker and ORCA_5.4.1 for the reconstruction of the pion tracks.
- The reconstruction efficiencies of tracks and secondary vertices and the b -tagging efficiencies for b -jets with transverse energies of $E_T = 50, 100, 200 \text{ GeV}$ were determined both for the DAF and for the KF and compared with each other. The CMS version of the simulation was CMS 122, and ORCA_5.4.1 was used for the reconstruction of the tracks in the b -jets.
- The mass of the $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$ decay was reconstructed both for the low luminosity ($2 \times 10^{33} \text{ cm}^{-2} \text{ s}^{-1}$) and for the high luminosity ($10^{34} \text{ cm}^{-2} \text{ s}^{-1}$) scenario at CMS, and the results were compared with previous studies [26]. This channel is particularly sensitive to the precision of the reconstructed parameters. The events were simulated in a dedicated production, using the CMS 123 version of the CMS Tracker and ORCA_5.4.1 for the reconstruction.

The results are put into a context with the ones of the CMS Tracker Technical Design Report [2] in the final conclusion.

5.1 Track selection and parameter settings

In the efficiency and fake rate analysis the following parameters for track selection were chosen:

- For the efficiency:
 - Simulated tracks were selected according to:
 - * $p_T > 0.9 \text{ GeV}/c$,

- * $|\eta| < 2.5$,
- * radial distance of simulated vertex < 3 cm,
- * longitudinal distance of simulated vertex < 30 cm.
- Reconstructed tracks were selected according to:
 - * $p_T > 0.7$ GeV/ c ,
 - * $|\eta| < 2.6$,
 - * transverse impact parameter < 120 cm,
 - * longitudinal impact parameter < 170 cm,
 - * minimum number of 8 reconstructed hits.
- For the fake rate:
 - Simulated tracks were selected according to:
 - * $p_T > 0.7$ GeV/ c ,
 - * $|\eta| < 2.6$,
 - * radial distance of simulated vertex < 300 cm,
 - * longitudinal distance of simulated vertex < 300 cm,
 - * minimum number of 8 simulated hits.
 - Reconstructed tracks were selected according to:
 - * $p_T > 0.9$ GeV/ c ,
 - * $|\eta| < 2.5$,
 - * transverse impact parameter < 3 cm,
 - * longitudinal impact parameter < 30 cm,
 - * minimum number of 8 reconstructed hits.

For the association of a reconstructed track to a simulated track at least 50% of the reconstructed hits must originate from simulated hits of the simulated track.

In the entire analysis, the DAF annealing schema was set to (81, 9, 4, 1, 1, 1). In addition to the number of reconstructed hits, the DAF has another way to qualify the track reconstruction, the number of effective hits. The number of effective hits is the sum of all assignment probabilities of the reconstructed hits in one track. This sum is equal to or smaller than the number of reconstructed hits, and it corresponds to the number of hits actually used in the track fit. For the analysis, reconstructed tracks from the DAF are required to have a minimum number of 8 effective hits.

5.2 Single muon track reconstruction performance

Fig. 5.1 shows examples of histograms of the five track parameters and the χ^2/n_{df} , for muons with $p_T = 10$ GeV/ c , using the DAF. The histograms are fitted with a mixture of two Gaussians. Parameters P1–P3 are the fitted values of the first Gaussian for

the height, the mean and the standard deviation of the distribution, P4–P6 are the corresponding values for the second Gaussian. A statistically significant shift of the mean values of the distributions for the transverse track parameters p_T^{-1} , φ and x can be observed.

Fig. 5.2 shows the corresponding pull histograms and the χ^2 -probability. The pull of p_T^{-1} has a RMS which is too large (1.86), whereas the RMS of the pulls of the other track parameters are between 0.9 and 1. The χ^2 -probability has a mean value of 0.42. The pulls of p_T^{-1} , φ and x are shifted by a significant amount.

Fig. 5.3 shows an overview of the track parameter residuals as a function of η , both for the Kalman Filter (KF) and the Deterministic Annealing Filter (DAF) and for muons with a $p_T = 1, 10, 100$ GeV/c. Both DAF and KF produce almost identical results in the case of the residuals. The mean χ^2/n_{df} is larger for the DAF and for $p_T = 1$ GeV/c tracks, and it is equal or smaller for the other tracks.

Fig. 5.4 shows the corresponding overview for the pulls of the track parameters. Except for p_T^{-1} , the RMS of the pulls are in general between 0.8 and 1.2. Concerning p_T^{-1} , a significant difference between DAF and KF can be observed for the pulls of muons with $p_T = 1$ GeV/c. For muons with $p_T = 100$ GeV/c, the pulls of p_T^{-1} are smallest but still between 1.2 and 1.5. The mean χ^2 -probability is too small for $p_T = 1$ GeV/c tracks, and it is between 0.4 and 0.6 for the other tracks.

During the analysis a systematic bias especially for the transverse track parameters p_T^{-1} , φ and x has been observed. Fig. 5.5 shows the mean values of the residuals as a function of η . The shifts in λ and z are statistically not significant. φ and x show a dependency both on η and on p_T . The bias is strongest for muons with small p_T and high η , and almost vanishes for tracks with $p_T = 100$ GeV/c. The bias of p_T^{-1} virtually depends only on η but not on p_T itself. The relative bias grows up to 1% in the very forward region, although is less dominant in the barrel region of the CMS Tracker. This means that the bias is of the same order of magnitude as the resolution itself. The inhomogenous magnetic field and a suboptimal treatment of the Tracker material are most probably the reason for these biases.

Fig. 5.6 gives an overview of the track reconstruction efficiency for single muons. The efficiency is close to 1 for the Tracker region up to $|\eta| = 2$, with a very small difference in favour of the DAF for muons with $p_T = 1$ GeV/c. A significant difference is observed for muons with $|\eta|$ larger than 2.3, where the efficiency of the Kalman Filter drops to 20%, whereas the one of the DAF stays above 55%.

5.3 Track reconstruction efficiency of single pions

Fig. 5.7 gives an overview of the track reconstruction efficiency for single pions of $p_T = 1, 10, 100$ GeV/c. The DAF has a higher reconstruction efficiency in all cases. In the case of $p_T = 1$ GeV/c pions the difference is of the order of 5%, in the case of $p_T = 10$ GeV/c pions it is 1%–2%, and for of $p_T = 100$ GeV/c pions it is 1% or below.

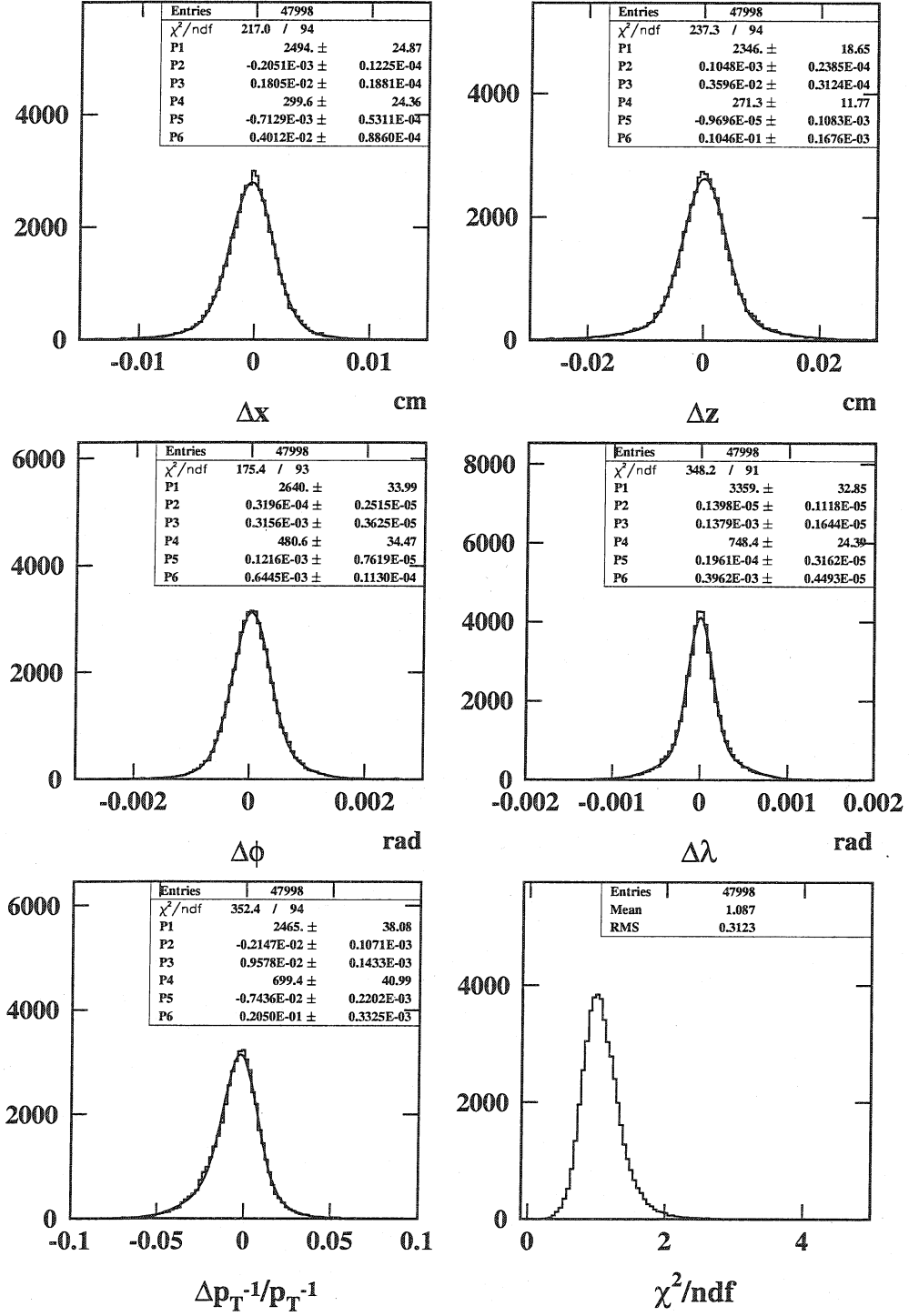


Figure 5.1: Residuals of the DAF for $p_T = 10$ GeV/c muon tracks. P1-P3 are the fitted values of the first Gaussian for the height, the mean and the standard deviation of the distribution, P4-P6 are the corresponding values for the second Gaussian.

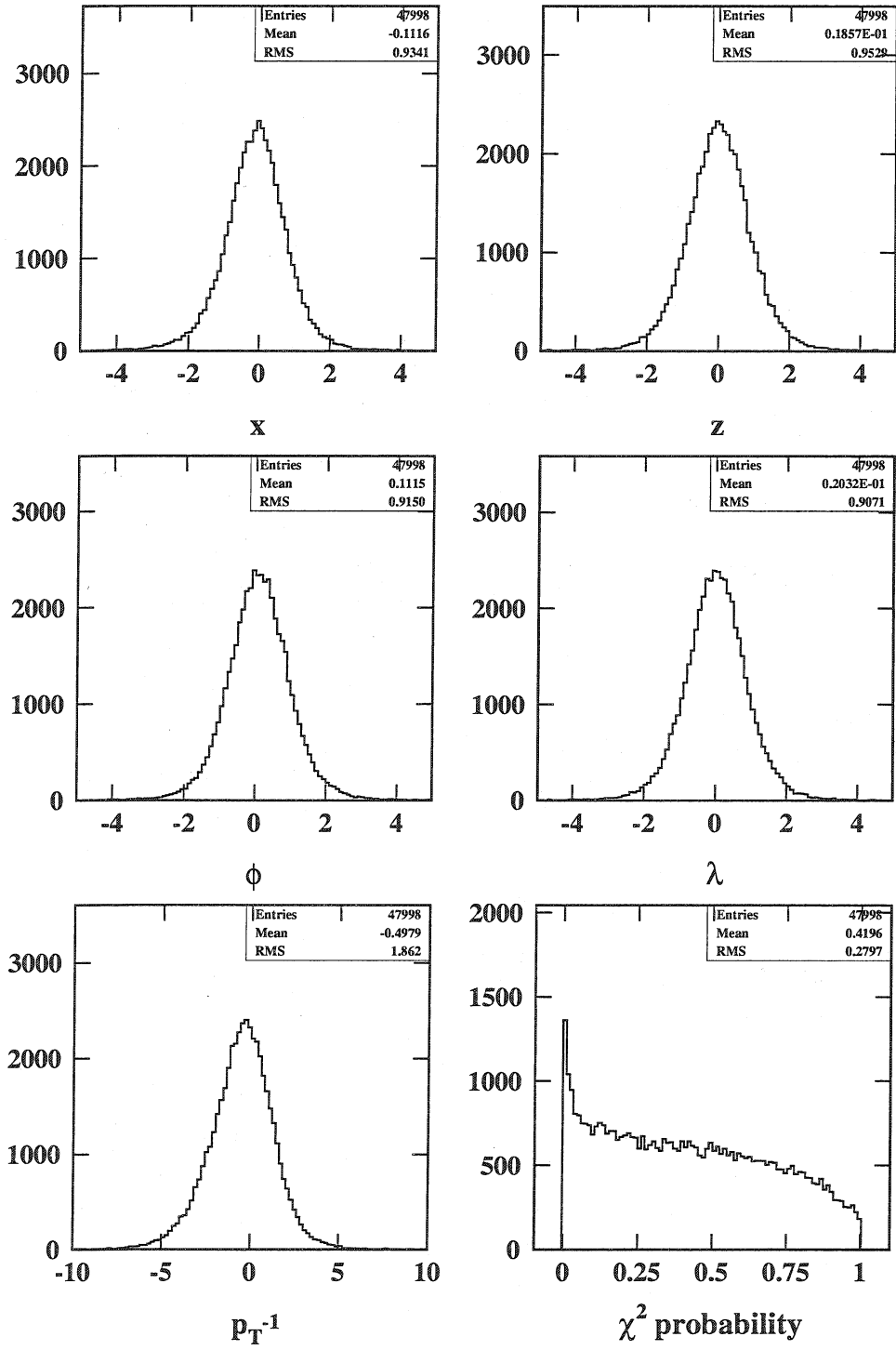


Figure 5.2: Pulls of the DAF for $p_T = 10$ GeV/ c muon tracks. The pull of p_T^{-1} has a RMS which is too large (1.86), and the pulls of p_T^{-1} , ϕ and x are shifted by a significant amount.

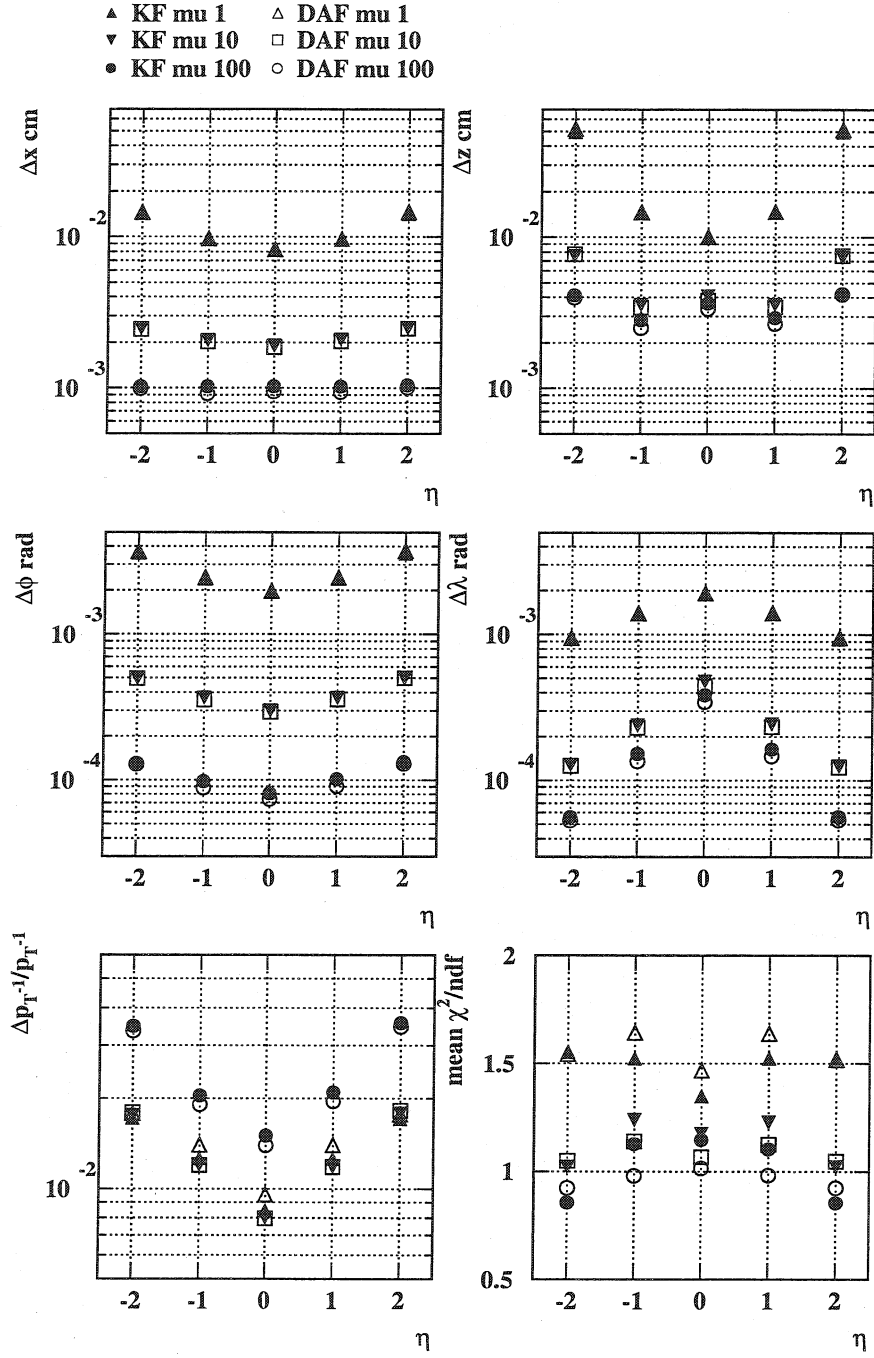


Figure 5.3: Comparison of the track parameter resolutions of single muons. No difference is observed in the resolutions of the track parameters. Concerning the average χ^2/n_{df} , the values of the DAF are closer to 1 than the ones of the Kalman Filter for the same type of muon tracks.

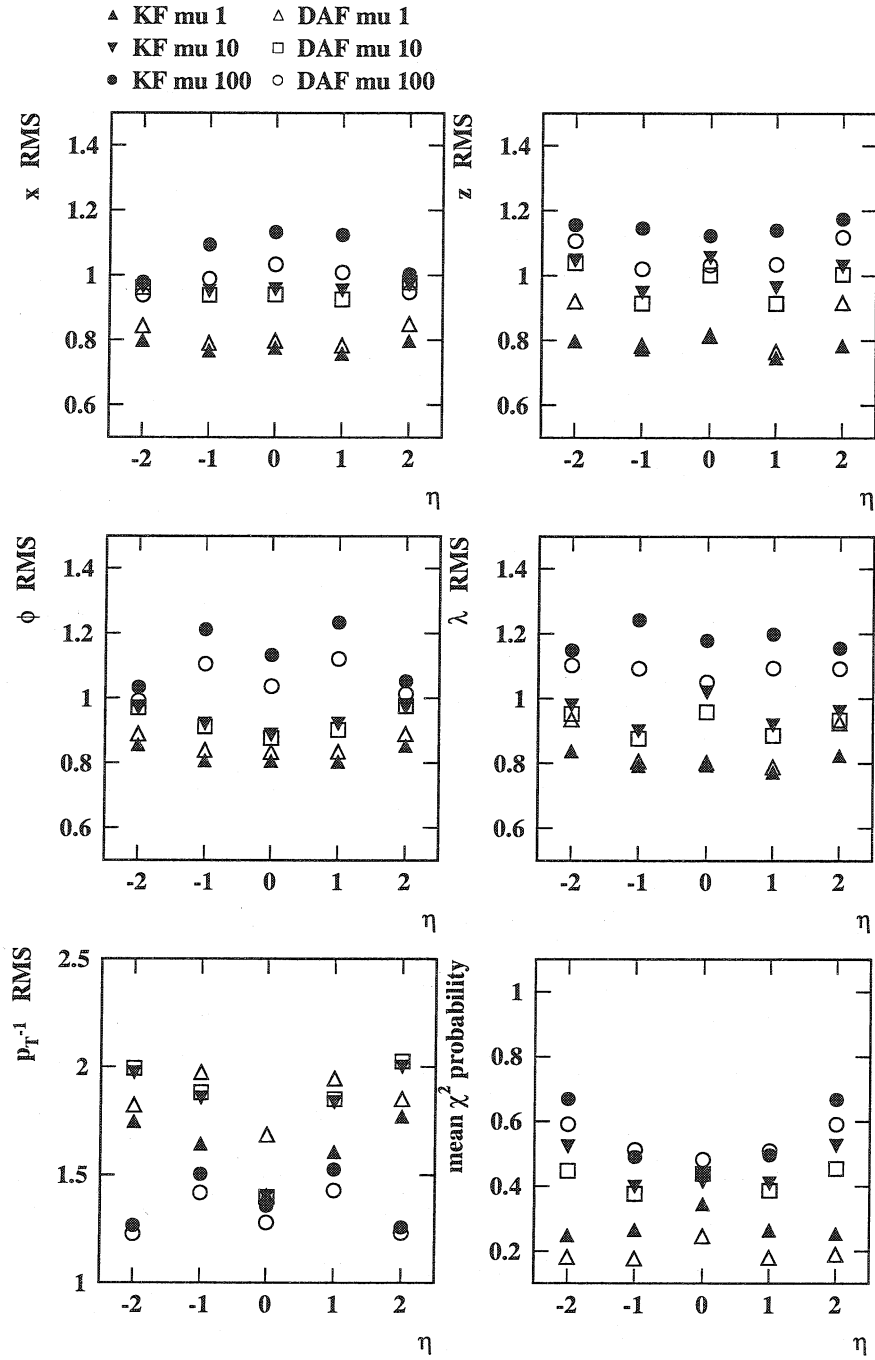


Figure 5.4: Comparison of the track parameter pulls of single muons. Except for p_T^{-1} , the RMS of the pulls are in general between 0.8 and 1.2. The average χ^2 -probability is too small for $p_T = 1 \text{ GeV}/c$ tracks, and is between 0.4 and 0.6 for the other tracks.

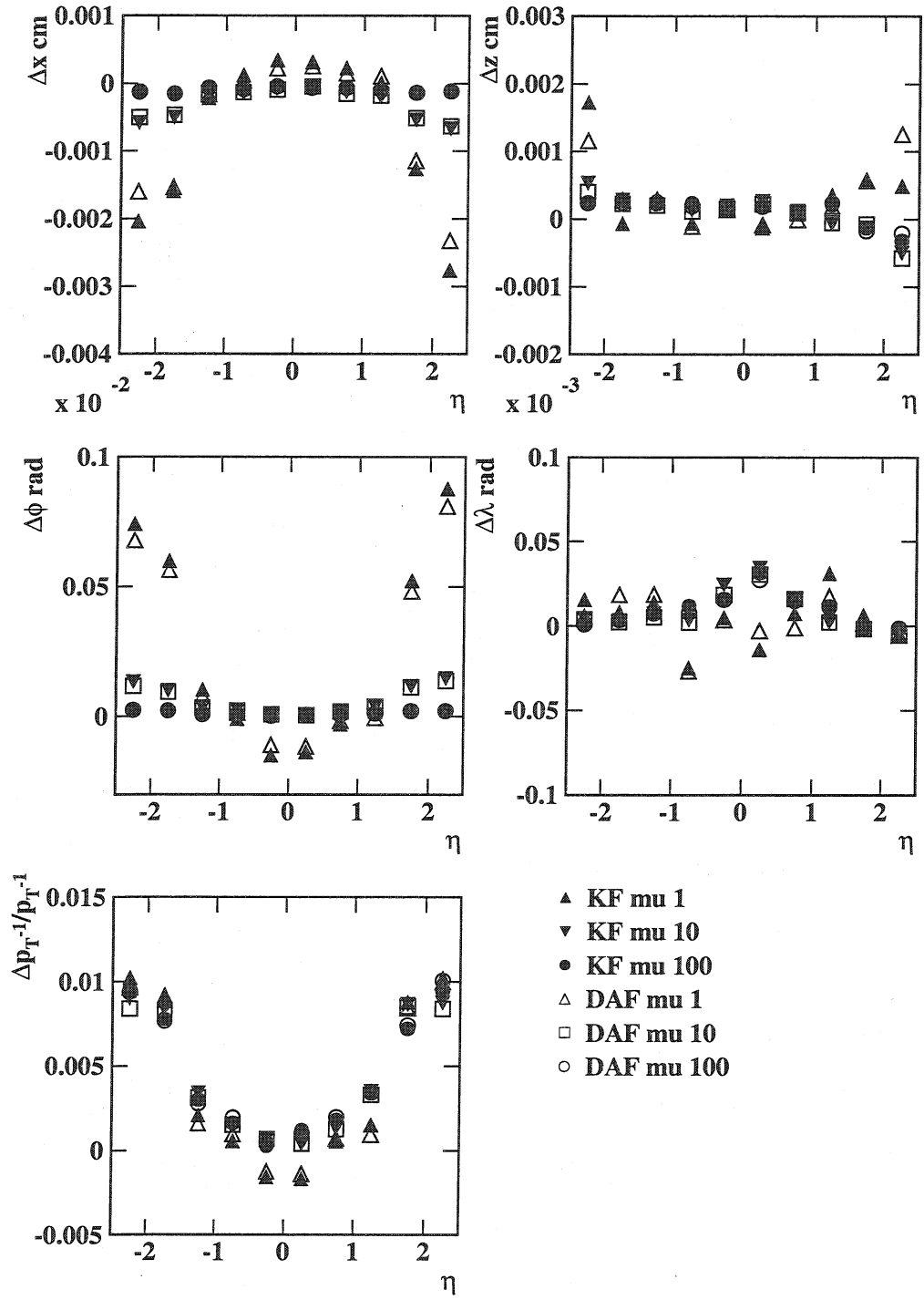


Figure 5.5: Bias of residuals for single muons. ϕ and x show a dependency on both η and p_T . The relative bias of p_T^{-1} depends only on η but not on p_T itself. The bias of λ and z is statistically not significant.

muon efficiency

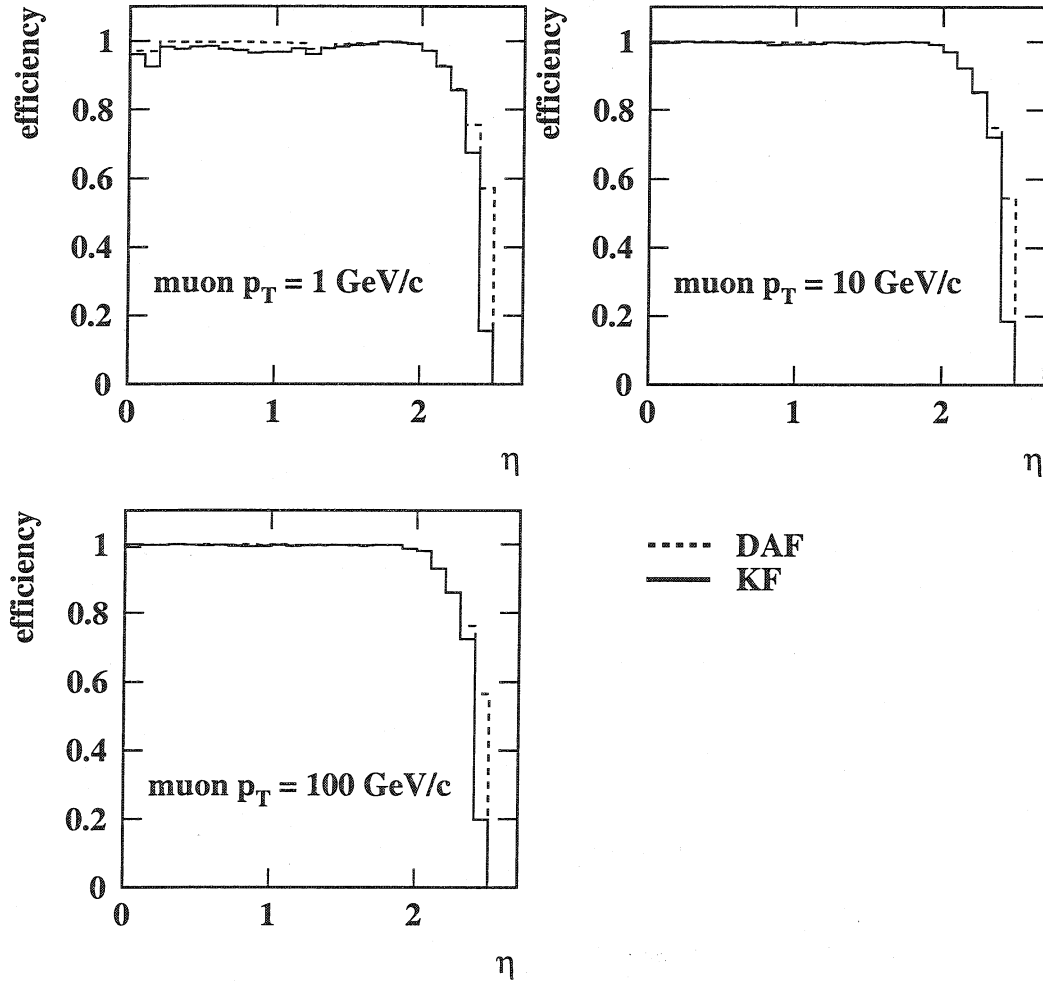


Figure 5.6: Track reconstruction efficiency of single muons. The efficiency is close to 1 for the Tracker region up to $|\eta| = 2$, with a very small difference in favour of the DAF for muons with $p_T = 1 \text{ GeV/c}$. A significant difference is observed for muons with $|\eta|$ larger than 2.3, where the efficiency of the Kalman Filter drops to 20%, whereas the one of the DAF stays above 55%.

pion efficiency

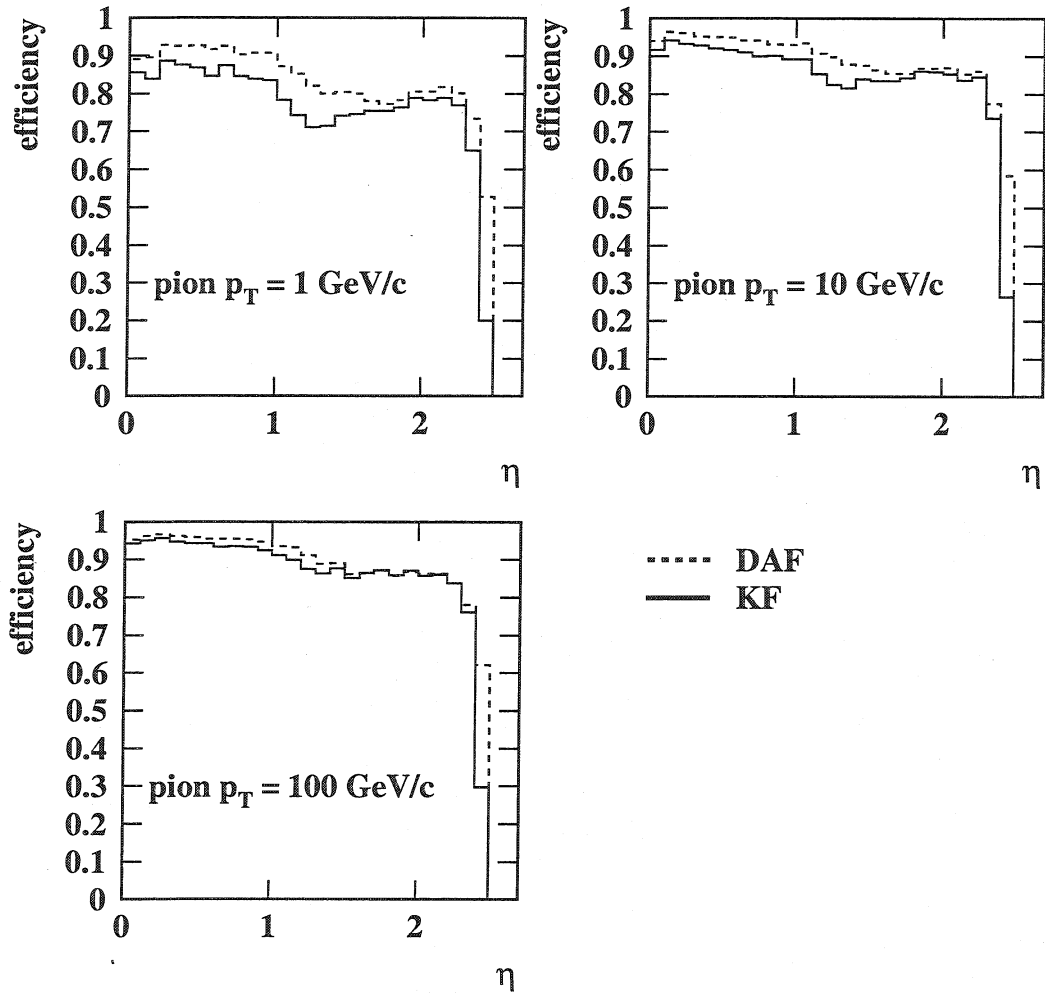


Figure 5.7: Track reconstruction efficiency of single pions. The DAF has a higher reconstruction efficiency in all cases. The difference is most dominant for low p_T pions, where it is of the order of 5% for most of the η region.

The reconstruction efficiency for the DAF and for $p_T = 1$ GeV/ c pions is 90% in the barrel, and 80% in the forward region. For $p_T = 10, 100$ GeV/ c pions it is about 95% in the barrel, and 85% in the forward.

The difference can be explained by the fact that for the same number of reconstructed hits in the track fit, the Kalman Filter traverses more layers than the DAF, as the Kalman Filter requests exactly one hit per layer. The DAF, on the contrary, treats the hits on double sided detector layers separately, and it takes into account hits in the overlap of the detector layers as well.

5.4 Impact parameters, efficiencies and fake rates in b -jets

The analysis of track reconstruction performance in b -jets consists of two parts:

- The first part compares the track reconstruction performance between DAF and KF based on:
 - The impact parameter resolutions and pulls for high momentum tracks ($p_T > 15 \text{ GeV}/c$) in b -jets with $E_T = 200 \text{ GeV}$,
 - the track finding efficiency, and
 - the track finding fake rate.
- The second part deals with the impact of track reconstruction performance on the vertex reconstruction:
 - The secondary vertex finding efficiency is analysed as well as
 - the b -tagging efficiency based on secondary vertex reconstruction in trigger level-2 jets.
 - Finally, the mistagging rate of b -jets is analysed.

For the analysis three samples of $b\bar{b}$ events with fixed transverse energy were generated, at $E_T = 50, 100, 200 \text{ GeV}$. For each E_T , jets in four different pseudo-rapidity intervals of the jet-axis were simulated: $|\eta| = 0\text{--}0.7, 1.2\text{--}1.6, 1.6\text{--}2.0, 2.0\text{--}2.4$. For the mistagging analysis $u\bar{u}$ events with the same kinematical constraints as the $b\bar{b}$ events were generated.

The histograms of the residuals and pulls were fitted with a sum of two Gaussians which have the same mean value. The fitted parameters are the mean value of both Gaussians (P1), the height of the first Gaussian (P2), the sigma of the first Gaussian (P3), the height of the second Gaussian (P4) and the sigma of the second Gaussian (P5). The variance of the mixture can thus be computed in the following way:

$$\sigma^2 = \frac{\sigma_1 P2}{\sigma_1 P2 + \sigma_2 P4} \sigma_1^2 + \frac{\sigma_2 P4}{\sigma_1 P2 + \sigma_2 P4} \sigma_2^2.$$

5.4.1 Vertex selection

The required purity above which a reconstructed vertex was associated to its simulated vertex was set to 0.55. This means that at least 55% of the tracks used in the vertex fit must have an associated simulated track from the true vertex.

Δx

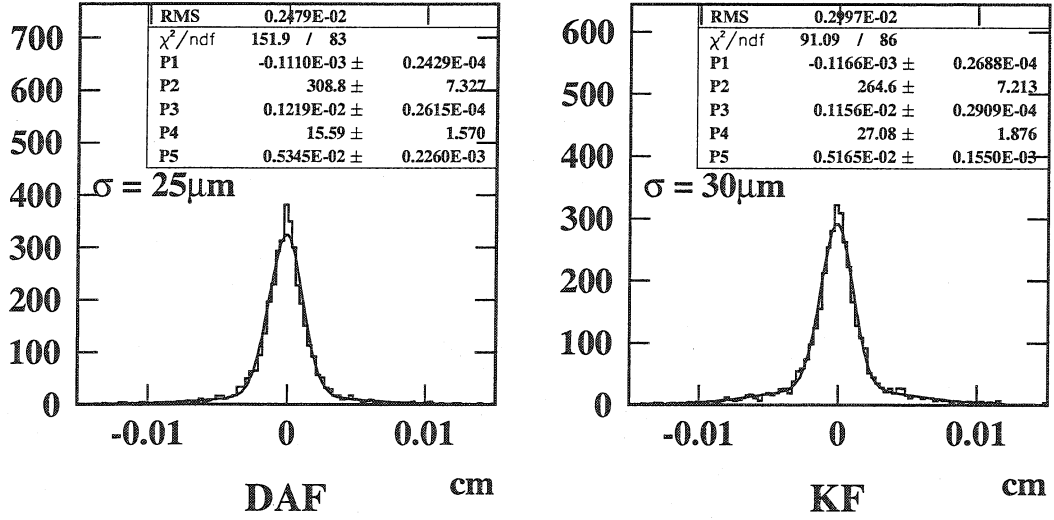


Figure 5.8: Transverse impact parameter resolution for tracks with $p_T > 15 \text{ GeV}/c$ in $E_T = 200 \text{ GeV}$, $0 < |\eta| < 0.7$ b -jets.

5.4.2 Impact parameter resolutions

Fig. 5.8 shows the transverse impact parameter residuals of KF and DAF for tracks with $p_T > 15 \text{ GeV}/c$ in $b\bar{b}$ events with $E_T = 200 \text{ GeV}$, $0 < |\eta| < 0.7$. Although the fitted standard deviations are comparable, the tails are reduced by the DAF as compared with the KF. This is an important improvement with regard to the subsequent vertex fit. The fit of the mixture gives a resolution of $26 \mu\text{m}$ for the DAF and $30 \mu\text{m}$ for the KF.

Fig. 5.9 shows the longitudinal impact parameter residuals of KF and DAF for the same tracks. Again the fitted standard deviations are comparable, and the tails are reduced by the DAF as compared to the KF. The fit of the mixture gives a resolution of $53 \mu\text{m}$ for the DAF and $65 \mu\text{m}$ for the KF.

Fig. 5.10 shows an overview of transverse and longitudinal impact parameter resolutions achieved by the KF and DAF for tracks with $p_T > 15 \text{ GeV}/c$ and in $E_T = 200 \text{ GeV}$ $b\bar{b}$ events with four different intervals of the jet axis. In all cases, the DAF has better resolution than the KF.

5.4.3 Impact parameter pulls and χ^2 -probability

Fig. 5.11 shows the transverse impact parameter pulls of KF and DAF for tracks with $p_T > 15 \text{ GeV}/c$ in $b\bar{b}$ events with $E_T = 200 \text{ GeV}$, $0 < |\eta| < 0.7$. The RMS of both pulls is too large. The RMS of the pull of the DAF is about 1.8, the one of the

Δz

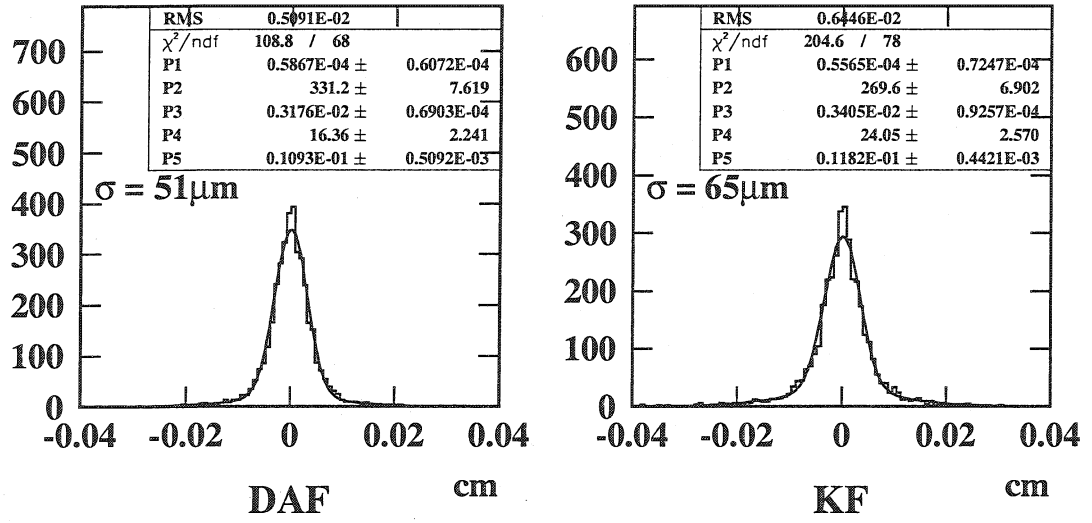


Figure 5.9: Longitudinal impact parameter resolution for tracks with $p_T > 15$ GeV/ c in $E_T = 200$ GeV, $0 < |\eta| < 0.7$ b -jets.

$E_T = 200$ GeV

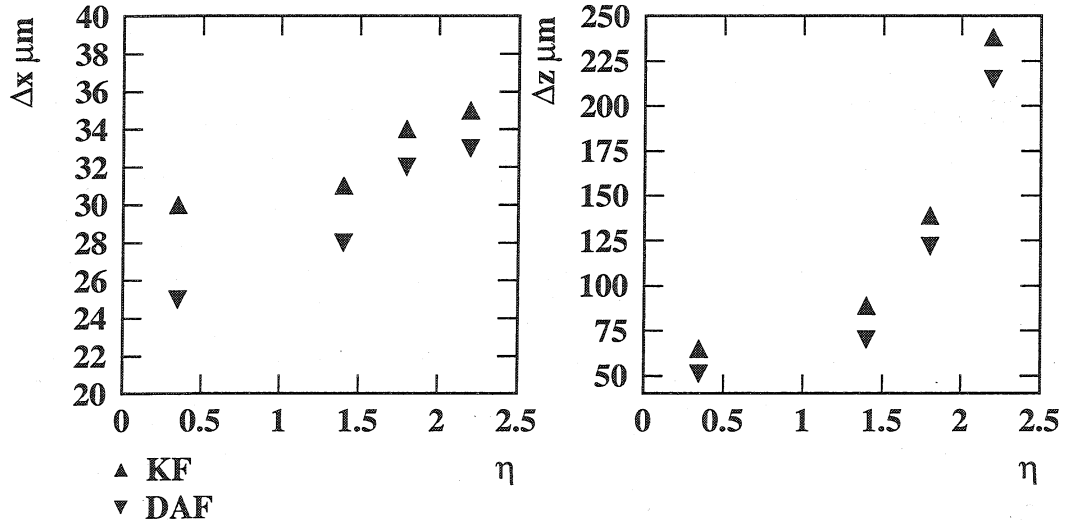


Figure 5.10: Impact parameter resolution versus η for tracks with $p_T > 15$ GeV/ c in $E_T = 200$ GeV b -jets.

$$\Delta x/\sigma(x)$$

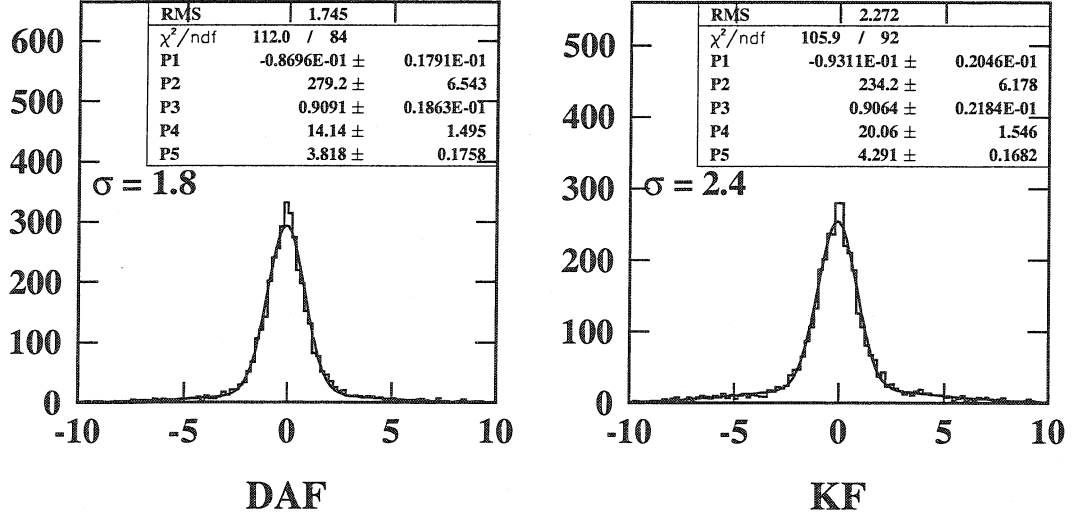


Figure 5.11: Transverse impact parameter pulls for tracks with $p_T > 15$ GeV/c in $E_T = 200$ GeV, $0 < |\eta| < 0.7$ b -jets.

KF is 2.3. The standard deviations of the mixtures are in good agreement with the RMS of the histogram.

Fig. 5.12 shows the longitudinal impact parameter pulls of KF and DAF for the same tracks. As in the case of the transverse impact parameter pulls, the RMS of both pulls is too large. The RMS of the pull of the DAF is about 1.6, the one of the KF is 2.1. The standard deviations of the mixtures are in good agreement with the RMS of the histogram.

Fig. 5.13 shows the χ^2 -probability distribution for DAF and KF for the same tracks. The distribution of the KF shows a huge peak at 0, and its mean value is too small (0.33 instead of 0.5). The distribution of the DAF is tilted towards low χ^2 -probability values, and the mean value is around 0.43. This is somewhat too small but significantly better than the one of the KF.

Fig. 5.14 shows an overview of transverse and longitudinal impact parameter pulls of the KF and DAF for tracks with $p_T > 15$ GeV/c in $E_T = 200$ GeV $b\bar{b}$ events, in four different η -intervals of the jet axis. In all cases, the RMS of the pulls is too large both for the DAF and the KF. Compared to the KF, the DAF has better (lower) RMS values and therefore tracks with better error estimation.

$$\Delta z/\sigma(z)$$

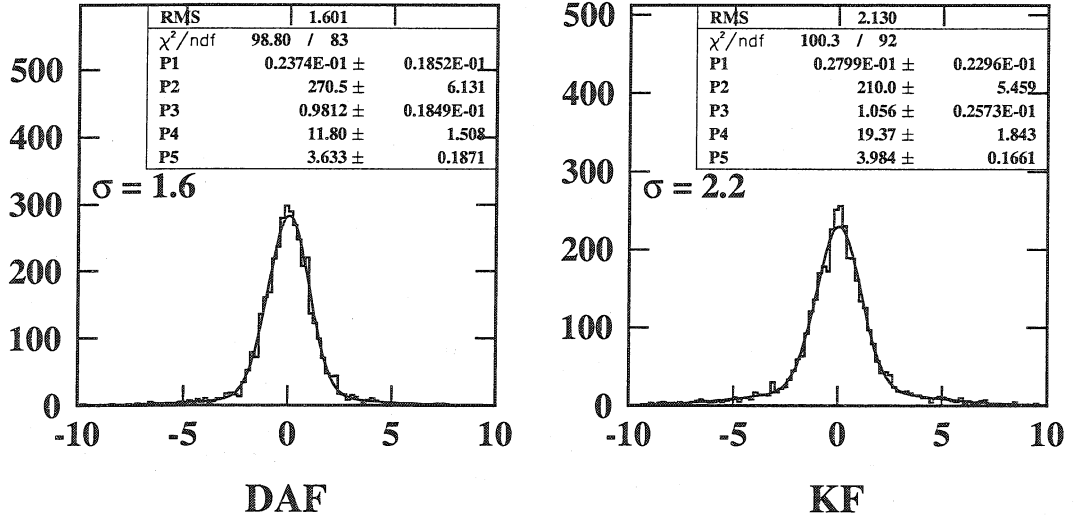


Figure 5.12: Longitudinal impact parameter pull for tracks with $p_T > 15$ GeV/c in $E_T = 200$ GeV, $0 < |\eta| < 0.7$ b -jets.

$$\chi^2 \text{ probability}$$

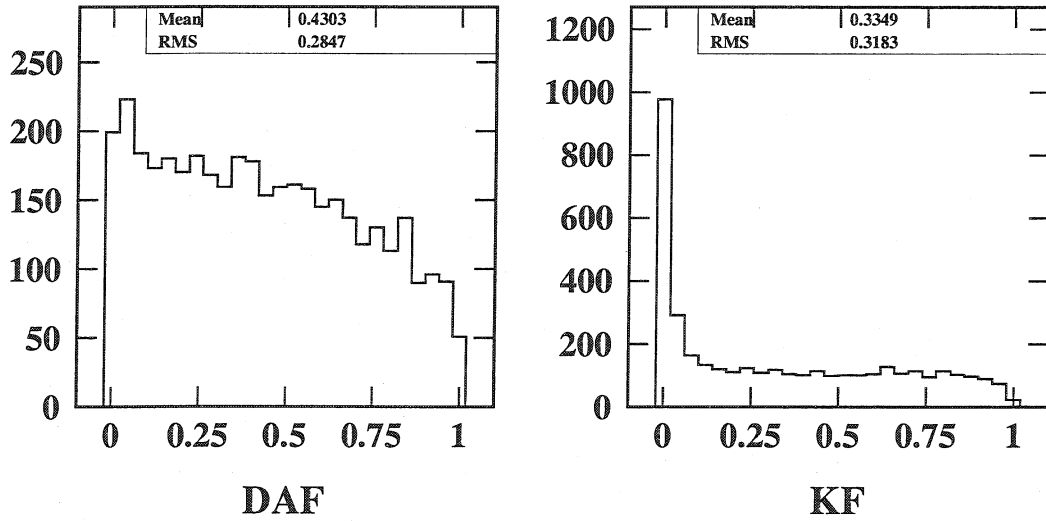


Figure 5.13: χ^2 -probability for tracks with $p_T > 15$ GeV/c in $E_T = 200$ GeV, $0 < |\eta| < 0.7$ b -jets.

$$E_T = 200 \text{ GeV}$$

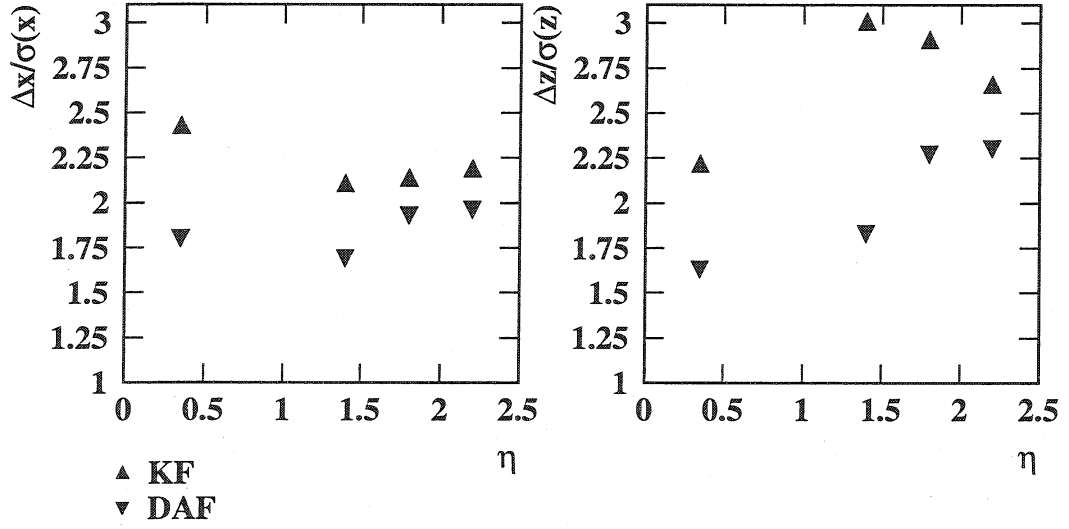


Figure 5.14: Impact parameter pulls for tracks with $p_T > 15 \text{ GeV}/c$ in $E_T = 200 \text{ GeV}$ b -jets.

5.4.4 Track reconstruction efficiency

Fig. 5.15 shows the track reconstruction efficiency in b -jets with $E_T = 50 \text{ GeV}$ for the four different intervals of the jet axis. For most of the η -region, the DAF has about 5% better efficiency. The track reconstruction efficiency for the DAF is about 90% in the barrel region of the CMS Tracker, and it is about 80%–85% in the forward region. In the very forward region the efficiency of the DAF drops to 40%–50%, whereas the one of the KF drops to 10%–20%.

Fig. 5.16 shows the track reconstruction efficiency in b -jets with $E_T = 100 \text{ GeV}$. As for b -jets with $E_T = 50 \text{ GeV}$, the DAF has a better efficiency than the KF. For most of the η region, the efficiency of the DAF is between 80% and 90%, the one of the KF between 70% and 85%.

Fig. 5.17 shows the track reconstruction efficiency in b -jets with $E_T = 200 \text{ GeV}$. As in the previous cases, the DAF is slightly better by about 5% with respect to the KF. In general, the efficiency for the DAF is between 80% and 95% for most of the η region, and it is between 70% and 90% for the KF.

The global track reconstruction efficiency for the three values of E_T is shown in Fig. 5.18. It shows a higher track reconstruction efficiency for the DAF by about 2%–5% with respect to the KF. It is between 82% and 92% for the DAF and between 75% and 90% for the KF for $|\eta| < 2$.

$b\bar{b} E_T = 50\text{GeV}$

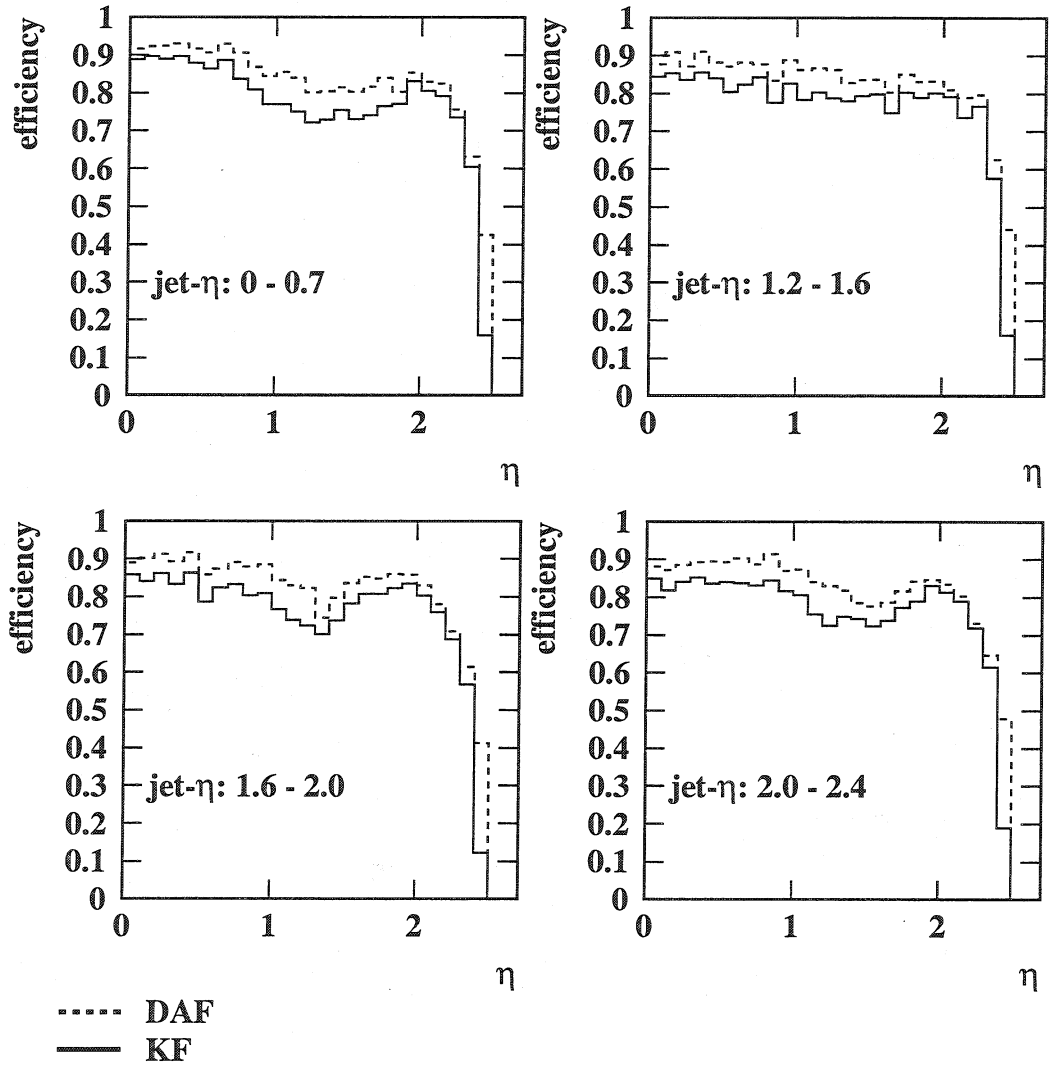


Figure 5.15: Track reconstruction efficiency in $E_T = 50\text{ GeV}$ b -jets. For most of the η -region, the DAF has about 5% better efficiency than the KF.

5.4.5 Track reconstruction fake rate

Fig. 5.19 shows the fake rate in b -jets with $E_T = 50\text{ GeV}$ for the four different intervals of the jet axis. The fake rates of DAF and KF are comparable. It is below 1% in the barrel and between 1% and 2.5% in the forward.

Fig. 5.20 shows the fake rate in b -jets with $E_T = 100\text{ GeV}$. The fake rates of DAF and KF are comparable for jets in the barrel (below 1%). For the jets in the forward region, the fake rates are higher for the KF. They are between 3% and 6% for the KF and between 2% and 3% for the DAF.

$b\bar{b} E_T = 100\text{GeV}$

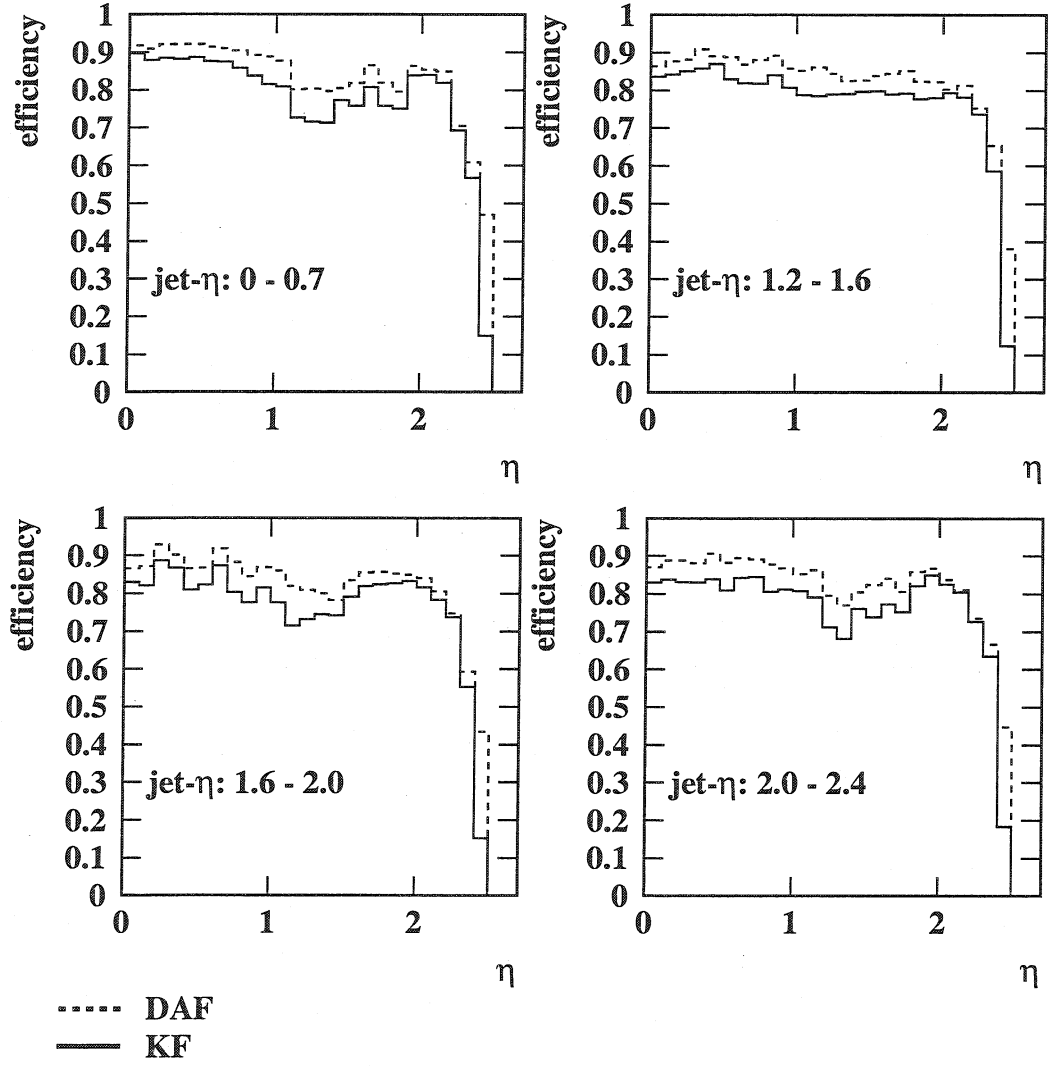


Figure 5.16: Track reconstruction efficiency in $E_T = 100\text{ GeV}$ b -jets. For most of the η region, the efficiency of the DAF is between 80% and 90%, the one of the KF between 70% and 85%

$b\bar{b} E_T = 200\text{GeV}$

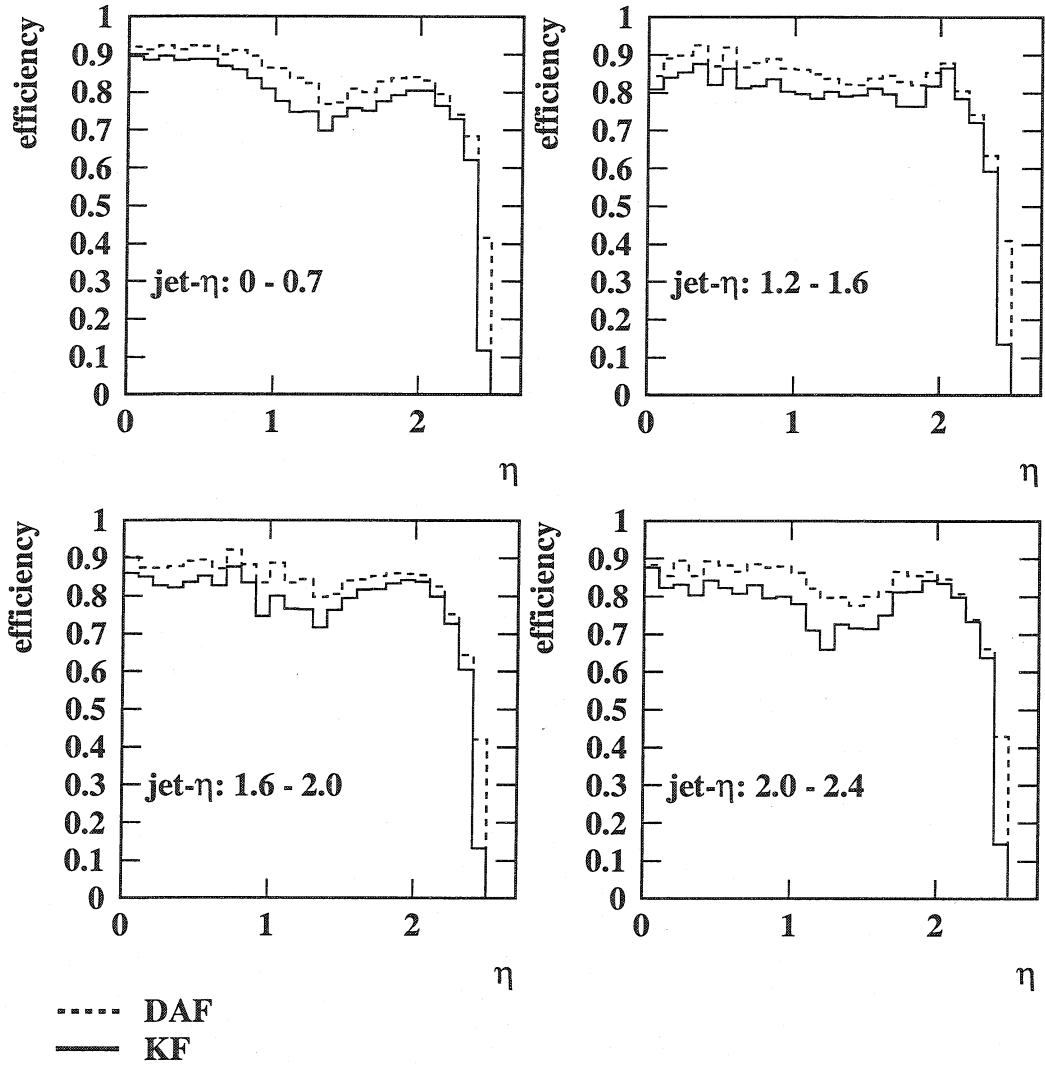


Figure 5.17: Track reconstruction efficiency in $E_T = 200\text{ GeV}$ b -jets. In general, the efficiency is between 70% and 90%.

Fig. 5.21 shows the fake rate in b -jets with $E_T = 200\text{ GeV}$. Whereas the fake rates of DAF and KF are still comparable for barrel jets (below 2°), there is a significant difference in the forward jets. The fake rates of the KF are between 10% and 17%, whereas the ones of the DAF stay between 3% and 8%.

Fig. 5.22 shows the overall fake rate for the three values of E_T . The results are comparable for $E_T = 50\text{ GeV}$, otherwise the KF has significantly higher fake rates than the DAF. The fake rates are much higher in the forward than in the barrel.

In addition to the number of reconstructed hits (RecHits), the DAF has another way to qualify the track reconstruction: the number of effective hits, i.e. the sum of all

efficiency in b-jets

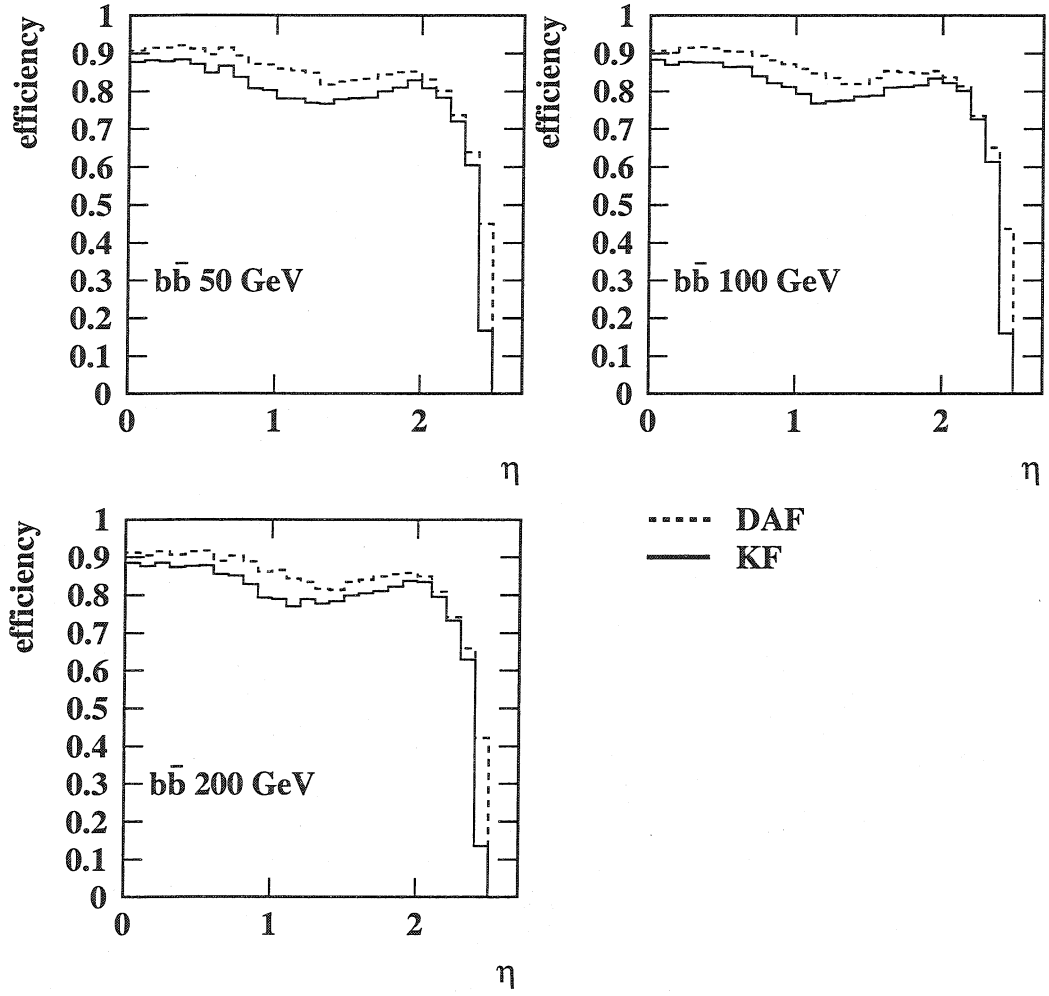


Figure 5.18: Overall track reconstruction efficiency in b -jets for DAF and KF. The track reconstruction efficiency for the DAF is 2%–5% higher than the one of the KF. In general, the track reconstruction efficiency for the DAF in b -jets is between 85% and 95%.

assignment probabilities of the reconstructed hits used in the track. This sum is equal to or smaller than the number of RecHits, and it reflects the number of hits actually used in the track fit.

Fig. 5.23 shows the normalized integral of fake tracks with a certain number of effective hits as a function of the number of effective hits. For $E_T = 50$ GeV jets about 35% of the fake tracks have up to 9 effective hits. A cut a 9 effective hits would lower the fake rate accordingly. For $E_T = 100$ GeV and 200 GeV jets, about 40%–50% of the fake tracks could be sorted out by this. On the other hand, about 50% of the fake tracks of the DAF have more than 10 effective hits. These fake tracks have a priori a good reconstruction quality.

$b\bar{b} E_T = 50\text{GeV}$

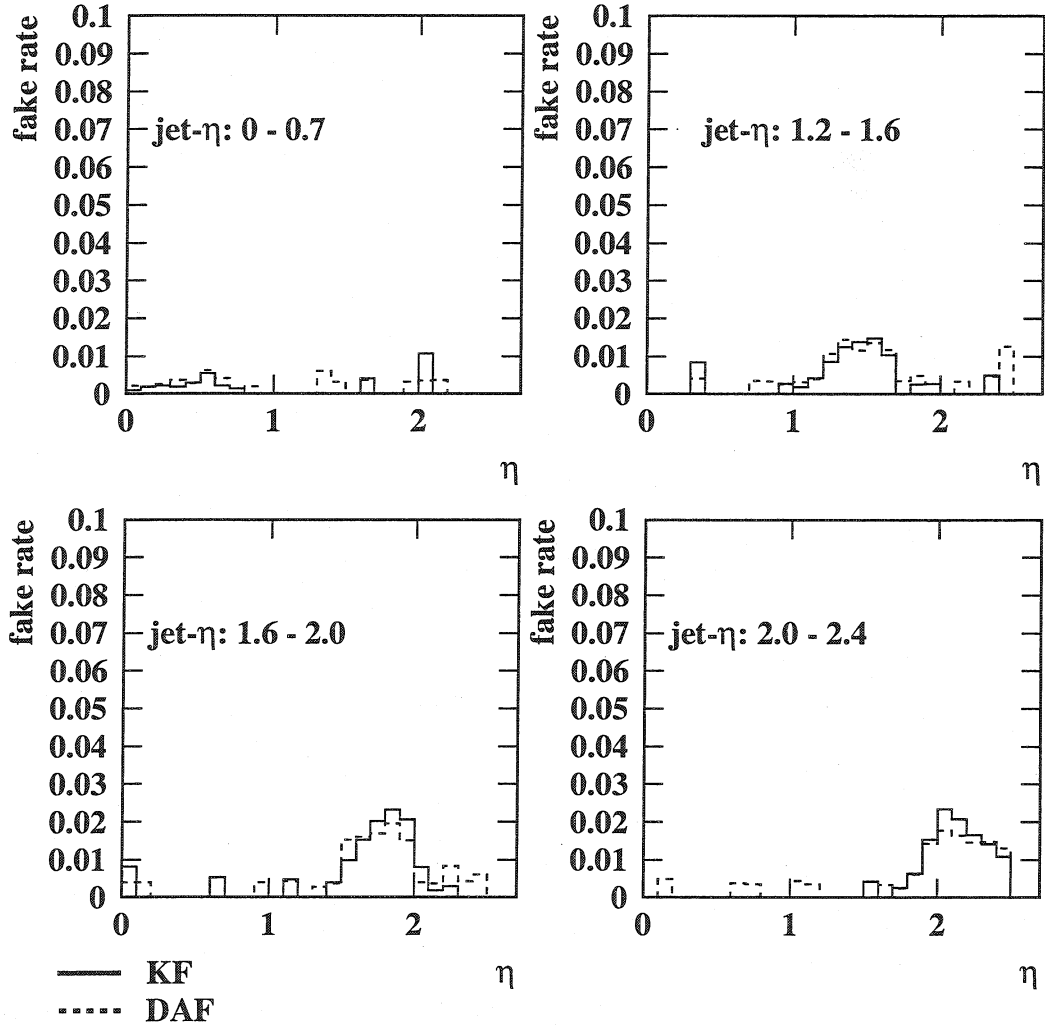


Figure 5.19: Track reconstruction fake rate in $E_T = 50\text{GeV}$ b -jets. The fake rates of DAF and KF are comparable. They are below 1% in the barrel and between 1% and 2.5% in the forward.

Fig. 5.24 shows the χ^2 -probability distributions for the same fake tracks. The peak at 0 is expected, as fake tracks are supposed to have a very low χ^2 -probability. On the other hand, all three distributions show a large and almost flat tail up to a χ^2 -probability of 1, which might indicate an unwanted feature in the selection of simulated tracks, which are missing in the analysis of the fake rate.

$b\bar{b} E_T = 100\text{GeV}$

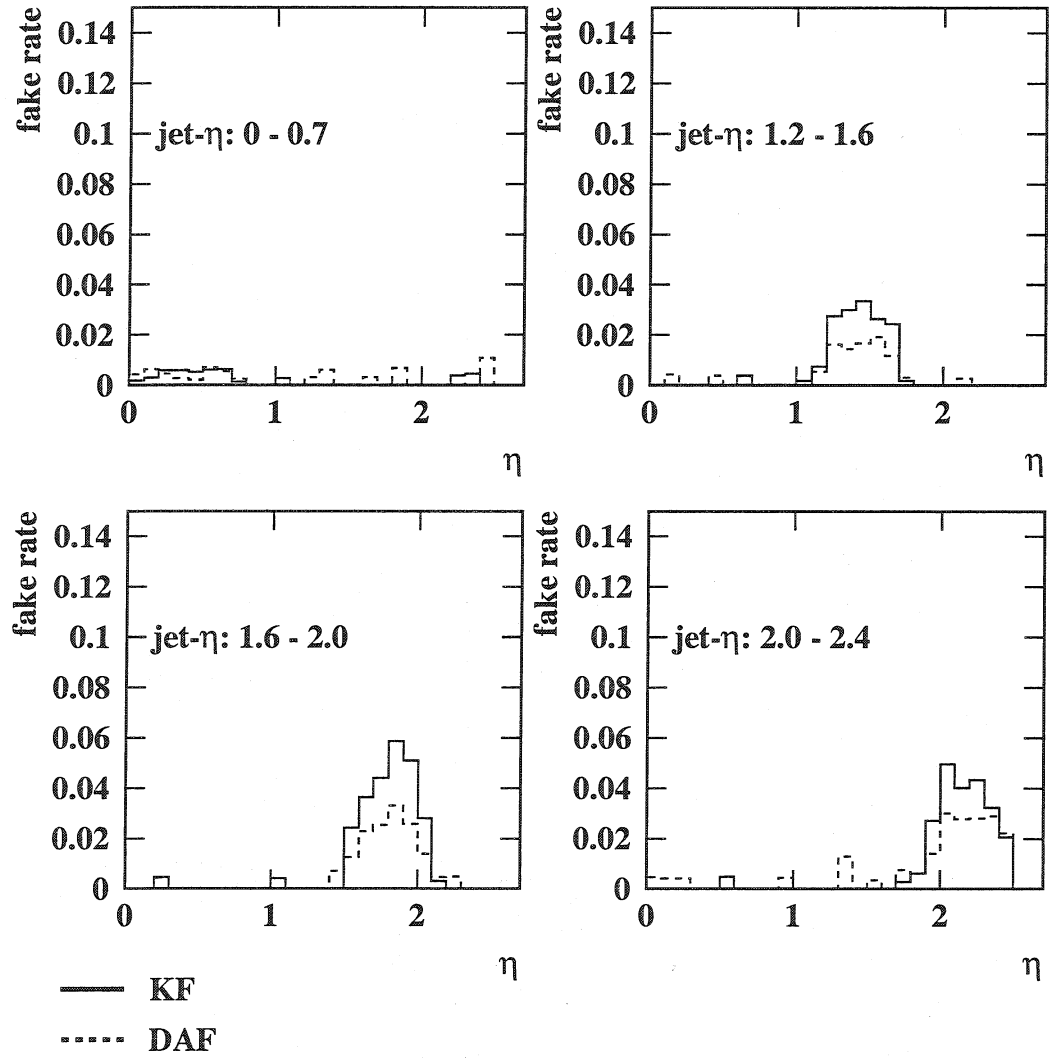


Figure 5.20: Track reconstruction fake rate in $E_T = 100\text{ GeV}$ b -jets. The fake rates of DAF and KF are comparable for jets in the barrel (below 1%). For the jets in the forward region, the fake rates are higher for the KF. They are between 3% and 6% for the KF and between 2% and 3% for the DAF.

$b\bar{b} E_T = 200\text{GeV}$

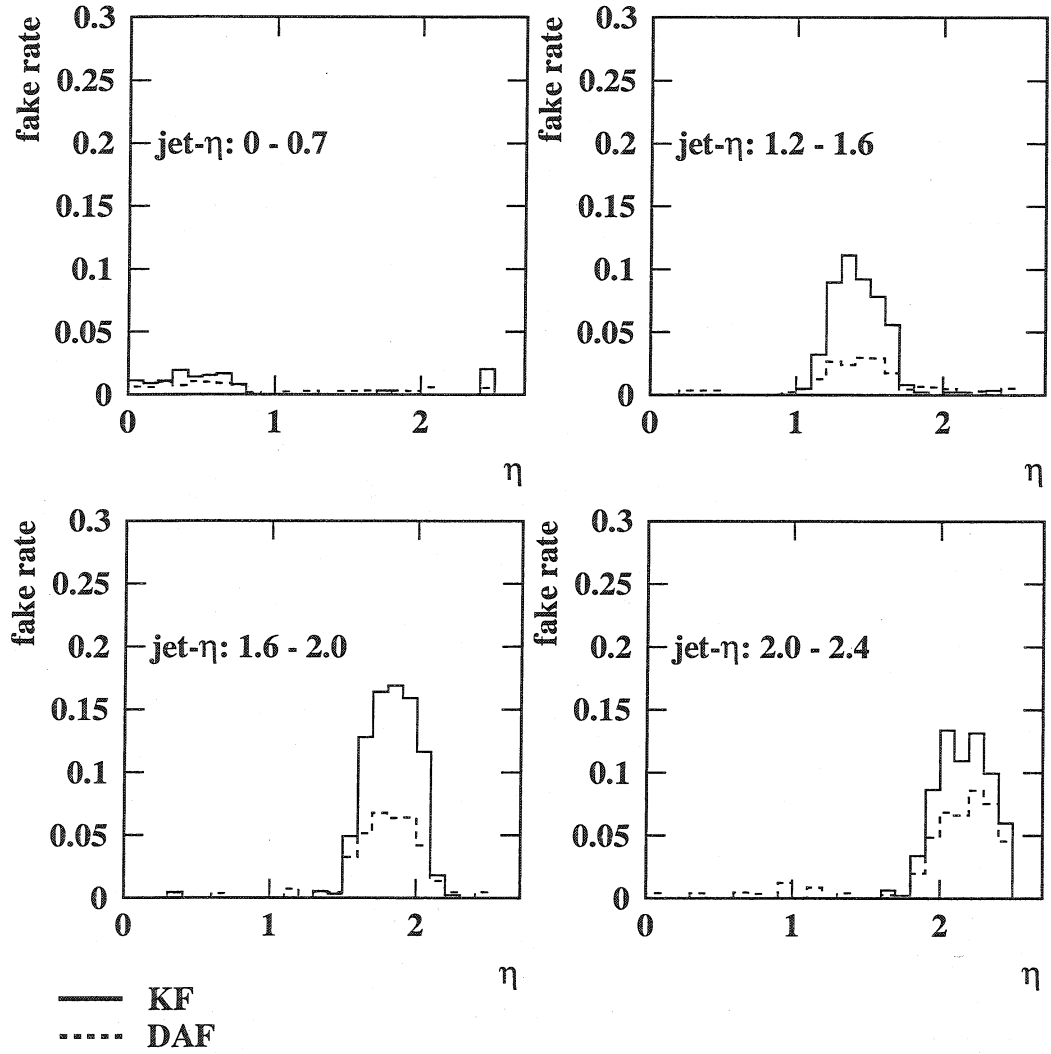


Figure 5.21: Track reconstruction fake rate in $E_T = 200\text{ GeV}$ b -jets. The fake rates of DAF and KF are still comparable for barrel jets (below 2%), but there is a significant difference in the forward jets. The fake rates of the KF are between 10% and 17%, whereas the ones of the DAF stay between 3% and 8%.

fake rate in b-jets

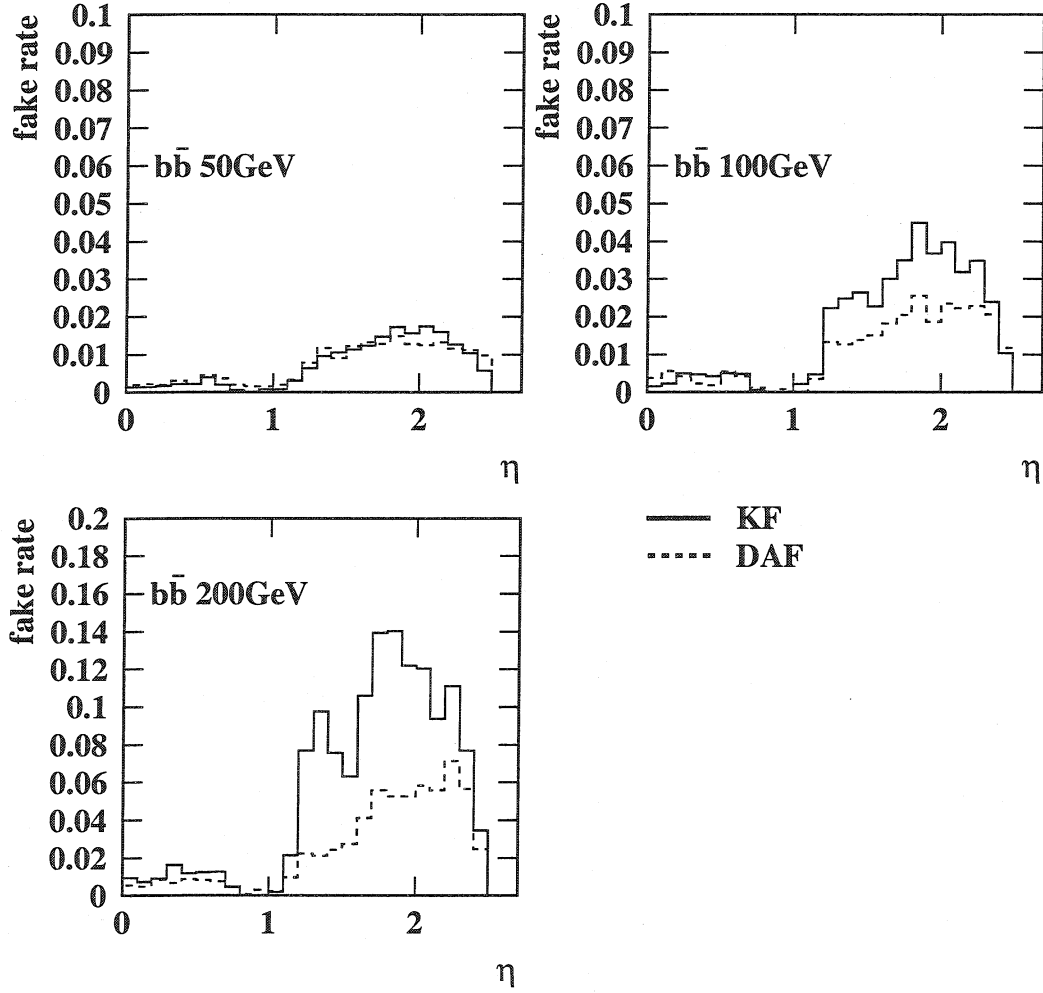


Figure 5.22: Overall track reconstruction fake rate in b -jets, combining the results of the four different jet- η bins. The fake rates are comparable for $E_T = 50$ GeV. In the two other cases the KF has significantly higher fake rates than the DAF. In general, the fake rates are much higher in the forward than in the barrel.

DAF fake track effective hits

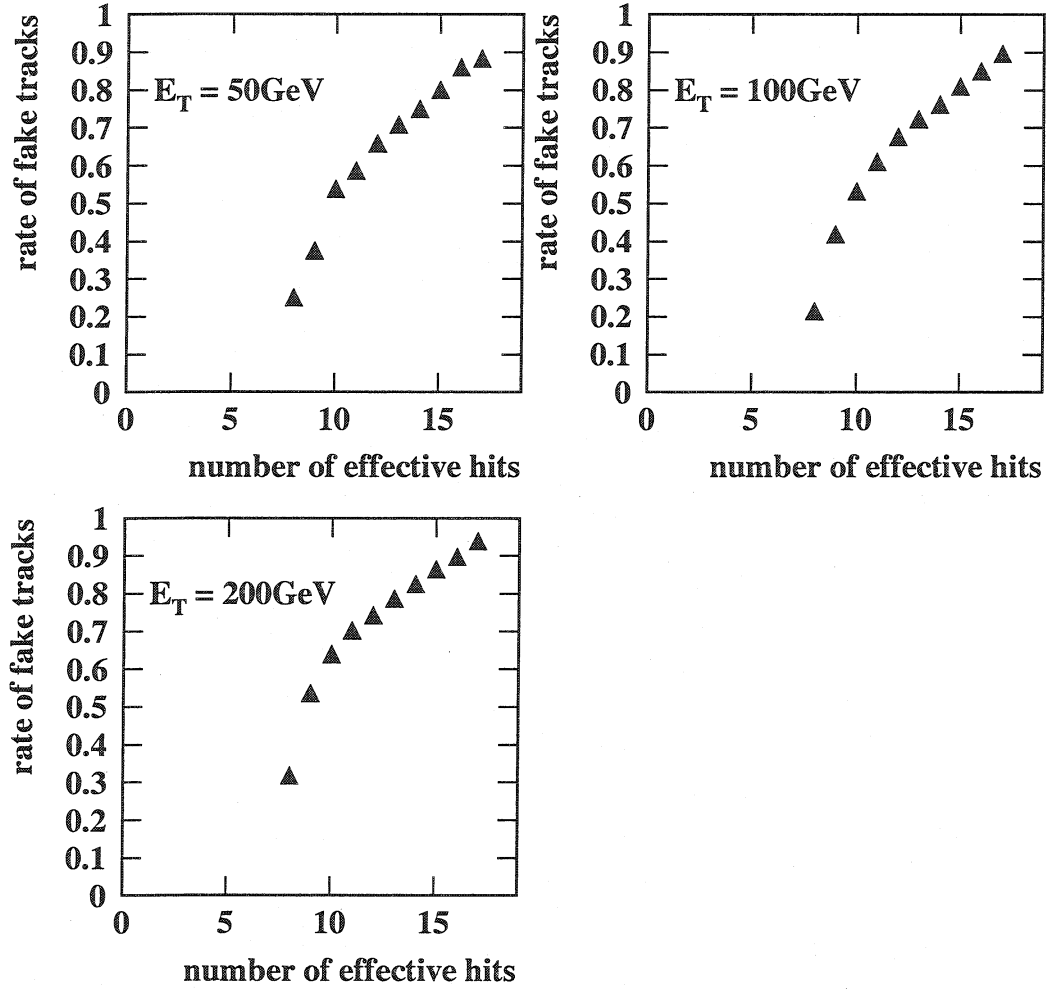


Figure 5.23: Integral of fake tracks as a function of the number of effective hits for the DAF. For $E_T = 50 \text{ GeV}$ jets about 35% of the fake tracks have up to 9 effective hits, a cut at 9 effective hits would reduce the fake rate of the DAF accordingly. For $E_T = 100 \text{ GeV}$ and 200 GeV jets, about 40%–50% of the fake tracks could be sorted out by this. On the other hand, about 40%–50% of the fake tracks of the DAF have more than 10 effective hits, which indicates a good reconstruction quality.

DAF fake track χ^2 probability

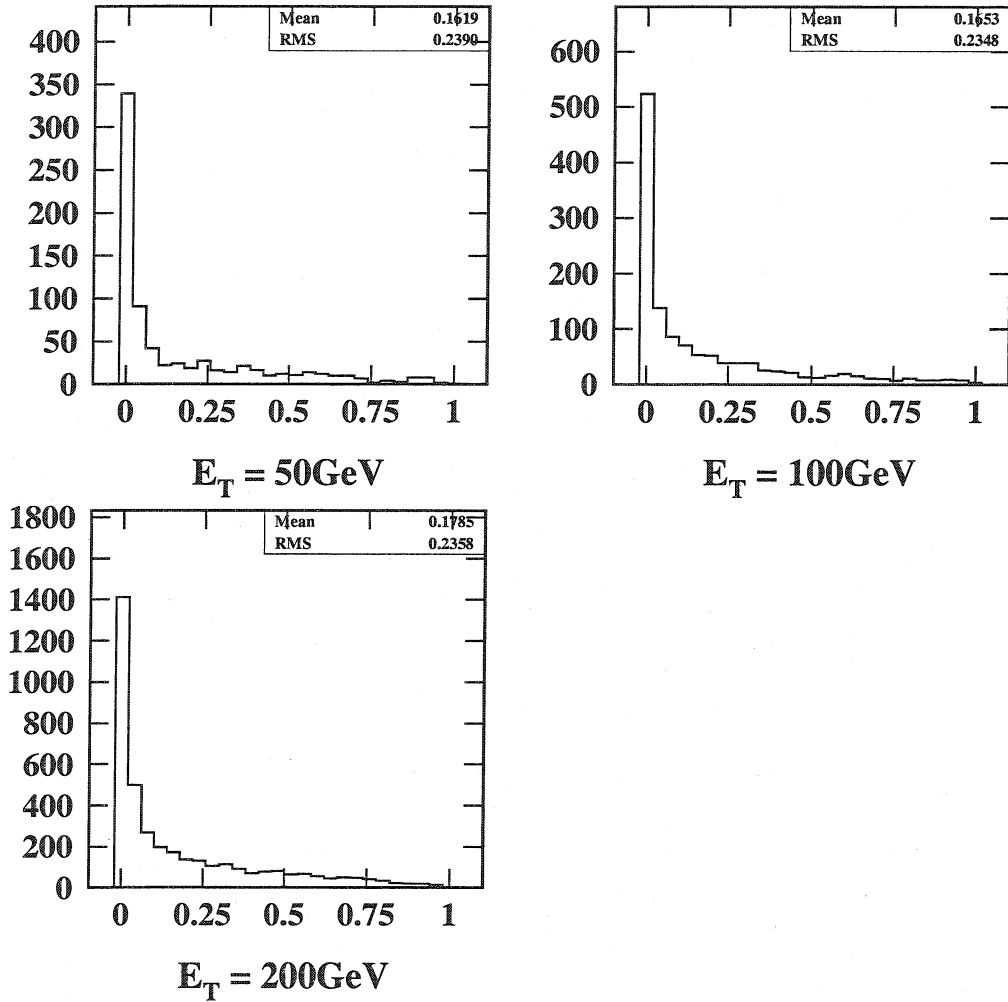


Figure 5.24: χ^2 -probability distributions of the DAF for fake tracks. The peak at 0 is expected, as fake tracks are supposed to have a very low χ^2 -probability. What is not expected is the large and almost flat tail up to a χ^2 -probability of 1, which indicates that these reconstructed tracks are actually good tracks. It might indicate an unwanted feature in the selection of simulated tracks, which are missing in the analysis of the fake rate.

secondary vertex efficiency

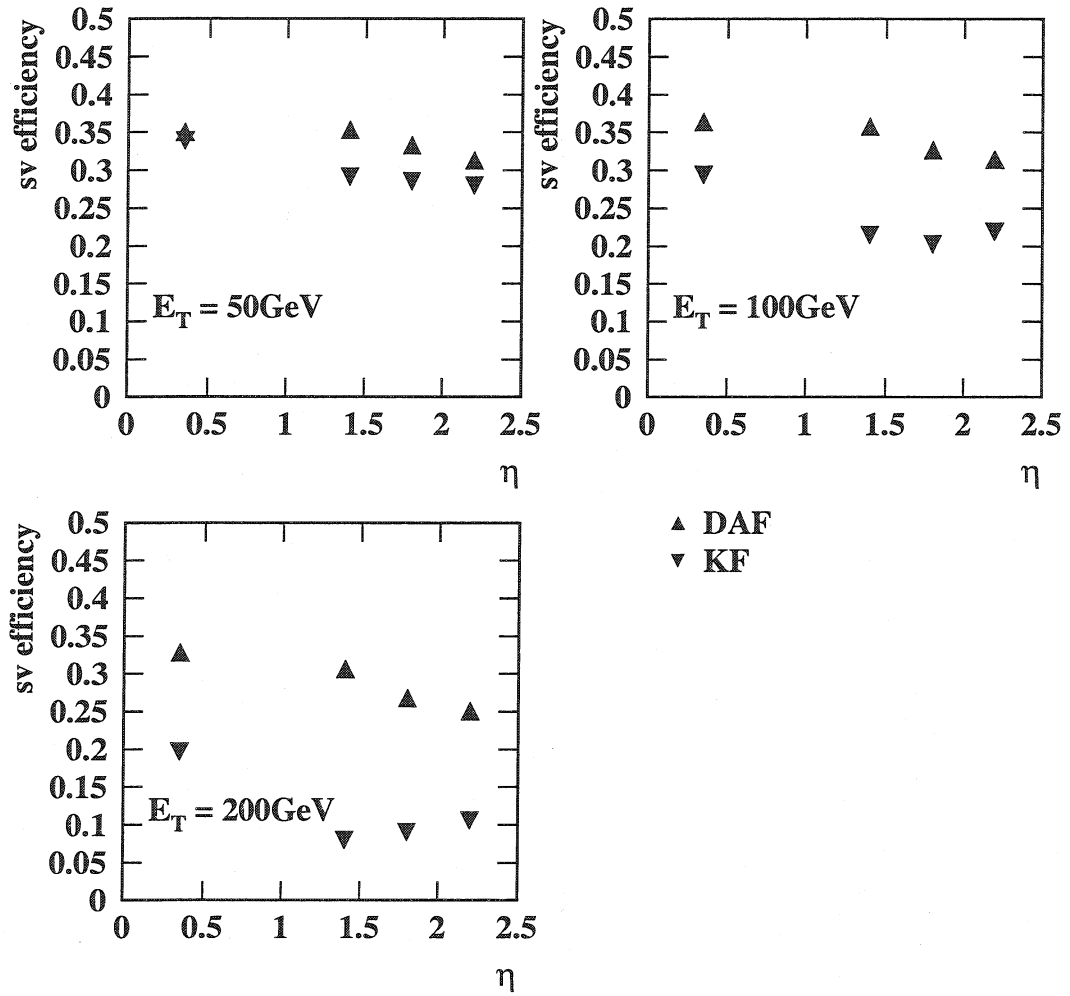


Figure 5.25: Secondary vertex finding efficiency in b -jets. In all cases, the DAF has better secondary vertex finding efficiency than the KF. The difference between DAF and KF is smallest in $E_T = 50$ GeV jets, and largest in $E_T = 200$ GeV jets.

5.4.6 Secondary vertex finding efficiency

Fig. 5.25 shows the secondary vertex finding efficiency. In all cases, the DAF has better secondary vertex finding efficiency than the KF. It is 30%–35% for b -jets with $E_T = 50$ GeV for the DAF and 28%–35% for the KF. For b -jets with $E_T = 100$ GeV it is 30%–36% for the DAF and 20%–30% for the KF. The secondary vertex finding efficiency is lowest in b -jets with $E_T = 200$ GeV, with 25%–33% for the DAF and 10%–20% for the KF.

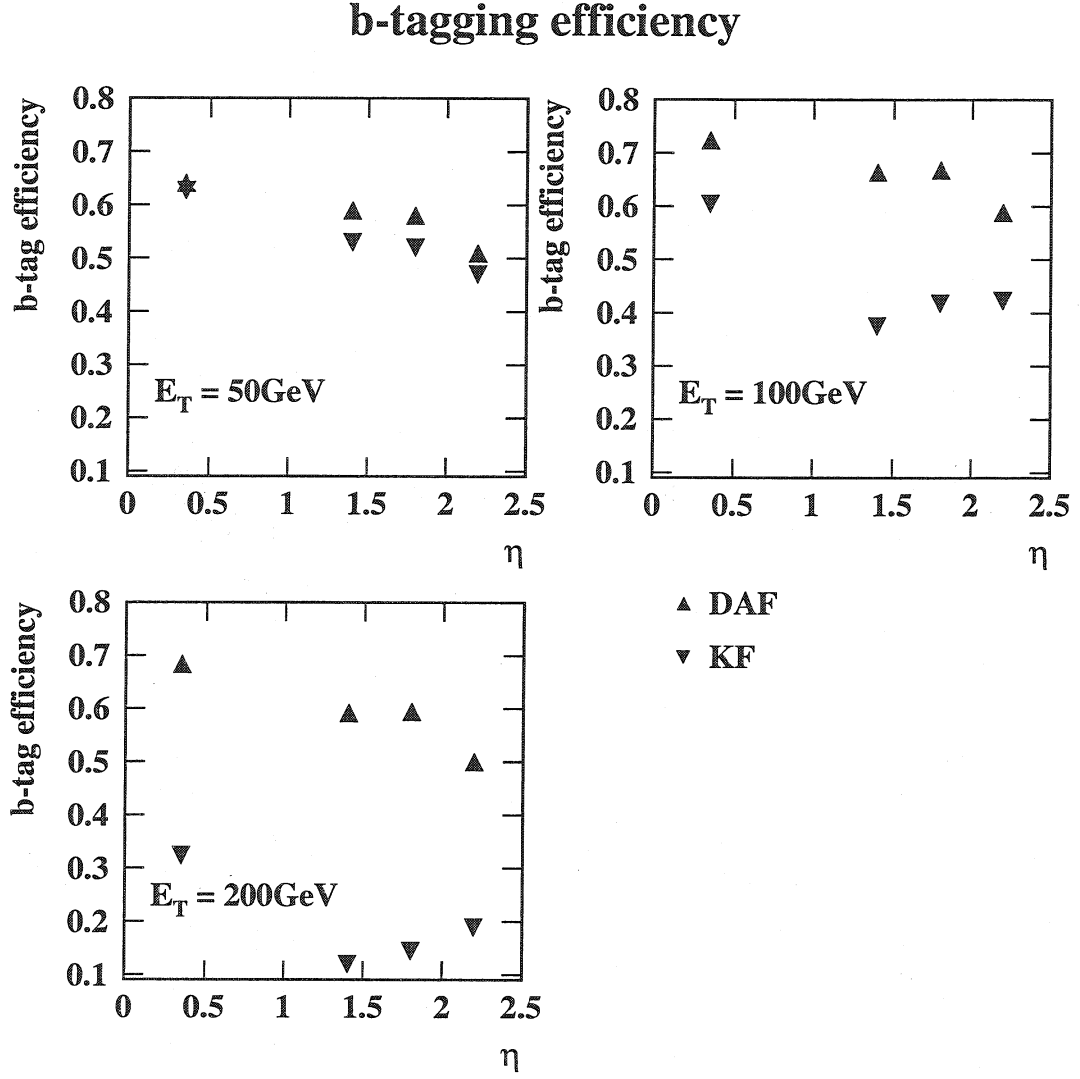


Figure 5.26: b -tagging efficiency in L2 jets. Especially for b -jets with high E_T , the b -tagging efficiency of the DAF is significantly higher than the one of the KF.

5.4.7 b -tagging efficiency by secondary vertex reconstruction in trigger level-2 jets

For the analysis of the b -tagging efficiency of the DAF and the KF, an algorithm based on secondary vertex reconstruction was chosen [27].

Fig. 5.26 shows the b -tagging efficiency in L2 jets. Especially for b -jets with high E_T , the b -tagging efficiency is significantly higher for the DAF than for the KF. The b -tagging efficiency of the DAF is above 50% everywhere, up to 72% in the barrel for b -jets with $E_T = 100 \text{ GeV}$. The difference with respect to the KF is within 5% for $E_T = 50 \text{ GeV}$ jets, and it is largest for $E_T = 200 \text{ GeV}$ jets.

Fig. 5.27 shows the b -mistagging in L2 jets. The mistagging was determined by

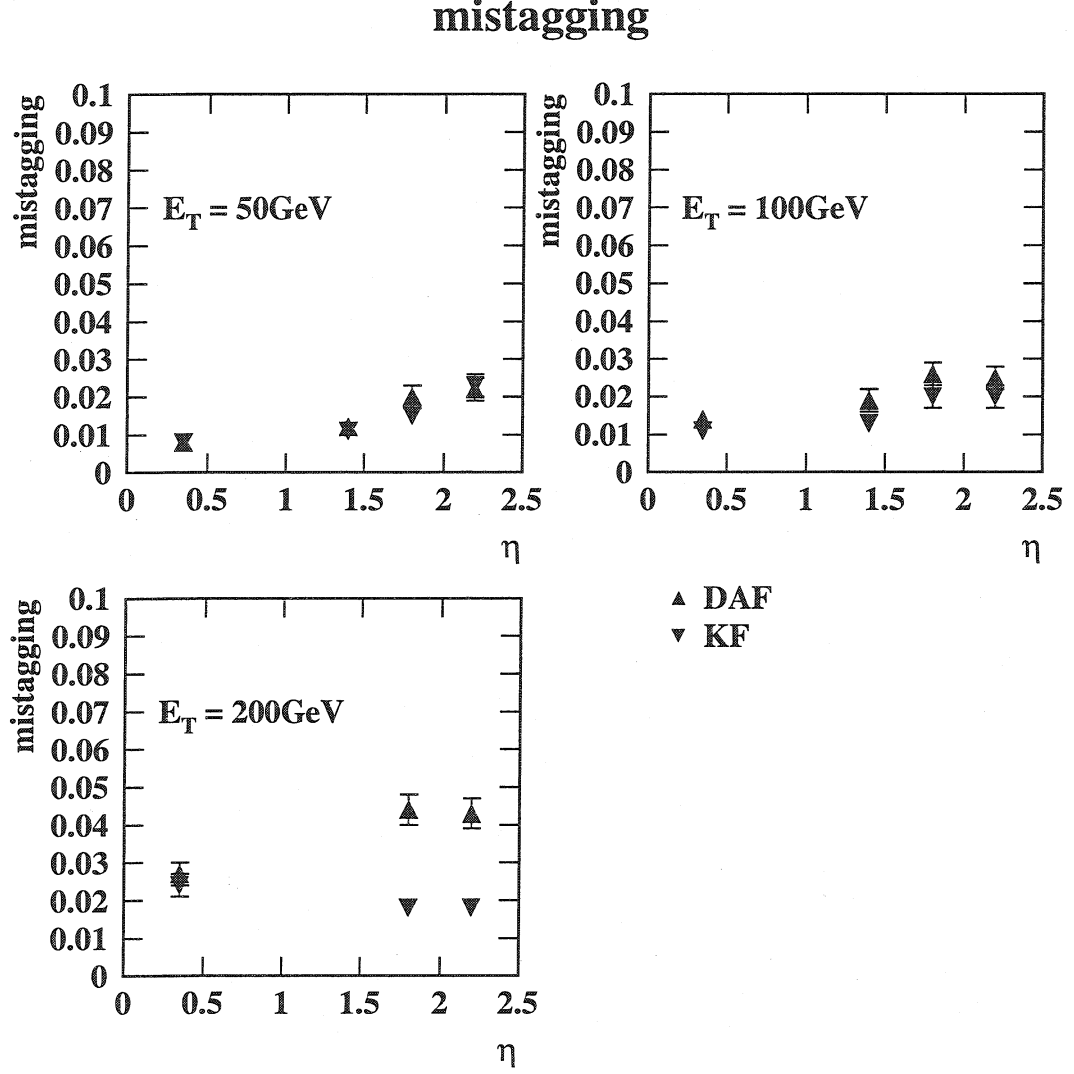


Figure 5.27: b mistagging in L2 jets. The mistagging is comparable for $E_T = 50, 100$ GeV u -jets. In the forward region of $E_T = 200$ GeV u -jets, the DAF has an about twice as high mistagging rate than the KF. It reaches up to 4.4% for the DAF, whereas it is 2% for the KF.

using the same b -tagging algorithm on $u\bar{u}$ events with identical kinematical cuts. In the barrel jets and for $E_T = 50, 100$ GeV u -jets the mistagging of DAF and KF is comparable. In the forward region of $E_T = 200$ GeV u -jets the DAF has about twice the mistagging rate than the KF. It is up to 4.4% while it stays at 2% for the KF.

Following the results of the b -tagging comparison, the KF seems to be less suitable for this particular b -tagging algorithm based on secondary vertex reconstruction.

5.5 Mass reconstruction of $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$

The $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$ channel is particular sensitive to the precision of the track reconstruction. In order to discover this channel at CMS, maximum precision of the mass reconstruction of the $B_{(d)s}^0$ is necessary in order to suppress the background. The two muons of the $B_{(d)s}^0$ are mainly low- p_T tracks.

In order to quantify the difference in reconstruction performance between the DAF and the KF in this particular channel, the mass of a $B_{(d)s}^0$ was reconstructed both in the low-luminosity and in the high-luminosity scenario at CMS. A complete study of this channel was previously done in [26].

Fig. 5.28 shows some important characteristics from the simulation. The simulated mass of the $B_{(d)s}^0$ is $5.369 \text{ GeV}/c^2$. The p_T distribution of the muons shows that almost all muons have a p_T less than $30 \text{ GeV}/c$. The tracks are selected within $|\eta| < 2.5$.

5.5.1 Low luminosity scenario

Fig. 5.29 shows the reconstructed mass and momentum residuals for the DAF and for the KF. The histograms are fitted with a mixture of two Gaussians. P1–P3 are the fitted values of the first Gaussian for the height, the mean and the sigma of the distribution, P4–P6 are the corresponding values for the second Gaussian. Both the KF and the DAF show a shift in the residuals towards higher reconstructed masses. The shift is of the order of $14 \text{ MeV}/c^2$. Both DAF and KF reconstruct the mass with a precision of about $41 \text{ MeV}/c^2$ for the core part of the distribution. The RMS of the distribution is about $56 \text{ MeV}/c^2$.

Fig. 5.30 shows the transverse and longitudinal impact parameters of the two muons at the $B_{(d)s}^0$ decay vertex. The residuals of the impact parameters were fitted with a mixture of two Gaussians with common mean value. The corresponding values of P1–P5 are P1 for the common mean value, P2 is the height of the core distribution, P3 is the sigma of the core distribution. P4 is the height of the tail distribution, P5 the corresponding sigma. The sigmas of the mixtures are printed as well. DAF and KF give similar results for both the transverse and the longitudinal impact parameter. The sigma of the mixture is about $40 \mu\text{m}$ for the transverse, and about $80 \mu\text{m}$ for the longitudinal impact parameter. The values of the DAF are slightly better but the difference is statistically not significant.

Fig. 5.31 shows the bias in the mass residuals as a function of η and φ . It can be seen that the bias depends on η , and that it is lower for muons in the central barrel

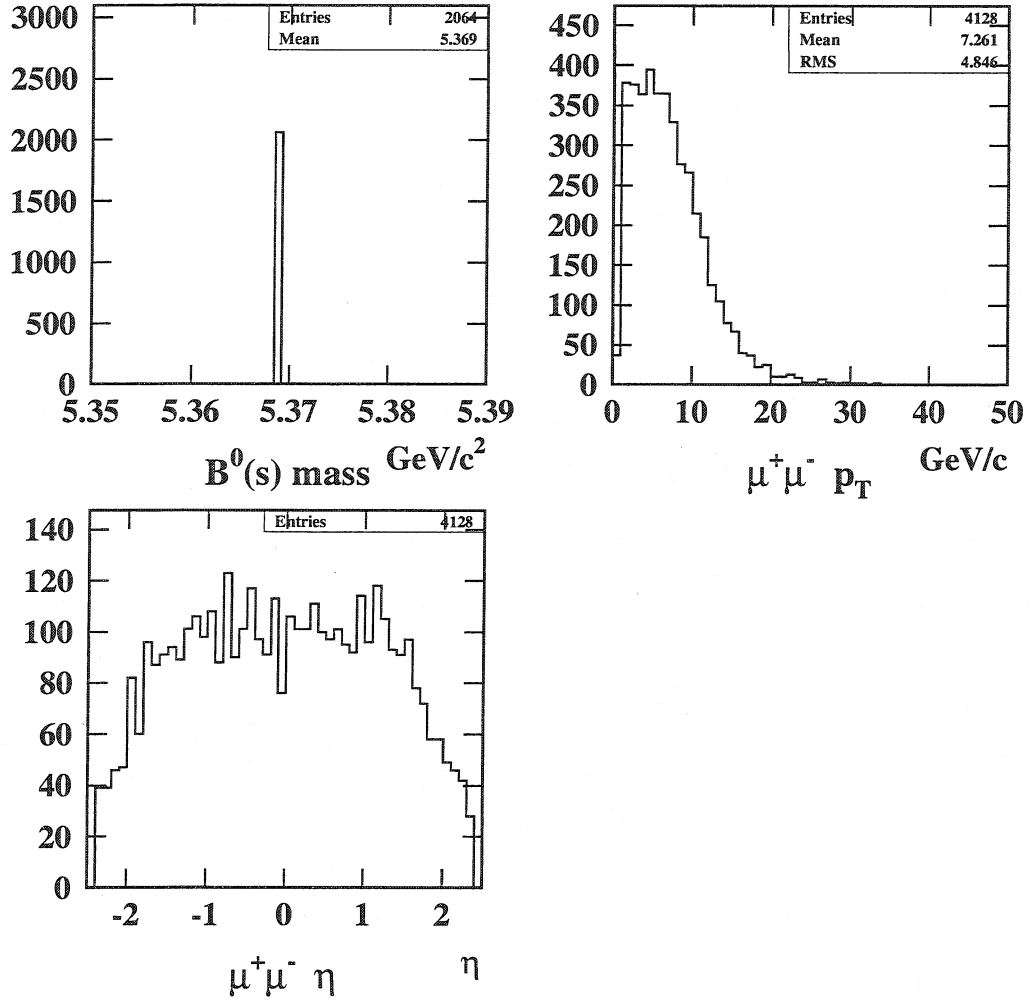


Figure 5.28: Simulation values of the decay $B^0_{(d)s} \rightarrow \mu^+\mu^-$. The simulated mass of the $B^0_{(d)s}$ is 5.369 GeV/c^2 . The p_T distribution of the muons shows that almost all muons have a p_T less than 30 GeV/c . The tracks are selected within $|\eta| < 2.5$

region of the CMS Tracker. The bias as function of φ shows a systematic shift of about 10 – 20 MeV/c^2 for the whole φ range.

5.5.2 High luminosity scenario

The same type of analysis has been performed in the high luminosity scenario at CMS.

Fig. 5.32 shows the residuals of mass and momentum both for the DAF and the KF for the same events. No statistically significant degradation in momentum and mass resolution is observed with respect to the results obtained in the low luminosity scenario. The difference between the DAF and the KF in the mass resolution is

low luminosity

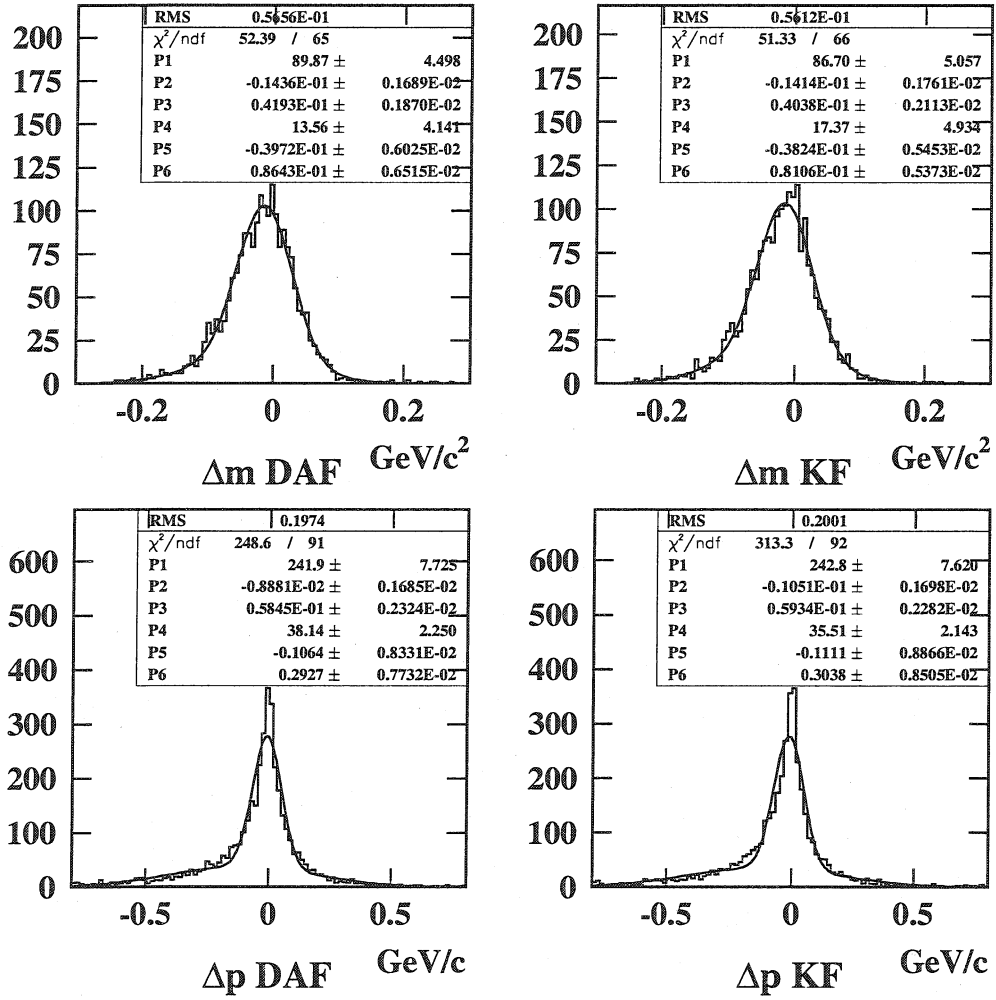


Figure 5.29: Reconstructed mass and momentum of the $B^0_{(d)s} \rightarrow \mu^+\mu^-$ decay for the low luminosity scenario at CMS. The mass of the $B^0_{(d)s}$ is reconstructed with a precision of about 41 MeV/c^2 both by the DAF and by the KF.

statistically not significant.

Fig. 5.33 shows the position resolution of the muons at high luminosity. Again, no degradation for both impact parameters is observed with respect to the results from the low luminosity scenario.

The mass resolution of 41 MeV/c^2 for the core distribution is significantly worse than the 26 MeV/c^2 published in [26]. In addition, the residuals have not zero mean. The results of DAF and KF are consistent, and there is no statistically significant gain in resolution by using the DAF instead of the KF. The difference in luminosity has virtually no effect on the mass and impact parameter resolutions.

low luminosity

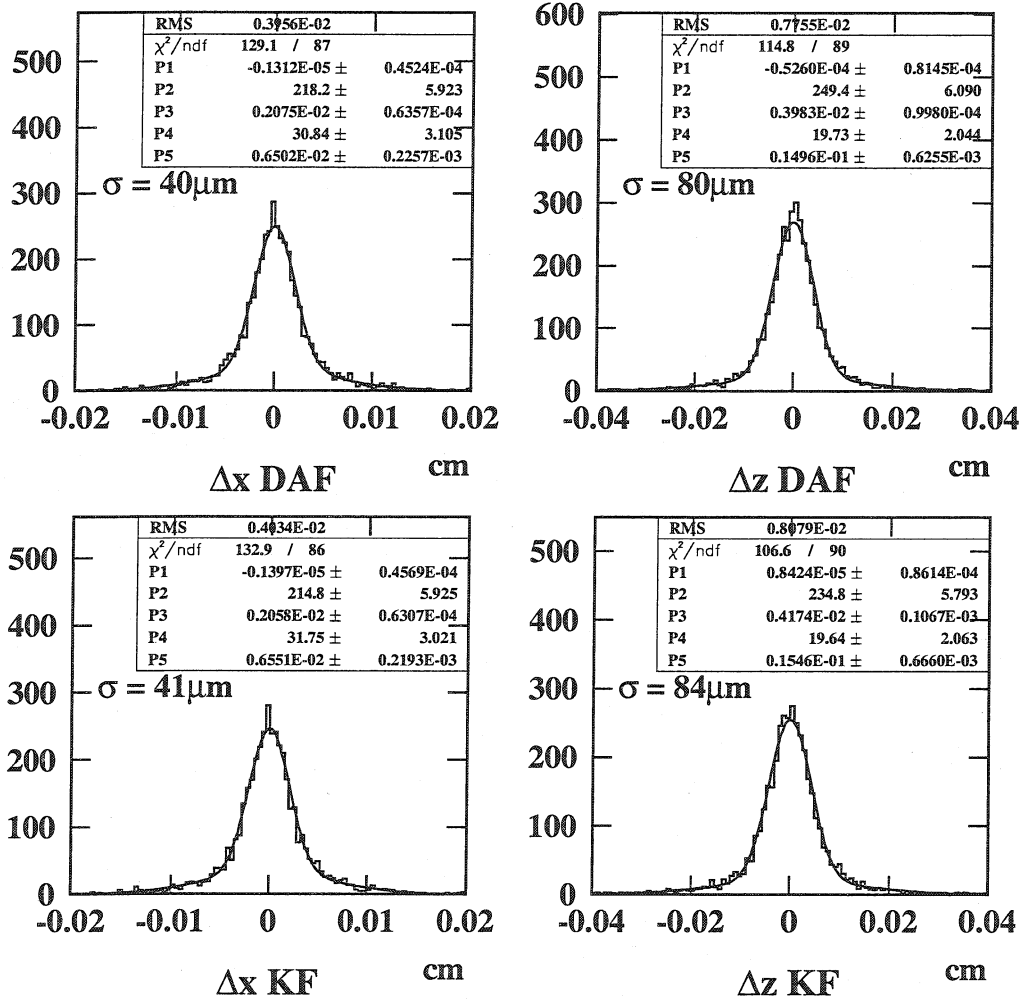


Figure 5.30: Transverse and longitudinal impact parameter of the muons at the $B^0_{(d)s} \rightarrow \mu^+\mu^-$ decay vertex. DAF and KF have the same transverse and longitudinal impact parameter resolution within the statistical uncertainty.

5.6 Conclusion and comparison of results with TDR [2]

The performance of the two different track reconstruction methods in the CMS Tracker can be summarized as follows:

- Fig. 5.34 shows a TDR-like overview of the resolutions for the two impact parameters and the inverse transverse momentum. Comparing the results with the high-luminosity pixel configuration in the TDR, the transverse impact parameter resolution is better by about 20%–30% for the three types of muon tracks and over the whole CMS Tracker region. For tracks with p_T above

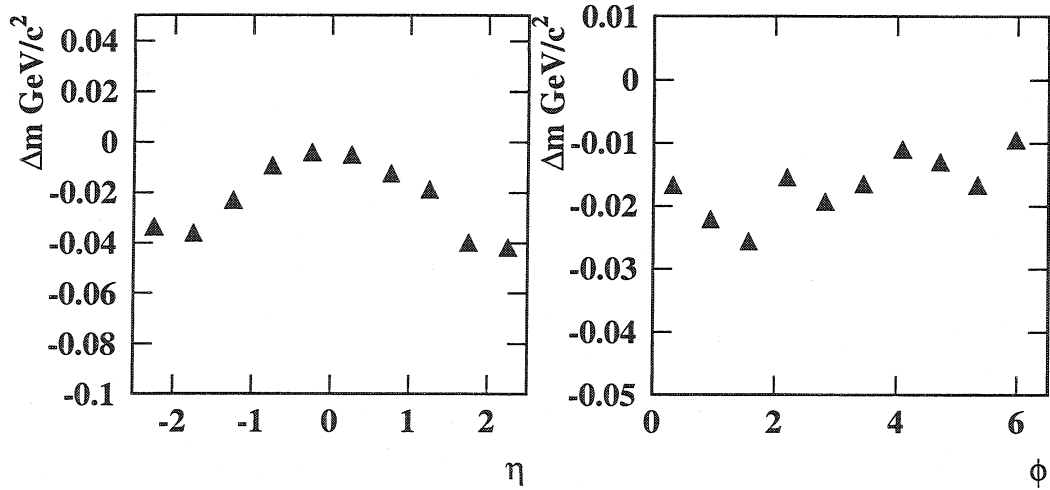


Figure 5.31: Bias in reconstructed mass as function of η and ϕ of the $B^0_{(d)s}$ mass. The bias of the mass depends on η , and the bias as function of ϕ shows a systematic shift of about $10 - 20 \text{ MeV}/c^2$ for the whole ϕ range.

$10 \text{ GeV}/c$, the transverse impact parameter resolution is better than $25\mu\text{m}$ over the full η -range (TDR: $35\mu\text{m}$). The longitudinal impact parameter is better as well by about 20%–30% compared to the one of the TDR. It is $40\mu\text{m}$ – $70\mu\text{m}$ for the η -range up to 2 (TDR: $80\mu\text{m}$ – $120\mu\text{m}$). On the other hand, the p_T resolution is comparable in the barrel region of the tracker, but it is worse by about 20%–30% in the η -regions larger than 1–1.5 to 2.5.

The changes in the values can be explained by the fact that the layout of the CMS Tracker has significantly changed since the publication of the Tracker TDR, with respect both to the type of sensors (silicon instead of gas detectors) and to the layout (reduced number of layers and radius of the tracker).

- As shown in Fig. 5.5, the residuals of the transverse track parameters (p_T , ϕ , x) have a bias which varies with η . The relative bias of p_T is independent of the transverse momentum, approaching 1% in the forward region. For the ϕ and x coordinate the bias is smaller for higher momenta. Suboptimal material description in the track reconstruction and an inhomogeneous magnetic field which is not properly taken into account are potential sources of these biases. No significant bias is observed for the longitudinal track parameters λ and z .
- Muons are reconstructed with an efficiency close to 100% in the η -region up to 2 (see Fig. 5.6). The DAF has a slightly better efficiency for $1 \text{ GeV}/c$ muons in the barrel, and a significantly better efficiency in the very forward region $|\eta| > 2.3$. The reconstruction efficiency better than 98% mentioned in the TDR can be confirmed for the $|\eta|$ -region up to 2.
- Isolated pions with $p_T = 1 \text{ GeV}/c$ are reconstructed with an efficiency above 90% (DAF) and 85% (KF) in the barrel region, and with an efficiency of about

high luminosity

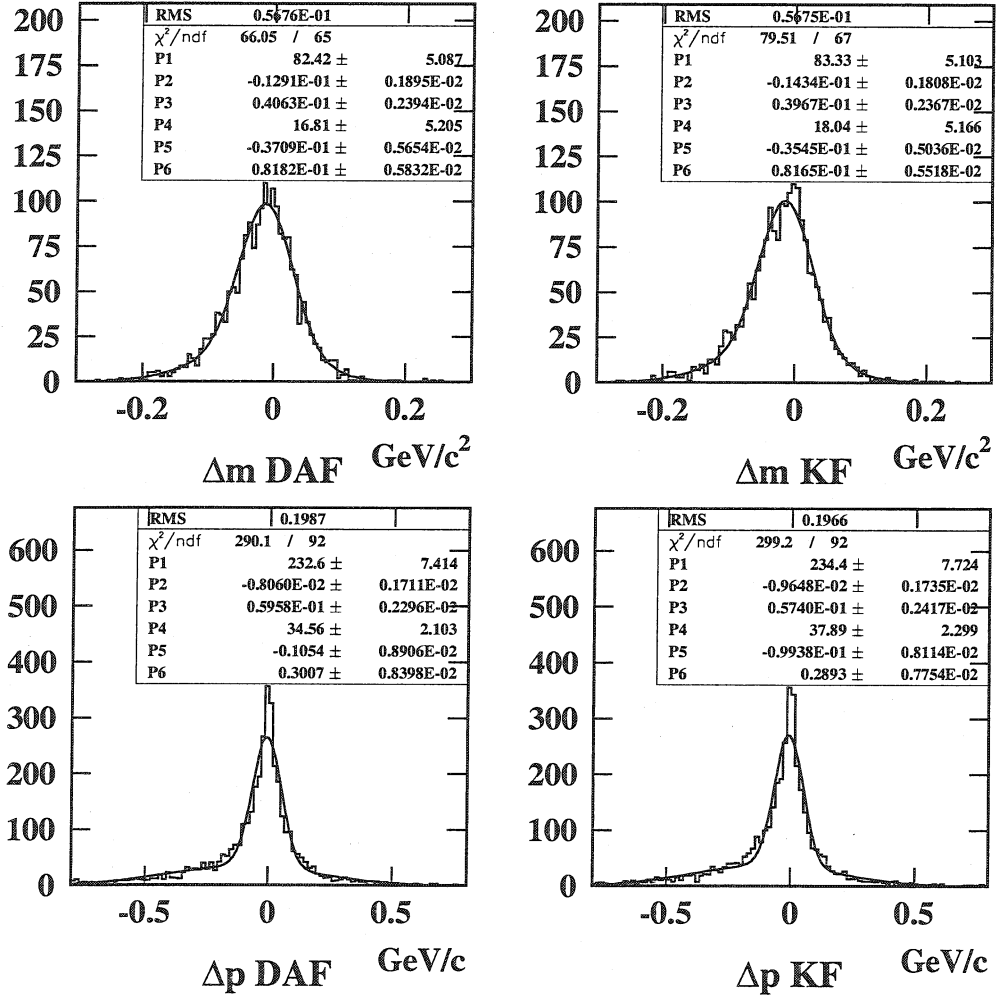


Figure 5.32: Reconstructed mass and momentum of the $B_{(d)s}^0$ decay for high luminosity. No degradation in momentum and mass resolution is observed with respect to the results obtained for the low luminosity scenario.

80% (DAF) and 70%–80% (KF) in the forward region (see Fig. 5.7). The value given in the TDR (for dense jets) is 85%, which can be confirmed for the barrel region and for the KF, and which is 5% better for the DAF. Pions with $p_T = 10, 100 \text{ GeV}/c$ are reconstructed with an efficiency of 95% or better for the DAF in the barrel (90%–95% for the KF), and about 85%–90% both for the DAF and the KF in the forward region. Compared with the value of 95% given in the TDR (for dense jets), this can only be confirmed in the barrel region of the CMS Tracker. In general, the DAF has a better reconstruction efficiency than the KF for $p_T = 1 \text{ GeV}/c$ pions (about 5%), and a slightly better efficiency (a few percent) also for the high- p_T pions. In all cases the efficiency

high luminosity

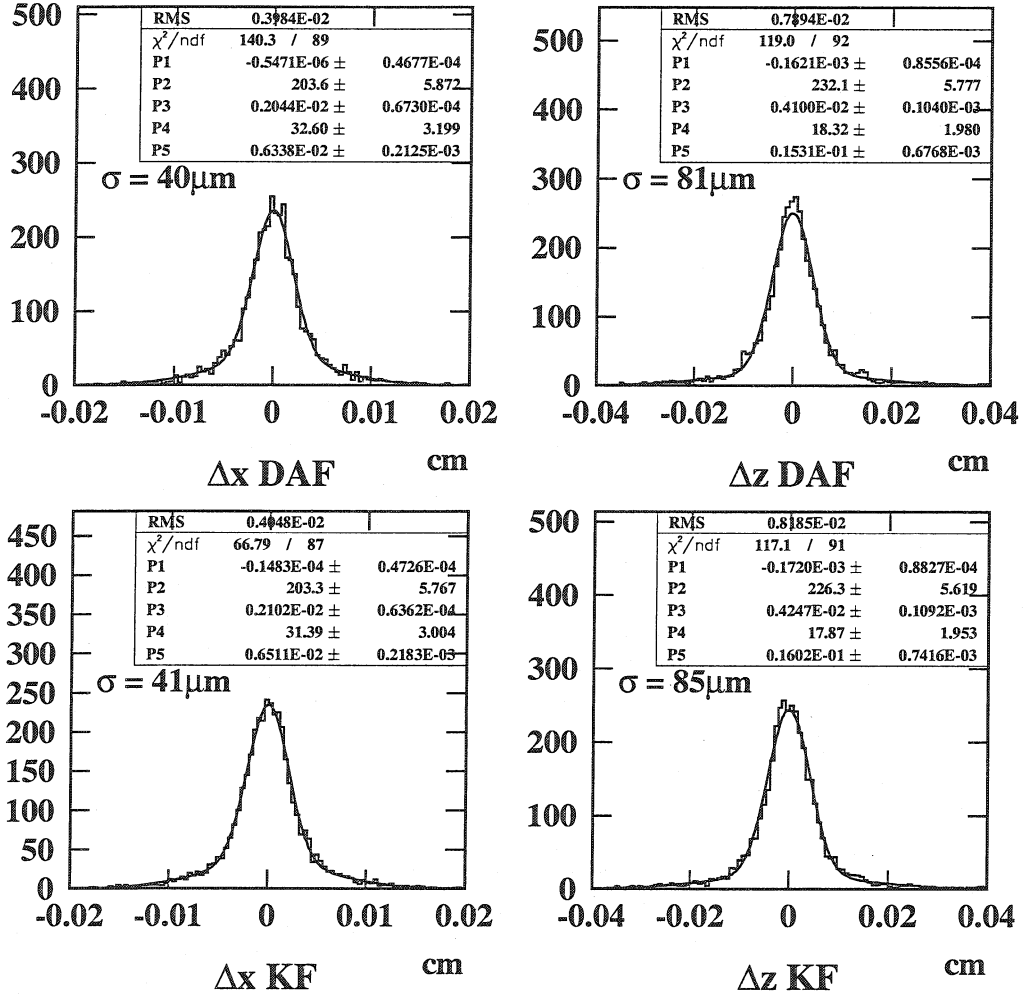


Figure 5.33: Transverse and longitudinal impact parameter of the muons in the $B^0_{(d)s} \rightarrow \mu^+ \mu^-$ decay for high luminosity. No degradation for both impact parameters is observed with respect to the results from the low luminosity background.

for very high η tracks (2.2–2.5) is about twice as high for the DAF than for the KF (50%–60% compared to 20%–30%).

- Pions in b -jets are reconstructed with an efficiency above 82% for the DAF and above 78% for the KF for $|\eta| < 2$ (see Fig. 5.18). The overall reconstruction efficiency compared to the one of the TDR (above 85%) is comparable for the DAF and slightly worse for the KF.
- The fake rate in b -jets (see Fig. 5.22) is comparable for tracks in $E_T = 50$ GeV jets (below 1%). In jets with higher E_T , the fake rate is considerably higher for the KF (up to 15% in the forward) compared to the DAF (up to 8%). A

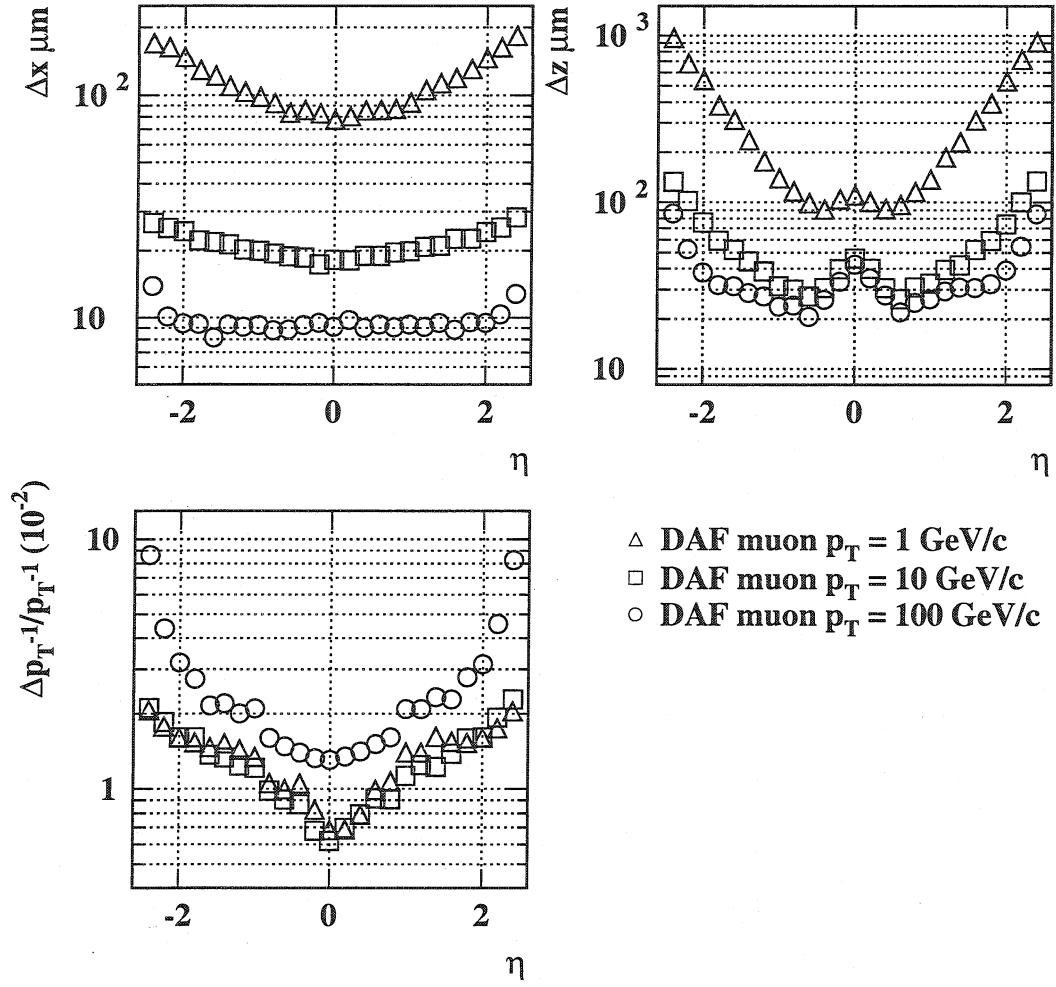


Figure 5.34: TDR-like overview of resolution of the transverse and longitudinal impact parameter and of the inverse transverse momentum.

fake track analysis for the DAF (see Fig. 5.23) shows that with a cut on the number of effective hits in the fit, the number of fake tracks can be reduced by 40%–50% by requesting at least 9 effective hits. However, a large fraction of fake tracks of the DAF has a high number of effective hits and also a good χ^2 probability (see Fig. 5.24). This might indicate a problem in the selection of simulated tracks for the analysis.

- The analysis of $p_T > 15 \text{ GeV}/c$ tracks in $b\bar{b}$ events with $E_T = 200 \text{ GeV}$ shows that the DAF has significantly better impact parameter resolutions and pulls (see Figs. 5.10 and 5.14).
- The secondary vertex finding efficiency in b -jets is 25%–35% for the DAF everywhere (see Fig. 5.25). The KF has a lower secondary vertex finding efficiency than the DAF, which is a few percent less in $E_T = 50 \text{ GeV}$ jets, and which is 10%–15% less in $E_T = 200 \text{ GeV}$ jets.

- The b -tagging efficiency (see Fig. 5.26) was analysed using an algorithm based on secondary vertex reconstruction [27]. In the barrel it is in general between 65% and 72% for the DAF and between 30% and 65% for the KF. In the forward region, for jets with $E_T = 50, 100$ GeV the b -tagging efficiency is 50%–65% for DAF and 40%–55% for the KF. For b -jets with $E_T = 200$ GeV it is 50%–55% in the forward region (10%–20% for the KF). The b -tagging efficiency of the DAF is considerably better compared to the TDR (50% efficiency for jets in the barrel and 40% efficiency for jets in the forward region). Currently the KF is not suitable for this kind of b -tagging algorithm.
- The mistagging rate (see Fig. 5.27) was estimated using $u\bar{u}$ events with identical kinematical constraints. In general, for $E_T = 50$ GeV jets in the barrel DAF and KF have comparable mistagging rates (1%–2.5%). A significant difference is observed in $E_T = 200$ GeV jets for the forward region, where the mistagging rate of the DAF (4.4%) is about twice as large than the one of the KF (2%).
- The mass of the $B_{(d)s}^0$ ($5.369 \text{ GeV}/c^2$) decaying into $\mu^+\mu^-$ can be reconstructed with a core precision of about $41 \text{ MeV}/c^2$, both for the DAF and the Kalman Filter. The precision of the reconstructed mass is not affected by high luminosity background. On the other hand, there is a systematic bias of the mean value of the mass residuals, of the order of $14 \text{ MeV}/c^2$. This bias may come from suboptimal material treatment in the track fit and the inhomogeneous magnetic field in the forward region (see Fig. 5.31).

In general, in absence of ambiguities the DAF and KF behave equally well. In dense jet environments, the DAF has both higher efficiencies and lower fake rates than the KF. The mass reconstruction of $B_{(d)s}^0 \rightarrow \mu^+\mu^-$ is equal for the DAF and the KF.

A comparison with the results presented in the TDR shows that the DAF can achieve better results in terms of impact parameter resolution and b -tagging efficiency. The reconstruction efficiency of isolated muons is as good as in the TDR, and the reconstruction efficiency of isolated pions is as good as in the TDR for the barrel region. It is worse by 5% in the forward region for isolated pions. The main deterioration in the p_T resolution is most probably due to a better knowledge of the CMS Tracker material budget (an increase of 50%) and a different layout (smaller lever arm).

6 Performance studies of multitrack reconstruction

The analysis of 3-prong τ decays originating from heavy SUSY-Higgs with $m_H = 500$ GeV produced by $gg \rightarrow b\bar{b}H^0$, $H^0 \rightarrow \tau^+\tau^-$, shows that in such events the tracks of the τ jet are very collimated and potential candidates for the Multi Track Filter (MTF). By reconstructing 3-prong τ s and taking them into account in the analysis process, the number of $H^0 \rightarrow \tau^+\tau^-$ events could be increased by a factor of about 1.7 [28, 29, 30].

The main difficulty for the track reconstruction in these events is that the τ jets have to be reconstructed with exactly three tracks and with an optimal estimation of the impact parameters for the vertex reconstruction and background rejection.

In order to quantify and qualify the performance in 3-prong τ jets, results are compared with the Kalman Filter [20] and Deterministic Annealing Filter [23].

6.1 Track selection and parameter settings

For the track selection the following parameters were chosen:

- Simulated tracks were selected according to:
 - $p_T > 0.9$ GeV/ c ,
 - $|\eta| < 2.5$,
 - radial distance of simulated vertex < 3 cm,
 - longitudinal distance of simulated vertex < 30 cm.
- Reconstructed tracks were selected according to:
 - $p_T > 0.7$ GeV/ c ,
 - $|\eta| < 2.6$,
 - transverse impact parameter < 120 cm,
 - longitudinal impact parameter < 170 cm,
 - minimum number of 8 reconstructed hits.

For the comparison of the performance of the different track reconstruction methods, the following approach was selected:

- All tracks are reconstructed by KF and DAF in the full CMS Tracker.
- The annealing schema of the DAF was set to (81, 9, 4, 1, 1, 1).

- After the reconstruction, 3-prong τ decays were selected which fulfill the requirements for the use of the MTF. Only such events are used for the later analysis and comparison of track reconstructors.
- The requirements for the use of the MTF are three reconstructed tracks of a τ decay, with two or three tracks having a $p_T > 30 \text{ GeV}/c$.
- The MTF is run on selected pairs or triplets of tracks after the KF and after the DAF.
- The annealing schema of the MTF was set to (4, 1, 1, 1).
- The comparison is done between the KF, the DAF, the MTF run after the KF and the MTF run after the DAF.
- The residuals are fitted with a sum of two Gaussians which have the same mean value. The fitted parameters are the mean value of both Gaussians (P1), the height of the first Gaussian (P2), the sigma of the first Gaussian (P3), the height of the second Gaussian (P4) and the sigma of the second Gaussian (P5). The variance of the mixture can thus be computed in the following way:

$$\sigma^2 = \frac{\sigma_1 P2}{\sigma_1 P2 + \sigma_2 P4} \sigma_1^2 + \frac{\sigma_2 P4}{\sigma_1 P2 + \sigma_2 P4} \sigma_2^2.$$

- The transverse impact parameter (x), the longitudinal impact parameter (z) and the inverse transverse momentum (p_T^{-1}) are compared at the decay vertex of the τ .

6.2 Simulation information

The data sample [31] was simulated using the CMS 121 version of CMS for the low luminosity scenario, and ORCA_5_4_1 was used for the reconstruction. In order to illustrate some important physical characteristics of the data, the following plots have been made using the simulation information:

- Fig. 6.1 shows the distribution of the simulated radial distance of the τ lepton, the simulated η distribution of the τ lepton, the ΔR distribution of pairs of two tracks within a τ jet and the p_T distribution of the τ s with three selected simulated tracks.
 - The τ radial distance follows an exponential distribution and reaches up to 3 cm and more. For the analysis, τ s with a radial distance less than 3 cm have been selected.
 - τ leptons are selected within the η region $-2.4 < \eta < 2.4$.

- ΔR is defined as

$$\Delta R = \sqrt{((\Delta\varphi)^2 + (\Delta\eta)^2)}$$

and is a measure of the collimation of the τ jet at the τ vertex. The ΔR distribution shows that basically all pairs of tracks within a τ jet have a ΔR less than 0.075. For the separation of tracks in the CMS Tracker it is also important to know their transverse momentum p_T , as only pairs or triplets of tracks with a large enough p_T are sufficiently collimated to be potential candidates for MTF smoothing.

- Finally the p_T distribution of the τ s with 3 simulated tracks is shown.
- Fig. 6.2 shows the number of simulated tracks of the τ s which were selected (upper left), the fraction of tracks in τ jets with at least 8 simulated hits per track (“reconstructable” fraction of tracks), and the fraction of tracks in such “reconstructable” τ s with 3 tracks which are candidates for the MTF.
 - In the particular sample used for reconstruction and analysis, about 20% of the τ s have 3 selected simulated tracks, and about 70% have 1 selected track (upper left).
 - The fraction of tracks which originate from a τ with 3 selected tracks and which have at least 8 simulated hits per track with respect to all tracks from τ s with 3 selected tracks is shown in the upper right picture. In the barrel region of the tracker, about 90%–95% of the tracks coming from a τ with 3 tracks have at least 8 simulated hits per track and should therefore be reconstructable. The ratio drops to about 65%–70% percent in the forward region of the CMS Tracker.
 - The fraction of tracks which are potential candidates for the MTF is shown in the third distribution. In addition to the requirement of 8 simulated hits per track, it is also required that at least two tracks in such jets have a p_T larger than 30 GeV/c. About 50% to 70% of the tracks of “reconstructable” τ s with 3 tracks would also be candidates for the MTF.

6.3 Efficiency of reconstructing 3 tracks in τ jets

For the reconstruction efficiency of 3 tracks, τ jets with 3 simulated tracks are pre-selected. For the qualification as a “reconstructable” τ jet, at least 8 simulated hits per track are required for all three tracks. This definition will be used in order to quantify the algorithmic efficiency of reconstructing 3 tracks in τ jets. Tracks are reconstructed with a minimum number of 8 reconstructed hits in the CMS Tracker. The Kalman Filter accepts exactly one reconstructed hit per detector layer, therefore in this specific case the number of reconstructed hits is equal to the number of layers in the reconstructed track.

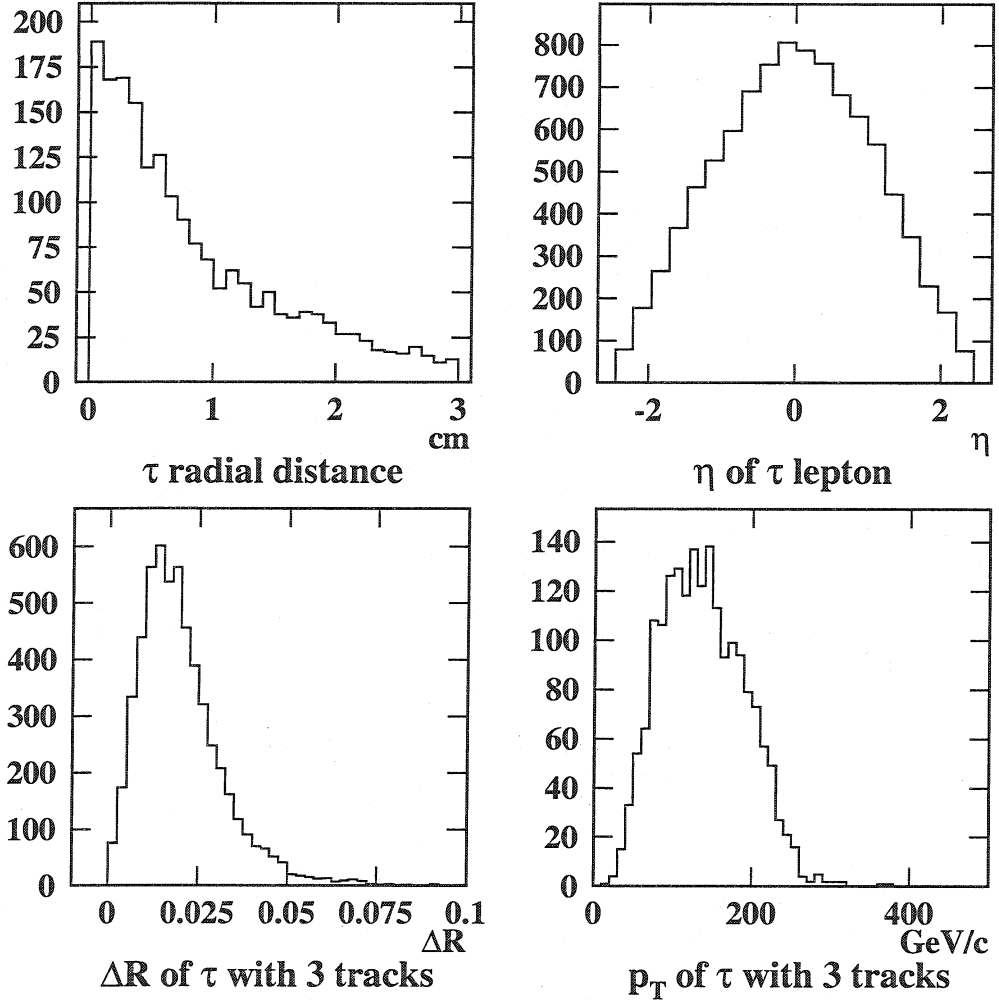


Figure 6.1: The τ radial distance follows an exponential distribution and reaches up to 3 cm and more. The η distribution of the τ leptons shows that τ s are selected within $-2.4 < \eta < 2.4$. The ΔR distribution shows that basically all pairs of tracks have a ΔR smaller than 0.075. For the separation of the tracks in the CMS Tracker also their p_T is of importance. Finally, the p_T distribution of the τ s with three simulated tracks shows that jets from τ s have a p_T up to 350 GeV/c.

Fig. 6.3 shows the fraction of reconstructed tracks in selected 3-prong τ jets (dashed line), both for the Kalman Filter (KF) and for the Deterministic Annealing Filter (DAF). About 75%–80% of all of the selected 3-prong τ s are reconstructed with three tracks. Around 20% of the τ jets are reconstructed with two tracks only. The fraction of “reconstructable” τ jets is represented by the fine dotted line. It is of the order of 85% of all selected 3-prong τ jets.

Fig. 6.4 shows the global track reconstruction efficiency of three tracks in τ jets both for the KF and the DAF. The solid line represents the fraction of reconstructed tracks of 3-prong τ s with respect to all simulated tracks from selected 3-prong τ s. The

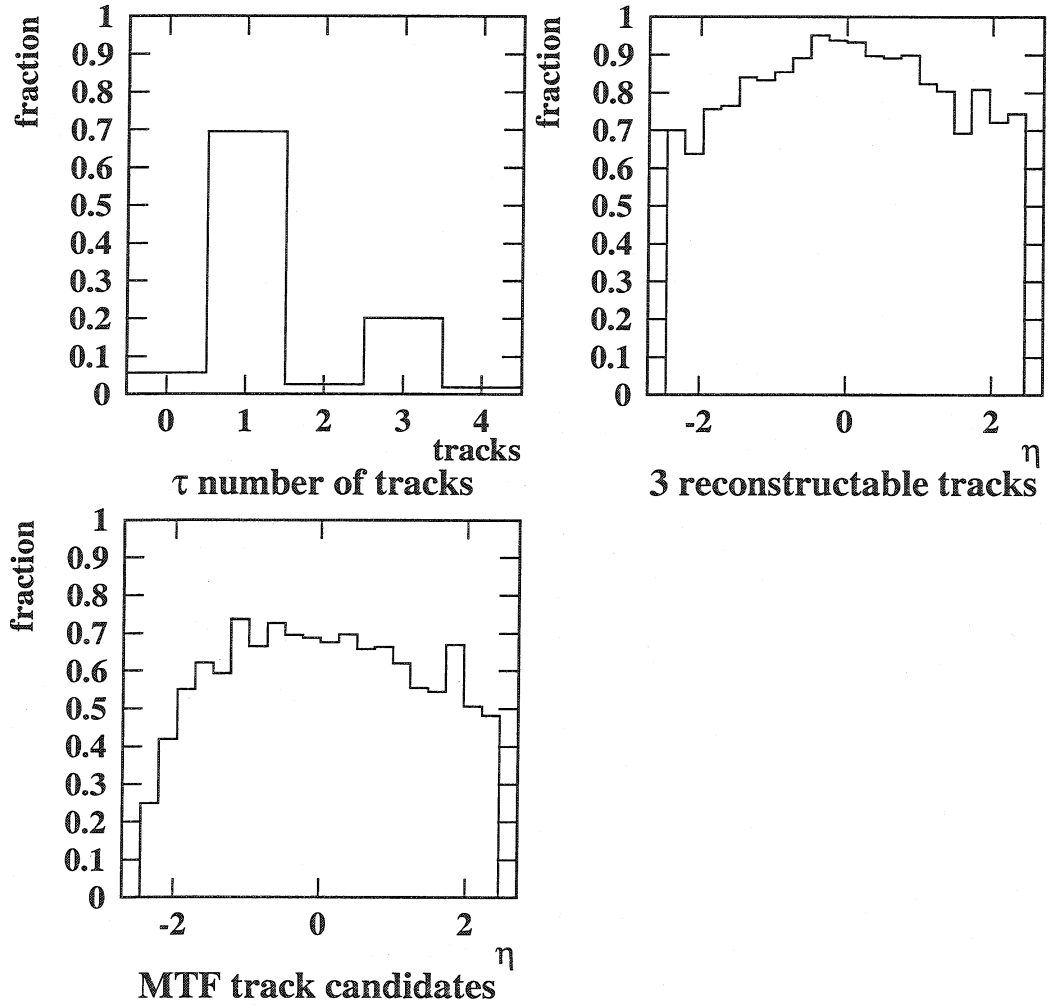


Figure 6.2: The upper left figure shows the number of selected simulated tracks of τ s. About 20% of the τ s have 3 selected tracks. The upper right figure shows the fraction of tracks originating from 3-track τ s with at least 8 hits per track with respect to all tracks from such τ s. This figure gives an idea of the upper limit of reconstructable tracks in τ jets. For example, in the barrel part of the CMS Tracker about 90% of the tracks from 3-track τ jets originate from a τ with at least 8 hits per track. The bottom left figure shows the fraction of tracks of 3-track τ s which are also candidates for the MTF smoothing, with respect to all the tracks of “reconstructable” 3-track τ s. For this, at least two tracks have to have a p_T above 30 GeV/ c in a 3-track τ .

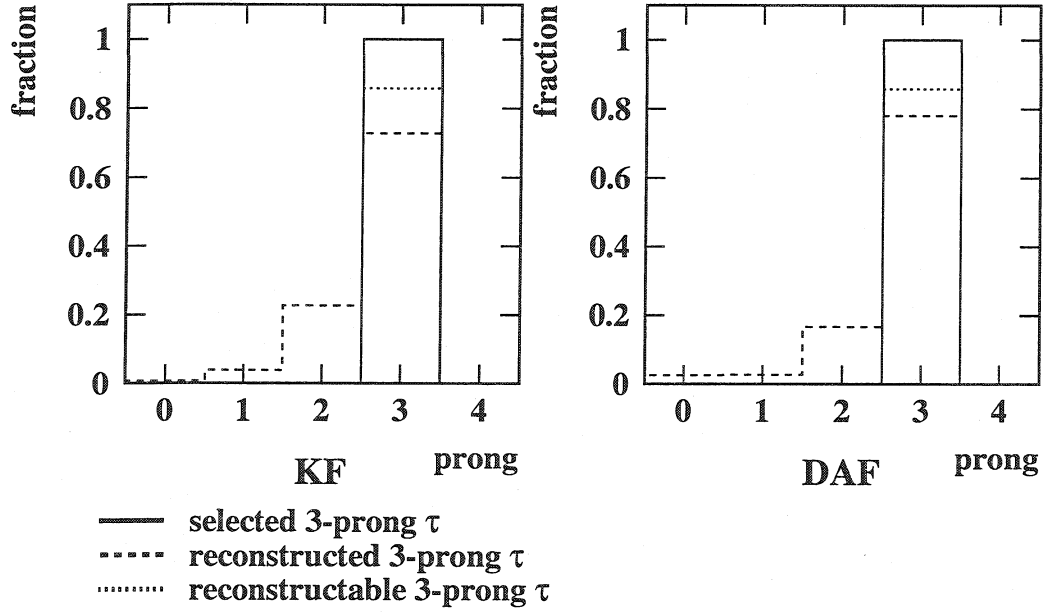


Figure 6.3: Fraction of reconstructed 3-prong τ s with three tracks, two tracks or one track, both for the KF and for the DAF, represented by the dashed line. KF and DAF reconstruct about 75%–80% of the selected 3-prong τ jets with three tracks. Around 20% of the 3-prong τ jets have two reconstructed tracks. The fraction of “reconstructable” τ jets is represented by the fine dotted line (about 85% of all selected 3-prong τ s).

dashed line represents the fraction of tracks of 3-prong τ s which are reconstructed with two tracks only. The fine dotted line represents the fraction of simulated tracks from reconstructable 3-prong τ s with respect to all simulated tracks from 3-prong τ s, which represents an upper limit of the reconstructability.

The KF has a global efficiency between 50% in the very forward region and a peak of about 90% around $\eta = 0$. The DAF has a global efficiency of about 55% in the very forward region and about 85%–90% in the central region of the CMS Tracker.

Fig. 6.5 shows the algorithmic track reconstruction efficiency for both track finders. This measure is a rather conservative one, as the request of eight simulated hits per track does not automatically translate into eight corresponding reconstructed hits in the CMS Tracker due to noise fluctuations and readout losses.

The algorithmic track finding efficiency of three tracks is 85% or better for the DAF for almost all of the region of the CMS Tracker, and it is 90% or better in the central barrel region. For the KF, the algorithmic efficiency is better than 75% for most of the Tracker, and reaches 90% and more around $\eta = 0$.

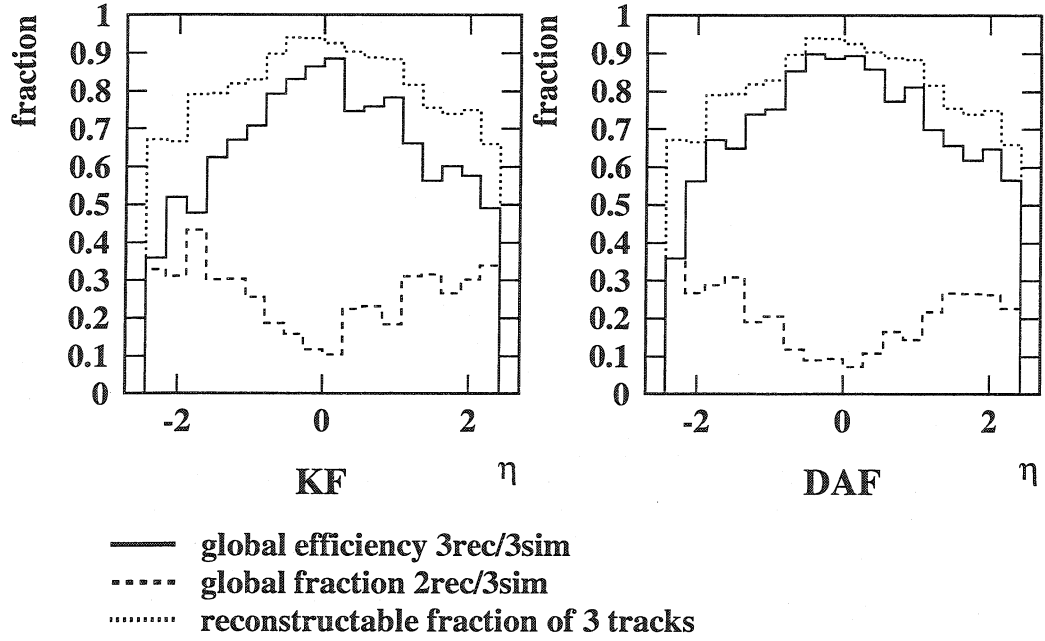


Figure 6.4: Global track reconstruction efficiency of three tracks in 3-prong τ decays (solid line), KF and DAF. The global track reconstruction efficiency for the DAF is about 80%–90% in the central barrel region of the CMS Tracker, and it is between 55%–75% in the forward region. The reconstruction efficiency of three tracks of the KF is comparable for the barrel, but it is worse by about 5%–10% in the forward region. The dashed line represents the fraction of two tracks reconstructed for selected 3-prong τ jets. The fine dotted line gives a measure the upper limit of reconstructable three tracks by requiring at least 8 simulated hits for each simulated track in the τ jet.

6.4 Track parameter resolution in 3-prong τ jets

Fig. 6.6 shows the transverse impact parameter resolution at the τ vertex. The core resolution of $10\mu\text{m}$ is about the same in all four cases. The standard deviation of the tails is of the order of $55\mu\text{m}$. The standard deviation of the mixture is $36\mu\text{m}$ for the KF, $30\mu\text{m}$ for the DAF and the MTF run after the KF and $29\mu\text{m}$ for the MTF run after the DAF. Relative to the KF, the DAF and the MTF have 17%–20% better resolution, which means that the amount of the tails could be reduced by a significant factor with respect to the KF.

Fig. 6.7 shows the longitudinal impact parameter resolution at the τ vertex. The core resolution of about $35\mu\text{m}$ is the same in all four cases. The standard deviation of the tails is of the order of $150\mu\text{m}$. Like in the case of the transverse impact parameter resolution, the tails are largest for the Kalman Filter. The standard deviation of the mixture is $98\mu\text{m}$ for the KF, and between $73\mu\text{m}$ and $76\mu\text{m}$ for the DAF and the MTF. Relative to the KF, the DAF and the MTF have about 25% better longitudinal impact parameter resolution.

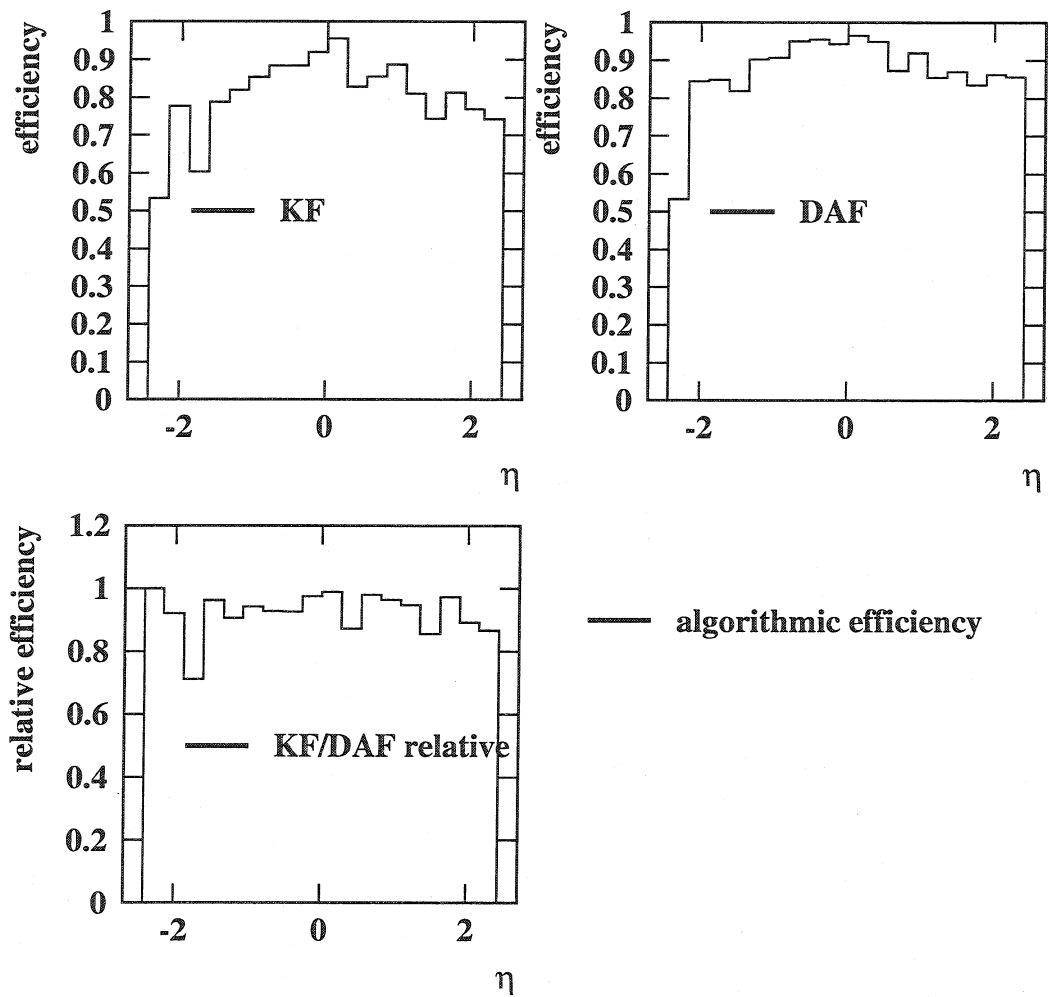


Figure 6.5: Algorithmic reconstruction efficiency of three tracks in 3-prong τ jets (solid line), both for the Kalman Filter (KF) and for the Deterministic Annealing Filter (DAF). In this analysis the algorithmic efficiency is defined by requiring at least eight simulated hits per track for all three simulated tracks in the τ jet. The DAF has an efficiency equal to or better than 85% over almost all of the η region of the CMS Tracker. In the central part the algorithmic efficiency is better than 90%. The KF has an efficiency better than 70% for most of the η region and up to 90% in the barrel. The relative efficiency of the KF with respect to the DAF shows a small advantage for the DAF.

Δx

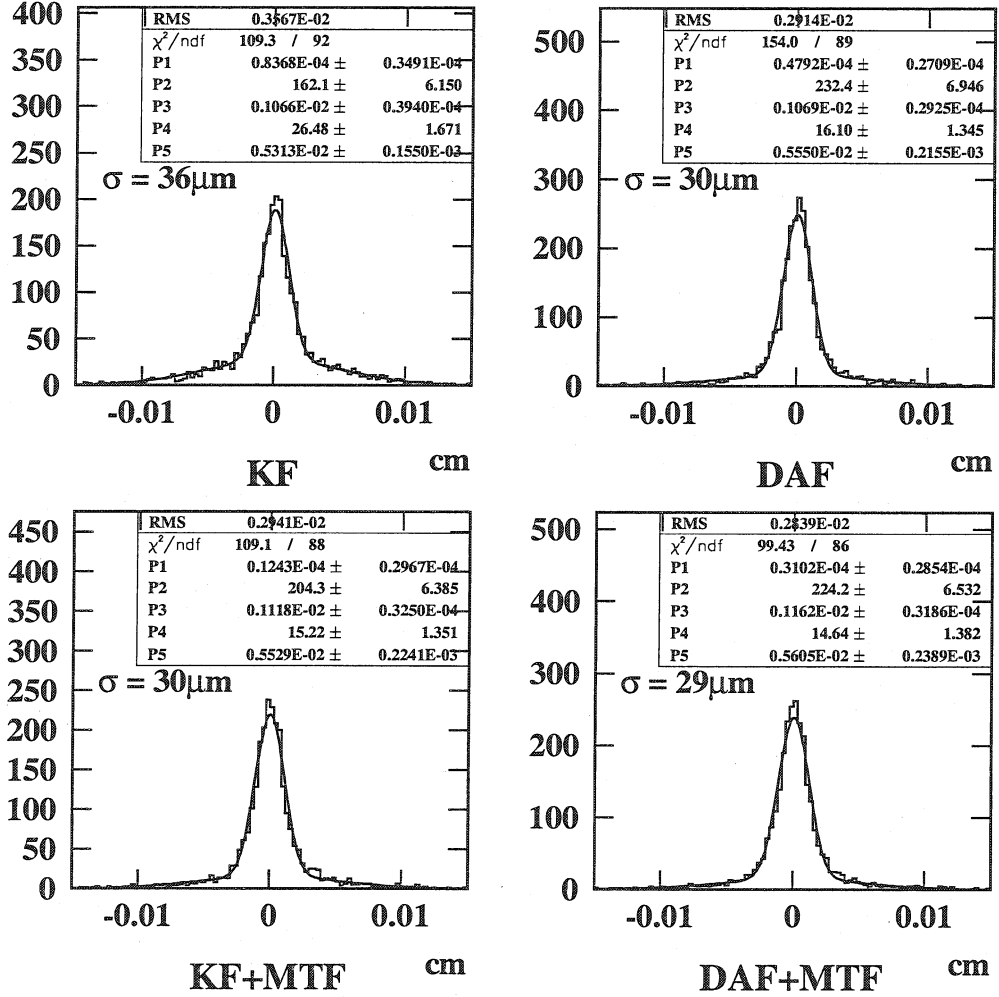


Figure 6.6: Transverse impact parameter resolution at the τ vertex. The core resolution of about $10\mu\text{m}$ is the same in all four cases. The standard deviation of the tails is about $55\mu\text{m}$. The sigma of the mixture is $36\mu\text{m}$ for the KF, $30\mu\text{m}$ for the DAF and the MTF run after the KF and $29\mu\text{m}$ for MTF run after the DAF. Relative to the KF, the other track fits have a 17%–20% better resolution.

Δz

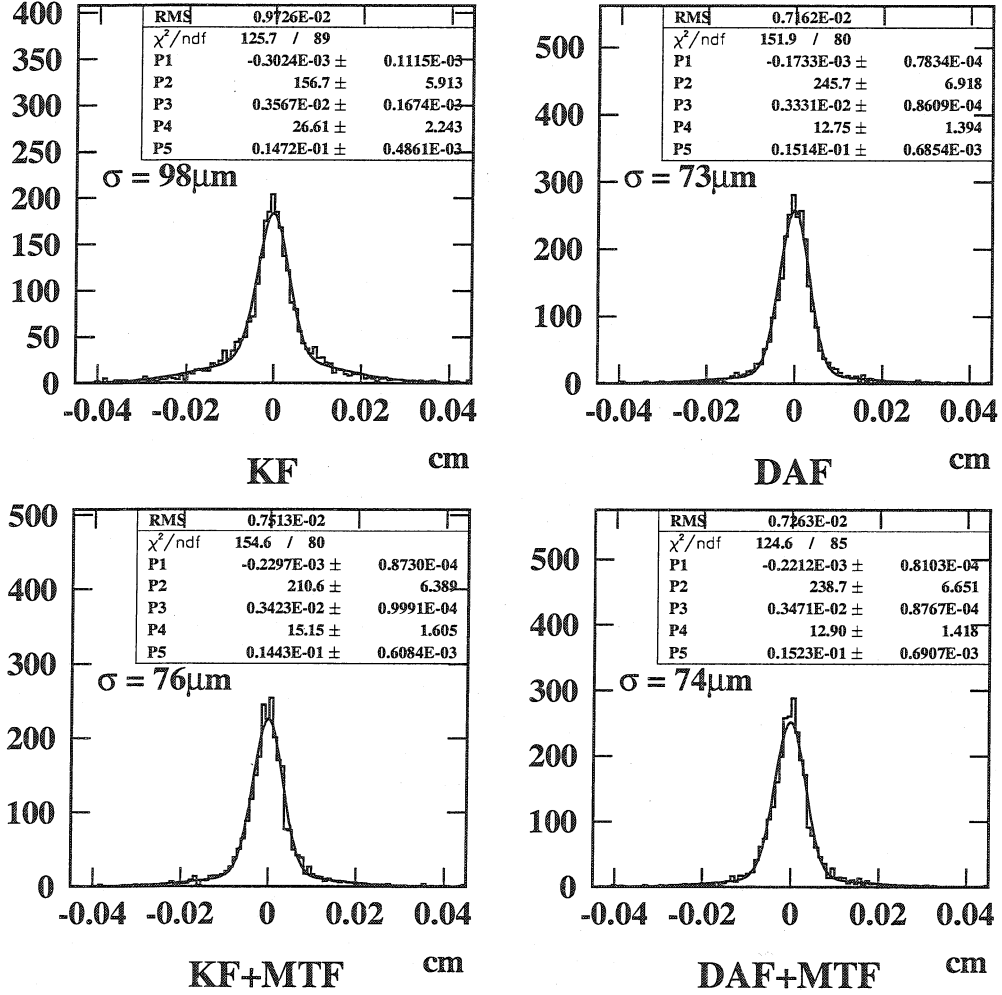


Figure 6.7: Longitudinal impact parameter resolution at the τ vertex. The core resolution about $35\mu\text{m}$ in all four cases. The standard deviation of the tails is of the order of $150\mu\text{m}$. Like in the case of the transverse impact parameter resolution, the tails are largest for the Kalman Filter. The standard deviation of the mixture is $98\mu\text{m}$ for the KF, and between $73\mu\text{m}$ and $76\mu\text{m}$ for the DAF and the MTF. Relative to the KF, the DAF and the MTF have about 25% better longitudinal impact parameter resolution.

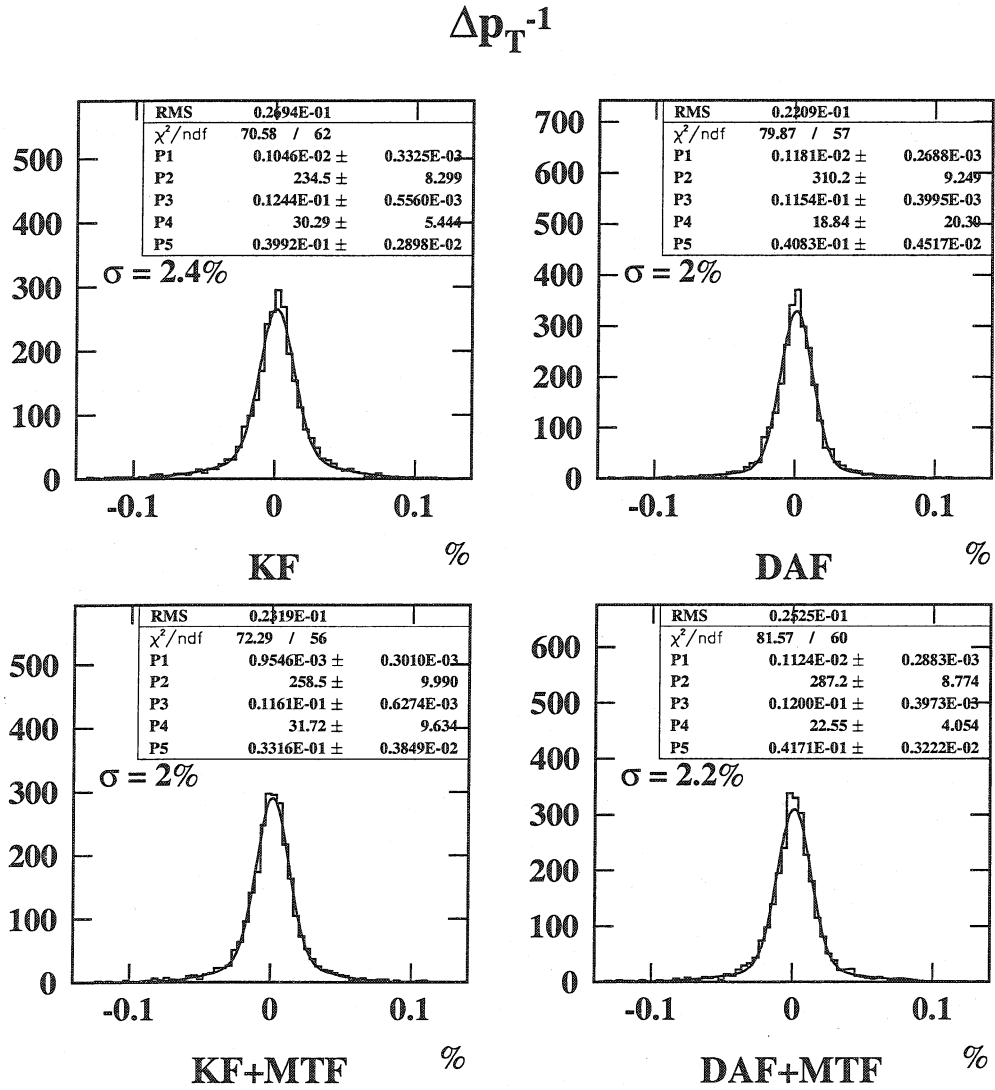


Figure 6.8: Transverse momentum resolution at the τ vertex. The core resolution is between 1.1% and 1.2% in all four cases and statistically compatible. The overall resolution is 2.4% for the KF, 2% for the DAF and for the MTF run after the KF and after DAF and 2.2% for the MTF run after the DAF. Relative to the KF, the DAF and the MTF have about 10%–17% better transverse momentum resolution.

Fig. 6.8 shows the relative p_T^{-1} resolution at the τ vertex. The core resolution is between 1.1% and 1.2% in all four cases. The tail resolution is about 3.3%–4.1%. The standard deviation of the mixture is 2.4% for the KF, and it is between 2% and 2.2% for the DAF and the MTF. Relative to the KF, the DAF and the MTF have about 10%–17% better transverse momentum resolution.

Both DAF and MTF achieve significantly better resolutions than the KF, but there is no visible improvement by the MTF if it is run after the DAF.

6.5 Track parameter pulls and χ^2 -probability in 3-prong τ jets

In order to compare the quality of the track reconstruction, the pulls and the χ^2 probabilities are compared.

Fig. 6.9 shows the pulls of the transverse impact parameter at the τ vertex. The RMS of the pull of the KF is 3 and is the worst of all. The RMS of the pulls for the DAF and the MTF are between 2 (MTF) and 2.2 (DAF), which are still too large but significantly better than the one of the KF. The fitted standard deviations of the Gaussian mixture are slightly better, with 1.8 for the MTF run after the DAF, 2 for the DAF and 2.9 for the KF. The DAF has about 30% better pulls than the KF, and the MTF has up to 38% better pulls. The pulls of the MTF run after the DAF are 10% better than the ones of the DAF.

Fig. 6.10 shows the pulls of the longitudinal impact parameter. In all cases the RMS of the pulls is significantly larger than 1. The RMS is worst for the Kalman Filter (3.2) and is between 1.9 (MTF after DAF) and 2.1 (DAF) for the others. The fitted standard deviations of the Gaussian mixture are 3.2 for the KF, 1.9 for the DAF and 1.7 for the MTF run after the DAF. This means that pulls are about 40% better for the DAF and 47% better for the MTF run after the DAF. The MTF run after the DAF has still 10% better pulls than the DAF itself.

Fig. 6.11 shows the pulls of the inverse transverse momentum. Like in the other cases they are worst for the KF, with a RMS of 2.4. They are best for the MTF run after the DAF with a RMS of 1.8. The MTF run after the KF has comparable pulls with respect to the MTF run after the DAF. The MTF has pulls which are about 10% better than the one of the DAF.

Fig. 6.12 shows the χ^2 -probability distributions. The χ^2 -probability distribution of the KF shows a huge peak at zero and has a mean value of 0.24. This peak can be significantly reduced if a MTF is run after the KF, with a mean of 0.43, but still with a peak at zero. The DAF does not have a sharp peak at zero like the KF, but has a small hump instead. Its mean value is 0.43. The MTF run after the DAF has by far the best χ^2 probability distribution among the four. It is flat and has a mean value of 0.48.

6.6 Conclusion

The results can be summarized in the following way:

- About 75%–80% of the selected 3-prong τ jets are reconstructed with three tracks globally. About 85% of the selected 3-prong τ s are reconstructable for the choice of minimum number of eight hits for all three tracks.
- The DAF has a slightly better efficiency than the KF.

- A fit with a mixture of two Gaussians for the residuals shows that the sigmas of both Gaussians are about the same for the DAF and the KF. When looking at the sigma of the mixture, the DAF has about 20%–25% better impact parameter resolutions than the KF which is due to the reduction of the tails. The p_T resolution is about 10%–17% better than the one of the KF.
- Using the MTF after the KF or the DAF does not improve the resolution achieved by the DAF. It is equal to the DAF for the impact parameter resolutions and it is the same or slightly worse in one case for the p_T resolution, but still better than the one of the KF.
- The pulls are too large in all cases. The KF has the pulls with the highest RMS. The errors are best estimated by the MTF, showing an improvement of about 10% with respect to the DAF for all track parameters. With respect to the KF, the pulls of the MTF are between 38% and 47% better for the impact parameters, and they are about 22% better for the p_T^{-1} .
- The χ^2 -probability distribution is nearly perfect for the MTF run after the DAF. It is the worst for the KF with a sharp peak at zero. The MTF run after the KF can improve the χ^2 probability significantly, making it comparable to the one of the DAF.

Finally, the fraction of merged clusters due to close or overlapping tracks was briefly analysed for the barrel part of the CMS Tracker .

Fig. 6.13 shows the average number of simulated and reconstructed hits for the barrel pixel and first silicon layers. Only tracks in reconstructed 3-prong τ jets which are candidates for MTF smoothing have been selected. Hits in overlap regions and stereo detectors are not counted. While the average number of simulated hits is three for all layers, the mean number of reconstructed hits is slightly above two in the first pixel barrel layer and between two and a half and three in the second pixel layer. There are three reconstructed hits on average for all other outer barrel layers. Therefore clusters merge most likely in the first two pixel layers, and virtually all of the tracks are well separated in the silicon part of the CMS Tracker.

$$\Delta x/\sigma(x)$$

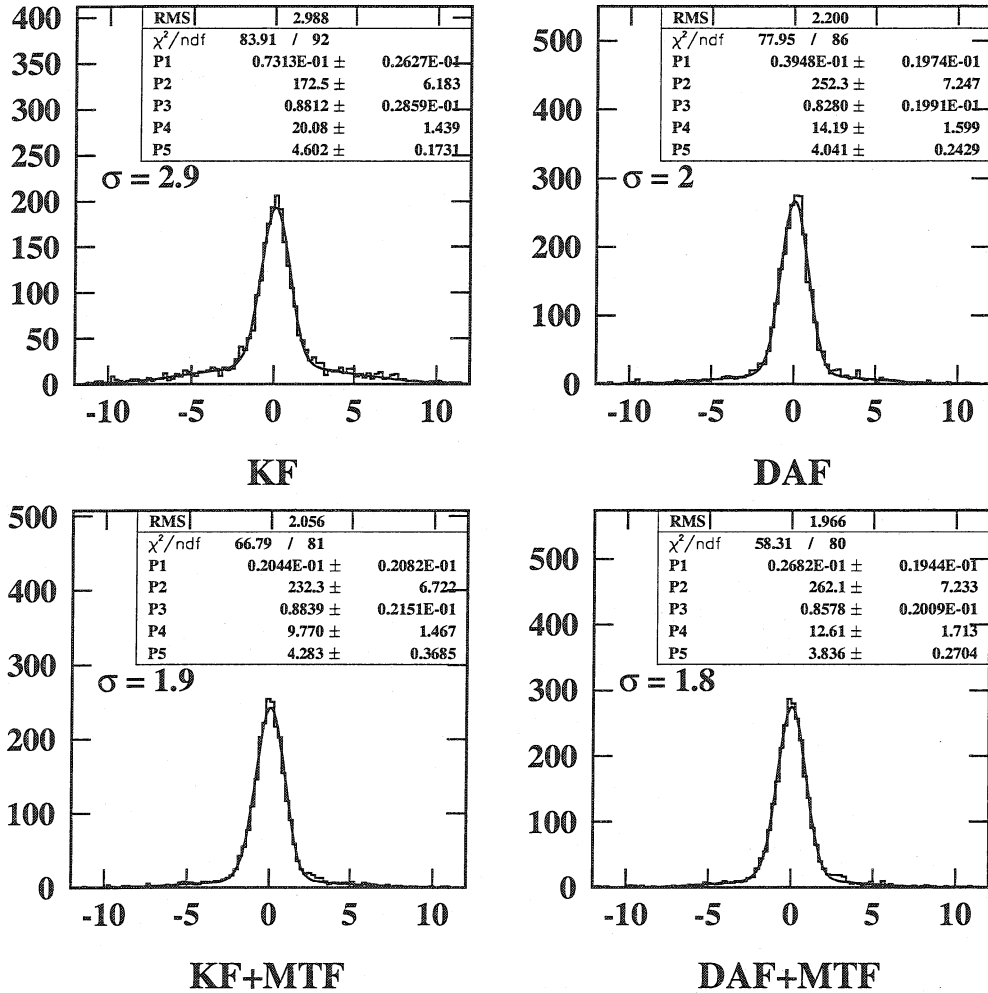


Figure 6.9: Pull distributions of the transverse impact parameter. The RMS of the pull of the KF is 3 and is the worst of all. The RMS of the pulls for the DAF and the MTF are between 2 (MTF) and 2.2 (DAF), which is still too large but significantly better than the one of the KF. The fitted standard deviations of the Gaussian mixture are slightly better than the RMS. The DAF has about 30% better pulls than the KF, and the MTF has up to 38% better pulls. The pulls of the MTF run after the DAF are 10% better than the ones of the DAF.

$$\Delta z/\sigma(z)$$

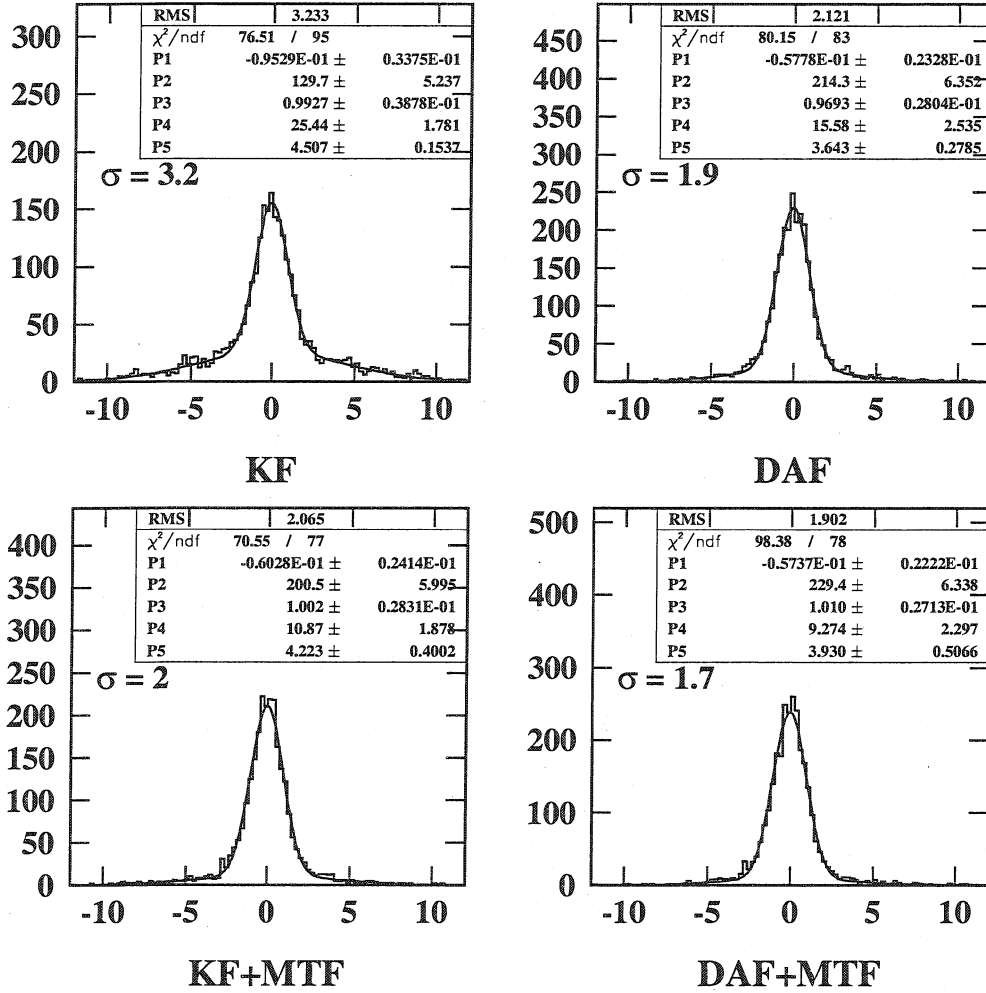


Figure 6.10: Pull distributions of the longitudinal impact parameter. As for the transverse impact parameter pulls, their RMS is significantly larger than 1. The best is the MTF run after the DAF, with a RMS of 1.9 compared to 3.2 for the KF. The total standard deviation obtained by a fit of a mixture of two Gaussians gives 3.2 for the KF, 1.9 for the DAF and about 1.7 for the MTF after the DAF. The improvement is between 40% and 47% percent with respect to the KF.

$$\Delta p_T^{-1}/\sigma(p_T^{-1})$$

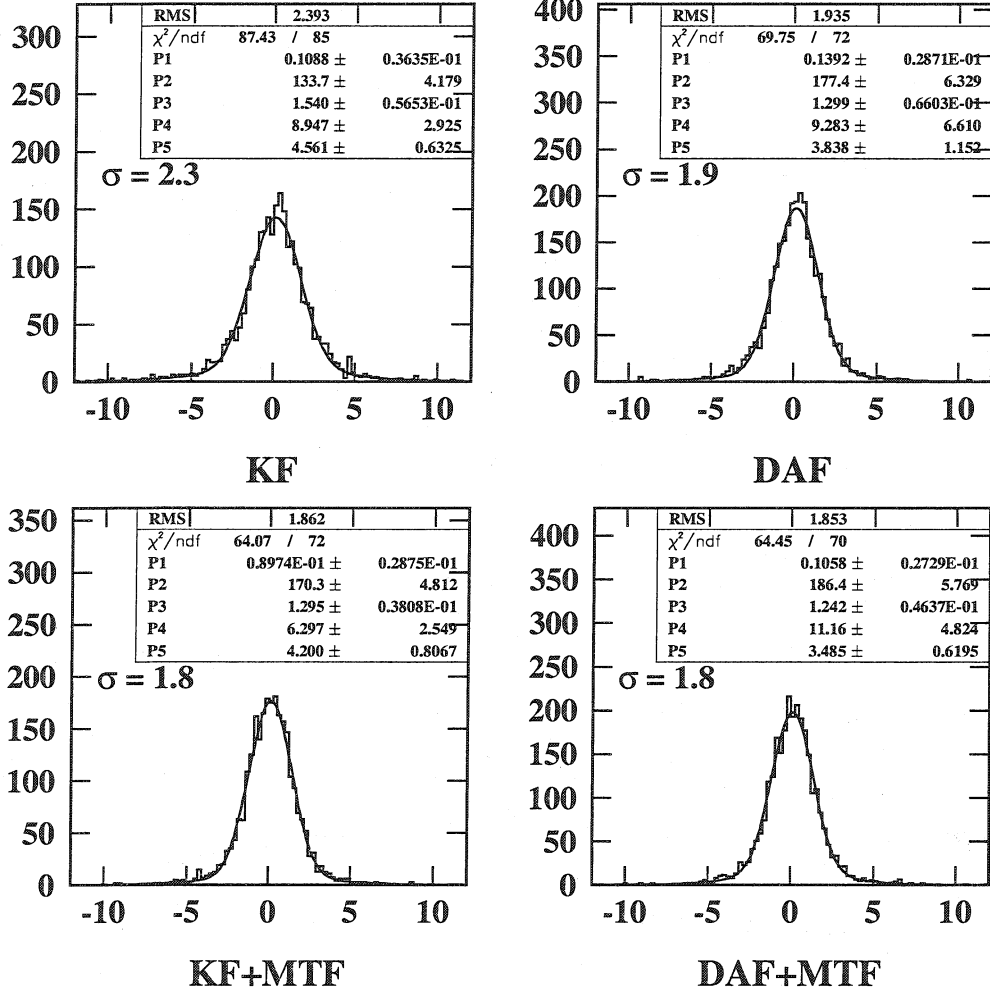


Figure 6.11: Pull distributions of the inverse transverse momentum. The KF is the worst with a RMS of about 2.4. The MTF which is run after the DAF has pulls with a RMS of about 1.8. The pulls of the DAF have a RMS of about 2, which is 10% worse than the MTF.

χ^2 probability

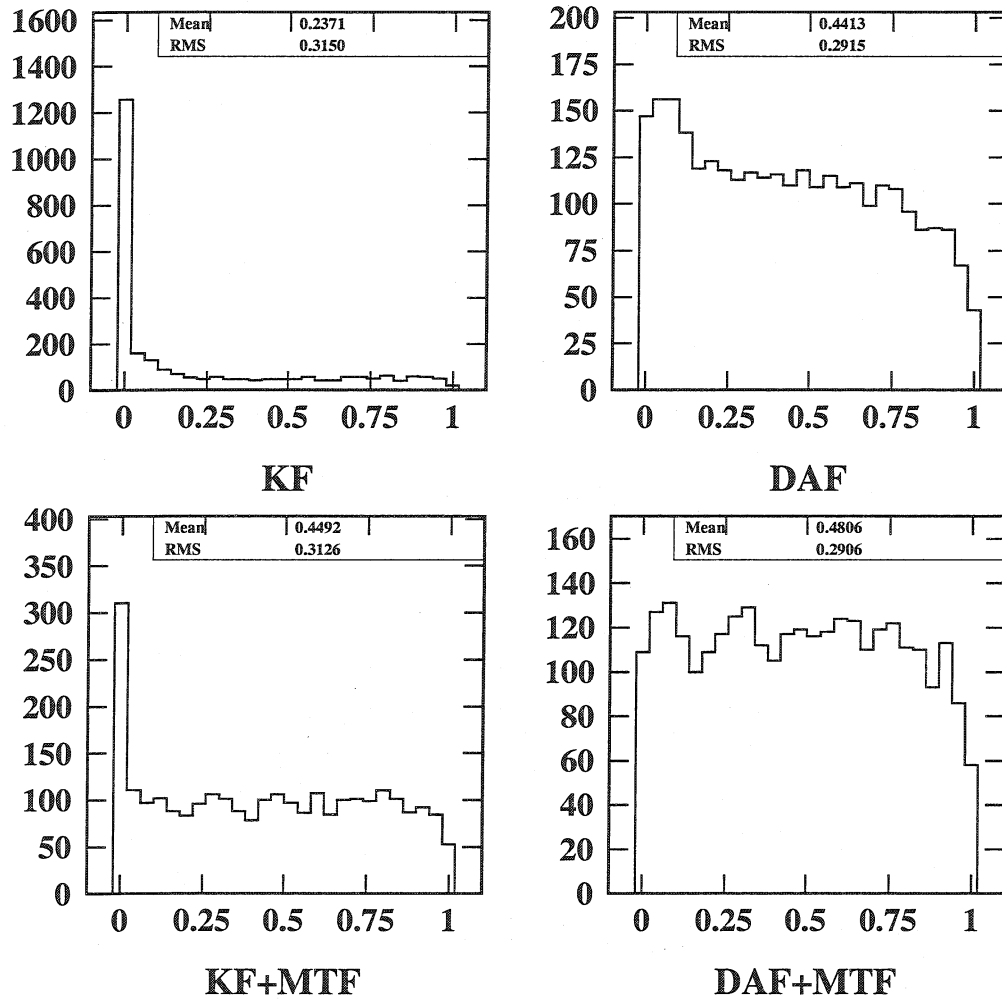


Figure 6.12: χ^2 -probability distributions for the KF, the DAF, the MTF run after the KF and the MTF run after the DAF. The χ^2 -probability distribution of the KF is worst, having a sharp peak at zero and a mean value of 0.23. The MTF run after the DAF has the best χ^2 -probability distribution among the four. The distribution is flat and has a mean value of 0.48. Both the DAF and the MTF run after the KF have better distributions than the KF but still have a small peak at or around zero.

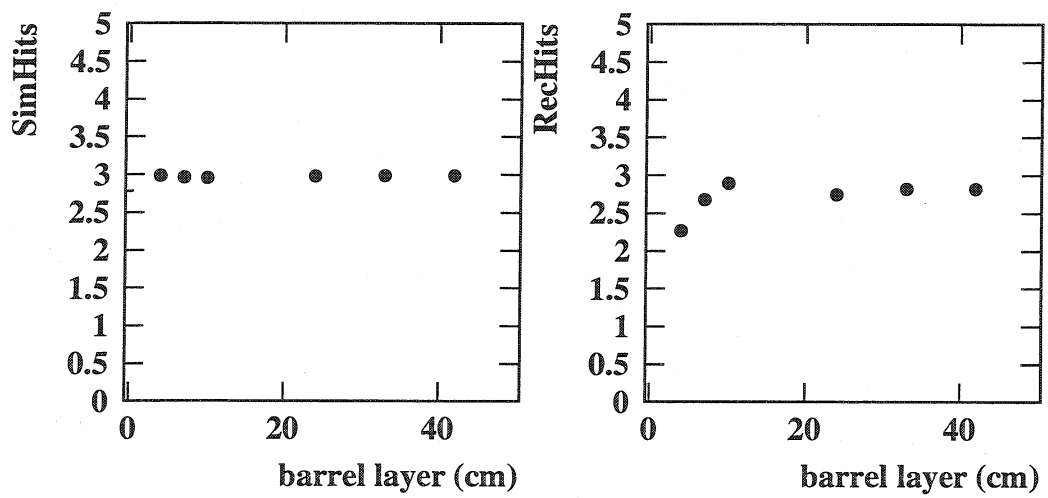


Figure 6.13: Average number of simulated and reconstructed hits per layer in a τ jet. While the number of simulated hits is always three (one per simulated track), the number of reconstructed hits is less than three in the barrel pixels. It is between 2 and 2.5 in the innermost barrel layer, and between 2.5 and 3 for the second pixel barrel layer.

7 Alignment

Alignment of tracking detectors is an essential step in the task of track reconstruction. Without proper alignment it is impossible to reach the ultimate position and momentum resolution. Besides finding the alignment of the tracking detectors it is also important to constantly monitor the alignment and to update the alignment constants whenever required.

In the CMS Tracker this task is particularly challenging because of the large number of independent silicon sensors, about 20000, and their high resolution, between 10 and 40 microns. In order to exploit fully the resolution of the tracker the sensors must be aligned to a precision significantly better than their intrinsic resolution. This precision can only be achieved in an alignment procedure based on charged tracks, since the expected precision from the mechanical mounting and laser beam alignment is significantly worse than the intrinsic sensor resolution.

The large number of alignment parameters implies a very large number of tracks needed to align the tracker. Even at nominal LHC luminosity and the expected trigger rate of the order of 10 Hz for high p_T muons the time to accumulate the necessary statistics is of the order of a week. It is therefore important to make efficient use of the track information in the alignment algorithm.

A unified framework for the simultaneous unbiased estimation of the alignment parameters of several detectors is presented, which fully uses the track information. The method is a straightforward extension of the standard Kalman Filter [20]. Since the dimension of the parameter space can be large the occasional convergence to local minima cannot be excluded. Strong empirical evidence will be presented below that this actually occurs. This problem is solved by introducing annealing, i.e. by gradually turning on the weights of the observations in the course of the estimation process. The resulting algorithm closely resembles the Deterministic Annealing Filter invented for robust track reconstruction in the presence of noise and ambiguities [23].

For the sake of simplicity the proposed alignment procedure has been studied using a simple track and detector model, neglecting material effects and pattern recognition issues. It should be stressed, however, that the method is completely general and can be applied with any kind of track and detector model.

7.1 Formalism

The problem of aligning a track detector can be stated in the following generic way. The observation \mathbf{m} recorded by the detector depends both on the vector of track parameters (track state) \mathbf{p} of the track crossing the detector and on the vector of alignment parameters (alignment state) \mathbf{a} . The alignment state may contain both shifts and rotations. The dependence can be written down in a generalized

measurement equation:

$$\mathbf{m} = \mathbf{f}(\mathbf{p}, \mathbf{a}) + \boldsymbol{\epsilon}$$

where $\boldsymbol{\epsilon}$ is the vector of measurement errors. The covariance matrix \mathbf{V} of $\boldsymbol{\epsilon}$ is assumed to be known.

The function \mathbf{f} may be linear or non-linear in either argument. In the presence of rotations in the alignment state, \mathbf{f} is non-linear in \mathbf{a} . While the track parameters \mathbf{p} are different from event to event, the alignment parameters \mathbf{a} of a detector are common for all tracks. In the linear approximation the measurement equation can be written as:

$$\mathbf{m} = \mathbf{c} + \mathbf{H}\mathbf{p} + \mathbf{D}\mathbf{a} + \boldsymbol{\epsilon},$$

where \mathbf{H} and \mathbf{D} are the Jacobians:

$$\mathbf{H} = \frac{\partial \mathbf{f}}{\partial \mathbf{p}}(\mathbf{p}_0, \mathbf{a}_0)$$

$$\mathbf{D} = \frac{\partial \mathbf{f}}{\partial \mathbf{a}}(\mathbf{p}_0, \mathbf{a}_0).$$

It is now assumed that there is a predicted track state \mathbf{p}_0 along with its covariance matrix \mathbf{C}_0 , as well as a predicted alignment state \mathbf{a}_0 along with its covariance matrix \mathbf{E}_0 . The recipe for updating both the track state and the alignment state can be derived in a manner which is analogous to the derivation of the standard Kalman Filter. The resulting update formulas are:

$$\mathbf{a}_1 = \mathbf{a}_0 + \mathbf{E}_0 \mathbf{D}^T \mathbf{W} [\mathbf{m} - \mathbf{f}(\mathbf{p}_0, \mathbf{a}_0)]$$

$$\mathbf{E}_1 = \mathbf{E}_0 - \mathbf{E}_0 \mathbf{D}^T \mathbf{W} \mathbf{D} \mathbf{E}_0$$

for the alignment state plus covariance matrix, and

$$\mathbf{p}_1 = \mathbf{p}_0 + \mathbf{C}_0 \mathbf{H}^T \mathbf{W} [\mathbf{m} - \mathbf{f}(\mathbf{p}_0, \mathbf{a}_0)]$$

$$\mathbf{C}_1 = \mathbf{C}_0 - \mathbf{C}_0 \mathbf{H}^T \mathbf{W} \mathbf{H} \mathbf{C}_0$$

for the track state plus covariance matrix. The following auxiliary matrix needs to be computed:

$$\mathbf{W} = [\mathbf{V} + \mathbf{H} \mathbf{C}_0 \mathbf{H}^T + \mathbf{D} \mathbf{E}_0 \mathbf{D}^T]^{-1}$$

Note that the update formulas for the track and alignment parameters decouple into two separate ones. We will come back to this point in section 7.4. In a similar way it is also possible to decouple the estimation of shift and rotation parameters, although this is tantamount to neglecting the correlations between the two (see subsection 7.5).

Information about \mathbf{a} is accumulated continually, increasing after each track. In addition, at each step in the alignment procedure the full covariance matrix \mathbf{E} of \mathbf{a} is known, which can be used as a criterion for stopping the alignment procedure. The convergence of \mathbf{a} clearly depends on the precision of its starting value. It can be taken from mechanical measurements, from laser beam alignment, or from a previous alignment with tracks.

Smoothing is done in the usual way, by running two filters in opposite directions and combining the results on each detector surface. Both in the forward and in the backward filter the alignment states are not updated, but the knowledge of the current alignment state is used for the update of the track state vector (see above). The alignment parameters are updated in the smoothing step, which combines the predictions from the forward and the backward filter with the measurements. This means that at each update of \mathbf{a} both the full information about the track and the current alignment of all other detectors enter into the update.

The predicted alignment states contain the current knowledge of the alignment parameters, based on the tracks already processed. The predicted track states, on the other hand, are based on the observations from the current track only, but depend also on the current alignment. As long as the alignment parameters are not known to sufficient precision the predicted track states are therefore biased, especially if several detectors are aligned simultaneously. As a consequence, it may happen that alignment parameters converge to a suboptimal solution (local minimum) of the alignment problem. This can be prevented by introducing annealing in a manner similar to the Deterministic Annealing Filter developed for robust track reconstruction [23]. This means that the observations in all detectors to be aligned are downweighted by a large factor in the beginning. This factor is then gradually decreased, until it reaches 1 after a prescribed number of tracks. Strong empirical evidence will be presented that this annealing procedure solves entirely the problem of convergence to local minima.

7.2 Design and implementation for a test-beam like setup

The object oriented design of the track and alignment reconstruction is strongly related to the existing one of the track reconstruction environment for the CMS Tracker in ORCA [32] (see also sections 2 and 3). Wherever possible, existing classes (e.g. the `RecHit`) are used directly, or provided base class interfaces are implemented (e.g. the `DetLayer` or the `DetUnit`). Therefore the alignment specific algorithms are isolated in specific classes for the minimization (fitter, smoother), similar to the way it is done in track reconstruction only for the CMS Tracker.

7.2.1 Overview

The task of track reconstruction (with or without alignment) can be split in four major components:

- a tracker geometry
- an alignment geometry
- tracker hits and detector readout
- a track reconstructor

This global layout (modularity) allows that one can change the track reconstructor while keeping the geometry the same or vice versa. Alternatively, one might want to study a given geometry first with simulated hits and then switch to real test beam data, only by changing the way hits are produced.

7.2.2 Tracker geometry

A common base class `Tracker` defines an interface for accessing specific tracker geometries (CMS or test beam) in a unique way. The requirement of a tracker geometry is to provide access to its sub-components, enabling the reconstruction of tracks. Therefore the `Tracker` base class interface prescribes a method that allows the access to the `DetLayers`. A second method allows the access to all sensitive volumes in a tracker — the `DetUnits`. In the case of a test beam a `DetUnit` most likely represents also the surface of a `DetLayer`. The object model with its basic components can be seen in Figure 7.1.

7.2.3 Alignment geometry

The geometrical structure used for track finding (`Tracker` geometry) is different from the geometrical structure for aligning components. An alignment geometry has to provide access to the specific alignment structures and, in the case of a demand, act on the current `Tracker` geometry by moving and rotating parts. This mechanism can be used for both aligning and misaligning, if one wants to introduce misalignment into an existing `Tracker` geometry.

In the case of this simple test beam example the alignment components coincide with the ones used in the track reconstruction — each `DetUnit` has its associated `AlignableDetUnit` and thus can be aligned. The current `AlignmentStates` (an `AlignmentState` is composed of `AlignmentParameters` and `AlignmentErrors`) are stored within the `AlignableDetUnits`. In addition, the alignment geometry has to know about the reference detectors and the alignable detectors (see Figure 7.2).

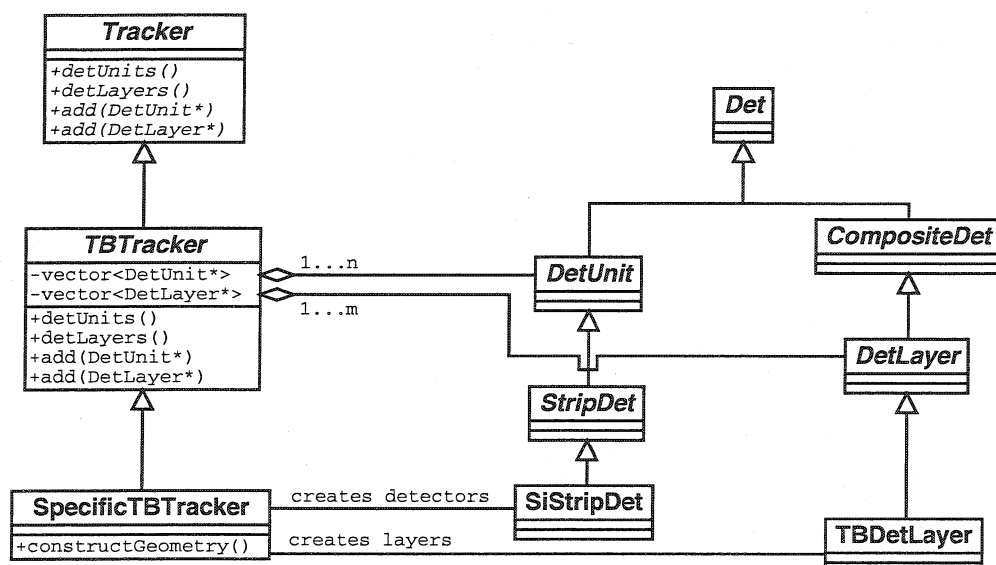


Figure 7.1: Object model of the Tracker geometry. The TBTracker class implements the interface of the abstract base class Tracker and keeps control of the whole geometry. A SpecificTBTracker reads configuration files and constructs DetLayers and DetUnits which are then added to the TBTracker.

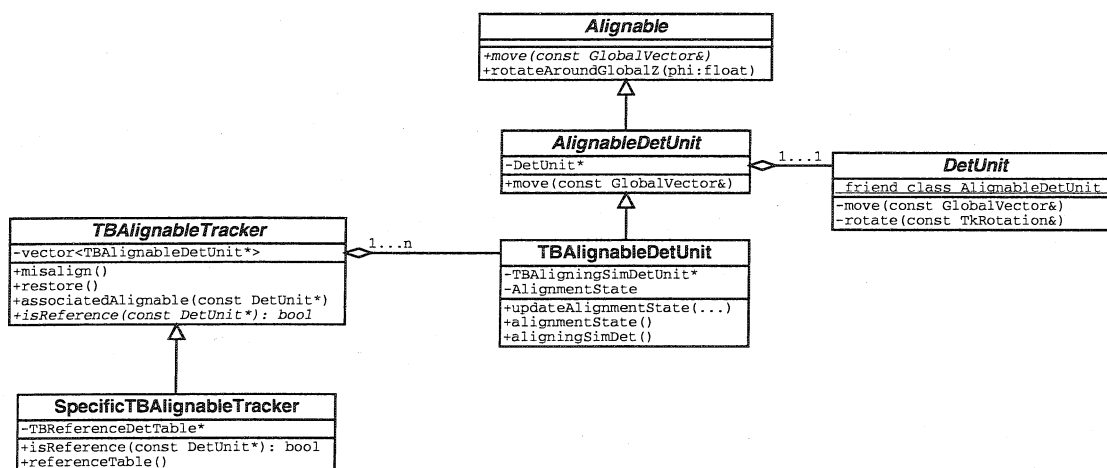


Figure 7.2: Object model of the alignment geometry. In the case of the test beam like setup the alignment geometry is simple, as the structure of the track reconstruction geometry matches the structure of the alignment geometry.

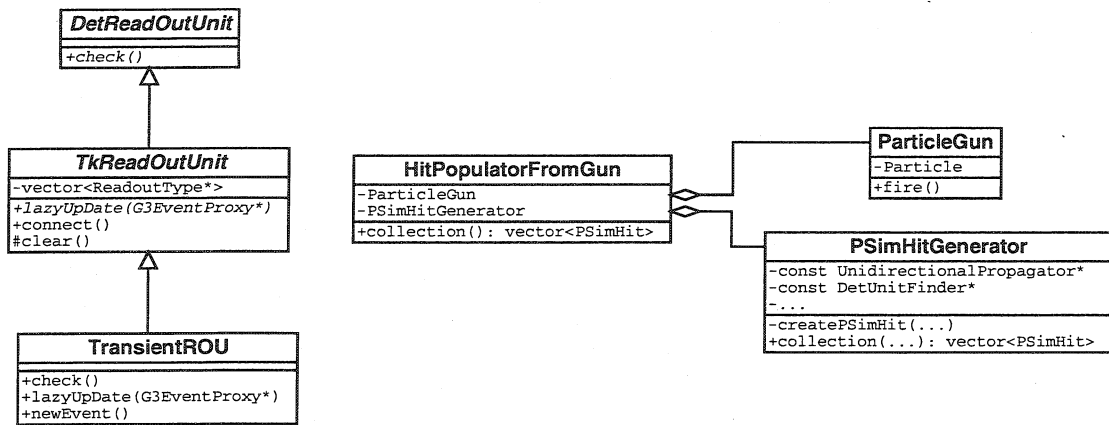


Figure 7.3: Class collaboration diagram for the detector readout and the HitPopulatorFromGun. In the case of a simulation, the Digis (signal per strip) of a detector have to be created from a detector response simulation. This is done by a Digitizer, which creates Digis based on the information provided by SimHits.

7.2.4 Tracker hits and detector readout

Each DetUnit is connected to a detector readout. It reads out the Digis (digi = signal of one strip or pixel) of a detector event by event. These Digis are then used by a Clusterizer in order to create the RecHits — the fundamental measurements needed by the track reconstruction. However, when simulating tracks one does not have a “real” readout and one needs to simulate the detector response. In ORCA this is done by a Digitizer which creates Digis from the information it gets from simulated hits (the SimHits) and thus fills the readout. In addition, the readouts have to be reset at the beginning of each event, cleaning up the Digis and SimHits from the previous event. In the simulation experiment a HitPopulatorFromGun is used in order to create SimHits (see Figure 7.3).

7.2.5 Track reconstructor with alignment

It is required to reconstruct tracks on the base of the available RecHits taking into account misalignment of detectors. Therefore the reconstruction of tracks together with the estimation of alignment parameters is performed by a dedicated high level object. The track reconstructor contains a track SeedGenerator and a TrajectoryBuilder with alignment. The SeedGenerator creates initial starting values for potential track candidates, e.g. by combining pairs of RecHits. The AlignmentTrajectoryBuilder creates a Trajectory starting from a seed and estimates both track and alignment parameters. An overview is given in Figure 7.4.

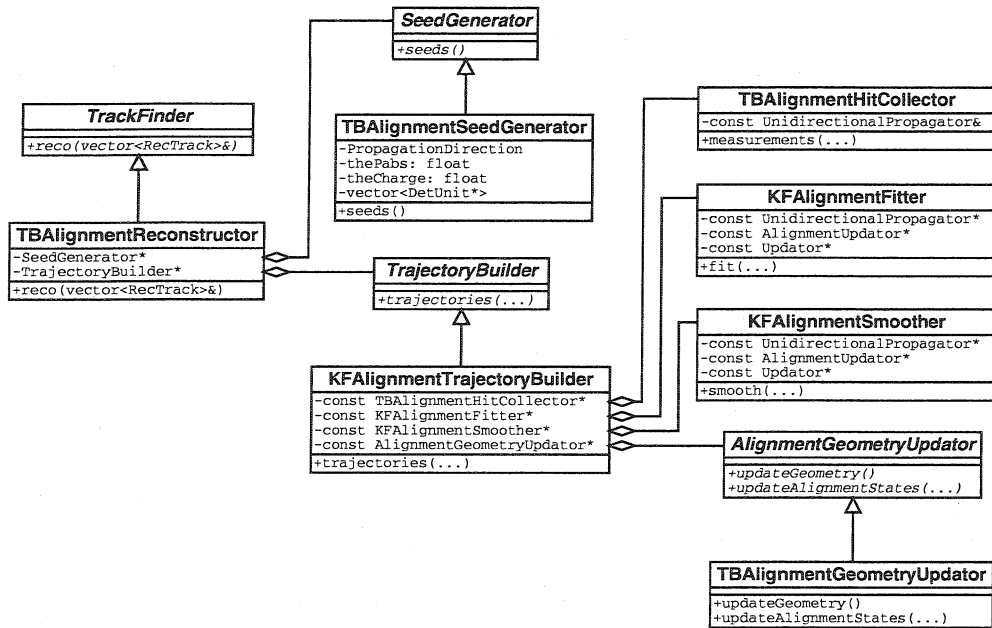


Figure 7.4: Object model of the track reconstructor. The most important components are the SeedGenerator and the TrajectoryBuilder.

7.3 Results of a simulation experiment

The algorithm has been tested and verified in a simulation experiment. The setup is a somewhat simplified model of a typical test beam configuration.

7.3.1 The detector model

A layout has been used with five consecutive pairs of x - y silicon strip detectors along the global z -(beam-)axis. Each detector has a pitch of $60\mu\text{m}$ and a strip length of 6 cm. The RecHits are gaussian smeared SimHits with a standard deviation corresponding to the pitch of the detectors ($\text{pitch}/\sqrt{12}$). The observation along the strip is always set to 0 in the local frame of the detector (center of the strip); its standard deviation is set to the strip length divided by $\sqrt{12}$.

The first and the last pair of detectors define the reference frame, assuming that their true positions and orientations are known. The three intermediate pairs of detectors are misaligned (see Figure 7.5).

7.3.2 Global and local frame

The global coordinate system or frame has been chosen such that the global z -axis points into the direction of the beam (see Figure 7.5). The definition of the local

7.3.3 Track simulation and track model

The beam is generated at $z = 0$ covering the full $6\text{cm} \times 6\text{cm}$ area of sensitive detector volumes. Positively charged muons with a momentum of 100 GeV are simulated. The beam direction is defined by two angles, ϑ and φ , where

$$\tan \vartheta = \frac{p_T}{p_z} \quad \text{and} \quad \tan \varphi = \frac{p_y}{p_x}$$

ϑ has been generated in the range $0 < \vartheta < 0.014$ in order to avoid tracks being perfectly parallel with z , and φ has been generated in the range $0 \leq \varphi < 2\pi$.

In the absence of a magnetic field the track model is very simple. The following set of track parameters has been used for a fixed z :

$$\mathbf{p} = \begin{pmatrix} t_x \\ t_y \\ x \\ y \end{pmatrix}, \quad \text{with} \quad t_x = \frac{dx}{dz} = \frac{p_x}{p_z} = \tan \vartheta \cos \varphi, \quad t_y = \frac{dy}{dz} = \frac{p_y}{p_z} = \tan \vartheta \sin \varphi$$

Track propagation from $z = z_1$ to $z = z_2$ is then described by a simple matrix:

$$\mathbf{p}|_{z=z_2} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ z_2 - z_1 & 0 & 1 & 0 \\ 0 & z_2 - z_1 & 0 & 1 \end{pmatrix} \mathbf{p}|_{z=z_1}$$

7.3.4 The misalignment model

Misalignment can be seen as the difference between the assumed ideal position and orientation of a given detector with respect to its true parameters in global space. Therefore each transformation from the local detector coordinate system into global space and vice versa has errors if the assumed ideal parameters are not the true ones. Misalignment is introduced by moving and rotating the detector planes before shooting particles, and they are restored to the assumed ideal (but now wrong) positions and rotations before the start of reconstructing tracks. While the local position of a hit on a detector stays unchanged (the strip of the hit does not change), its global position will be wrong as the transformation from the local to the global frame is wrong. The task of the alignment procedure to estimate the relative offset and rotation with respect to the assumed position and rotation.

Out of the six possible alignment parameters — three shifts and three rotations — two shifts and one rotation in the plane of the detector (2-dimensional space) are simulated and estimated, those three being the ones to which track reconstruction is most sensitive. Figure 7.7 illustrates the two shifts and the rotation.

The transformation of a point $\mathbf{m}(x, y) \rightarrow \mathbf{m}'(x', y')$ from the global frame into the global misaligned frame is therefore:

$$\begin{aligned}\mathbf{m}'(x', y') &= R(\Delta\phi)(\mathbf{m}(x, y) - \mathbf{s}_{\text{global}}) \\ &= R(\Delta\phi) \left(\mathbf{m}(x, y) - \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right)\end{aligned}$$

with

$$\mathbf{s}_{\text{global}} = \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

being the shift in $\Delta x, \Delta y$ and $R(\Delta\phi)$ a small rotation around z .

The following ranges for the shifts Δx and Δy and the rotation $\Delta\phi$ in the global frame have been chosen for the simulation:

$$\begin{aligned}\text{shift in global } x\text{-direction:} & \quad -0.2 \text{ cm} < \Delta x < 0.2 \text{ cm} \\ \text{shift in global } y\text{-direction:} & \quad -0.2 \text{ cm} < \Delta y < 0.2 \text{ cm} \\ \text{rotation around global } z\text{-axis:} & \quad -0.02 \text{ rad} < \Delta\phi < 0.02 \text{ rad}\end{aligned}$$

For each run the alignment parameters are randomly generated assuming a flat distribution in the above ranges.

It should be noted that instead of estimating $(\Delta x, \Delta y, \Delta\phi)$ directly a slightly different set of alignment parameters has been chosen :

$$\mathbf{a} = \begin{pmatrix} \Delta x' \\ \Delta y' \\ \Delta\phi \end{pmatrix}, \quad \text{with} \quad \begin{pmatrix} \Delta x' \\ \Delta y' \end{pmatrix} = R(\Delta\phi) \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

where the shift rotated by $\Delta\phi$ is estimated rather than the simulated global shift. In the following it will be referred to

$$\mathbf{s}' = \begin{pmatrix} \Delta x' \\ \Delta y' \end{pmatrix}$$

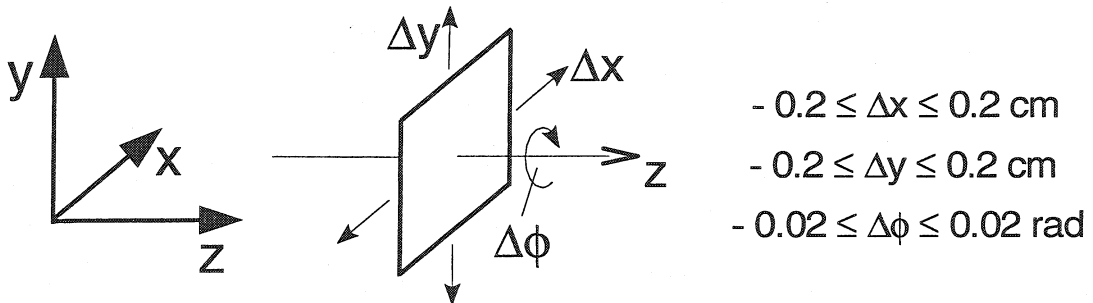


Figure 7.7: Illustration of shifting and rotating the planes for misalignment.

as the global rotated shift. The reason for this choice will be explained after the derivation of the Jacobians.

The measurement model can then be formulated in the most general way:

$$\begin{aligned}\mathbf{m}_{\text{local}} &= \mathbf{f}(\mathbf{p}, \mathbf{a}) + \boldsymbol{\epsilon} \\ &= \mathbf{T}(\phi)_{\text{global} \rightarrow \text{local}} \mathbf{R}(\Delta\phi) [\mathbf{P}\mathbf{p}_{\text{global}} - \mathbf{s}_{\text{global}}] + \boldsymbol{\epsilon} \\ &= \mathbf{R}(\Delta\phi) \mathbf{P}\mathbf{p}_{\text{local}} - \mathbf{T}(\phi)\mathbf{s}' + \boldsymbol{\epsilon}\end{aligned}$$

where \mathbf{m} is the observation (hit) in the local frame, \mathbf{p} is the track state ($\mathbf{p}_{\text{global}}$ in the global and $\mathbf{p}_{\text{local}}$ in the local frame), \mathbf{s}' is the global rotated shift, and $\boldsymbol{\epsilon}$ is the measurement error in the local frame. The last expression of the measurement equation will be used for deriving the essential Jacobians.

With this choice of track parameters, the matrix \mathbf{P} is a simple projection:

$$\mathbf{P} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

\mathbf{T} is the transformation from the global to the “ideal” local frame; in this case it is either the identity or a rotation by 90 degrees around the global z -axis. \mathbf{R} is a small rotation around the z -axis in addition to the “ideal” one:

$$\mathbf{R} = \begin{pmatrix} \cos(\Delta\phi) & \sin(\Delta\phi) \\ -\sin(\Delta\phi) & \cos(\Delta\phi) \end{pmatrix}$$

7.3.5 The estimation procedure

For the estimation procedure it is assumed that the assignment of hits to tracks has already been done.

Track fitting is performed by a Kalman Filter plus smoother in order to get optimal predicted track states \mathbf{p}_0 in all detectors. The current state \mathbf{a}_0 of the alignment parameters in all detectors is used in the fit of the track. Note that the alignment parameters are estimated independently in all detectors. In order to avoid cluttering the notation the index to the detector is dropped.

Using the misalignment model described above, the derivative matrices \mathbf{H} and \mathbf{D} can be computed without difficulty in any particular detector:

$$\begin{aligned}\mathbf{H} &= \frac{\partial \mathbf{f}}{\partial \mathbf{p}}(\mathbf{p}_0, \mathbf{a}_0) = \mathbf{R}\mathbf{P} \\ \mathbf{D} &= \frac{\partial \mathbf{f}}{\partial \mathbf{a}}(\mathbf{p}_0, \mathbf{a}_0) = (-\mathbf{T} \mid \mathbf{R}'\mathbf{P}\mathbf{p}_0)\end{aligned}$$

where \mathbf{R}' is the derivative of the current rotation matrix:

$$\mathbf{R}' = \begin{pmatrix} -\sin(\Delta\phi_0) & \cos(\Delta\phi_0) \\ -\cos(\Delta\phi_0) & -\sin(\Delta\phi_0) \end{pmatrix}$$

and \mathbf{p}_0 is the predicted track state in the local system.

It can be seen that the third column of \mathbf{D} depends on the rotation angle $\Delta\phi$, thus making the equation non-linear in $\Delta\phi$. By choosing the rotated global shifts $\Delta x'$, $\Delta y'$ as alignment parameters, it is avoided that also the other elements of \mathbf{D} depend on $\Delta\phi$. In order to simplify the notation the rotated global shifts will henceforth be denoted by Δx and Δy .

From this the matrix \mathbf{W} is computed, using the current annealing factor $\alpha^{(k)}$ which depends on the current track number k :

$$\mathbf{W} = [\alpha^{(k)}\mathbf{V} + \mathbf{H}\mathbf{C}_0\mathbf{H}^T + \mathbf{D}\mathbf{E}_0\mathbf{D}^T]^{-1}$$

The update of the track state and of the alignment state then proceeds as described in section 7.1. Three annealing schedules have been investigated:

Schedule	$\alpha^{(1)}$	$\alpha^{(n)}$	$\alpha^{(k)}$
A	1	1	1
B	10000	10000	10000
C	10000	1	$10000 \frac{n-k}{n-1}$

Schedule A is the standard Kalman Filter, schedule B is a Kalman Filter which effectively uses only the predictions of the tracks from the reference detectors, and schedule C is a Deterministic Annealing Filter [23] with a geometric cooling schedule. In all cases the initial values of the alignment parameters have been taken to be zero with sufficiently large errors.

7.3.6 Sensitivity and convergence of alignment parameters

Not all alignment parameters can be reliably determined as not all of the observations are equally affected by shifting and rotating the detector. The sensitivity of the alignment parameters with respect to the observations is given by the matrix $\mathbf{E}_0\mathbf{D}^T\mathbf{W}$. The last factor \mathbf{W} is dominated by the information content of the observations. Therefore in a detector measuring only x a shift Δy in y cannot be estimated very reliably because the sensitivity is dominated by the information content of the y -measurement. In this case this information content is smaller by a factor of 1 million [square of (length /pitch)] than the information content of the x -measurement. It should be noted, however, that this effect is irrelevant in track reconstruction as long as the y -coordinate of the prediction is sufficiently precise so that the precision of the x -coordinate is not spoiled by the rotation correction.

The different sensitivities of the alignment parameters are clearly visible if their evolution is plotted as a function of the number of tracks. As an example, Figure 7.8 shows the development of Δx , Δy , and $\Delta\phi$ in an x -detector, for a run with 2000 tracks. The behaviour depends on the annealing schedule. Convergence of Δx is

satisfactory with all schedules, although with schedule B it is somewhat slower. Estimates of Δy are biased with all schedules. The differences between the three schedules can be seen most clearly in the convergence of $\Delta\phi$. Schedule A converges very quickly, but is off by about 0.2 mrad. Schedule B converges much more slowly and is off by 0.4 mrad after 2000 tracks. Schedule C is clearly the best, being off by less than 20 μ rad after 2000 tracks in this particular run.

7.3.7 Precision of the estimated alignment parameters

Due to statistical fluctuations a single run cannot determine the absolute performance of a single schedule. To this end 2000 runs with random misalignment configurations and performing alignment over 2000 tracks for each run have been generated, which means that a total amount of 4 million tracks is generated and reconstructed per annealing schedule. The resulting histograms of residuals for schedule C are shown in Figure 7.9.

Figure 7.10 shows the standard deviations of the residuals (estimated alignment parameters minus true ones) for all annealing schedules. The shifts are shown only for the precise coordinates (Δx in x -detectors, Δy in y -detectors). Schedule A suffers from occasional convergence to local minima, whereas schedule B only uses information from the reference detectors. Schedule C (annealing) gives the best results in all cases. With runs of 2000 tracks each, the standard deviations of the shifts are about 0.6 μ m, and the standard deviations of the rotations angles are below 50 μ rad.

Finally, a check whether the computed errors on the alignment parameters correspond to the actual spread around the true values was done. The resulting histograms of standardized residuals (pulls) for schedule C are shown in Figure 7.11.

Figure 7.12 shows the standard deviations of the pulls for all annealing schedules. With schedule C, the standard deviations are indeed reasonably close to 1, whereas they are far too large with schedule A and far too small with schedule B. The mean values are compatible with 0 in all cases.

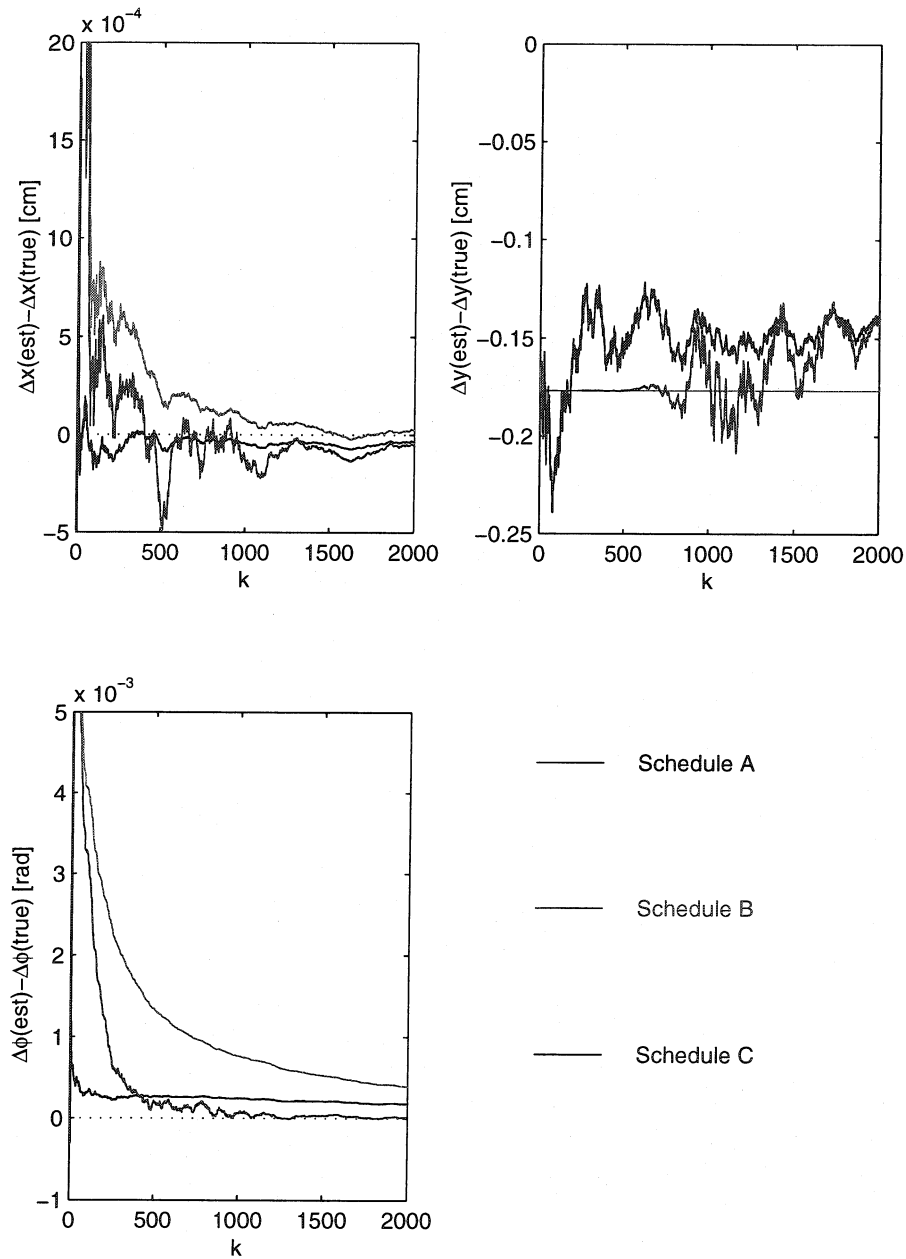


Figure 7.8: Development of estimated alignment parameters as a function of the track number for an x -detector. While convergence of Δx (shift in precise coordinate, scale in micron) is satisfactory for all three schedules, there is no convergence for Δy (shift in unprecise coordinate, scale in mm). The convergence of $\Delta \phi$ (scale in mrad) is best for schedule C.

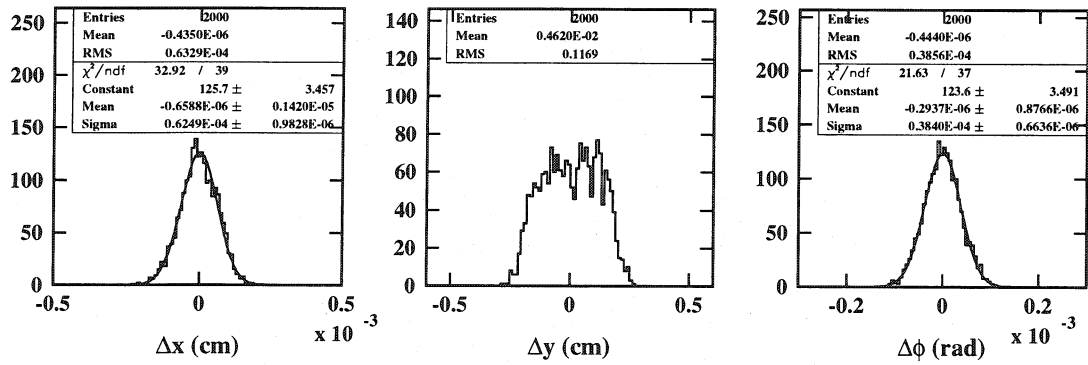


Figure 7.9: Residual histograms of the three alignment parameters for an x -detector and for annealing schedule C. As one can see, the shift along the unprecise coordinate cannot be estimated reliably.

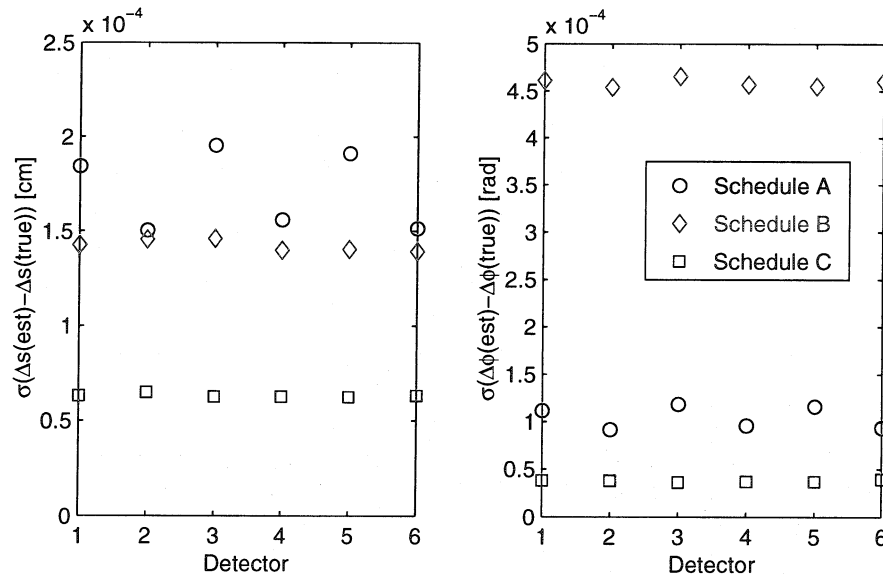


Figure 7.10: Standard deviation of the residuals (estimated alignment parameters minus true ones). Left hand side: shift in precise coordinate, right hand side: angle of rotation. The scale of the residuals of the shifts is in micron. The shifts are of the order of about $0.6\ \mu\text{m}$ for schedule C, and $1.5\text{--}2\ \mu\text{m}$ for schedule A and B. The scale of the residuals of the rotation angle $\Delta\phi$ is tenths of a millirad. Schedule C achieves resolutions of about $50\ \mu\text{rad}$, schedule A about $100\ \mu\text{rad}$ and schedule B about $450\ \mu\text{rad}$.

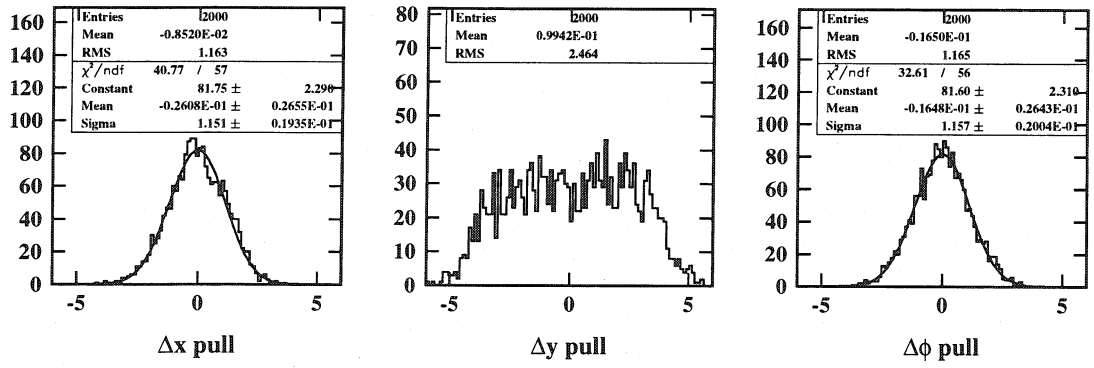


Figure 7.11: Pull histograms of the three alignment parameters for an x -detector and for annealing schedule C. The standard deviation of the pulls is reasonably close to 1 for the shift Δx and for the rotation $\Delta\phi$, whereas it is too large for the shift in Δy .

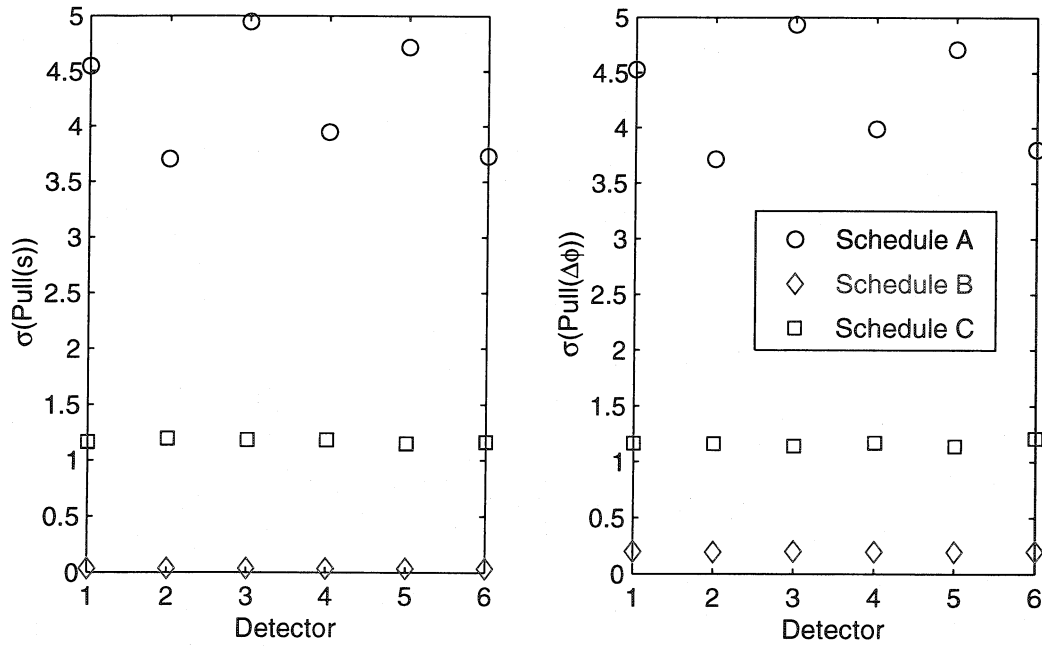


Figure 7.12: Standard deviation of the standardized residuals (pulls). Left hand side: shift in precise coordinate, right hand side: angle of rotation. Only in the case of schedule C the calculated errors of the reconstructed alignment parameters correspond reasonably good to the actual spread around the true values. In the case of schedule A these errors are too small, in the case of schedule B they are too big.

7.3.8 Track reconstruction after alignment

In order to quantify the quality of the different alignment strategies, for each misalignment configuration 2500 tracks are generated, out of which the first 2000 tracks are used for estimating the alignment parameters. Then the current alignment state is “frozen” and for the last 500 tracks where only the track parameters are estimated. For each annealing schedule this is done with 1000 different random misalignment configurations, resulting in a total amount of 2.5 million simulated tracks out of which 0.5 million reconstructed tracks are used for track analysis.

7.3.9 Resolution of the track parameters

For the resolution in the four track parameters $(t_x, t_y, x, y)^T$ the following results are obtained:

Schedule	x [μm]	y [μm]	$dx/dz \times 10^{-4}$	$dy/dz \times 10^{-4}$
no misalignment	14.45 ± 0.01	14.56 ± 0.01	1.372 ± 0.001	1.372 ± 0.001
A	14.55 ± 0.01	14.64 ± 0.01	1.37 ± 0.001	1.371 ± 0.001
B	16.13 ± 0.02	16.19 ± 0.02	1.448 ± 0.001	1.444 ± 0.001
C	14.45 ± 0.01	14.59 ± 0.01	1.37 ± 0.001	1.371 ± 0.001

Although alignment schedule C achieves best results for the estimation of the alignment parameters, this does not translate into a significantly better track parameter resolution compared with results from alignment schedule A. In both cases the parameter resolutions are equivalent to the ones achieved without misaligning the detectors. The differences in track position resolution between schedules A (Kalman Filter) and C (Deterministic Annealing Filter) are less than one tenth of a micron in favour of schedule C, and there is no difference at all in direction resolution. Track parameter resolution after strategy B is about 10% worse compared to the resolution obtained without misalignment (or strategy A and C). An example of two resolution distributions of the x coordinate is given in Figure 7.13.

7.3.10 Pulls of the track parameters

For the standard deviations of the pulls of the four track parameters $(t_x, t_y, x, y)^T$ the following results are obtained:

Schedule	x	y	dx/dz	dy/dz
no misalignment	1 ± 0.001	1 ± 0.001	1.002 ± 0.001	1.001 ± 0.001
A	1.007 ± 0.001	1.006 ± 0.001	1 ± 0.001	1.001 ± 0.001
B	1.009 ± 0.001	1.016 ± 0.002	1.001 ± 0.001	1.002 ± 0.001
C	1 ± 0.001	1.002 ± 0.001	1 ± 0.001	1.001 ± 0.001

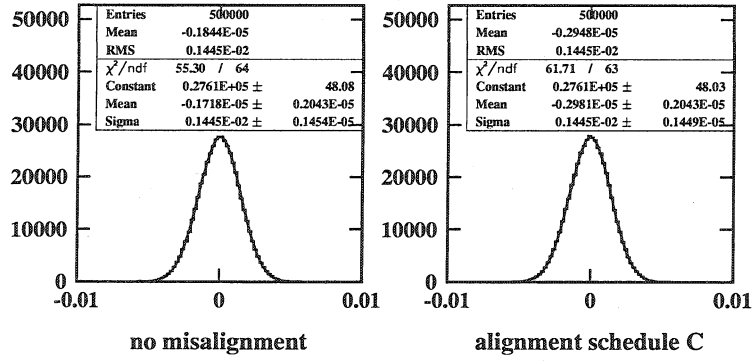


Figure 7.13: Histograms of the position resolution x after track reconstruction without initial misalignment of detectors (left) and track reconstruction after alignment with annealing schedule C (right). As can be seen, full track parameter resolution is achieved with schedule C.

In all cases, the calculated errors of the reconstructed track parameters are compatible with the actual spread around the true values. The pull distributions are mean-value free with a standard deviation of 1 in all cases. The bad error estimation of alignment parameters for alignment schedules A and B seem to have almost no effect on the error estimation of the track parameters. Examples of two pull distributions of the x coordinate is given in Figure 7.14.

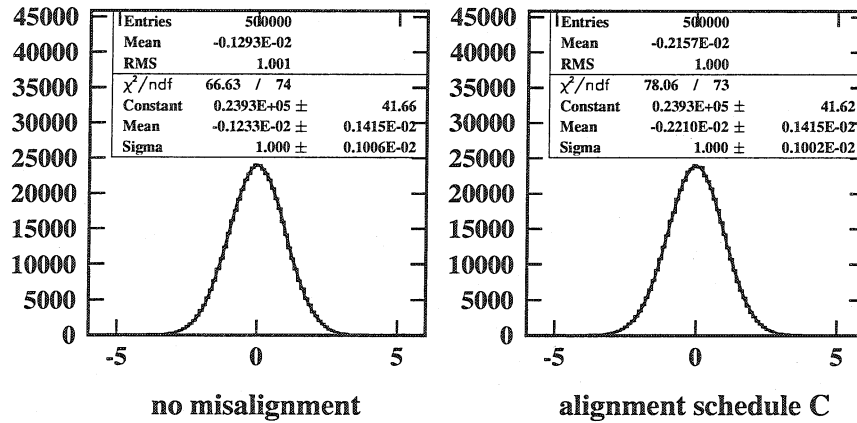


Figure 7.14: Histograms of the pulls of the x position coordinate after track reconstruction without initial misalignment (left picture) and alignment schedule C. The histograms are mean-value free and the standard deviations are exactly 1 in both cases.

7.4 Summary and conclusion

The results can be summarized as follows:

- A method was presented for the simultaneous estimation of alignment parameters of several detectors with respect to a set of reference detectors in parallel

to track reconstruction. The reference detectors are needed in order to fix the reference frame with respect to which the other detectors are aligned. This method (“alignment with tracks”) has been formulated in the Kalman Filter formalism which is already known for track reconstruction.

- Three different alignment strategies have been investigated (annealing schedules). From a simulation experiment it can be concluded that a Kalman Filter with annealing (schedule C) gives quantitatively and qualitatively the best results for the estimation of the alignment parameters. The necessary number of tracks and the annealing schedule can be tailored to the specific properties of the setup.
- Differences in track reconstruction after using alignment schedule A or C are negligible. In both cases the precision of track reconstruction is as good as for a perfectly aligned tracker. The performance of alignment schedule B can be expected to get worse with a more realistic track model using multiple scattering.
- The algorithms have been implemented and tested in ORCA for a test-beam like setup. The formulation of alignment reconstruction in the Filter formalism has made it easy to implement the algorithms in the existing logic and framework for track reconstruction in ORCA. In fact, this was one of the main motivations to formulate “alignment with tracks” in the Kalman Filter formalism for track reconstruction.

It therefore can be concluded that:

- It is possible to estimate simultaneously track and alignment parameters with the Kalman Filter.
- The method can be used with any kind of track model, provided that the magnetic field is known to sufficient precision. The result is a kind of “local” alignment, giving positions and orientations of a set of detector elements with respect to a fixed set of reference detectors.
- The algorithms have been verified in a simple and fully controlled simulation environment. This can be considered as a small but essential step towards an application in a realistic environment, e.g. for the CMS Tracker. Global alignment of the CMS Tracker is of course much more complex, requiring relative alignment of detector elements which are never crossed by one and the same track. In addition, the effects of wrong hit-to-track assignment and material effects on the convergence of the alignment parameters have to be investigated. This, however, is beyond the scope of this work.
- In the simulation experiment alignments with methods A and C are both precise enough to achieve full (intrinsic) resolution in track parameters.

- The decoupling of the update formulas into one update for track parameters and one for alignment parameters has additional advantages. They can be used together or separately in the following way:
 - One can update both track and alignment parameters and still have a fully valid track with consistent error estimates after smoothing.
 - For a certain number of tracks one does only update the alignment parameters without using the track for further analysis after smoothing (standard alignment procedure).
 - After alignment over a certain number of tracks one freezes the current alignment states. For the following track reconstruction one uses the generalized update formulas derived in section 7.1 in order to take into account properly results from the preceding alignment. This is an alternative way of taking into account results from the alignment, without moving hits or detectors “by hand”. The shifts and rotations will enter statistically correctly via the alignment parameters in the update of the track state.

7.5 Addendum to the formalism

In the case of the estimation of all six alignment parameters — three shifts and three rotations — it might be of advantage to separate explicitly the shifts \mathbf{s} and the rotations \mathbf{r} in the linear approximation:

$$\mathbf{m} = \mathbf{f}(\mathbf{p}, \mathbf{s}, \mathbf{r}) + \boldsymbol{\epsilon}$$

In the linear approximation \mathbf{m} can be expressed as

$$\mathbf{m} = \mathbf{c} + \mathbf{H}\mathbf{p} + \mathbf{A}\mathbf{s} + \mathbf{B}\mathbf{r} + \boldsymbol{\epsilon}$$

with

$$\begin{aligned} \mathbf{H} &= \frac{\partial \mathbf{f}}{\partial \mathbf{p}}(\mathbf{p}_0, \mathbf{s}_0, \mathbf{r}_0) \\ \mathbf{A} &= \frac{\partial \mathbf{f}}{\partial \mathbf{s}}(\mathbf{p}_0, \mathbf{s}_0, \mathbf{r}_0) \\ \mathbf{B} &= \frac{\partial \mathbf{f}}{\partial \mathbf{r}}(\mathbf{p}_0, \mathbf{s}_0, \mathbf{r}_0) \end{aligned}$$

With $\text{cov}(\mathbf{p}_0) = \mathbf{C}_0$, $\text{cov}(\mathbf{s}_0) = \mathbf{S}_0$ and $\text{cov}(\mathbf{r}_0) = \mathbf{U}_0$ and

$$\mathbf{W} = [\mathbf{V} + \mathbf{H}\mathbf{C}_0\mathbf{H}^T + \mathbf{A}\mathbf{S}_0\mathbf{A}^T + \mathbf{B}\mathbf{U}_0\mathbf{B}^T]^{-1}$$

the update formulas are:

a) for the track parameters \mathbf{p} :

$$\begin{aligned}\mathbf{p}_1 &= \mathbf{p}_0 + \mathbf{C}_0 \mathbf{H}^T \mathbf{W} [\mathbf{m} - \mathbf{f}(\mathbf{p}_0, \mathbf{s}_0, \mathbf{r}_0)] \\ \mathbf{C}_1 &= \mathbf{C}_0 - \mathbf{C}_0 \mathbf{H}^T \mathbf{W} \mathbf{H} \mathbf{C}_0\end{aligned}$$

b) for the shifts \mathbf{s} :

$$\begin{aligned}\mathbf{s}_1 &= \mathbf{s}_0 + \mathbf{S}_0 \mathbf{A}^T \mathbf{W} [\mathbf{m} - \mathbf{f}(\mathbf{p}_0, \mathbf{s}_0, \mathbf{r}_0)] \\ \mathbf{S}_1 &= \mathbf{S}_0 - \mathbf{S}_0 \mathbf{A}^T \mathbf{W} \mathbf{A} \mathbf{S}_0\end{aligned}$$

c) for the rotations \mathbf{r} :

$$\begin{aligned}\mathbf{r}_1 &= \mathbf{r}_0 + \mathbf{U}_0 \mathbf{B}^T \mathbf{W} [\mathbf{m} - \mathbf{f}(\mathbf{p}_0, \mathbf{s}_0, \mathbf{r}_0)] \\ \mathbf{U}_1 &= \mathbf{U}_0 - \mathbf{U}_0 \mathbf{B}^T \mathbf{W} \mathbf{B} \mathbf{U}_0\end{aligned}$$

8 Summary and outlook

The Deterministic Annealing Filter (DAF), the Multi Track Filter (MTF) and the Kalman Filter (KF) have been compared on different event topologies in the CMS Tracker. Some of these event types are considered to be among the most difficult ones for track reconstruction in CMS.

All three track finders have a solid implementation in ORCA, and they have been extensively tested. For the purpose of algorithm verification the ParticleGun was created in order to have a fully controlled simulation environment for testing and validating.

As expected, it has turned out that there are event topologies where Kalman Filter and Deterministic Annealing Filter produce comparable results, and that there are event topologies where the DAF systematically gives better results than the KF, especially in “difficult” environments for track reconstruction. The MTF does not improve the precision of the DAF in dense bundles of tracks, but at least gives better error estimates and a nearly perfect χ^2 -distribution. This is important in the subsequent vertex fit.

“Alignment with tracks” was formulated as an extended Kalman Filter and therefore put on a formal mathematical base. This was motivated for by the fact that the Kalman Filter for track reconstruction is implemented and well understood in ORCA. With this new approach the implementation of a track fit with parallel estimation of alignment parameters in ORCA is straightforward. Basic class-components can be directly used and existing design principles can be followed.

- The implementations of the track fits were verified using the ParticleGun in a fully controlled environment. In the limit of Gaussian errors of the measurements, DAF and MTF produce statistically correct results. The effect of non-Gaussian measurement errors on the estimated errors was investigated as well as the behaviour of the DAF and the KF in a controlled noisy environment. In such noisy environments the DAF achieves qualitatively and a quantitatively better results than the KF.
- In realistic simulation environments of isolated tracks including multiple scattering and energy loss, both KF and DAF produce compatible result within the statistical limits. The DAF has a higher efficiency of reconstructing pions. The performance of reconstruction of isolated muon tracks (mainly low p_T) was evaluated for the $B_{(d)s}^0 \rightarrow \mu^+ \mu^-$ channel, a channel particularly sensitive to the precision of the reconstructed tracks for discovery at CMS (see 5). The precision of the mass reconstruction of the $B_{(d)s}^0$ ($5.369 \text{ GeV}/c^2$) is the same for the Kalman Filter and the Deterministic Annealing Filter, both for the low and the high luminosity scenario at CMS. It is about 50% worse with respect to previous publications [26], with a core mass resolution of $40 \text{ MeV}/c^2$ instead of $26 \text{ MeV}/c^2$. The degradation can be explained by a different layout of the

CMS Tracker, a better understanding of the material budget (50% increase) and a more realistic description of the magnetic field with respect to previous studies.

- In b -jets the DAF has higher track reconstruction efficiencies, lower fake rates and qualitatively and quantitatively better reconstructed high p_T tracks ($> 15 \text{ GeV}/c$) than the KF. The secondary vertex finding efficiency and the b -tagging efficiency is better when using the DAF than the KF, and it is substantially better for b -jets with high E_T where the tracks are collimated.
- The performance of the MTF was evaluated for τ -jets from $500 \text{ GeV}/c^2$ Higgs decays, a topology which is supposed to be particular difficult at CMS. Results are compared with the ones of DAF and KF. The DAF has 10%–20% better resolutions and 20%–30% better pulls than the KF. The MTF has resolutions comparable to the DAF, but the pulls are 10% better, and the χ^2 is nearly perfect. About 80% of the τ -jets are reconstructed with three tracks.
- “Alignment with tracks” was implemented in ORCA and was verified using a simplified test-beam setup. However, the method is completely general and can be used with any track and detector model. A geometrical cooling schema produced qualitatively and quantitatively best results. The advantage of this approach is that it smoothly follows the logic of the standard track reconstruction in ORCA, which allows a straightforward implementation.

For the track reconstruction in general a better material description and a propagator which is capable to track through inhomogenous magnetic fields is desirable. It might be of interest to have a dedicated pattern recognition (track finding) for the DAF in order to profit as much as possible from the internal hit-to-track association capability of the DAF. This could significantly speed up the entire reconstruction with the DAF, as less time needs to be spent for combinatorics. Although advanced track reconstruction methods for the CMS Tracker are a long term project, the available implementations are already sufficiently mature to be used in parallel to the standard KF. Especially vertex reconstruction of collimated jets profits a lot when using the DAF or the MTF. It is therefore safe to conclude that there is a vital interest in further exploring advanced track fitting methods.

We have also seen that the distributions of the track parameters sometimes have substantial tails. A mixture of two Gaussians seems to be a very appropriate model in this case. It would be very interesting to develop a vertex fit which is capable of handling such mixtures, either as a Gaussian-sum filter or as an adaptive version of the usual linear filter.

References

- [1] A. Strandlie, J. Wroldsen, R. Frühwirth, and B. Lillekjendlie. Particle tracks fitted on the Riemann sphere. *Computer Physics Communications*, 131:95–108, 2000.
- [2] *The Tracker Project - Technical Design Report*. CERN, 1998.
- [3] <http://cmsinfo.cern.ch/welcome.html>.
- [4] <http://cmsdoc.cern.ch/cms.html>.
- [5] *The Compact Muon Solenoid - Letter of Intent*. CERN, 1992.
- [6] *The Compact Muon Solenoid - Technical Proposal*. CERN, 1994.
- [7] <http://lhc-new-homepage.web.cern.ch/lhc-new-homepage/>.
- [8] <http://www.cern.ch>.
- [9] <http://cmsdoc.cern.ch/cms/physics/btau/management/top/btau.html>.
- [10] <http://cmsdoc.cern.ch/tk.html>.
- [11] *Addendum to the CMS Tracker TDR*. CERN, 2000.
- [12] <http://www.cern.ch/~duccio/layout>.
- [13] *The Magnet Project - Technical Design Report*. CERN, 1997.
- [14] http://cmsdoc.cern.ch/~afrey/material_budget/.
- [15] J. D. Jackson. *Classical Electrodynamics*. John Wiley and Sons, 1998.
- [16] R. Frühwirth, M. Regler, R.K. Bock, H. Grote, and D. Notz. *Data Analysis Techniques for High-Energy Physics*. Cambridge University Press, 2000.
- [17] <file:/cern/pro/src/geant321/doc/gedoc>.
- [18] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns*. Addison-Wesley, 1995.
- [19] Thomas Erlner. *UML Das Einsteigerseminar*. bhv Verlag, 2000.
- [20] R. Frühwirth. Application of Kalman filtering to track and vertex fitting. *Nucl. Instrum. Meth.*, A262:444, 1987.
- [21] R. Frühwirth. Track fitting with non-Gaussian noise. *Comp. Phys. Comm.*, 100:1–16, 1997.
- [22] M. Ohlsson, C. Peterson, and A. Yuille. Track finding with deformable templates — the elastic arms approach. *Computer Physics Communications*, 71:77, 1992.

- [23] R. Frühwirth and A. Strandlie. Track fitting with ambiguities and noise: a study of elastic tracking and non-linear filters. *Comp. Phys. Comm.*, 120:197, 1999.
- [24] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1, 1977.
- [25] R. Frühwirth and A. Strandlie. Adaptive multitrack fitting. *Comp. Phys. Comm.*, 133:34–42, 2000.
- [26] A. Nikitenko, A. Starodumov, and N. Stepanov. Observability of $B_{(d)s}^0 \rightarrow \mu\mu$ decay with the CMS detector. *CMS Note*, 039, 1999.
- [27] Pascal Vanlaer. private communications, CMS Note in preparation.
- [28] R. Kinnunen and A. Nikitenko. Study of $H(\text{SUSY}) \rightarrow \tau\tau \rightarrow l^\pm + \tau jet + E_t^{miss}$ in CMS. *CMS Note*, 106, 1997.
- [29] R. Kinnunen and D. Denegri. The $H(\text{SUSY}) \rightarrow \tau\tau \rightarrow h^\pm + h^\mp + X$ channel, its advantages and potential instrumental drawbacks. *CMS Note*, 037, 1999.
- [30] R. Kinnunen and A. Nikitenko. Study of $H(\text{SUSY}) \rightarrow \tau\tau \rightarrow 2\tau$ jets in CMS. *CMS Note in preparation*.
- [31] <http://cmsdoc.cern.ch/~anikiten/taustudy>.
- [32] <http://cmsdoc.cern.ch/orca>.

Lebenslauf

Matthias WINKLER

Engelsdorferweg 13
A-3730 Eggenburg

Persönliche Daten

geboren am 29.09.1973 in Eggenburg, Österreich
Familienstand verheiratet, röm.-kath.
Wehrdienst 10/1992 – 05/1993

Berufliche Erfahrung

seit 08/1999 CERN in Genf; Entwicklung von objektorientierter
Software in C++ zur Mustererkennung und
Statistischen Datenanalyse
10/1998 – 08/1999 Institut für Hochenergiephysik der Österreichischen
Akademie der Wissenschaften in Wien; Software-
entwicklung und Datenanalyse von Siliziumdetektoren
Betriebssysteme UNIX (Linux, Solaris), MS Windows

Ausbildung

06/2002 Doktorat der Technischen Naturwissenschaften an der
TU Wien
10/1998 Diplom-Ingenieur der Technischen Physik an der
TU Wien; 1995/1996 Austauschjahr mit der
ETH Lausanne (EPFL), Schweiz
05/1992 Deutsche Reifeprüfung an der Deutschen Schule
Istanbul, Türkei
09/1984 – 07/1985 Bundesgymnasium Horn
09/1980 – 07/1984 Volksschule Eggenburg

Praktika, Kurse Design Patterns
C++ for particle physicists
CERN School of Computing 2000, Marathon,
Griechenland
Siemens Wien, Sommer 1996 und 1997

Fremdsprachen Englisch, Französisch, Türkisch