

Chapter 6

A LVL2 pre-processing algorithm for the precision tracker

Contents

6.1 Introduction	144
6.2 SCTFEX	144
6.2.1 Implementation	144
6.1.2 Implementation of timing measurements.....	145
6.3 Input files for SCTFEX	146
6.3.1 Detector geometry database	146
6.1.2 Data files	146
6.4 Input data format	148
6.5 Selection of hits from modules containing RoI data	149
6.5.1 Modules within the RoI	149
6.1.2 Implementation of algorithm	150
6.1.3 Performance results.....	150
6.1.4 Calculation times	153
6.6 Grouping of hits into clusters.....	154
6.6.1 Definition of a cluster	154
6.6.2 SCT implementation	154
6.6.3 Pixel detector implementation	155
6.6.4 Cluster size.....	157
6.6.5 Calculation times	158
6.7 Conversion from clusters to space-points	160
6.7.1 Local and global co-ordinates	160
6.1.2 Barrel SCT	161
6.1.3 End-cap SCT	163
6.1.4 Pixel detector	165
6.1.5 Performance results.....	166
6.1.6 Calculation times	168
6.8 Algorithm for selection of space-points.....	170
6.8.1 Implementation	170
6.1.2 Performance results.....	171
6.9 Summary and conclusions	172
6.9.1 Total pre-processing calculation times	172
6.1.2 Addition of pixel detector	173
6.1.3 Implementation of pre-processing algorithm.....	173

6.1 Introduction

In this chapter I present a pre-processing algorithm for the precision tracker. Pre-processing is an additional processing step before the LVL2 feature extraction algorithm. In this processing step the hits to be used in the feature extraction (track finding) algorithm are selected and transformed from the format in which the hits are received from the RODs (see chapter 5) into a format suitable for the feature extraction algorithm. Because the pre-processing algorithm must operate in an online environment, a very important aspect is its calculation time.

The pre-processing algorithm presented consists of four parts:

- Selection of hits from modules within the RoI;
- Combination of hits into clusters;
- Combination of clusters from “ ϕ -layer” and “stereo-layer” (SCT) and conversion into (ϕ, r, z) co-ordinates (space-points);
- Selection of space-points belonging to the RoI (optionally);

The pre-processing algorithm is implemented in the software package SCTFEX, described in section 6.2. The input files used by this program are described in section 6.3. It was necessary to make certain assumptions on the format of the raw data [1]. These are described in section 6.4. Each step of the algorithm is described in more detail in a separate section, together with performance and timing measurements. The algorithm for the selection of hits from modules within the RoI is presented in section 6.5. The clustering algorithm is presented in section 6.6. The algorithm for the creation of space-points is presented in section 6.7. The algorithm for the selection of space-points belonging to the RoI is presented in section 6.8. The results are summarised in section 6.9, together with suggestions for implementation of the pre-processing algorithm in the LVL2 system.

6.2 SCTFEX

6.2.1 Implementation

To study both the physics performance and calculation time¹ of the LVL2 trigger algorithm for the precision tracker, I have written the C++ program SCTFEX. This program both includes a pre-processing and track reconstruction algorithm. The track reconstruction algorithm is described in the next chapter. SCTFEX does not make use of any external non ANSI C++ libraries (except some platform dependent timing routines) and can be implemented on any platform with a C++ compiler. A very important difference between SCTFEX and a real implementation of the algorithms in the LVL2 system is that the communication of the LVL2 processor with the ROB, global part and supervisor² is not implemented in SCTFEX. In this program the events are read from a data file.

Optimisation

The implementation of the algorithms can have a large impact on their execution speed. An improved version of the clustering algorithm for the data from the pixel detectors is for example about a factor 100 faster than the original version (see section 6.6.3 for the currently

¹ A study of the calculation time of an algorithm is also referred to as a **benchmark** of the algorithm.

² This is dependent on the architecture of the LVL2 system.

implemented clustering algorithm). Attention is paid to implement the algorithms as efficiently as possible, and especially to avoid repetition of calculations. Sometimes however a less efficient implementation is chosen for reasons of flexibility. To be able to easily test various versions of an algorithm, for example so-called virtual functions are sometimes used. Which implementation of the virtual function is called is decided at run time. A virtual function can be less efficient than a normal function because it can not be expanded inline³.

Whenever possible and making sense, **LUTs (LookUp Tables)** are used to speed up the algorithms. A LUT is a table containing pre-calculated values of a function. Instead of performing the calculation, only a table index is calculated and the function result is read from the table. LUTs are especially useful when the function can only have a restricted number of inputs, and the table index can be easily calculated, e.g. the input is given by an integer index. A function calculating the position of a silicon strip compared to the centre of the module can for example very effectively be implemented in a LUT. The strip number is used as table index in this case.

In the case of a (in principle) continuous input distribution, a LUT can only be used by making a discrete distribution of it. The results of $\cos x$ for example, can only be stored in a LUT for a restricted set of input points, e.g. 1000 input values homogeneously distributed between $-\pi$ and π . The LUT will in general not exactly reproduce full calculation in this case. Also the calculation of the table index can be more complicated. In the current implementation of the algorithm, LUTs are only used in the case of a discrete set of input points. They are not used in the case of a continuous set of input points.

6.2.2 Implementation of timing measurements

For the timing measurements a Dell PC with a 400 MHz Pentium II processor is used. This system has 64 MB internal memory, a cache memory of 512 kB and a 100 MHz system bus. It runs with Windows NT 4.00. The compiler used is Microsoft Visual C++ (version 5.1). With this compiler the highest possible optimisation has been used (small non virtual functions are for example automatically expanded inline).

A dedicated board with a **DSP (Digital Signal Processor)** is used for timing measurements. The resolution is better than 1 μ s. Occasionally a timing measurement can be much higher than normal due to interrupts of the operating system. To prevent a potential influence on the timing results, the measurement is repeated a few times and unrealistic high numbers are skipped. The mean value of the remaining measurements is taken as the final value.

The tests on error conditions (array overflow, illegal data) are all based on the C++ `assert` macro. These tests can be completely switched off using the NDEBUB **C/C++ pre-processor**⁴ definition. All timing results presented in this and the following chapter are based on the version without any checking on error conditions. Most of the efficiency and statistics results presented in this and the next chapter are only calculated if NDEBUB is not defined.

³ Each call of an inline function is replaced by the expansion of its body, preventing the overhead of a function call.

⁴ The C/C++ pre-processor is a processing step before the actual compilation. Depending on the used pre-processor definitions, the C/C++ code to be compiled is selected, macros are expanded to normal C/C++ statements and header files, containing C/C++ definitions, are included. If NDEBUB is not defined, `assert` is expanded to a routine stopping the program with an error message if the passed argument is false. If NDEBUB is defined, `assert` is expanded to nothing.

Only the calculation time of the algorithms is taken into account. Any file input/output or output to screen is excluded from the measurements or completely turned off. Initialisation is only taken into account if it has to be repeated for each RoI. Initialisation at the beginning of the program is not taken into account. All timing measurements presented in this chapter are in μs .

Timing measurements on other computer platforms

On each computer platform with a C/C++ compiler available, the ANSI C `clock` routine can be used. Because the resolution of this routine is in general rather bad (on the Pentium II 1 ms), the routine to be measured has to be repeated many (10^3 or 10^4) times and only the total time has to be measured. The calculation time can now be defined as the sum value divided by the number of repetitions.

6.3 Input files for SCTFEX

6.3.1 Detector geometry database

SCTFEX uses a geometry database, stored in an external ASCII data file. The layout of the detector used has been described in chapter 2 and is shown in figure 2.7.

Each end-cap part of the SCT in this layout consists of nine wheels, each containing two or three rings with modules. Each detector module consists of two layers, shifted from each other in z . The strips on the first layer are radially placed (the “ ϕ -layer”). The second layer is rotated so that the strips make a stereo angle α of about 40 mrad with the radial axis (the “stereo-layer”). An end-cap part covers the range $82 \text{ cm} \leq |z| \leq 277 \text{ cm}$, corresponding to $1.4 \leq |\eta| \leq 2.5$. The barrel part of the SCT consists of four layers, each containing 12 rings with modules. The innermost detector layer has a radius of 30 cm, the outermost detector layer has a radius of 52 cm. The barrel part covers the range $|z| < 74.5 \text{ cm}$, corresponding to $|\eta| < 1.4$. Also here each module consists of a “ ϕ -layer” and a “stereo-layer”. The total number of silicon strips is about 6.2×10^6 .

The barrel pixel detector consists of three layers, each containing 13 rings with modules. The innermost layer (the B-layer) has a radius of 4 cm, the outermost layer has a radius of 14 cm. The B-layer covers the range $|z| \leq 35 \text{ cm}$, corresponding to $|\eta| \leq 2.5$. The other two barrel layers cover the range $|\eta| \leq 1.7$. Each end-cap part of the pixel detector consists of four disks, each containing one or two rings with modules. The end-cap part covers the range $47 \text{ cm} \leq |z| \leq 107 \text{ cm}$, corresponding to $1.7 \leq |\eta| \leq 2.5$. A module of the pixel detector consists of only one layer. The total number of pixels is about 1.4×10^8 .

6.3.2 Data files

The data samples used are based on full detector simulation (simulation with GEANT and DICE, see chapter 3) of the inner detector and calorimeter. A simulation of the LVL1 and LVL2 calorimeter trigger is provided by the package ATRIG [3]. The detector simulation includes the effects of inefficiency and noise in the SCT and pixel layers. The effects of misalignment of the detector modules are not included in the current study. For each RoI, all the strips and pixels with a signal are stored. These strips and pixels are also referred to as **hits**. For the current study I have used the following sets of data samples:

- 1421 RoIs of electrons of $p_T = 30 \text{ GeV}$ with minimum bias background (high luminosity). These RoIs are referred to as electron RoIs in the remaining part of this chapter. The distribution in pseudorapidity η (RoI centre position) is given in figure 6.1. From the figure it follows that the data sample has a homogeneous distribution with $|\eta| < 2.5$. The distri-

bution in ϕ (RoI centre position) is given in figure 6.2. The (uncompressed) size of this file is about 182 MB.

- 1179 RoIs of jets with minimum bias background (high luminosity). Those jets were identified by LVL1 as electron RoIs and are referred to as jet RoIs in the remaining part of this chapter. The size of this file is about 136 MB.

The sample with electron RoIs has been used for efficiency studies of the track reconstruction algorithm described in the next chapter. The jet RoIs have been used as background sample to study the rejection efficiency of the track reconstruction algorithm against background events. For the pre-processing algorithm these data samples have been used to make a study of the calculation time and a study of the RoI data size.

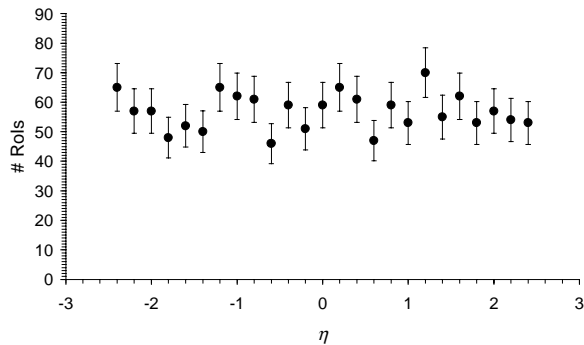


Figure 6.1 RoI η distribution, electrons.

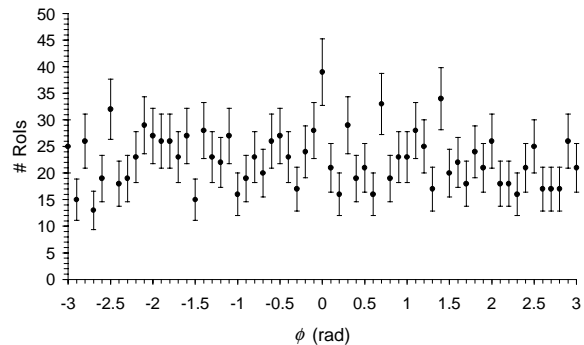


Figure 6.2 RoI ϕ distribution, electrons.

File format

The events are stored in ASCII files in a format very similar to the format described in [4]. All numbers are written in hexadecimal format, to prevent loss of accuracy of floating-point values. A drawback of storing numbers in hexadecimal format is that the data files are much less readable by the human reader. Another drawback is that floating-point numbers can have a different representation on different platforms. On the platforms tested however the hexadecimal numbers read from the file can be immediately interpreted as floating-point values without any transformation.

RoI occupancy

The RoI occupancy can be calculated from the total number of hits in modules within the RoI per detector part, divided by the total number of strips or pixels in these modules. The RoI occupancy values in the different detector parts are given in table 6.1. The corresponding histograms for the electron RoIs are given in figure 6.3 to figure 6.6.

Both for the SCT and pixel detectors, the RoI occupancy can approximately be described by a Gaussian distribution, however with a large tail. The RoI occupancy in the end-cap area is significantly less than in the barrel area. The pixel detectors have a much lower RoI occupancy than the SCT. The average RoI occupancy for jet RoIs is slightly larger than for electron RoIs. Because the hits are not completely uncorrelated, the local occupancy in a small part of a module can be higher than the values presented in the table.

The values for the RoI occupancy given in the table are slightly different from the values for the occupancy given in the inner detector TDR [5]. In the TDR the given average occupancy for the barrel pixel detector varies between 0.011% and 0.045% (B-layer). For the end-cap pixel disks an average occupancy of 0.012% is given. For the barrel SCT the average occupancy varies between 1.4% (layer 4) and 1.7% (layer 1). These numbers are based on full

detector simulation of b jets from $H^0 \rightarrow b\bar{b}$ with minimum bias background (high luminosity).

Table 6.1 RoI occupancy in the different sub-detectors.

	electron RoIs		jet RoIs	
	mean [%]	max [%]	mean [%]	max [%]
barrel SCT	0.644	4.329	0.856	3.906
end-cap SCT	0.542	1.628	0.697	3.125
barrel pixel detector	0.019	0.087	0.023	0.089
end-cap pixel detector	0.004	0.032	0.009	0.054

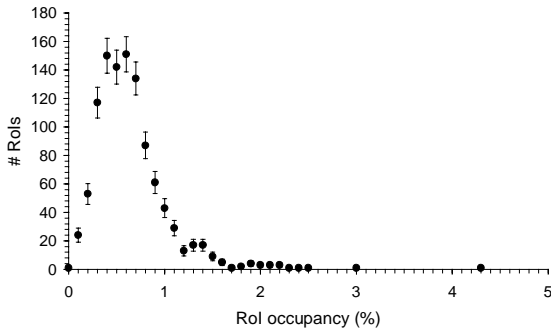


Figure 6.3 RoI occupancy, barrel SCT, electrons.

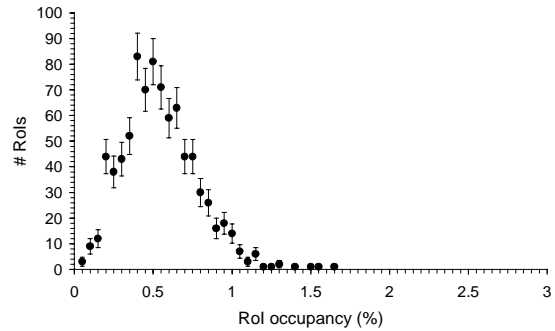


Figure 6.4 RoI occupancy, end-cap SCT, electrons.

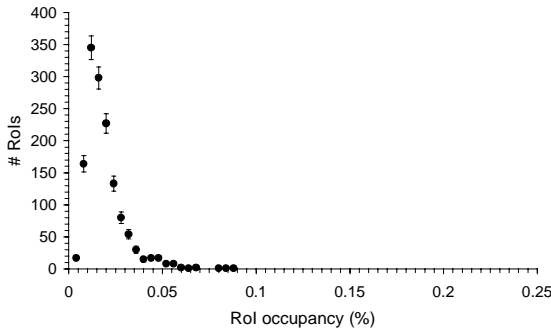


Figure 6.5 RoI occupancy, barrel pixel detector, electrons.

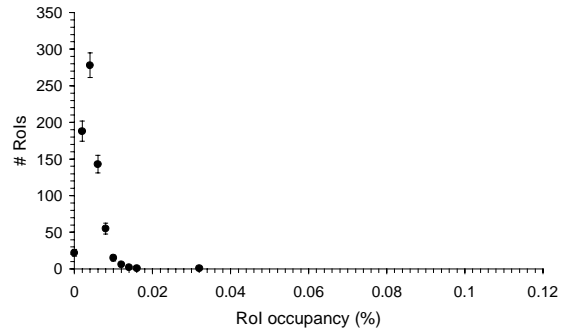


Figure 6.6 RoI occupancy, end-cap pixel detector, electrons.

6.4 Input data format

A 32-bit word describes each hit [1]. From this address it can be found which strip or pixel in the detector has been hit. This scheme is based on a binary read out in which no pulse height information is available. The individual bits for the SCT, the barrel pixel detector and the end-cap pixel detector have a different meaning (figure 6.7 to figure 6.9). For the SCT and end-cap pixel detector, one bit of the address word is not used. The `part` field indicates the detector part. The detector part can be barrel SCT, forward SCT, backward SCT, barrel pixel detector, forward pixel detector or backward pixel detector. The `plane` field indicates the number of the cylindrical layer in the barrel and the number of the disk in the end-cap area. The `layer` field (SCT only) indicates whether the layer is a “ ϕ -layer” or “stereo-layer”.

For each sub-detector (SCT/pixel detector) and detector area (barrel/end-cap), all the hits of one RoI are stored in one long list. In a real implementation on the FEX processors, the hits will be received in fragments, consisting of the data of one or a few ROB. A header containing information about the current RoI will precede each fragment [6]. The merging of the fragments in one list is regarded as part of the communication overhead and is not taken into account in the current implementation.

31	29	27	22	18	12	10	0
part		plane	ring	module	layer		strip

Figure 6.7 Meaning of the individual bits in the address word, SCT.

31	29	27	23	17	9	0
part	plane	ring	module	column		row

Figure 6.8 Meaning of the individual bits in the address word, barrel pixel detector.

31	29	26	24	16	8	0
part	plane	ring	module	column		row

Figure 6.9 Meaning of the individual bits in the address word, end-cap pixel detector.

6.5 Selection of hits from modules containing RoI data

6.5.1 Modules within the RoI

Even if only a small fraction of the event fragment in a ROB belongs to an RoI, the complete event fragment stored in the ROB (data from 16 modules for the barrel and end-cap SCT) is sent to the FEX processors. The FEX processors have to select the part of the event fragment that belongs to the RoI. In the first part of the pre-processing algorithm this selection is based on the detector modules. All the data of a module is regarded to belong to an RoI or not. A barrel module is assumed to be within an RoI if:

$$\begin{aligned} \phi_{modMax} &\geq \phi_{RoIMin} \wedge \phi_{RoIMax} \geq \phi_{modMin} \\ z_{modMax} &\geq z_{RoIMin}(r_{mod}) \wedge z_{RoIMax}(r_{mod}) \geq z_{modMin} \end{aligned} \quad (6.1)$$

with $(\phi_{modMin}, \phi_{modMax}, z_{modMin}, z_{modMax})$ the (ϕ, z) range spanned by the module, $(\phi_{RoIMin}, \phi_{RoIMax}, z_{RoIMin}, z_{RoIMax})$ the (ϕ, z) range spanned by the RoI and r_{mod} the radial position of the module, see also section 5.8. An end-cap module is assumed to be within an RoI if:

$$\begin{aligned} \phi_{modMax} &\geq \phi_{RoIMin} \wedge \phi_{RoIMax} \geq \phi_{modMin} \\ r_{modMax} &\geq r_{RoIMin}(z_{mod}) \wedge r_{RoIMax}(z_{mod}) \geq r_{modMin} \end{aligned} \quad (6.2)$$

In the (x, y) plane a constant RoI size is used (0.2 rad). In the (r, z) plane, the RoI size depends on the position (r in the barrel, z in the end-cap), taking into account the length of the interaction region along the beam axis, see section 5.8. The ROB mapping scheme determines which detector module sends data to which ROB. For the (barrel) SCT several schemes for mapping the detector modules to the ROB are available. The layout based on simulated annealing optimisation (see section 5.8) has been used [8]. Because there are no

mapping schemes currently available for the pixel detectors, the data selection step is only implemented for the SCT.

6.5.2 Implementation of algorithm

The **PSD (Program Structure Diagram)** of the algorithm for the selection of hits from modules containing RoI data implemented in `SCTFEX` is given in figure 6.10. The implementation consists basically of a loop over the hits in the input lists. For each hit the corresponding detector module is determined based on the `module` bits of the address word and the hit is only copied to the output list if the module is within the RoI.

Because only a fixed number of RoI positions is possible (see chapter 5), a LUT can be used to define for each RoI the corresponding modules that are within this RoI. If 32 modules are packed in one word (bit set means module present), the size of this table is in principle (number of RoIs \times number of modules)/32 words (about 1500 KB for electron RoIs if the pixel detectors are also included). For practical reasons, a much smaller table of about 500 bytes (number of modules/32 words, SCT only) is at the moment used in `SCTFEX`. This table needs to be recalculated for each new RoI. This recalculation is not taken into account in the presented timing results. So the results are the same as if the big table were used.

The data files currently available contain only hits from modules that are within the RoI. This means that for the extra modules falling outside the RoI some dummy hits have to be generated, and put into the input list. The number of the dummy hits to be generated is based on the mean occupancy of the modules within the current RoI. In general the occupancy will be higher inside an RoI than outside an RoI⁵. This means that the number of additionally generated hits will be too large, and the calculation times will be too pessimistic.

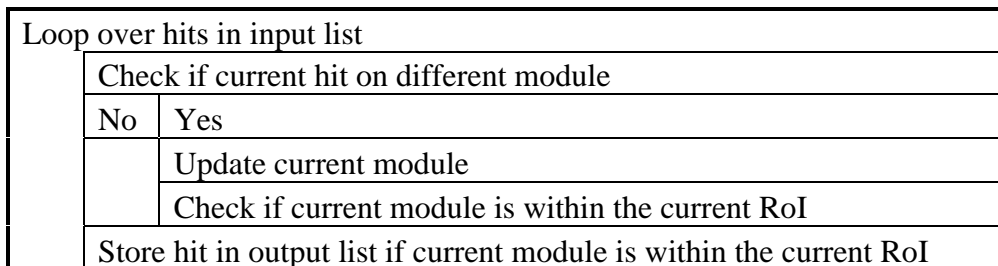


Figure 6.10 PSD for algorithm for the selection of hits from modules containing RoI data.

6.5.3 Performance results

Number of SCT ROBs receiving an RoI request per LVL1 trigger

The histogram of the total number of SCT ROBs receiving an RoI request, is given in figure 6.11. A ROB receives an RoI request if at least one of the SCT modules mapped to this ROB is within the RoI.

The number of ROBs differs significantly from the number given in some older studies [8, 9]. This is due to the fact that in the current study also the length of the LHC interaction region along the beam axis is taken into account. Also the size of the detector modules has changed since the study described in [9]. The numbers are slightly different from the numbers presented in the previous chapter (figure 5.17). This is mainly due to the fact that the centre positions in ϕ of the RoIs are shifted with $\pi/64$ compared to the definition used in the previous chapter.

⁵ The hits inside an RoI are created by the high- p_T electron plus minimum bias background. The hits outside an RoI are created by the minimum bias background only.

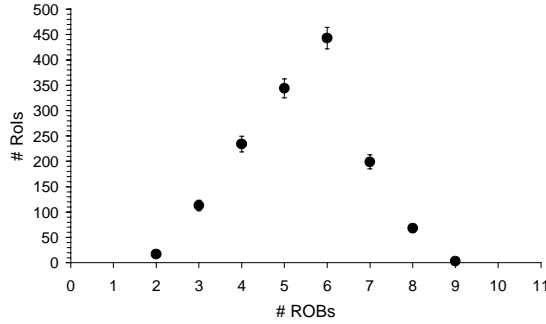


Figure 6.11 Number of SCT ROBs receiving an RoI request per LVL1 trigger, electron ROIs.

Number of modules within an RoI

The distribution of the numbers of modules within an RoI for the different detector parts are given in figure 6.12 to figure 6.15 (electron ROIs). Only ROIs that cover the corresponding sub-detector part are taken into account (An RoI with centre position $|\eta| = 2.5$ will for example cover zero modules of the barrel SCT).

In the barrel area, up to about 40 modules can be within an RoI, both for the SCT and pixel detectors. In the end-cap area, up to about 30 modules can be within an RoI. The distributions of the number of modules within an RoI for the barrel SCT, end-cap SCT and barrel pixel detector resemble Gaussian distributions, however with large tails on the left side. These tails are due to ROIs in the overlapping area and at the end of acceptance.

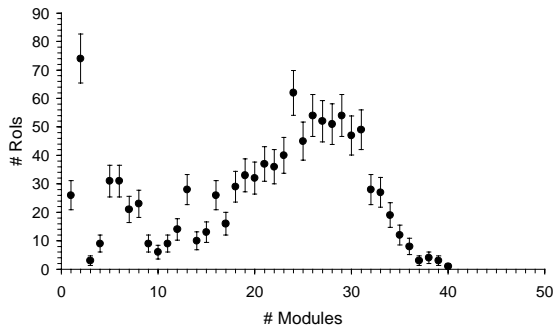


Figure 6.12 Number of modules within an RoI, barrel SCT, electron ROIs.

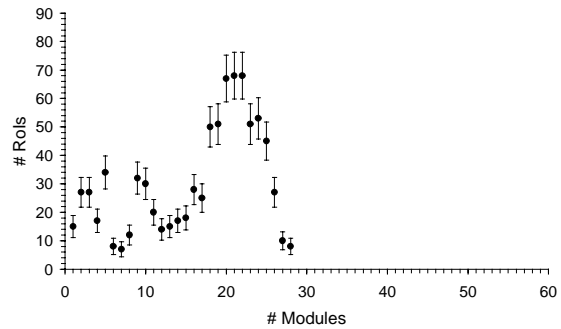


Figure 6.13 Number of modules within an RoI, end-cap SCT, electron ROIs.

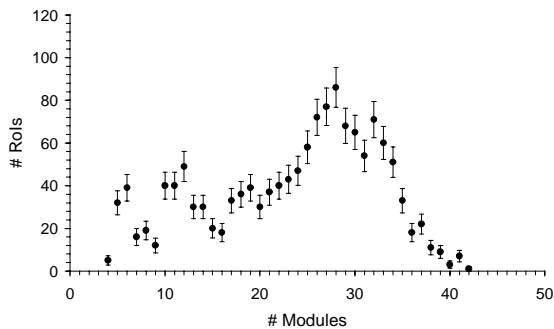


Figure 6.14 Number of modules within an RoI, barrel pixel detector, electron ROIs.

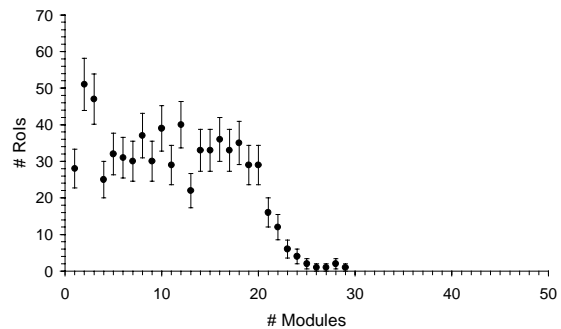


Figure 6.15 Number of modules within an RoI, end-cap pixel detector, electron ROIs.

Number of hits per RoI

The distributions of the number of hits per RoI for the different sub-detectors are given in figure 6.16 to figure 6.19 (electrons). Only RoIs that cover the corresponding sub-detector part are taken into account.

Although the occupancy in the barrel pixel detector is much lower than in the barrel SCT, the number of hits is roughly the same. The number of hits in the end-cap pixel disks is very low compared to the other detectors. For each detector part the distribution of the number of hits per RoI resembles a Gaussian distribution, however with a large tail, especially on the right side.

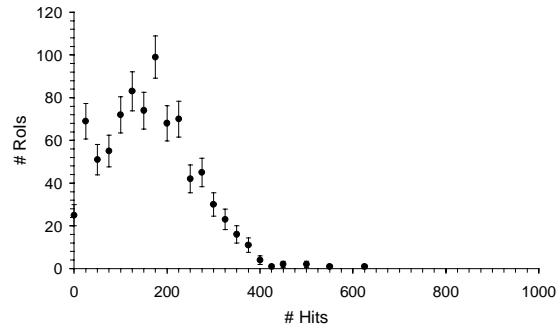
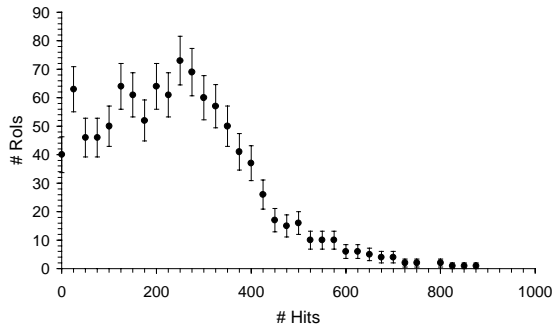


Figure 6.16 Number of hits per RoI, barrel SCT, **Figure 6.17** Number of hits per RoI, end-cap electron RoIs.

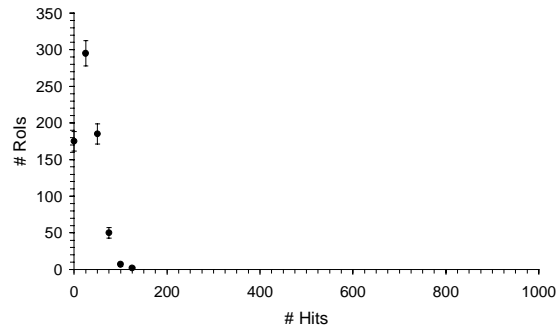
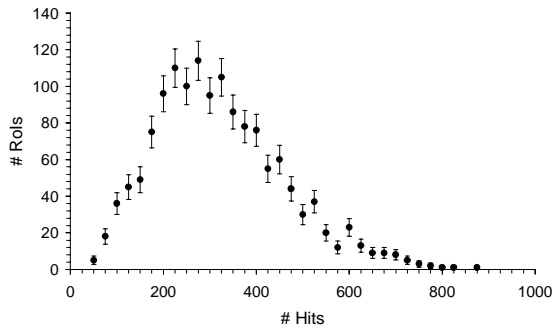


Figure 6.18 Number of hits per RoI, barrel pixel detector, **Figure 6.19** Number of hits per RoI, end-cap pixel detector, electron RoIs.

Summary of results

The results are summarised in table 6.2, together with the results for the jet RoIs. In this table values are given for the number of ROBs that receive an RoI request, the number of modules within an RoI, the number of hits per RoI and the total number of hits stored in the ROBs that receive an RoI request. The total number of modules of which data is stored in the ROBs that receive an RoI request is not given but can easily be calculated from the number of ROBs that receive an RoI request, taking into account that each ROB receives data from 16 modules. For the mean values presented only RoIs that cover the corresponding sub-detector part are taken into account.

The number of ROBs and modules for jet RoIs is almost the same as for electron RoIs. The number of hits is somewhat larger.

Comparing the number of ROBs receiving an RoI request with the number of modules within an RoI shows, that only a relatively small fraction of about 30% of the modules of which data is stored in the ROBs that receive an RoI request is within this RoI. This means

that the total number of hits transported through the switching network can be significantly higher than the number of hits in the RoI (up to about a factor 3).

Table 6.2 Number of ROBs, modules and hits.

	electron RoIs		jet RoIs	
	mean	max	mean	max
Number of ROBs total SCT	5.4	9	5.6	9
Number of ROBs barrel SCT	4.4	8	4.4	9
Number of ROBs end-cap SCT	3.4	7	3.3	7
Number of modules barrel SCT	20.3	40	21.2	42
Number of modules end-cap SCT	16.6	28	15.7	28
Number of modules barrel pixels	23.3	42	25.3	43
Number of modules end-cap pixels	11.0	29	10.5	30
Number of (RoI) hits barrel SCT	245	1155	275	1140
Total number of hits barrel SCT	690	3062	771	3212
Number of (RoI) hits end-cap SCT	164	633	173	555
Total number of hits end-cap SCT	451	1479	496	2304
Number of (RoI) hits barrel pixels	321	869	395	947
Number of (RoI) hits end-cap pixels	31	123	32	166

6.5.4 Calculation times

The calculation times of the data selection algorithm, described in figure 6.10, are given in table 6.3. The results for jet RoIs and electron RoIs are roughly the same. The calculation times presented are based on the assumption that the hits are ordered per module, which is not unrealistic because the readout is organised per module. If the hits are not ordered per module, the same algorithm will work but will be more time consuming (for each new module the LUT has to be checked whether this module is within the current RoI or not, see also figure 6.10).

Table 6.3 Calculation times of data selection algorithm [μ s].

	barrel SCT		end-cap SCT	
	mean	max	mean	max
electron RoIs	21	46	19	32
jet RoIs	22	46	19	30

Parameterisation of calculation times

For modelling studies of the trigger system it is desirable that a simple parameterisation can describe the calculation time. The calculation time versus the number of hits (electron RoIs) is given in figure 6.20 (barrel SCT) and figure 6.21 (end-cap SCT). From these figures it follows that in a very good approximation the calculation time scales linearly with the number of hits. The calculation time is larger than predicted by a linear fit in only a small proportion of cases. The time necessary for initialisation, the overhead of the function calls, and the time necessary for the timing measurements itself are all included in the offset.

For the barrel SCT, the calculation time versus the number of hits is given by:

$$\text{Time } [\mu\text{s}] = 13.7 + 0.030 \times (\#\text{Hits}) \tag{6.3}$$

For the end-cap SCT, the calculation time versus the number of hits is given by:

$$\text{Time } [\mu\text{s}] = 13.2 + 0.031 \times (\#\text{Hits}) \quad (6.4)$$

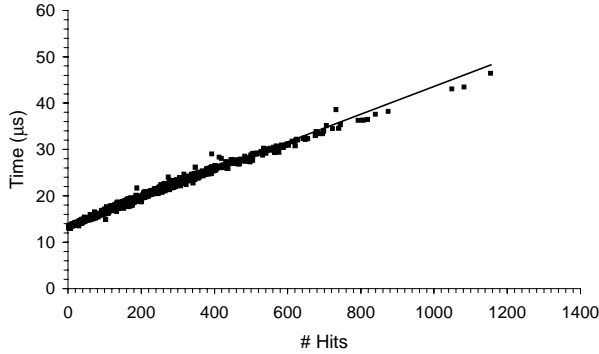


Figure 6.20 Calculation time of the data selection algorithm versus the number of hits, barrel SCT, electron RoIs.

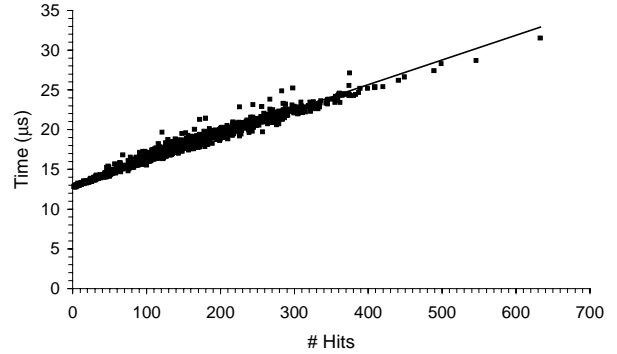


Figure 6.21 Calculation time of the data selection algorithm versus the number of hits, end-cap SCT, electron RoIs.

6.6 Grouping of hits into clusters

6.6.1 Definition of a cluster

When passing through a detection layer, the same high- p_T particle can create several neighbour hits due to for example the diffusion of electrons in the silicon. For this reason, the hits need to be grouped into **clusters**.

In the current implementation the definition of a cluster is very simple: a continuous group of strips or pixels with a hit. Gaps between these strips or pixels are not allowed. Very big clusters are not split. For the SCT this is the only possible definition of a cluster if no pulse height information is available. In the offline analysis probably a more complicated definition of a cluster will be used for clusters from the pixel detectors.

The definition of a pixel cluster is slightly different from the definition used to produce the LVL2 results presented in the inner detector TDR [5]. In the TDR definition clusters are limited to a maximum size and very big clusters are split in two or more parts.

6.6.2 SCT implementation

An example of two clusters in a part of an SCT module is given in figure 6.22. The first cluster consists of four strips, the second one of two strips. For the SCT, a cluster is determined by its (one-dimensional) position t and size s :

$$\begin{aligned} t &= \frac{1}{2}(t_1 + t_N) \\ s &= N \end{aligned} \quad (6.5)$$

with t_1 the integer strip index of the first hit in the cluster, t_N of the last hit in the cluster and N the number of strips in the cluster. From this equation it follows that the number of potential cluster positions is two times the number of strips. The resolution of the centre position of a cluster is half the resolution of a strip (the fractional part of a cluster position is always 0.5 or 0.0). This means that the cluster position can easily be used as index for a LUT⁶.

⁶ This LUT can be used for the conversion from clusters into space-points (section 6.7).

The PSD for the algorithm implemented in SCTFEX is given in figure 6.23. The implementation consists of a loop over the hits, checking if each hit is adjacent to the next hit or not. The assumption is made that the hits are grouped per module and that the hits per module are ordered by increasing (or decreasing) strip number.



Figure 6.22 An example of ten SCT strips with two clusters.

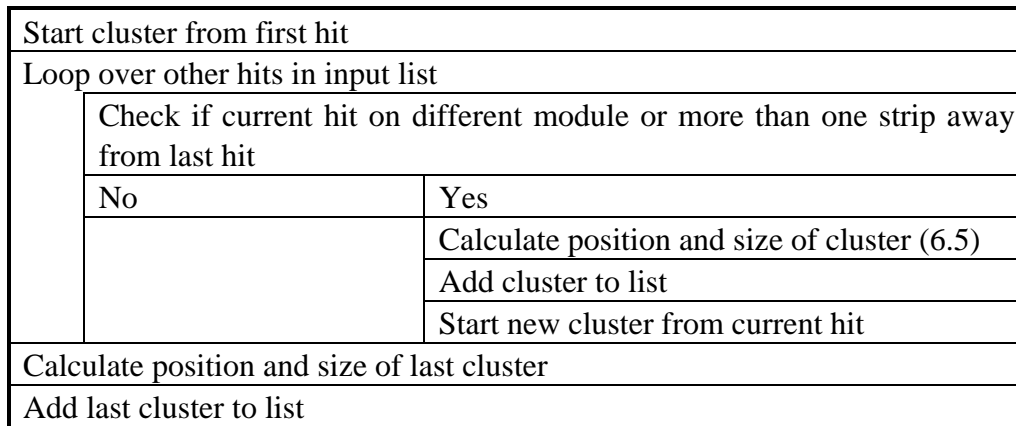


Figure 6.23 PSD for SCT cluster classification algorithm.

6.6.3 Pixel detector implementation

An example of two clusters in a pixel module is given in figure 6.26. For the pixel detectors a cluster is given by its (two-dimensional) position (t, l) and size (s_t, s_l) , and the number of pixels in the cluster N :

$$t = \frac{1}{N} \sum_{i=1}^N t_i$$

$$s_t = \max(t_1, \dots, t_N) - \min(t_1, \dots, t_N)$$

$$l = \frac{1}{N} \sum_{i=1}^N l_i$$

$$s_l = \max(l_1, \dots, l_N) - \min(l_1, \dots, l_N)$$
(6.6)

with t_i and l_i the row and column index of hit i . In the example the size of the first cluster according to this definition is $(5, 3)$. The size of the second cluster is $(4, 3)$. From equation (6.6) it follows that the number of potential cluster positions is very large and it is not very attractive to use the cluster position as index for a LUT.

For the pixel detectors a much more complicated cluster finding algorithm is necessary. Each hit can have neighbours in adjacent rows or columns, giving eight instead of two possible neighbours. The hits are also not ordered per cluster anymore, between two hits of the same cluster can be several hits belonging to other clusters.

In the current implementation a two-dimensional map of a pixel module (**pixel map**) is used (implemented in a one-dimensional array) together with a list of hits stored in the map. The pixel map contains a one if a hit is present on the current row and column position, and a

zero otherwise. For each hit stored in the map, the unpacked row and column position and the corresponding (one-dimensional) array index of the pixel map is stored in the list.

The clusters are reconstructed each time the next hit read from the input is on a different module or more than one row and column away from the last hit (so definitely not belonging to the same cluster). If only one hit is stored in the list, the cluster consists of only one single hit, and is reconstructed directly from the list (in the end-cap region this happens frequently). If several hits are stored in the list, the pixel map is filled and searched for clusters via a recursive cluster classification algorithm, where each pixel with a hit checks its surrounding pixels. The surrounding pixels are checked in order of increasing column and row index (lowest column and row first). First the column index is increased, then the row index.

In each step of the recursive cluster classification algorithm, information about the cluster position and size is updated, according to (6.6). The assumption is made that the hits are first ordered by row and then by column in the input data. In this case it is not necessary to check all eight neighbours. For the first pixel of a cluster at maximum four surrounding pixels have to be checked. For the other pixels, the number of neighbour cells to be checked varies between three and five. The PSD for the pixel cluster search algorithm is given in figure 6.24. Attention has been paid to implement the recursive cluster classification algorithm as efficiently as possible, with a minimal function interface. In the example of figure 6.26 with two clusters the order in which the cells are checked is given, together with the number of the cell from which the current cell is checked and the number indicating the order of the hits in the input data. The PSD for the recursive cluster classification algorithm is given in figure 6.25.

Loop over hits in input data			
Check if current hit on different module or more than one row and column away from last hit			
No	Yes		
Check if only one hit is stored in list			
No		Yes	
Loop over hits stored in list		One cluster consisting of one pixel found. Update list with clusters.	
Set pixel in pixel map (value 1)			
Loop over hits stored in list			
Check if pixel in pixel map is set (value 1)			
No	Yes		
Recursive cluster classification algorithm starting from current pixel (figure 6.25)			
Add found cluster to list with clusters			
Reset list with hits			
Update list with current hit			
Create last cluster and update list with found clusters			

Figure 6.24 PSD for pixel cluster search algorithm.

Update information about cluster position and size	
Reset current pixel (value zero)	
Loop over possible neighbour pixels	
Check if neighbour pixel contains hit	
No	Yes
	Start recursive cluster classification algorithm from neighbour pixel

Figure 6.25 PSD for recursive cluster classification algorithm.

1 (*, 1)	2 (1)	5 (4)	7 (6)				24 (23)	26 (25)	
3 (1)	4 (1, 2)	6 (4, 3)	8 (6)		22 (*, 4)	23 (22, 5)	25 (23, 6)	27 (25)	
10 (4)	11 (4, 7)	21 (4)	9 (6)	30 (22)	31 (22, 8)	41 (22)	29 (23)	28 (25)	
12 (11)	13 (11, 9)	20 (11)		32 (31)	33 (31, 10)	40 (31)			
14 (13)	15 (13, 11)	19 (13)		34 (33)	35 (33, 12)	39 (33)			
16 (15)	17 (15)	18 (15)		36 (35)	37 (35)	38 (35)			

Figure 6.26 An example of a part of a pixel module containing two clusters. The first number gives the order in which the cells are checked. Between brackets the number of the cell is given from which the current cell is checked (first number) and the number indicating the order of hits in the input data (second number). A ‘*’ means that the cell is an initiator of a recursive cluster search and taken from the list with hits.

6.6.4 Cluster size

The mean numbers of strips and pixels in the cluster are given in table 6.4, the corresponding histograms are given in figure 6.27 to figure 6.30 (electron RoIs). Due to for example loopers, the number of strips or pixels in the cluster can be very large, especially in the barrel area. However, this occurs very rarely. The number of strips and pixels in the cluster for electron RoIs and jet RoIs are almost the same.

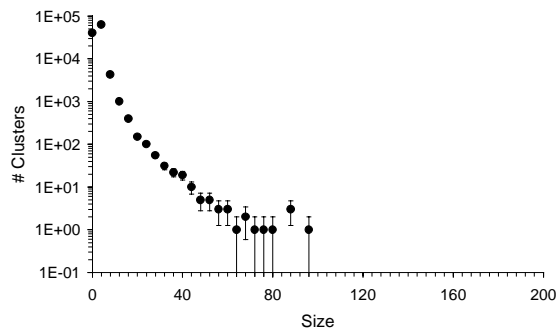


Figure 6.27 Number of strips in cluster, barrel, electron RoIs.

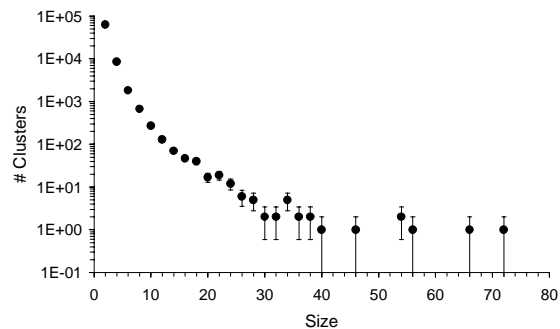


Figure 6.28 Number of strips in cluster, end-cap, electron RoIs.

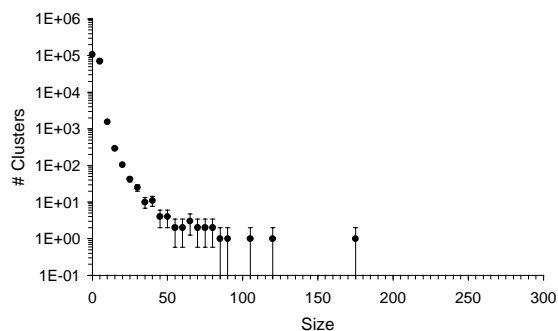


Figure 6.29 Number of pixels in cluster, barrel, electron RoIs.

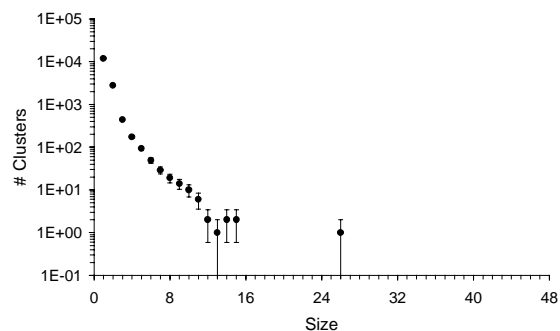


Figure 6.30 Number of pixels in cluster, end-cap, electron RoIs.

Table 6.4 Number of strips and pixels in cluster.

	electron RoIs		jet RoIs	
	mean	max	mean	max
barrel SCT	2.4	94	2.3	121
end-cap SCT	1.8	72	1.8	76
barrel pixel detector	2.5	175	2.5	154
end-cap pixel detector	1.4	26	1.4	31

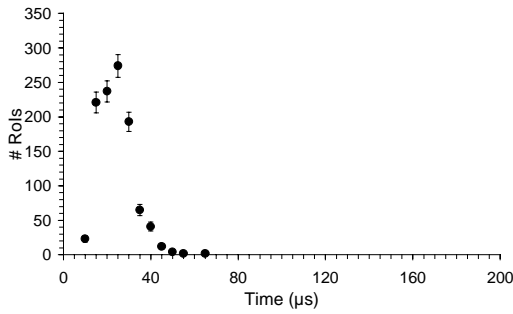
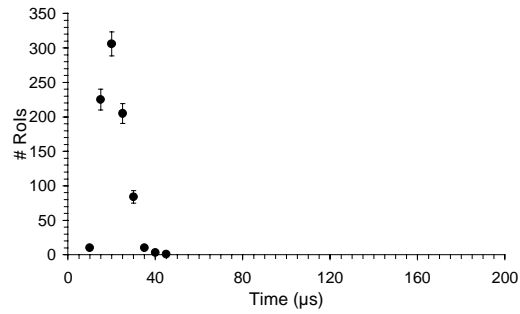
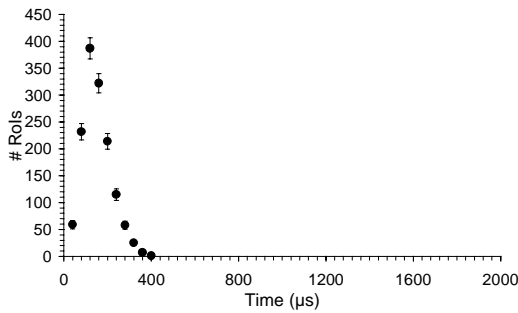
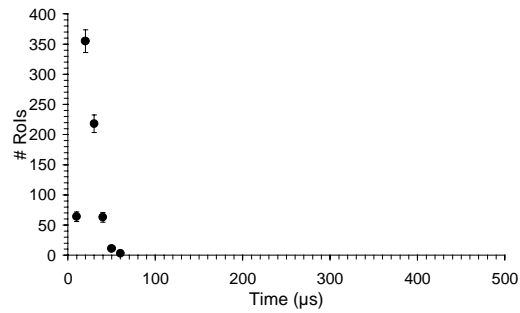
6.6.5 Calculation times

The calculation times are given in table 6.5. The corresponding distributions are given in figure 6.31 to figure 6.34 (electron RoIs). These distributions resemble Gaussian distributions. The calculation times in the barrel pixel detector are significantly higher than in the other detectors. The calculation times for jet RoIs are roughly 10% higher than for electron RoIs.

The pixel cluster search algorithm described is about a factor 100 faster than the algorithm used to produce the results for the inner detector TDR [5], although the definition of a cluster is not exactly the same. The TDR algorithm has a maximum size for a cluster which is not the case for the algorithm presented.

Table 6.5 Calculation times of the clustering algorithm [μs].

	barrel SCT		end-cap SCT		barrel pixels		end-cap pixels	
	mean	max	mean	max	mean	max	mean	max
electron RoIs	24	67	21	41	154	393	25	61
jet RoIs	26	64	22	43	173	428	26	76


Figure 6.31 Calculation times of clustering algorithm, barrel SCT, electron RoIs.

Figure 6.32 Calculation times of clustering algorithm, end-cap SCT, electron RoIs.

Figure 6.33 Calculation times of clustering algorithm, barrel pixel detector, electron RoIs.

Figure 6.34 Calculation times of clustering algorithm, end-cap pixel detector, electron RoIs.

Parameterisation of calculation times

The calculation times versus the number of RoI hits for the different sub-detectors are given in figure 6.35 to figure 6.38. In a good approximation, the calculation time scales linearly with the number of RoI hits, both for the SCT and pixel detectors. For the barrel SCT, the calculation time versus the number of RoI hits is given by (electron RoIs):

$$\text{Time } [\mu\text{s}] = 12.5 + 0.047 \times (\#\text{RoI hits}) \quad (6.7)$$

For the end-cap SCT, the calculation time versus the number of RoI hits is given by (electron RoIs):

$$\text{Time } [\mu\text{s}] = 12.2 + 0.053 \times (\#\text{RoI hits}) \quad (6.8)$$

For the barrel pixel detector, the calculation time versus the number of RoI hits is given by (electron RoIs):

$$\text{Time } [\mu\text{s}] = 9.89 + 0.446 \times (\#\text{RoI hits}) \quad (6.9)$$

For the end-cap pixel detector, the calculation time versus the number of ROI hits is given by (electron RoIs):

$$\text{Time } [\mu\text{s}] = 12.98 + 0.373 \times (\#\text{RoI hits}) \quad (6.10)$$

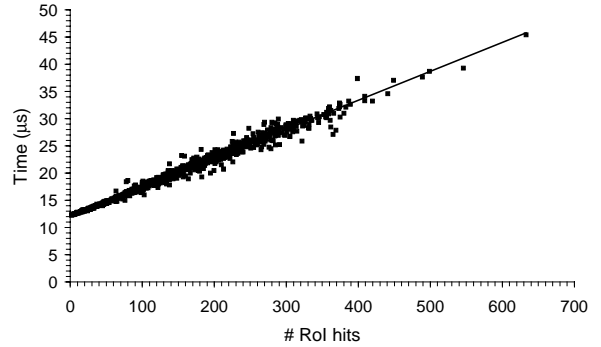
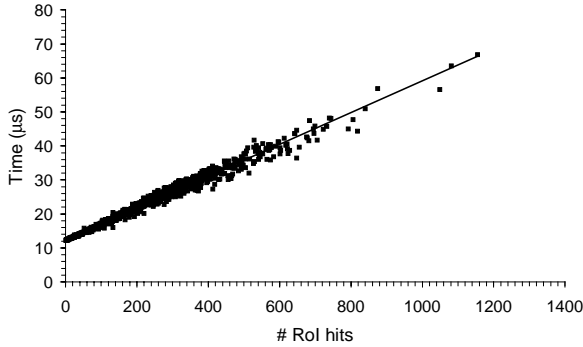


Figure 6.35 Calculation time of clustering algorithm versus the number of ROI hits, barrel SCT, electron RoIs.

Figure 6.36 Calculation time of clustering algorithm versus the number of ROI hits, end-cap SCT, electron RoIs.

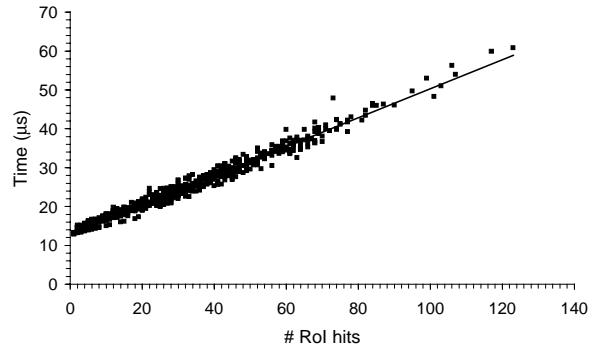
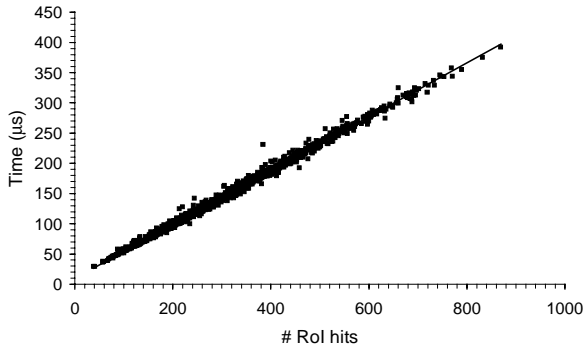


Figure 6.37 Calculation time of clustering algorithm versus the number of ROI hits, barrel pixel detector, electron RoIs.

Figure 6.38 Calculation time of clustering algorithm versus the number of ROI hits, end-cap pixel detector, electron RoIs.

6.7 Conversion from clusters to space-points

6.7.1 Local and global co-ordinates

The clusters need to be combined (for the SCT) and converted to **space-points**. One can define **local** and **global co-ordinates** of a space-point. The local co-ordinates are defined in the co-ordinate frame of a module, with position (0, 0) equal to the centre of the module. The global co-ordinates (ϕ, r, z) or (x, y, z) are defined in the co-ordinate frame of the ATLAS detector. The track finding algorithm described in the next chapter is based on global (ϕ, r, z) space-points. The transformation from (x, y) co-ordinates to (ϕ, r) co-ordinates is given by:

$$\begin{aligned} \phi &= \text{atan}(y/x) \\ r &= x \oplus y \end{aligned} \quad (6.11)$$

The position of a module in global co-ordinates is given by $(\phi_{mod}, r_{mod}, z_{mod})$ or $(x_{mod}, y_{mod}, z_{mod})$. As the specific implementation of the conversion algorithm depends strongly on the detector layout, each detector is handled separately.

6.7.2 Barrel SCT

For the barrel SCT, the local co-ordinates are given by (T, L) . T (cm) gives the position transverse to the strips; L (cm) gives the position parallel to the strips. The cluster position t can easily be converted into the corresponding local co-ordinate T :

$$T = (t - t_{centre}) T_{strip} \quad (6.12)$$

with T_{strip} the strip pitch (cm) and t_{centre} the index of the most central strip. For calculating the L -co-ordinate, one needs to combine the cluster on the “ ϕ -layer” (T_1) with a corresponding cluster on the “stereo-layer” (T_2). For each corresponding cluster a space-point is created. The **degree of ambiguity** is defined as the number of clusters on the “stereo-layer” matching with a cluster on the “ ϕ -layer”. These space-points have the same ϕ -position and r -position, however a different z -position. The local position (T, L) of the space-point is given by (appendix E.1):

$$\begin{aligned} T &= T_1 \\ L &= \frac{T_2 - T_1 \cos \alpha}{\sin \alpha} \end{aligned} \quad (6.13)$$

with α the stereo angle (see also section 6.3.1). A cluster pair is only accepted if L is in the allowed range:

$$|L| \leq \frac{1}{2} L_{strip} \quad (6.14)$$

with L_{strip} equal to the strip length. Clusters on the “ ϕ -layer” without a corresponding cluster on the “stereo-layer” are skipped.

The conversion from local to global variables is given by:

$$\begin{aligned} x &= x_{mod} + e_x (T + T_{shift}) \\ y &= y_{mod} + e_y (T + T_{shift}) \\ z &= z_{mod} + L + z_{shift} \end{aligned} \quad (6.15)$$

with e_x and e_y given by:

$$\begin{aligned} e_x &= -\sin(\phi_{mod} + \tau) \\ e_y &= \cos(\phi_{mod} + \tau) \end{aligned} \quad (6.16)$$

with τ the tilt angle of the module. The tilt angle of the modules can be seen in figure 2.10. The values of the tilt angle are given in table 2.4. T_{shift} is a correction for the movement of the electrons inside the detector in the transverse direction due to the Lorentz force in the presence of a magnetic field that is not collinear with the electric field in the detector. T_{shift} has a fixed value 0.004 cm [1]. z_{shift} is a correction for a shift due to the staggered structure of the modules [10]. For barrel layer 1 and 3, z_{shift} is given by [1]:

$$z_{shift} = 0.18 + 0.03(T + T_{shift}) + \frac{0.04}{r} \quad (6.17)$$

with all dimensions in cm. For barrel layer 2 and 4, z_{shift} is given by:

$$z_{shift} = -0.18 - 0.03(T + T_{shift}) + \frac{0.04}{r} \quad (6.18)$$

The standard deviations of the space-points are given by [11]:

$$\begin{aligned}\sigma_\phi &= \frac{T_{\text{strip}}}{\sqrt{12}r} \\ \sigma_r &= T_{\text{strip}} \left(\frac{x e_x}{r} \oplus \frac{y e_y}{r} \right) \\ \sigma_z &= \frac{T_{\text{strip}}}{\sqrt{6} \sin|\alpha|}\end{aligned}\tag{6.19}$$

Implementation

First a list with modules containing clusters is build, to make a more efficient search for potential partner clusters. This list contains for each module a pointer⁷ to the first and last cluster on the “ ϕ -layer” and on the “stereo-layer”. Modules containing only clusters on the “stereo-layer” are skipped. The other part of the algorithm is based on three nested loops: a loop over the modules from the “ ϕ -layer”, a loop over the clusters in the current module and finally a loop over the clusters in the corresponding module from the “stereo-layer”. If the two clusters are partners, a space-point is created. Clusters on the “ ϕ -layer” without a partner on the “stereo-layer” are skipped.

LUTs are used to speed up the calculations. For each potential cluster position, the corresponding ϕ , r , σ_ϕ , σ_z , and σ_r -values are stored. In this way calculating these values with (6.12), (6.11), (6.15), (6.16) and (6.19) can be avoided. In principle a table of ($2 \times$ number of strips \times number of rings \times number of planes \times number of modules) entries is necessary, corresponding to about 18 MB. The factor 2 in the expression for the table size is coming from the fact that there are two times more potential cluster positions than strips.

To reduce the memory usage of the program, in the current implementation a somewhat smaller table of ($2 \times 2 \times$ number of strips \times number of planes \times number of modules) entries for ϕ and σ_r , and a table of ($2 \times 2 \times$ number of strips \times number of planes) entries for r and σ_ϕ , is used. The total size of the two tables is about 3 MB using a 32 bits float implementation. This reduction is based on the fact that a detector plane contains only modules on two different radial positions, organised in such a way that adjacent rings have always a different radial position. This means that a factor 2 can replace the number of rings in the expression for the table size. The values of r and σ_ϕ are independent from the module number. The value of σ_z is always the same.

This reduction is only possible if alignment errors are not taken into account. Using the full size LUT, storing corrected information in the LUT can compensate the alignment errors of an individual module.

The z -co-ordinate is calculated according to (6.12), (6.13), (6.15), (6.17) and (6.18). A LUT is not used. The PSD of the implemented algorithm is given in figure 6.39 and figure 6.40.

⁷ A **pointer** is a variable containing as contents the memory address of another variable. The use of pointers is very common in C/C++ programming.

Search for first cluster on “ ϕ -layer”		
Check if cluster found		
No	Yes	
Create first entry in module list		
Set pointer to first cluster on “ ϕ -layer”		
Loop over clusters (starting from next cluster)		
Check if current cluster on different module		
No	Yes	
Check if current cluster on different layer		Set pointer to last cluster
No	Yes	Create new entry in module list
	Set pointer to last cluster on “ ϕ -layer”	Skip all clusters on “stereo-layer”
	Set pointer to first cluster on “stereo-layer”	Set pointer to first cluster on “ ϕ -layer”
Set pointer to last cluster		

Figure 6.39 PSD for building list with modules.

Create list of modules (figure 6.39)		
Loop over modules in list		
Calculate LUT offset for current module		
Loop over clusters in “ ϕ -layer”		
Calculate LUT index for current cluster		
Loop over clusters in “stereo-layer”		
Calculate position L according to (6.12) and (6.13)		
Check if position L in allowed range		
No	Yes	
	Create new space-point	
	Calculate ϕ , σ_ϕ , r , σ_r , σ_z -values via LUT.	
	Calculate z -position via (6.15)	

Figure 6.40 PSD for barrel SCT space-point conversion algorithm.

6.7.3 End-cap SCT

In the end-cap SCT, the local co-ordinates are given by (Φ, P) . Φ (mrad) gives the local position opposite to the strips; P (cm) gives the local position along the strips. For each cluster on the “ ϕ -layer” one can directly calculate the local co-ordinate Φ :

$$\Phi = (t - t_{centre}) \Phi_{strip} \quad (6.20)$$

with Φ_{strip} the strip pitch in mrad and t_{centre} the index of the most central strip. For calculating the local co-ordinate P one needs to combine a cluster from the “ ϕ -layer” (Φ_1) with a cluster from the “stereo-layer” (Φ_2). The local co-ordinates of the created space-point are given by (appendix E.2):

$$\begin{aligned} \bar{\Phi} &= \Phi_1 \\ P &= r_{mod} \frac{(\sin \Phi_2 - \sin(\Phi_2 + \alpha))}{\sin(\Phi_1 - (\Phi_2 + \alpha))} - r_{mod} \end{aligned} \quad (6.21)$$

with α the stereo angle of the strips on the “stereo layer”.

The conversion from local to global space-points (ϕ, r, z) is given by:

$$\begin{aligned} \phi &= \phi_{mod} + \bar{\Phi}_1 \\ r &= r_{mod} + P + r_{shift} \\ z &= z_{mod} \end{aligned} \quad (6.22)$$

r_{shift} is a correction for a shift due to the staggered structure of the modules [10]. For odd wheels, r_{shift} is given by [1]:

$$r_{shift} = -0.17 - 0.008 \times F \quad (6.23)$$

with all dimensions in cm. For even wheels, r_{shift} is given by:

$$r_{shift} = 0.22 + 0.008 \times F \quad (6.24)$$

The corresponding standard deviations are given by:

$$\begin{aligned} \sigma_\phi &= \frac{\Phi_{strip}}{\sqrt{12}} \\ \sigma_r &= 0.1 \\ \sigma_z &= 0.005 \end{aligned} \quad (6.25)$$

Implementation

The implementation is in principle very similar to the algorithm used for the barrel SCT. The size of the LUTs can however be much smaller because the conversion from local (Φ, P) co-ordinates to global (ϕ, r, z) co-ordinates is much easier. An extra step via global (x, y, z) co-ordinates, as for the barrel SCT, is not necessary. For each possible cluster position, the corresponding local position Φ , $\cos \Phi$ and $\sin \Phi$ are stored in a LUT. To make an implementation in LUTs easier, expression (6.21) is replaced by:

$$P = r_{mod} \frac{\sin \Phi_2 - \sin \Phi_2 \cos \alpha - \cos \Phi_2 \sin \alpha}{\sin \Phi_1 (\cos \Phi_2 \cos \alpha - \sin \Phi_2 \sin \alpha) - \cos \Phi_1 (\sin \Phi_2 \cos \alpha + \cos \Phi_2 \sin \alpha)} - r_{mod} \quad (6.26)$$

In the current implementation, a table of ($2 \times$ number of rings \times number of planes \times number of strips) items is needed (total size about 100 KB). The factor 2 in the expression for the table size is coming from the fact that there are two times more potential cluster positions than strips.

6.7.4 Pixel detector

Barrel pixel detector

For the pixel detectors, the cluster position (t, l) can be converted easily into the corresponding local co-ordinates (T, L) , with the position along the pixel columns given by L , and the position along the rows given by T (both in cm). The conversion is given by:

$$\begin{aligned} T &= (t - t_{centre}) T_{pixel} \\ L &= (l - l_{centre}) L_{pixel} \end{aligned} \quad (6.27)$$

with (T_{pixel}, L_{pixel}) the row and column pitch of the pixel detector and (t_{centre}, l_{centre}) the indices of the most central pixel.

The conversion from local to global co-ordinates is given by:

$$\begin{aligned} x &= x_{mod} - e_x (T + T_{shift}) \\ y &= y_{mod} + e_y (T + T_{shift}) \\ z &= z_{mod} + L \end{aligned} \quad (6.28)$$

with T_{shift} a correction for the Lorentz angle (fixed value 0.002 cm). The terms e_x and e_y are given by:

$$\begin{aligned} e_x &= \sin(\phi_{mod} + \tau) \\ e_y &= \cos(\phi_{mod} + \tau) \end{aligned} \quad (6.29)$$

with τ the tilt angle of the module.

The standard deviations are given by:

$$\begin{aligned} \sigma_\phi &= \frac{T_{pixel}}{r\sqrt{12}} \\ \sigma_r &= T_{pixel} \left(\frac{x e_x}{r} \oplus \frac{y e_y}{r} \right) \\ \sigma_z &= \frac{L_{pixel}}{\sqrt{12}} \end{aligned} \quad (6.30)$$

End-cap pixel detector

For disk 1 and 3, equation (6.27) gives the conversion between the cluster position and local co-ordinate. For disk 2, the readout direction is different and the conversion is given by:

$$\begin{aligned} T &= (t_{centre} - t) T_{pixel} \\ L &= (l - l_{centre}) L_{pixel} \end{aligned} \quad (6.31)$$

The conversion from local to global variables is given by:

$$\begin{aligned} x &= x_{mod} - e_x (T + T_{shift}) + e_y (L + L_{shift}) \\ y &= y_{mod} + e_y (T + T_{shift}) + e_x (L + L_{shift}) \\ z &= z_{mod} \end{aligned} \quad (6.32)$$

The meaning of the variables e_x and e_y is the same as for the barrel pixel detector⁸. T_{shift} and L_{shift} are empirical corrections to reproduce the results of full detector simulation. L_{shift} is given by -0.03 cm. For odd modules T_{shift} is given by -0.0023 cm, for even modules T_{shift} is given by 0.0023 cm.

The standard deviations are given by:

$$\begin{aligned}\sigma_\phi &= \frac{T_{pixel}}{r\sqrt{12}} \\ \sigma_r &= \left(\frac{x}{r} (e_x T_{pixel} - e_y L_{pixel}) \oplus \frac{y}{r} (e_y T_{pixel} + e_x L_{pixel}) \right) \\ \sigma_z &= 0.005\end{aligned}\tag{6.33}$$

Implementation

The implementation for the pixel detectors can be much simpler than the implementation for the SCT because there is no search for partner clusters. The implementation is based on a loop over the clusters, converting each cluster into a local (T, L) co-ordinate and a (x, y, z) space-point according to (6.28) and (6.32). The (x, y, z) co-ordinates are converted to (ϕ, r, z) co-ordinates via (6.11). To speed up the calculations, a small LUT is used to store for each module x_{mod} , y_{mod} , z_{mod} , e_x and e_y . The corresponding PSD is given in figure 6.41.

Loop over clusters	
Check if current cluster on different module	
No	Yes
Calculate LUT index for current module	
Transform cluster to global (x, y, z) space-point	
Transformation to global (ϕ, r, z) space-point	

Figure 6.41 PSD for pixel pre-processing algorithm.

6.7.5 Performance results

Degree of ambiguity

The degree of ambiguity (number of cluster on the “stereo-layer” matching with a cluster on the “ ϕ -layer”) for the barrel SCT is given in figure 6.42 (electron RoIs) and figure 6.43 (jet RoIs). The corresponding plots for the end-cap part are given in figure 6.44 (electron RoIs) and figure 6.45 (jet RoIs).

Frequently (about 20% of the clusters) the degree of ambiguity is more than one. About 9% of the clusters does not contain a partner cluster on the “stereo-layer”. In most cases these clusters are due to noise or the partner cluster is missing due to detector inefficiencies. The degree of ambiguity for jet RoIs and electron RoIs are almost the same.

⁸ A special readout option for the pixel detectors is a so-called “FAST OR”, summing the pixels along one row. In this case the pixel detectors behaves like a kind of strip detectors and the knowledge of the position along the rows L is lost. From (6.28) it follows that the barrel pixel detector provides enough information to calculate two-dimensional (x, y) or (ϕ, r) space-points. The end-cap can however not be used anymore. From (6.32) it follows that it is not possible to reconstruct with a high accuracy the positions (x, y) or (ϕ, r) .

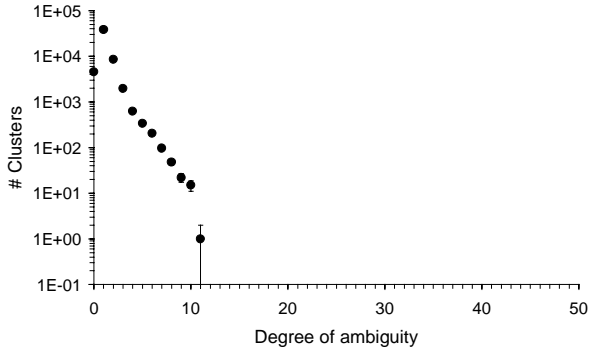


Figure 6.42 Degree of ambiguity, barrel SCT, electron RoIs.

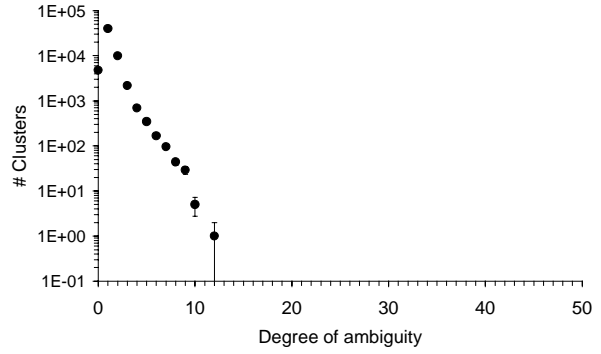


Figure 6.43 Degree of ambiguity, barrel SCT, jet RoIs.

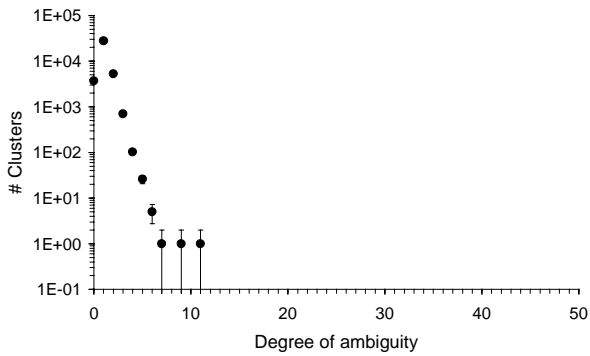


Figure 6.44 Degree of ambiguity, end-cap SCT, electron RoIs.

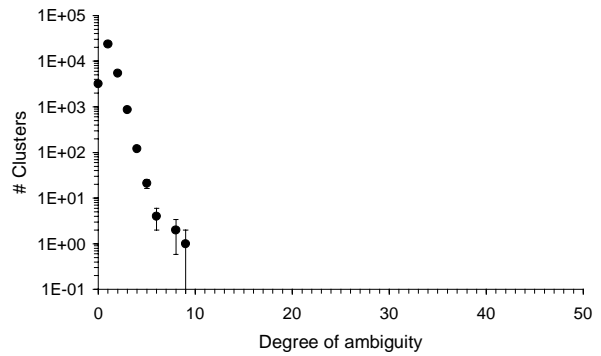


Figure 6.45 Degree of ambiguity, end-cap SCT, jet RoIs.

Average number of space-points per RoI

Table 6.6 summarises the average number of space-points in an RoI. The distribution for the total number of space-points in an RoI is also given in figure 6.46 (electron RoIs). This distribution resembles a Gaussian distribution, however with a large tail on the right side. The number of space-points for jet RoIs is slightly higher than for electron RoIs. On average only 8-10 space-points per RoI belong to the high- p_T electron. The other space-points are due to minimum bias background, noise and wrong combinations of clusters from the “ ϕ -layer” and “stereo-layer”.

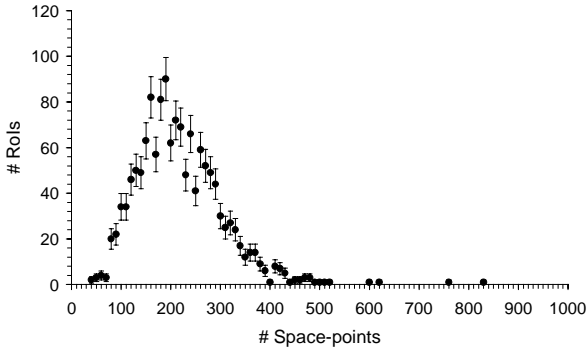


Figure 6.46 Total number of space-points in an RoI, electron RoIs.

Table 6.6 Number of space-points in a RoI.

	electron RoIs		jet RoIs	
	mean	max	mean	max
total	217	1034	251	757
barrel SCT	64	758	74	538
end-cap SCT	49	194	55	213
barrel pixels	128	360	147	387
end-cap pixels	23	79	22	17

6.7.6 Calculation times

The calculation times of the space-point conversion algorithm are given in table 6.7. The corresponding distributions are given in figure 6.47 to figure 6.50. A Gaussian distribution approximately describes the calculation times for the pixel detectors. Very large tails dominate the distributions for the calculation times of the SCT.

Although much simpler, the barrel pixel algorithm is on average much more time consuming than the SCT algorithm. This is mainly due to the time consuming calculation of the inverse tangent and square root in the transformation from (x, y, z) to (ϕ, r, z) co-ordinates. A LUT can be used to speed up the calculations for the pixel detectors although the results of full calculations would not be exactly reproduced in this case. Some preliminary timing measurements show that the mean calculation time for the barrel pixel detector will drop from 128 μs to about 77 μs (electron RoIs) when a LUT is used in the transformation from (x, y, z) to (ϕ, r, z) co-ordinates.

Due to the very low occupancy, resulting in a small number of clusters, the end-cap pixel calculation time is much less time consuming.

Table 6.7 Calculation times of the space-point conversion algorithm [μs].

	barrel SCT		end-cap SCT		barrel pixels		end-cap pixels	
	mean	max	mean	max	mean	max	mean	max
electron RoIs	77	580	93	372	128	335	33	86
jet RoIs	88	509	106	436	146	357	33	97

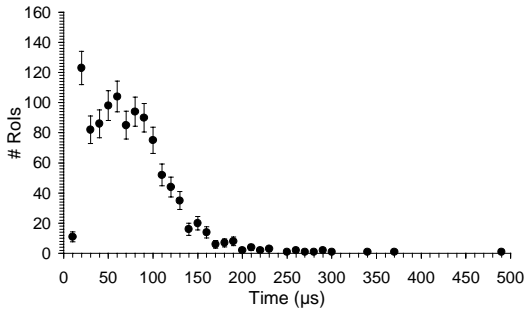


Figure 6.47 Calculation times of space-point conversion algorithm, barrel SCT, electron RoIs.

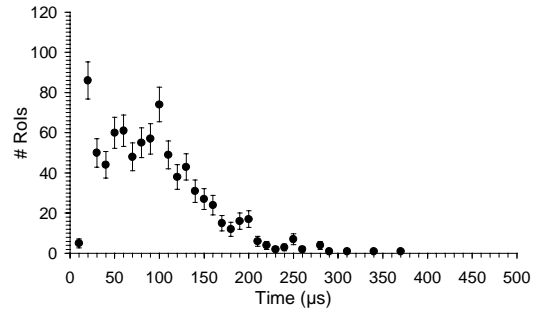


Figure 6.48 Calculation times of space-point conversion algorithm, end-cap SCT, electron RoIs.

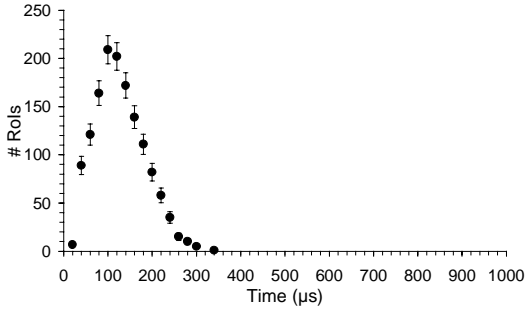


Figure 6.49 Calculation times of space-point conversion algorithm, barrel pixel detector, electron RoIs.

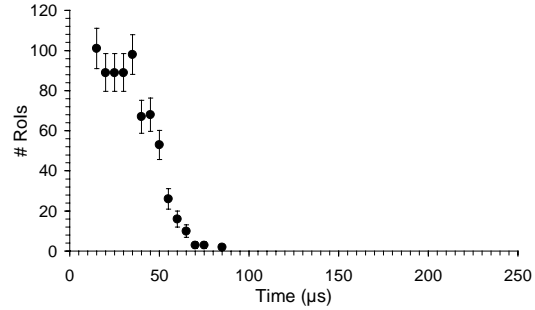


Figure 6.50 Calculation times of space-point conversion algorithm, end-cap pixel detector, electron RoIs.

Parameterisation of calculation times

The calculation time against the number of clusters for the different sub-detectors are given in figure 6.51 to figure 6.54. Theoretically the SCT cluster search algorithm scales quadratically with the occupancy. From the figures it follows that a parabola approximately describes the calculation time versus the number of clusters. For the pixel detectors, the calculation time scales approximately linearly with the number of clusters.

For the barrel SCT, the calculation time versus the number of clusters is given by (electron RoIs):

$$\text{time } [\mu\text{s}] = 20.58 + 0.244 \times (\#\text{Clusters}) + 0.0021 \times (\#\text{Clusters})^2 \quad (6.34)$$

For the end-cap SCT, the calculation time versus the number of clusters is given by (electron RoIs):

$$\text{time } [\mu\text{s}] = 13.09 + 0.6432 \times (\#\text{Clusters}) + 0.0021 \times (\#\text{Clusters})^2 \quad (6.35)$$

For the barrel pixel detector, the calculation time versus the number of clusters is given by (electron RoIs):

$$\text{time } [\mu\text{s}] = 7.61 + 0.939 \times (\#\text{Clusters}) \quad (6.36)$$

For the end-cap pixel detector, the calculation time versus the number of clusters is given by (electron RoIs):

$$\text{time } [\mu\text{s}] = 12.59 + 0.901 \times (\#\text{Clusters}) \quad (6.37)$$

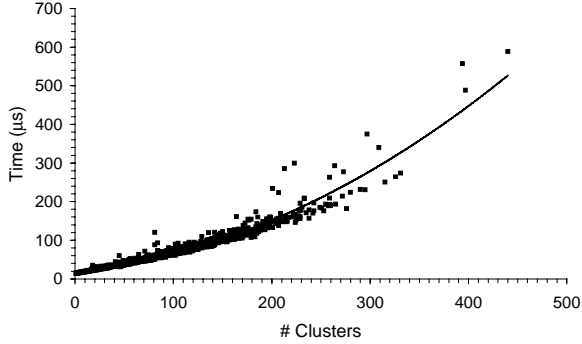


Figure 6.51 Calculation time space-point conversion algorithm versus number of clusters, barrel SCT, electron RoIs.

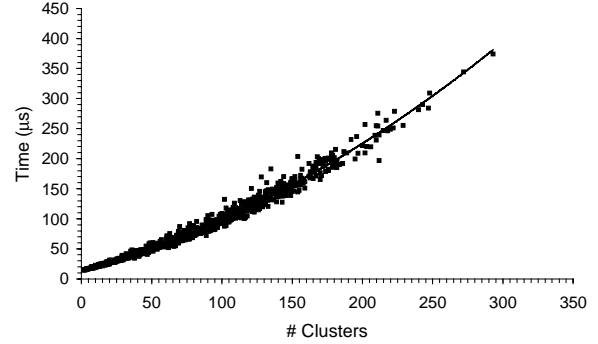


Figure 6.52 Calculation time space-point conversion algorithm versus number of clusters, end-cap SCT, electron RoIs.

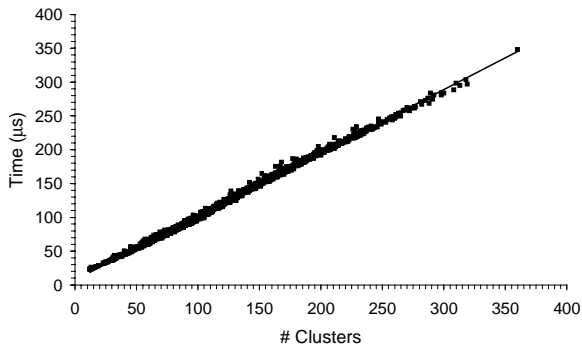


Figure 6.53 Calculation time space-point conversion algorithm versus number of clusters, barrel pixel detector, electron RoIs.

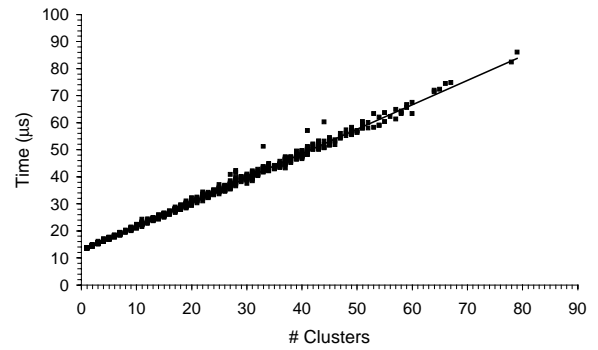


Figure 6.54 Calculation time space-point conversion algorithm versus number of clusters, end-cap pixel detector, electron RoIs.

6.8 Algorithm for selection of space-points

6.8.1 Implementation

Optionally, space-points are only stored, if their (ϕ, r, z) co-ordinates are within the (ϕ, r, z) range spanned by the RoI, taking into account the length of the interaction region along the beam axis.

Range in z barrel

For the barrel part the allowed range in z is a function of r and given by (see section 5.8):

$$\frac{r}{r_{cal}}(z_{calMin} + \Delta z) - \Delta z \leq z \leq \Delta z + \frac{r}{r_{cal}}(z_{calMax} - \Delta z) \quad (6.38)$$

with r_{cal} the radial position of the calorimeter and (z_{calMin}, z_{calMax}) the range in z spanned by the RoI for $r = r_{cal}$. The 2σ -value of the axial length of the LHC interaction region along the beam axis is given by $\Delta z = 11.2$ cm (see also section 5.8). For $r = 0$, the allowed z -range is equal to the 2σ -value of the axial length of the interaction region $(-\Delta z, \Delta z)$.

Range in r end-cap

For the end-cap part the allowed range in r is a function of z and given by:

$$r_{calMin} \frac{|z| - \Delta z}{z_{cal} - \Delta z} \leq r \leq r_{calMax} \frac{|z| + \Delta z}{z_{cal} + \Delta z} \quad (6.39)$$

The radial range at the calorimeter is given by (r_{calMin}, r_{calMax}) .

Range in ϕ

For each space-point the ϕ -value interpolated at the calorimeter is required to be in the range in ϕ spanned by the RoI. For the barrel part this requirement is given by:

$$\phi_{RoIMin} - \alpha_{rMax}(r_{cal} - r) \leq \phi \leq \phi_{RoIMax} + \alpha_{rMax}(r_{cal} - r) \quad (6.40)$$

with α_{rMax} the maximum allowed slope in the (ϕ, r) plane (based on the minimum p_T -value). For $r = r_{cal}$, the allowed range is equal to $(\phi_{RoIMin}, \phi_{RoIMax})$. For $r = 0$, the allowed range is equal to $(\phi_{RoIMin} - \alpha_{rMax}r_{cal}, \phi_{RoIMax} + \alpha_{rMax}r_{cal})$.

For the end-cap part a similar expression is valid with r replaced by z , and α_{rMax} replaced by α_{zmax} (maximum slope in the (ϕ, z) plane).

6.8.2 Performance results

The space-points created by the high- p_T electron can be deduced from a table stored in the data files. Only space-points matching in all three (ϕ, r, z) co-ordinates with the space-points stored in this table are regarded as created by the high- p_T electron. For space-points created by the high- p_T electron, the offset Δ_z in z with respect to the centre of the RoI is given in figure 6.55, with Δ_z defined as (see (6.38)):

$$\Delta_z = z - r \left(\frac{z_{calMin} + z_{calMax}}{2r_{cal}} \right) \quad (6.41)$$

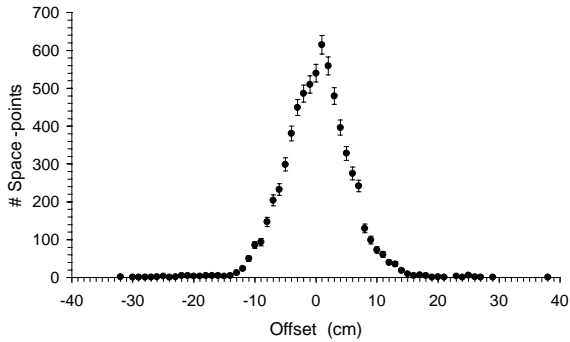
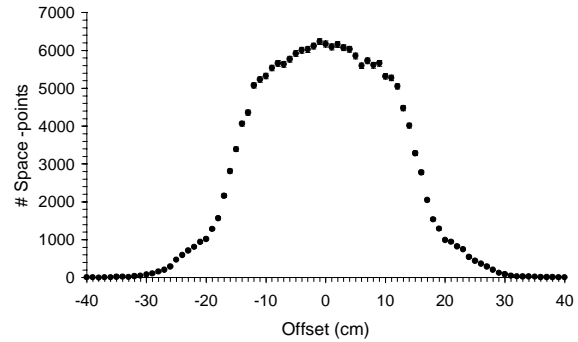
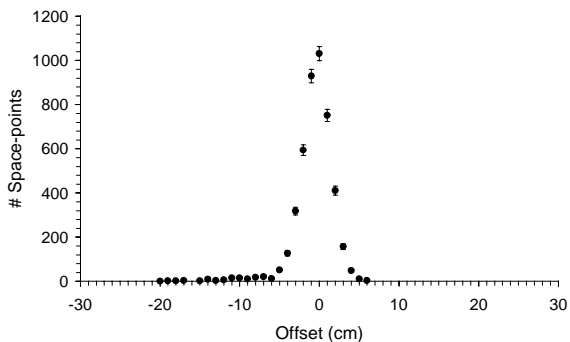
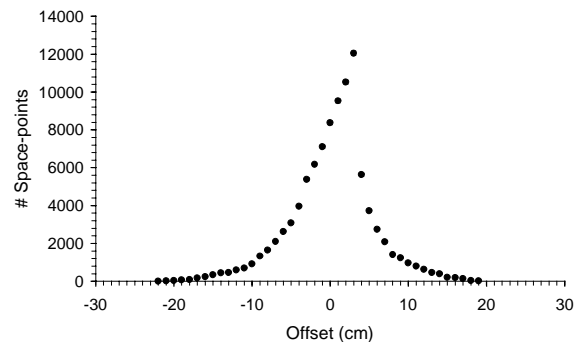
In figure 6.56 the offset is given for the space-points created by noise, the minimum bias background and the space-points created by a wrong combination of clusters from the “ ϕ -layer” and the “stereo-layer”. The corresponding plots for the end-cap region are given in figure 6.57 and figure 6.58 (all electron RoIs). In the figures it can be seen very clearly that the space-points created by the high- p_T electron are much more concentrated around the centre of the RoI.

The fraction of space-points in the allowed range, passing selection cuts (6.38), (6.39) and (6.40), is given in table 6.8. From this table it can be concluded that the selection algorithm is effective. Only 3% of the space-points created by the high- p_T electron do not satisfy the requirements, while almost 50% of the background space-points are rejected. These results can however be too optimistic because the axial length of the LHC interaction region along the beam axis is not correctly taken into account in the data samples currently available. More detailed studies with larger data samples correctly taking into account the LHC interaction region are necessary.

No separate calculation times of the selection algorithm are presented because it is integrated in the conversion to space-point algorithm further described in the previous section.

Table 6.8 Number of space-points passing the selection cuts.

	electron RoIs			jet RoIs		
	total	accepted	ratio [%]	total	accepted	ratio [%]
high- p_T electron	1.15×10^4	1.12×10^4	97.0	-	-	-
background	2.92×10^5	1.56×10^5	53.3	2.92×10^5	166×10^5	56.9

**Figure 6.55** Offset with the centre of the RoI for the space-points created by the high- p_T electron, barrel.**Figure 6.56** Offset with the centre of the RoI for the space-points created by the background, barrel.**Figure 6.57** Offset with the centre of the RoI for the space-points created by the high- p_T electron, end-cap.**Figure 6.58** Offset with the centre of the RoI for the space-points created by the background, end-cap.

6.9 Summary and conclusions

6.9.1 Total pre-processing calculation times

The total pre-processing calculation times with and without the data from the pixel detectors are summarised in table 6.9. The corresponding histograms are given in figure 6.59 and figure 6.60 (electron RoIs), together with the integrated fraction. This integrated fraction shows the fraction of the RoIs that falls within a certain calculation time.

Table 6.9 Total pre-processing calculation times [μ s].

	SCT + pixel detectors		SCT	
	mean	max	mean	max
electron RoIs	482	1.28×10^3	171	694
jet RoIs	545	1.09×10^3	199	679

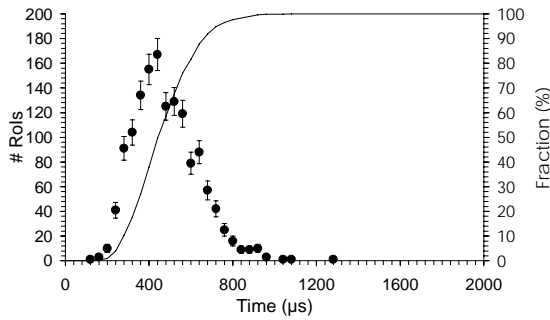


Figure 6.59 Calculation times of the total pre-processing, SCT + pixel detectors, electron RoIs.

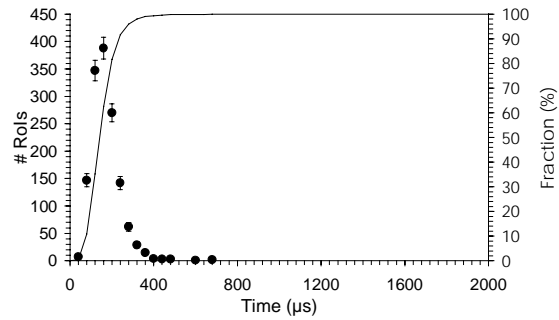


Figure 6.60 Calculation times of the total pre-processing, SCT, electron RoIs.

6.9.2 Addition of pixel detector

When the data from the pixel detectors is used in the LVL2 trigger, the pre-processing calculation time increases significantly (about a factor 3). With the data from the pixel detectors, 90% of the calculation times is less than about 650 μs (figure 6.59). Without the data from the pixel detectors, 90% is less than about 225 μs (figure 6.60).

The contribution from the pixel detector is mainly due to the barrel part where both the clustering and conversion to space-point algorithm are time consuming. The calculation time of the conversion to space-point algorithm can however be reduced significantly if a LUT is used for the transformation from (x, y, z) to (ϕ, r, z) co-ordinates.

The end-cap pixel detector has much less influence, due to the very low occupancy and small number of hits.

6.9.3 Implementation of pre-processing algorithm

The pre-processing can be implemented completely separately for each detector module⁹. This means that the pre-processing algorithm can be implemented on the FEX processors, on the processors of the ROBs, or on processors attached to the SFIs/RSIs (see chapter 5). A combination is also possible. At the moment it is assumed that the FEX processors do the complete pre-processing because the ROBs are already fully occupied with data input and output, and the pre-processing requires a lot of processing power, especially when the pixel detector is included.

Implementing the pre-processing on the FEX processors has some disadvantages. In this case a much bigger geometry table is needed containing information about the complete precision tracker instead of the modules attached to the current ROB only. Also the possibility exists that the same calculation has to be repeated because the same transformation may also be needed for the event filter. Depending on the architecture of the LVL2 system, the calculation has also to be repeated if one ROB receives more than one RoI request for the same event, but this is only a very small fraction (see also section 5.8.8).

Comparing table 6.2 with table 6.6 shows that for the electron RoIs the number of space-points is significantly less than the number of hits (a factor 10 for the SCT). The difference in data size is however much less because a hit can be stored in one word, while a space-point needs at least four words, i.e. the (ϕ, r, z) co-ordinates and an identification of the detector

⁹ The “ ϕ -layer” and “stereo-layer” of the SCT are regarded as one module in this case.

layer. This means that for the load on the switching network it does not make much difference whether the ROBs or the FEX processors do the pre-processing.

It can however be very attractive to implement the first step of the pre-processing algorithm (selection of hits from modules within the current RoI) in the ROBs or RSIs. Comparing the number of hits from modules within the RoI with the total number of hits stored in the ROBs that receive an RoI request in table 6.2 shows that this step reduces the data size with about a factor 3. The required processing power is minimal, even using a conservative estimate for the number of hits.

6.10 References

1. J.T.M. Baines, Private communication.
2. J.C. Vermeulen, Private communication.
3. ATLAS Trigger Performance Group, *Trigger Algorithms in ATRIG*, ATLAS DAQ-NO-060 (1996).
4. A. Bogaerts, R.J. Dankers, T. Hansl-Kozanecka, R. Hauser, J.C. Vermeulen and N. Woolley, *Physics Data Files for Trigger Level-2 Modelling, Emulation, Benchmarking of Algorithms and Demonstrators*, 1996.
5. ATLAS Inner Detector Community, *Inner Detector Technical Design Report Volume 1*, ATLAS TDR-4, CERN/LHCC 97-16 (1997).
6. F.J. Wickens, *Proposed Data Formats for T2 Demonstrator Program*, 1996.
7. R. Bock and P. LeDu, *Detector and Readout Specifications and Buffer-RoI Relations for the Level-2 Trigger Demonstrator Program*, ATLAS DAQ-NO-62 (1997).
8. V.A. Charlton, *Optimising the Grouping of Wafers in SCT Readout Buffers Using Simulated Annealing Methods*, ATLAS DAQ-NO-52 (1996).
9. V.A. Charlton, *Placing Wafers inside a Silicon Detector into Buffers*, 1996.
10. ATLAS Inner Detector Community, *ATLAS Inner Detector Technical Design Report Volume 2*, ATLAS TDR-5, CERN/LHCC 97-17 (1997).
11. S.J. Haywood, *Helix Fits with Stereo Measurements*, ATLAS INDET-NO-095 (1995).
12. S. George, J.R. Hubbard and J.C. Vermeulen, *Input Parameters for Modelling of the ATLAS Second Level Trigger*, ATLAS DAQ-NO-070 (1997).

Chapter 7

A LVL2 FEX algorithm for the precision tracker

Contents

7.1	Introduction	176
7.2	Trigger algorithm requirements for the inner detector	177
7.2.1	Introduction	177
7.2.2	High- p_T trigger	177
7.2.3	Low- p_T trigger	177
7.3	LVL2 track reconstruction algorithm implemented in SCTFEX	178
7.3.1	Introduction	178
7.3.2	Data files	178
7.3.3	Track equations	179
7.3.4	Histogramming algorithm	179
7.3.5	Least square fit algorithm	183
7.3.6	Algorithm options	186
7.3.7	Definition of acceptable and best fit combination	187
7.4	Determination of parameters	188
7.4.1	Introduction	188
7.4.2	Lower histogram threshold	189
7.4.3	Maximum allowed $\sigma^2 \chi^2$ -values	189
7.4.4	Maximum allowed longitudinal impact parameter z_0	191
7.4.5	Other parameters	192
7.5	Physics performance of the SCTFEX algorithm	192
7.5.1	Introduction	192
7.5.2	Number of bins selected by histogramming algorithm	193
7.5.3	Number of fit combinations	195
7.5.4	Quality of reconstructed tracks	196
7.5.5	Efficiency results	197
7.5.6	Conclusions on strategy	199
7.6	Benchmark results	199
7.7	Global algorithm	204
7.7.1	Introduction	204
7.7.2	Determination of parameters of global algorithm	205
7.7.3	Efficiency results	206
7.8	Comparison between the TDR algorithm and the SCTFEX algorithm	207
7.9	Comparison between the XKALMAN algorithm and the SCTFEX algorithm ..	209
7.9.1	Introduction	209

7.9.2	XKALMAN algorithm description	209
7.9.3	Differences between the XKALMAN and the SCTFEX algorithm.....	210
7.9.4	Performance of the XKALMAN algorithm.....	210
7.10	Conclusions and future work	210
7.11	References	211

7.1 Introduction

The ATLAS LVL2 algorithms can be divided in local parts and a global part. In the local parts the data corresponding to the RoIs defined by LVL1 or internally in LVL2 by the LVL2 TRT scan (chapter 5) is analysed, using track reconstruction algorithms for the inner detector and muon spectrometer data and clustering algorithms for the calorimeter data. These algorithms will in general be simpler and less iterative than the algorithms used for the offline analysis. They can not be too dependent on the ultimate resolution of the detector since the full set of calibration data (e.g. alignment offsets) may not be available at the trigger level. The data from each sub-detector is handled separately in the local part. The data from the different sub-detectors is not combined (except the pixel detector data and SCT data). To prevent loss of efficiency in the transition area between barrel and end-cap, the barrel and end-cap parts of the detector are regarded as one sub-detector. Some **features** of the reconstructed tracks and clusters are calculated, e.g. the pseudorapidity η , the transverse momentum p_T , the transverse energy E_T and the azimuth at the calorimeter ϕ_{cal} . In the global part the features belonging to the same event are combined into a final LVL2 acceptance/rejection decision.

In this chapter I present a LVL2 algorithm for the reconstruction of high- p_T tracks in the precision tracker, and the determination of track features. This algorithm is integrated with the pre-processing algorithm, described in the previous chapter, in the software package SCTFEX. The track reconstruction algorithm presented is developed for implementation in a processor farm. It is not directly suited for implementation on a FEX processor based on FPGA technology. A LVL2 algorithm, using data from the barrel SCT alone, developed for implementation on an FPGA processor is described in more detail in an older study [1]. In the algorithm presented in this chapter the barrel and end-cap parts are regarded as one sub-detector. This differs from some older studies where the barrel and end-cap parts of the precision tracker were based on different detector technologies [2, 3, 4, 5].

The LVL2 trigger requirements for the inner detector are described in section 7.2. The track reconstruction algorithm is described in section 7.3. The parameters determining the performance of the track reconstruction algorithm are described in section 7.4. The choice of parameter values used in the remaining part of the chapter is also motivated in this section. The efficiency of the algorithm for reconstructing high- p_T tracks and the rejection for background events are presented in section 7.5. Benchmarking results can be found in section 7.6. A very simple algorithm for the first global step is described in section 7.7. This algorithm combines the features calculated for the precision tracker and electromagnetic calorimeter and is suited for electron RoIs only. In section 7.8 the track reconstruction algorithm implemented in SCTFEX is compared with the FEX algorithm that has been used to produce the results for the TDR. In section 7.9 the algorithm is compared with the offline algorithm XKALMAN for the inner detector.

7.2 Trigger algorithm requirements for the inner detector

7.2.1 Introduction

The LVL2 trigger for the inner detector consists of a high- p_T trigger and a low- p_T trigger. It is important that the trigger selection criteria can be adapted when the LHC luminosity increases. It is foreseen that the physics channels will be given different weights depending on the luminosity. The low- p_T channels will be more important in the initial lower luminosity running.

7.2.2 High- p_T trigger

The high- p_T trigger consists of a search for high- p_T electron/muon tracks with $p_T > \sim 6$ GeV in RoIs defined by the LVL1 system. The corresponding trigger objects are described in chapter 5 and given in table 5.1 (MU6, EM20I, EM30I and MU20). This class of triggers is designed to accept channels containing high- p_T electrons or muons in the final stage, either from known physics sources, e.g. W , Z and top decay, or due to new physics, e.g. production of Higgs boson, production of new heavy gauge bosons W' and Z' , and SUSY. Within the trigger schemes currently under consideration, the triggers from single electromagnetic clusters dominate the LVL1 acceptance rate at high luminosity (table 5.2 and [6]). It is envisioned to use an electromagnetic calorimeter E_T -threshold of about 20 GeV (EM20I) for low luminosity running and about 30 GeV (EM30I) at the design luminosity.

At LVL2, the calorimeter trigger reduces the rate by a factor of about 10 (for an E_T -threshold of 30 GeV) compared to LVL1, while maintaining a high efficiency for isolated electrons and photons [6]. For the high- p_T triggers in the inner detector, the principal requirements arise from the need to reduce the rate of LVL1 high- E_T triggers from isolated electromagnetic clusters by rejecting background from minimum bias and jet events. This leads to the following two **physics requirements** [6]:

- Efficiency of better than 90% for reconstructing isolated high- p_T electron tracks, excluding the loss of tracks with very hard or multiple bremsstrahlung that may not be recovered offline.
- Reduction of the LVL2 acceptance rate by a factor of about 10 with respect to the rate of events passing the LVL2 calorimeter selection, at the design luminosity $L_{design} = 10^{34} \text{ cm}^{-2}\text{s}^{-1}$.

The efficiency of an algorithm for the reconstruction of isolated high- p_T tracks together with the efficiency for rejecting background events determine the **physics performance** of an algorithm.

7.2.3 Low- p_T trigger

The low- p_T trigger consists of an un-guided search (not based on RoIs) in the TRT for tracks with $p_T > \sim 0.5$ GeV, extrapolated into the precision detector when necessary. This trigger is designed for B-physics channels that will be accepted on the basis of topologies of low- p_T tracks. The goal is to maintain good efficiency for selected channels while keeping the LVL2 acceptance rate for these events below a maximum value of about 800 Hz. The precise requirements vary according to the channel to be selected [6]. A possible low- p_T LVL2 algorithm for the inner detector is described in more detail in the inner detector TDR [6].

7.3 LVL2 track reconstruction algorithm implemented in SCTFEX

7.3.1 Introduction

The track reconstruction algorithm implemented in SCTFEX is divided in two parts:

- A histogramming algorithm
- A combinatorial least square fit in the bin(s) selected by the histogramming algorithm

In the algorithm it is assumed that each RoI contains only one track created by a high- p_T particle. The histogramming algorithm is used to select sets of space-points (see chapter 6) to be passed on to the least square fit. The advantage of such an algorithm is that the execution time scales more slowly with occupancy than a combinatorial method. A similar histogramming algorithm developed for the TRT has been shown to scale typically as $M \log N$, with N the occupancy ([6], see also section 7.6). A combinatorial algorithm scales typically as N^2 . By fitting only to the space-points in selected bins of the histogram, the number of fit combinations and therefore the time required for the least square fit is significantly reduced. An alternative algorithm based purely on a least square fit and using only (r, ϕ) information from the barrel SCT has previously been studied [2, 7, 8].

In the following sections the track reconstruction algorithm is described in more detail. The data files used by the algorithm are described in section 7.3.2. The track equations on which the reconstruction algorithm is based are described in section 7.3.3. The histogramming algorithm is described in section 7.3.4. The combinatorial least square fit algorithm is described in section 7.3.5. Possible variations in the algorithm are finally described in section 7.3.6 and 7.3.7.

Attention has been paid to implement the algorithm as efficiently as possible and especially to avoid searches in large arrays and to avoid the necessity of resetting (initialising to zero) large arrays for each new RoI. On the computer platforms used, the time necessary for resetting large arrays can be several ms.

7.3.2 Data files

The data files are the same as those used for testing the pre-processing algorithm (chapter 6) and consist of a sample of 1421 electron RoIs and a background sample of 1179 jet RoIs.

The electron RoIs can be used for studying the efficiency of the algorithm for reconstructing high- p_T electron tracks. The efficiency of the algorithm for reconstructing high- p_T muon tracks is not investigated separately. The standalone (precision tracker only) efficiency will always be better for muon tracks than for electron tracks because muons do not suffer from bremsstrahlung energy losses.

The jet RoIs are identified by the LVL1 calorimeter trigger as electron RoIs and can be used for studying the rejection efficiency of the algorithm against background events. This rejection efficiency is an important factor for the reduction of the event rate that can be achieved by the LVL2 trigger, see also section 7.7.3. The LVL2 acceptance (output) rate can be calculated from the production cross section of the jet events and the reduction factor achieved by the algorithm implemented in SCTFEX. The LVL1 trigger rate for single electromagnetic clusters corresponding to this jet sample is about 16.7 kHz [9]. This is lower than the estimated trigger rate for high luminosity of 20 kHz given in table 5.2 due to single electromagnetic clusters (EM30I)¹.

¹ This contains the sum of the trigger menu items EM30I + J40, EM30I + 2J40, EM30I + 3J40, EM30I + 4J40 and EM30I + EM10I.

To make a detailed analysis of the algorithm performance possible, extra information is added to the data files. Especially useful is the information that associates each space-point with the particle that created it. That makes it possible to calculate the fraction of space-points created by the high- p_T electron and by the minimum bias background for each reconstructed track (see also section 7.5.4).

The results presented in this chapter are based on the full sample of space-points. The algorithm to reject a part of the space-points based on their (ϕ, r, z) co-ordinates, described in section 6.8 of the previous chapter, is not used.

7.3.3 Track equations

The trajectory of a charged high- p_T particle with parameters $(a_0, z_0, \phi_0, \cot\theta, Q/p_T)$ in a homogenous magnetic field along the beam axis is given by (see chapter 3, equation (3.6)):

$$\begin{aligned}\phi(r) &= \frac{-a_0}{r} + \phi_0 - \frac{r}{2R_{curv}} \\ z(r) &= z_0 + r \cot\theta\end{aligned}\tag{7.1}$$

With R_{curv} given by (equation (3.2)):

$$R_{curv} = \frac{p_T}{0.3 \cdot B \cdot Q}\tag{7.2}$$

with B the magnetic field along the beam axis.

In the algorithm implemented in SCTFEX, the trajectory is approximated by a straight line in the (r, ϕ) plane, the (z, ϕ) plane and the (z, r) plane. This is a valid approximation for a particle coming from the primary vertex ($r \gg a_0$):

$$\begin{aligned}\phi(r) &= \phi_0 - \frac{r}{2R_{curv}} = b_{r\phi} + a_{r\phi}r \\ r(z) &= -z_0 \tan\theta + z \tan\theta = b_{zr} + a_{zr}z \\ \phi(z) &= \phi_0 + \frac{z_0 \tan\theta}{2R_{curv}} - \frac{z \tan\theta}{2R_{curv}} = b_{z\phi} + a_{z\phi}z\end{aligned}\tag{7.3}$$

7.3.4 Histogramming algorithm

The track search is performed in the (r, ϕ) plane for RoIs defined by the barrel calorimeter $|\eta_{RoI}| \leq 1.5$, and in the (z, ϕ) plane for RoIs defined by the end-cap calorimeter $|\eta_{RoI}| > 1.5$. The reason is that for $|\eta_{RoI}| \leq 1.5$ the larger fraction of the space-points belongs to the barrel part with the best resolution in the (r, ϕ) plane, and for $|\eta_{RoI}| > 1.5$ the larger fraction belongs to the end-cap part with the best resolution in the (z, ϕ) plane.

A two-dimensional histogram is constructed of the number of detector layers² with at least one space-point in straight roads in the appropriate plane. Because a straight line can describe the trajectory of a charged high- p_T particle in the (z, ϕ) plane and the (r, ϕ) plane, the corresponding space-points should be found in the same histogram road. The roads are described by the ϕ -value of the intercept at the calorimeter (ϕ_{cal}) and the slope of their centre line (α). For an RoI in the barrel each histogram bin represents an interval in ϕ and an interval in

² The “ ϕ -layer” and “stereo-layer” are regarded as the same detection layer in this case.

$\alpha_r = d\phi/dr$. For an RoI in the end-cap each bin represents an interval in ϕ and an interval in $\alpha_z = d\phi/dz$. Each bin can be described by an index pair (i, j) . Index i determines the ϕ -interval. Index j determines the α -interval. The maximum absolute value of the slope α considered defines the minimum p_T -value of the search.

One has the possibility to use histogram bins overlapping or non-overlapping in ϕ . In the case of non-overlapping bins each bin represents an interval $\Delta\phi$ given by:

$$\Delta\phi = \frac{\phi_{RoIMax} - \phi_{RoIMin}}{N} \quad (7.4)$$

with N the number of ϕ -divisions and $(\phi_{RoIMin}, \phi_{RoIMax})$ the ϕ -range of the RoI (see chapter 5). In the case of overlapping bins the ϕ -interval spanned by a bin is two times larger (for the same number of ϕ -divisions), where each ϕ -value is always in two intervals (except at the outer ends of the RoI).

A drawback of non-overlapping bins is the eventual loss of tracks, when the track is reconstructed in two bins adjacent in ϕ . A drawback of using overlapping bins is the increase in calculation time of the histogramming algorithm. Also the calculation time of the least square fit algorithm will increase, because more bins will be selected (probably containing exactly the same space-points).

For even values of the bin index j , each bin represents a range in α around a central value:

$$\alpha(j) = j \frac{\alpha_{max}}{M} = j\Delta\alpha \quad (7.5)$$

with M the number of α -divisions and α_{max} the maximum allowed slope value (based on the minimum allowed p_T -value). For odd values of the bin index j , each bin represents a range in α around a central value:

$$\alpha(j) = -(j+1) \frac{\alpha_{max}}{M} = -(j+1)\Delta\alpha \quad (7.6)$$

This choice for assigning the j index causes an even j to correspond to a positively charged particle, while an odd j corresponds to a negatively charged particle. The bins are automatically ordered with respect to their p_T -value. Only the central values of α are used in the histogramming algorithm (discrete sampling). In the barrel area the maximum allowed slope value $\alpha_{r,max}$ is given by:

$$\alpha_{r,max} = \frac{0.15B}{p_{T,min}} \quad (7.7)$$

with $p_{T,min}$ the minimum p_T -value that can be reconstructed. In the end-cap area the maximum allowed value of the slope $\alpha_{z,max}$ is given by:

$$\alpha_{z,max} = \frac{0.15B \sinh|\eta_{RoI}|}{p_{T,min}} \quad (7.8)$$

with η_{RoI} the η -value of the RoI (centre position) delivered by the RoI builder.

A bin with index pair (i, j) represents an area in ϕ between (barrel):

$$\begin{aligned} \phi_{min}(i, j, r) &= \phi_{RoIMin} + \phi_{offset}(i) + i\Delta\phi + \alpha_r(j)(r_{cal} - r) \\ \phi_{max}(i, j, r) &= \phi_{RoIMin} + \phi_{offset}(i) + (i+1)\Delta\phi + \alpha_r(j)(r_{cal} - r) \end{aligned} \quad (7.9)$$

with r_{cal} the radial position of the calorimeter. In the case of overlapping bins, $\phi_{offset} = 0$ for even values of i and $\phi_{offset} = \Delta\phi/2$ for odd values of i . In the case of non-overlapping bins, $\phi_{offset} = 0$ for all values of i . A similar expression with r_{cal} replaced by z_{cal} , r replaced by z and α_r replaced by α_z , is valid in the end-cap part.

For RoIs defined by the barrel calorimeter, a space-point with co-ordinates $(r_{point}, \phi_{point}, z_{point})$ falls inside the range spanned by a bin with index pair (i, j) if:

$$\phi_{min}(i, j, r_{point}) \leq \phi_{point} \leq \phi_{max}(i, j, r_{point}) \tag{7.10}$$

For RoIs defined by the end-cap calorimeter, a space-point falls inside the range spanned by a bin with index pair (i, j) if:

$$\phi_{min}(i, j, z_{point}) \leq \phi_{point} \leq \phi_{max}(i, j, z_{point}) \tag{7.11}$$

An example of the histogramming algorithm is given in figure 7.1 showing five (numbered) detection layers in the (r, ϕ) plane containing eleven space-points (marked as black dots). Here, 2×8 ($\alpha \times \phi$) histogram roads are given, marked with dashed lines. The numbers of entries found in the histogram bins are given in figure 7.2. The histogram road with the maximum number of entries (four) is marked.

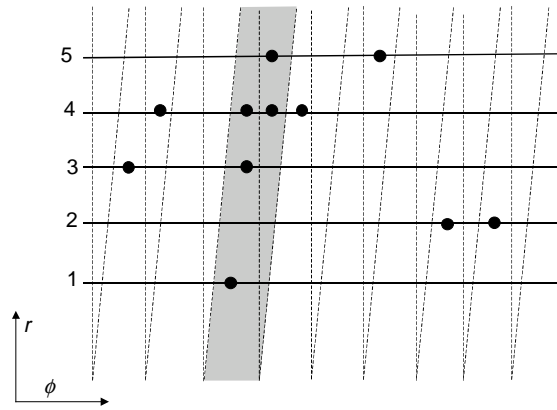


Figure 7.1 Schematic drawing of five (numbered) detection layers in the (r, ϕ) plane containing eleven space-points marked as black dots. Here 16 histogram roads (eight different offset values, two different slope values) are given, marked with dashed lines. The histogram road with the maximum number of entries is marked.

1	1	3	2	0	1	1	1
2	0	4	1	1	0	1	1

Figure 7.2 Number of entries (detection layers with at least one space-point in the current bin) found in the 2×8 histogram bins for the example of figure 7.1. The lower row corresponds to the larger α -value

Implementation of histogram

The histogram is implemented as an array of (number of ϕ -divisions \times number of α -divisions) elements, each containing the number of detection layers found with at least one

space-point in the current road and an identifier describing the current RoI. This identifier is added for efficiency reasons (otherwise the histogram has to be reinitialised completely for each new RoI).

Because a histogram is made of the number of detector layers with at least one space-point in the current road instead of a histogram of the total number of space-points found in the current road, an additional array is necessary. This array is used to check, before a histogram bin is updated, whether the same detector layer was already used in the current histogram bin. In the current implementation an array of (number of ϕ -divisions \times number of α -divisions \times number of planes) words is used for this test. In each array element an identifier describing the current RoI is stored, so that also this array needs to be initialised only once. For each space-point this array is checked whether the stored RoI identifier matches with the current RoI identifier. If not, the space-point is the first space-point found on the current detector layer for the current road, the histogram is updated and the RoI identifier is updated.

Selection of bins

One has the possibility to select all bins with at least the threshold value or only the bins with the highest number of entries (detection layers with at least one space-point in the current road). The selected bins are stored in a bi-directionally linked list. If the bin was not stored in the list, the bin is added at the end. If the bin was already stored, the bin is moved upwards in the list. The algorithm is implemented in such way that the selected bins are automatically ordered by number of entries. For the same number of entries, the bins are ordered by their index j (p_T -value). This means that the most promising bin (largest p_T -value with largest number of entries) is stored at the beginning of the list and the less promising bin (lowest p_T -value with the lowest number of entries) at the end of this list.

The **PSD (Program Structure Diagram)** for the histogramming algorithm is given in figure 7.3. The PSD for the linked list algorithm is given in figure 7.4.

Loop over j indices (α -positions, ordered by p_T -value)			
Loop over space-points			
Calculate interval of possible i indices (ϕ -positions) for current space-point and j index according to (7.10) and (7.11)			
Loop over possible i indices			
Calculate index of table			
Check if space-points on same detection layer already added to current histogram bin by checking RoI identifier stored in table			
No		Yes	
Update RoI identifier of table			
Calculate index of histogram			
Check if RoI identifier of histogram bin equal to current RoI identifier			
No		Yes	
First entry in current histogram bin for current RoI. Update RoI identifier of histogram bin		Increase number of entries in current histogram bin	
		Check if number of entries lower than threshold	
		No	Yes
		Update linked list of selected bins (figure 7.4)	
Loop over linked list of selected bins			
Calculate track parameters p_T, ϕ_0			
Increase current RoI identifier			

Figure 7.3 PSD for histogramming algorithm.

Check if number of entries is equal to the threshold			
No		Yes	
Bin was already stored in list. Determine old location		Check if linked list is empty	
Check if bin was already stored at beginning of list		No	Yes
No	Yes	Add bin to end of list	First bin found. Add it to list
Search upwards in list to first bin with higher number of entries	(bin is already stored at right location)		
Remove bin from old location. Re-map pointers. Special attention if bin was originally stored at end of list.			
Insert bin at new location. Re-map pointers. Special attention if bin now at beginning of list.			

Figure 7.4 PSD for algorithm for adding bins to the linked list of bins.

7.3.5 Least square fit algorithm

After the histogramming algorithm a least square fit algorithm is performed in the selected histogram bin(s). In the least square fit only one space-point per detection layer is used at the

same time. This means that in the case of several space-points on the same detection plane and reconstructed in the same bin, several combinations of space-points used in the least square fit are possible. When there are K bins found by the histogram algorithm, containing L_K detection planes, each of them containing N_{kl} space-points in the current bin ($N_{kl} > 0$), the total number of combinations N_{comb} is given by:

$$N_{comb} = \sum_{k=1}^K \left(\prod_{l=1}^{L_K} N_{kl} \right) \quad (7.12)$$

In most cases it is not necessary to investigate all combinations. The bins with a lower number of entries are only investigated if no acceptable combination has been found, see section 7.3.6. Depending on the strategy all combinations for a given number of entries are investigated and the best one is selected, or the algorithm stops as soon as an acceptable combination is found, see section 7.3.6.

A fast algorithm selects the space-points used in the current least square fit combination. This algorithm converts the number of the current combination into a set of indices each describing a space-point on a detection layer. It uses mainly division and modulus operations.

An example of the least square fit algorithm is given in figure 7.5, showing a (barrel) histogram road with six selected space-points in four layers, giving six combinations in total. Combination 4 is the best combination in this case (at least in the (r, ϕ) plane).

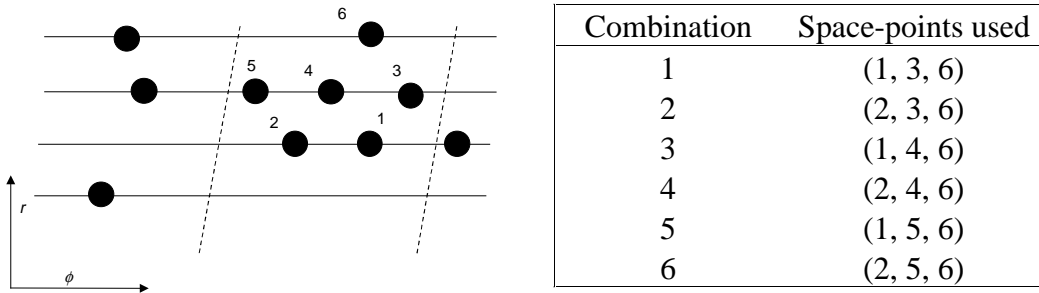


Figure 7.5 Schematic drawing of a (barrel) histogram road with four detector layers with six selected space-points in total, giving six least square fit combinations. The borders of the histogram road are marked with dashed lines. Four space-points falling outside the current histogram road are also shown.

For reasons of speed, all detection layers with at least one space-point found in the current histogram bin are currently used in the least square fit. It is not checked if the quality of the least square fit increases (smaller χ^2 -value per degree of freedom) if detection layers are left out. Due to detector inefficiencies potentially all space-points on a detection layer in a histogram bin can be created by minimum bias background. This means that even a track that is correctly reconstructed can contain wrong space-points (see also section 7.5.4). The PSD for the least square fit algorithm is given in figure 7.6.

Loop over selected histogram bins			
Stop or continue if current bin contains less entries and already a fit combination matching (7.18) and (7.19) found (depending on strategy used)			
Select space-points in current histogram bin			
Calculate number of combinations for current bin			
Check if number of combinations below maximum			
No		Yes	
		Loop over number of combinations	
		Select space-points used in current combination	
		Perform least square fit for current combination	
		Check if combination is current best one and matches (7.18) and (7.19)	
		No	Yes
			Update current best combination
		Stop or continue (depending on strategy used)	
Calculate parameters of final track for the best combination (if acceptable fit combination found)			

Figure 7.6 PSD for least square fit algorithm.

Least square fit equations

The fit itself is a standard least square fit for a data set consisting of N space-points with coordinates (r_i, z_i, ϕ_i) . The fit is performed in the (z, r) plane, the (r, ϕ) plane and the (z, ϕ) plane. In this case, the slope $a_{r\phi}$ and offset $b_{r\phi}$ in the (r, ϕ) plane are given by:

$$a_{r\phi} = \frac{1}{S_{tt}} \sum_{i=1}^N \phi_i \left(r_i - \frac{S_r}{N} \right) \quad (7.13)$$

$$b_{r\phi} = \frac{1}{N} \sum_{i=1}^N \phi_i - a_{r\phi} S_r$$

with S_r and S_{tt} given by:

$$S_r = \sum_{i=1}^N r_i \quad (7.14)$$

$$S_{tt} = \sum_{i=1}^N \left(r_i - \frac{S_r}{N} \right)^2$$

Similar expressions are valid in the (z, ϕ) plane and the (z, r) plane.

The quality of a least square fit is given by its χ^2 -value. In the (r, ϕ) plane, the $\chi_{r\phi}^2$ -value per degree of freedom is given by:

$$\chi_{r\phi}^2 = \frac{1}{N-2} \sum_{i=1}^N \frac{(\phi_i - (a_{r\phi} r_i + b_{r\phi}))^2}{\sigma_{\phi,i}^2} \quad (7.15)$$

The standard deviations of the space-points $\sigma_{\phi,i}$, $\sigma_{z,i}$ and $\sigma_{r,i}$ calculated by the pre-processing algorithm (equation (6.19), (6.25), (6.30) and (6.33) in chapter 6) and based on the intrinsic detector resolutions are not used in the current implementation. It is assumed that each space-point has the same standard deviations σ_{ϕ} , σ_z , and σ_r . In the algorithm currently implemented

in SCTFEX only the product $\sigma^2 \chi^2$ is calculated and used as measurement of the fit quality. In the (r, ϕ) plane, $\sigma_\phi^2 \chi_{r\phi}^2$ is given by:

$$\sigma_\phi^2 \chi_{r\phi}^2 = \frac{1}{N-2} \sum_{i=1}^N (\phi_i - (a_{r\phi} r_i + b_{r\phi}))^2 \quad (7.16)$$

In this case, the standard deviations of the slope $\sigma_{a,r\phi}$ and offset $\sigma_{b,r\phi}$ can be estimated using:

$$\begin{aligned} \sigma_{a,r\phi} &= \sqrt{\frac{\sigma_\phi^2 \chi_{r\phi}^2}{S_{tt}}} \\ \sigma_{b,r\phi} &= \sqrt{\frac{\sigma_\phi^2 \chi_{r\phi}^2}{N} \left(1 + \frac{S_r^2}{NS_{tt}}\right)} \end{aligned} \quad (7.17)$$

The reason to use $\sigma^2 \chi^2$ instead of χ^2 as a measurement of the fit quality is that it is very difficult to get realistic estimations for $\sigma_{\phi,i}$, $\sigma_{z,i}$ and $\sigma_{r,i}$. In this case $\sigma^2 \chi^2$ gives a more robust estimation of the fit quality than χ^2 [9]. The real values for $\sigma_{\phi,i}$, $\sigma_{z,i}$ and $\sigma_{r,i}$ can be much larger than the standard deviation values based on the intrinsic detector resolution and calculated in the previous chapter due to several reasons:

- In general the intrinsic detector resolution is only a small fraction of the total measurement error. The total measurement error is dominated by bremsstrahlung energy losses, detector inefficiencies, cluster reconstruction errors (the assignment of wrong hits to the reconstructed cluster), track reconstruction errors (the assignment of wrong space-points to the reconstructed track) and multiple scattering.
- Equation (7.15) is based on the assumption that the errors of the independent variable $\sigma_{r,i}$ are very small compared to the errors of the dependent (measured) variable $\sigma_{\phi,i}$. This is not the case when data from the barrel and end-cap detection layers is combined. For the end-cap detection layers, the independent variable is z instead of r and the $\sigma_{r,i}$ values can not be neglected as they have the same order of magnitude as the $\sigma_{\phi,i}$ values. A similar problem occurs for the fits in the (z, ϕ) plane and the (r, z) plane.

In chapter 3 this problem is solved via a transformation from z to r as independent variable for the end-cap detection layers using the relation $r = z \tan \theta$ (equation (3.18)). This method can not be used for the track reconstruction algorithm implemented in SCTFEX because $\tan \theta$ is not known. The value of $\tan \theta$ is actually determined via the fit.

The influence of $\sigma_{r,i}$ in equation (7.15) can also be included by using a modified (larger) value of $\sigma_{\phi,i}$. In this case the effective $\sigma_{\phi,i}$ value to be used in (7.15) can be much larger than the original $\sigma_{\phi,i}$ value following from the intrinsic detector resolution.

7.3.6 Algorithm options

Several variations in the algorithm are possible. Several numbers of histogram bins, two different detector layouts and several options to select the best fit combination have been tested.

Histogram bins

Both overlapping and not-overlapping bins in ϕ are tested. Two different bin sizes in α are tested, giving four combinations in total:

- The bins of the histogram are not overlapping in ϕ , 101×100 bins ($\alpha \times \phi$).
- The bins of the histogram are overlapping in ϕ , 101×199 bins.
- The bins of the histogram are not overlapping in ϕ , 25×100 bins.
- The bins of the histogram are overlapping in ϕ , 25×199 bins.

The limits in α and ϕ are the same, the bin sizes are changed. The numbers of bins are chosen to be compatible with the number of bins used in the TDR algorithm ([6], section 7.8).

Detector layout

- Both data from the SCT and pixel detectors is used. The barrel pixel detector always includes the B-layer (see chapter 2).
- Only data from the SCT is used.

Selection of best combination

- Only the histogram bins with the highest number of entries (detection layers with at least one space-point in the current road) are selected. All combinations in this bin are tested in the least square fit algorithm and the best combination is selected (see below).
- Only the histogram bins with the highest number of entries are selected. The search stops as soon as an acceptable combination is found (see below).
- All bins with at least the threshold value are selected (ordered by number of entries). If for a given number of entries no acceptable combination is found, the bins with a lower number of entries are investigated. For a given number of entries all combinations are tested in the least square fit algorithm and the best combination is selected.
- As above. The search stops as soon as an acceptable combination is found.

7.3.7 Definition of acceptable and best fit combination

Definition of acceptable combination

A combination of selected space-points is regarded as acceptable if the $\sigma^2 \chi^2$ -values of the corresponding least square fits in the individual planes do not exceed a maximum:

$$\begin{aligned} \sigma_\phi^2 \chi_{z\phi}^2 &< \sigma_\phi^2 \chi_{z\phi,max}^2 \\ \sigma_r^2 \chi_{zr}^2 &< \sigma_r^2 \chi_{zr,max}^2 \\ \sigma_\phi^2 \chi_{r\phi}^2 &< \sigma_\phi^2 \chi_{r\phi,max}^2 \end{aligned} \tag{7.18}$$

An extra requirement is given by the fact that high- p_T electrons are created in the primary vertex or very close to the primary vertex, resulting in a maximum allowed value for the longitudinal impact parameter z_0 (equation (7.3)):

$$\left| \frac{b_{zr}}{a_{zr}} \right| = \left| \frac{z_0 \tan \theta}{\tan \theta} \right| = |z_0| < z_{max} \tag{7.19}$$

A problem is to determine reasonable values for $\sigma_\phi^2 \chi_{z\phi,max}^2$, $\sigma_r^2 \chi_{zr,max}^2$, $\sigma_\phi^2 \chi_{r\phi,max}^2$ and z_{max} rejecting as much as possible of the background while keeping a good efficiency for RoIs from high- p_T electrons. The values actually used in SCTFEX are discussed in section 7.4.

Determination of best combination of selected space-points

A quantity determining the total quality of a combination of selected space-points (three least square fits) is necessary to be able to compare different combinations and to determine the combination of selected space-points giving the best least square fits in the three planes. If the

errors $\sigma_{\phi,i}$, $\sigma_{r,i}$ and $\sigma_{z,i}$ are known, the best combination is the combination with the smallest total χ^2 -value:

$$\chi_{total}^2 = \chi_{r\phi}^2 + \chi_{zr}^2 + \chi_{z\phi}^2 \quad (7.20)$$

Because only the values for $\sigma^2\chi^2$ are known this expression can not be used and a different method is necessary to compare the quality of different combinations.

In an older version of the SCTFEX algorithm combination 1 is regarded better as combination 2 if:

$$\frac{\sigma_{\phi}^2\chi_{r\phi,1}^2}{\sigma_{\phi}^2\chi_{r\phi,2}^2} + \frac{\sigma_r^2\chi_{zr,1}^2}{\sigma_r^2\chi_{zr,2}^2} + \frac{\sigma_{\phi}^2\chi_{z\phi,1}^2}{\sigma_{\phi}^2\chi_{z\phi,2}^2} < 3 \quad (7.21)$$

with combination 2 the currently best combination. A serious drawback of this method is that not only the $\sigma^2\chi^2$ -values of the corresponding fits determine which combination is finally selected but also the order in which the combinations are checked.

Another possibility is that the $\sigma^2\chi^2$ -value of the fit in the plane with the best resolution determines which combination is the best one. This means that for RoIs defined by the barrel calorimeter the best combination is the combination of space-points giving the best fit in the (r, ϕ) plane. For RoIs defined by the end-cap calorimeter, the best combination is the combination of space-points giving the best fit in the (z, ϕ) plane. A serious drawback of this method is that it is not possible to distinguish between combinations with for example almost the same values for $\sigma_{\phi}^2\chi_{r\phi}^2$, but significantly different values for $\sigma_{\phi}^2\chi_{z\phi}^2$ and $\sigma_r^2\chi_{zr}^2$. This occurs frequently due to the combination of clusters from the “ ϕ -layer” and “stereo-layer” of the SCT, potentially giving space-points with the same (r, ϕ) co-ordinates but different z -coordinates (barrel area).

A similar method is used by the algorithm described in the inner detector TDR ([6], see also section 7.8). In the TDR algorithm however, the best combination is always the combination of space-points giving the best fit in the (r, ϕ) plane, also for RoIs defined by the end-cap calorimeter.

In the algorithm currently implemented in SCTFEX, the best combination is the combination with the smallest quality factor Q [mrad^4cm^2], with Q defined as the multiplied $\sigma^2\chi^2$ -value for the corresponding least square fits in the three planes:

$$Q = \sigma_{\phi}^2\chi_{r\phi}^2\sigma_r^2\chi_{zr}^2\sigma_{\phi}^2\chi_{z\phi}^2 \quad (7.22)$$

This quality factor is somewhat empirical and only used for comparing different combinations.

7.4 Determination of parameters

7.4.1 Introduction

The performance of the algorithm implemented in SCTFEX depends strongly on the value of several parameters. For the best performance these parameters must be optimised. These parameters are the lower histogram threshold (section 7.4.2), the maximum allowed $\sigma^2\chi^2$ -values in the three planes (section 7.4.3) and the maximum value for the longitudinal impact parameter z_0 (section 7.4.4). The choices for the other parameters are described in section 7.4.5.

7.4.2 Lower histogram threshold

Distributions for the maximum number of entries found in a histogram bin are given in figure 7.7 (electron RoIs, 101×100 bins) and figure 7.8 (jet RoIs, 101×100 bins). The total number of entries in these distributions is equal to the total number of RoIs (1421 electron RoIs, 1179 jet RoIs).

The number of detection layers with at least one space-point in the current road is in general higher for electron RoIs than for jet RoIs. When the data from the pixel detectors is used in the LVL2 trigger one can use a lower threshold of four or five. A lower histogram threshold of five is the most attractive and used in the remaining part of this chapter, because 13% of the jet RoIs is rejected, while keeping a good efficiency for electron RoIs (99.5%). With a lower histogram threshold of four almost no jet RoIs (less than 1%) are rejected. Without using the data from the pixel detectors one has to use a lower threshold of three, otherwise the loss of efficiency for electron RoIs is too large (14%).

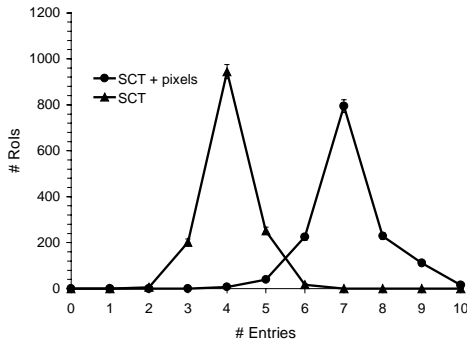


Figure 7.7 Maximum number of entries found in a histogram bin, 101×100 bins, electron RoIs.

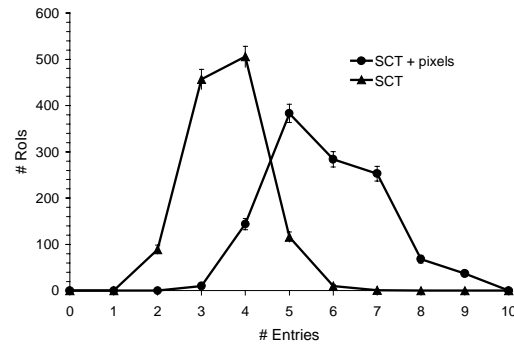


Figure 7.8 Maximum number of entries found in a histogram bin, 101×100 bins, jet RoIs.

7.4.3 Maximum allowed $\sigma^2 \chi^2$ -values

A combination of selected space-points is only accepted if the $\sigma^2 \chi^2$ -values of the corresponding least square fits in the individual planes do not exceed a maximum, see equation (7.18). A problem is to determine reasonable values for $\sigma_\phi^2 \chi_{r\phi, \max}^2$, $\sigma_\phi^2 \chi_{z\phi, \max}^2$ and $\sigma_r^2 \chi_{zr, \max}^2$. The largest possible fraction of the background should be rejected, while keeping a good efficiency for the RoIs created by high- p_T electrons. Different values for $\sigma^2 \chi_{\max}^2$ are used for RoIs defined by the barrel calorimeter and RoIs defined by the end-cap calorimeter. However the same values for $\sigma^2 \chi_{\max}^2$ are used for all the algorithm options described in section 7.3.6. This means that the values for $\sigma^2 \chi_{\max}^2$ used are not necessarily the most optimal choice for a specific option. The values for $\sigma^2 \chi_{\max}^2$ used are determined by comparing the distributions of the $\sigma^2 \chi^2$ -values in the three planes for electron RoIs with the corresponding distributions for jet RoIs.

The distributions for the $\sigma^2 \chi^2$ -values in the three planes ($\sigma_\phi^2 \chi_{r\phi}^2$, $\sigma_\phi^2 \chi_{z\phi}^2$ and $\sigma_r^2 \chi_{zr}^2$) are given in figure 7.9 to figure 7.14, both for electron RoIs and jet RoIs (101×100 bins, SCT + pixel detectors). RoIs defined by the barrel and end-cap calorimeter are presented separately. The total number of entries in these distributions is equal to the number of RoIs for which at least one bin with at least the threshold value is found by the histogramming algorithm. These

$\sigma^2\chi^2$ -values are found by investigating all combinations in the selected histogram bins (the bins with the maximum number of entries) and selecting the combination with the smallest quality factor Q (7.22) taking into account $|z_0| < z_{max}$ (see section 7.4.4).

For RoIs defined by the barrel calorimeter, the least square fit in (r, ϕ) plane has the smallest $\sigma^2\chi^2$ -value. This is what one can expect because the (r, ϕ) plane is the plane with the best resolution for the barrel detection layers. For RoIs defined by the end-cap calorimeter, the least square fit in the (z, ϕ) plane has the smallest $\sigma^2\chi^2$ -value. The $\sigma^2\chi^2$ -values for electron RoIs are slightly smaller than for jet RoIs. The biggest difference occurs for the (z, r) plane. However only rather soft cuts are possible, otherwise the loss of efficiency for the electron RoIs is too large. The average and maximal $\sigma^2\chi^2$ -values found are summarised in table 7.1, together with the values for $\sigma^2\chi_{max}^2$ used in the remaining part of this chapter, and the fraction of the RoIs with $\sigma^2\chi^2 > \sigma^2\chi_{max}^2$. With the values of $\sigma^2\chi_{max}^2$ currently used, this fraction is significantly larger for jet RoIs than for electron RoIs.

For RoIs defined by the end-cap calorimeter, the values for $\sigma^2\chi_{max}^2$ used are almost similar to the values used by the TDR algorithm, see also section 7.8. For RoIs defined by the barrel calorimeter, the cut on $\sigma_{\phi}^2\chi_{r\phi,max}^2$ is somewhat more stringent, while the cuts on $\sigma_{\phi}^2\chi_{z\phi,max}^2$ and $\sigma_r^2\chi_{zr,max}^2$ are somewhat less stringent, to take into account the better resolution in the (r, ϕ) plane and the worse resolution in the (z, ϕ) plane.

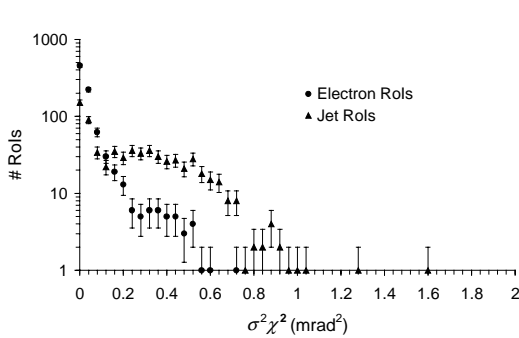


Figure 7.9 $\sigma_{\phi}^2\chi_{r\phi}^2$, barrel (101×100 bins, SCT + pixel detectors).

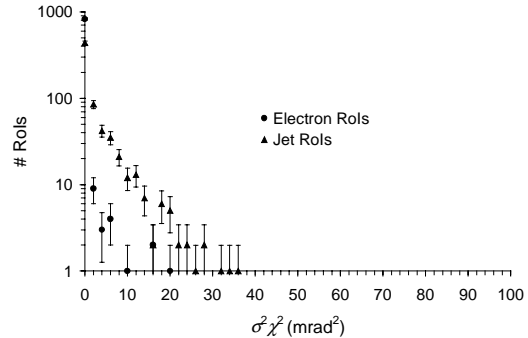


Figure 7.10 $\sigma_{\phi}^2\chi_{z\phi}^2$, barrel (101×100 bins, SCT + pixel detectors).

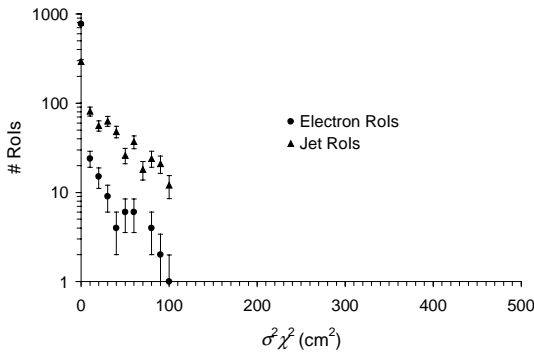


Figure 7.11 $\sigma_r^2\chi_{zr}^2$, barrel (101×100 bins, SCT + pixel detectors).

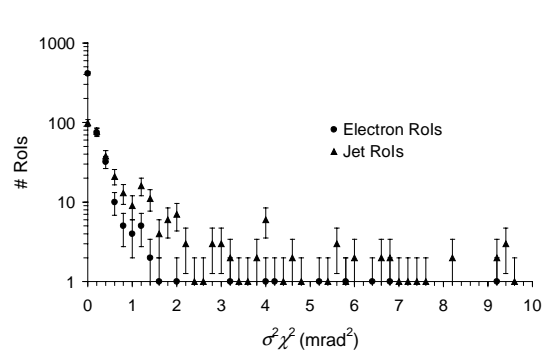


Figure 7.12 $\sigma_{\phi}^2\chi_{r\phi}^2$, end-cap (101×100 bins, SCT + pixel detectors).

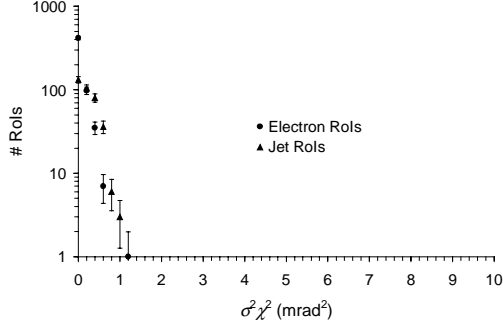


Figure 7.13 $\sigma_\phi^2 \chi_{z\phi}^2$, end-cap (101×100 bins, SCT + pixel detectors).

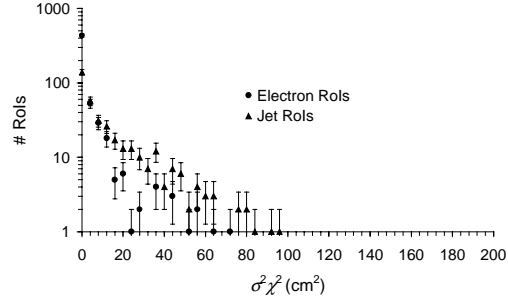


Figure 7.14 $\sigma_r^2 \chi_{zr}^2$, end-cap (101×100 bins, SCT + pixel detectors).

Table 7.1 Fit errors $\sigma^2 \chi^2$ (highest bin(s) selected, all combinations tested, 101×100 bins, SCT + pixels).

	electron RoIs			jet RoIs			$\sigma^2 \chi_{max}^2$
	mean	max	fraction [%] $\sigma^2 \chi^2 > \sigma^2 \chi_{max}^2$	mean	max	fraction [%] $\sigma^2 \chi^2 > \sigma^2 \chi_{max}^2$	
$\sigma_\phi^2 \chi_{r\phi}^2$ barrel [mrad ²]	0.05	0.74	0	0.24	1.60	0.5	1
$\sigma_\phi^2 \chi_{z\phi}^2$ barrel [mrad ²]	0.19	20.1	0.4	2.5	36.0	6.4	10
$\sigma_r^2 \chi_{zr}^2$ barrel [cm ²]	2.7	99.8	1.5	23.1	99.5	16.5	50
$\sigma_\phi^2 \chi_{r\phi}^2$ end-cap [mrad ²]	0.20	9.3	1.5	1.5	27.6	20.5	1.5
$\sigma_\phi^2 \chi_{z\phi}^2$ end-cap [mrad ²]	0.09	1.12	0	0.24	1.07	0	4
$\sigma_r^2 \chi_{zr}^2$ end-cap [cm ²]	2.8	71.1	4	12.9	94.6	25	15

7.4.4 Maximum allowed longitudinal impact parameter z_0

From the fit in the (z, r) plane the position of the longitudinal impact parameter z_0 can be reconstructed, see equation (7.19). The distributions for the reconstructed position of the longitudinal impact parameter z_0 for the combination with the smallest quality factor Q taking into account $\sigma_\phi^2 \chi_{r\phi, max}^2$, $\sigma_\phi^2 \chi_{z\phi, max}^2$ and $\sigma_r^2 \chi_{zr, max}^2$ are given in figure 7.15 (101×100 bins, SCT + pixels, only bins with maximum number of entries selected, all combinations tested). The total number of entries in these distributions is equal to the number of RoIs for which at least one histogram bin with at least the threshold value and a combination of least square fits matching (7.18) is found.

The fraction of events with $|z_0| > 20$ cm is slightly larger for jet RoIs than for electron RoIs. However only a rather soft cut is possible, otherwise the loss of electron RoIs will be too large. The results are summarised in table 7.2. In the remaining part of this chapter a cut $z_{max} = 25$ cm is used. This is larger than the 2σ -value of the LHC interaction region along the

beam axis (11.2 cm). For about 0.25% of the electron RoIs $|z_0| > z_{max}$. For about 1.5% of the jet RoIs $|z_0| > z_{max}$. The same cut is used for all the implementation options described in section 7.3.6. The value for z_{max} used is much more stringent than the value used by the algorithm described in the TDR ($z_{max} = 10$ m).

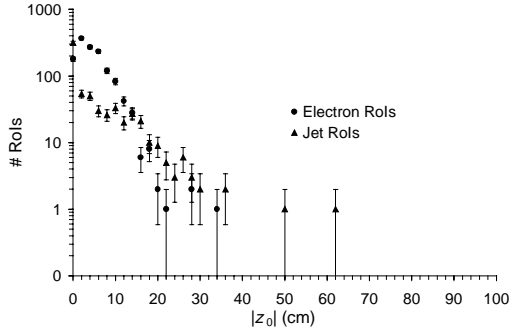


Figure 7.15 Longitudinal impact parameter $|z_0|$, 101×100 bins, highest bin(s) selected, all combinations tested, SCT + pixel detectors.

Table 7.2 Longitudinal impact parameter $|z_0|$ for electron RoIs and jet RoIs (101×100 bins, highest bin(s) selected, all combinations tested, SCT + pixel detectors).

electron RoIs		jet RoIs		z_{max} [cm]
mean [cm]	max [cm]	mean [cm]	max [cm]	
4.7	34.9	4.9	61.2	25.0

7.4.5 Other parameters

The radial position of the barrel calorimeter r_{cal} is given by 140 cm, the z -position of the end-cap calorimeter z_{cal} is given by 360 cm. To prevent very long calculation times the combinatorial least square fit algorithm is only performed if the total number of fit combinations in the current bin is below 1000. Otherwise the current bin is not investigated further. The minimum reconstructed $p_{T,min}$ determining the maximum slope in the histogram (equation (7.7) and (7.8)) is $p_{T,min} = 5$ GeV. A reconstructed track is accepted if the reconstructed p_T is above $p_T = 7$ GeV. This lower p_T -threshold of 7 GeV is chosen to be compatible with the algorithm described in the TDR [6] and different from the lower LVL1 p_T -threshold of 6 GeV mentioned in section 7.2 (MU6).

7.5 Physics performance of the SCTFEX algorithm

7.5.1 Introduction

In the following sections I present the physics performance of the track reconstruction algorithm implemented in SCTFEX. The number of bins selected by the histogramming algorithm and the number of least square fit combinations investigated are presented in section 7.5.2 and section 7.5.3. These numbers have an important impact on the calculation time of the algorithm. The quality of the reconstructed track is presented in section 7.5.4. The efficiency of the algorithm for reconstructing high- p_T tracks together with the efficiency for rejecting background events is presented in section 7.5.5. Conclusions on the best strategy are finally presented in section 7.5.6.

As described in section 7.3.6 the performance of the SCTFEX algorithm depends strongly on the value of several parameters. The values of the parameters used are summarised in table 7.3.

Table 7.3 Parameters used in the SCTFEX algorithm (see section 7.3.6).

	SCT + pixel detectors		SCT	
	barrel	end-cap	barrel	end-cap
lower histogram threshold	5	5	3	3
$\sigma_\phi^2 \chi_{r\phi,max}^2$ [mrad ²]	1	1.5	1	1.5
$\sigma_\phi^2 \chi_{z\phi,max}^2$ [mrad ²]	10	4	10	4
$\sigma_r^2 \chi_{zr,max}^2$ [cm ²]	50	15	50	15
z_{max} [cm]	25.0	25.0	25.0	25.0
r_{cal} [cm]	140	-	140	-
z_{cal} [cm]	-	360	-	360
$p_{T,min}$ [GeV]	5	5	5	5
$p_{T,threshold}$ [GeV]	7	7	7	7
maximum number of combinations per bin	1000	1000	1000	1000

7.5.2 Number of bins selected by histogramming algorithm

The distributions for the number of bins selected by the histogramming algorithm are given in figure 7.16 (SCT + pixel detectors, only bins with maximum number of entries³ selected), figure 7.17 (SCT + pixel detectors, all bins with at least the threshold value selected), figure 7.18 (SCT, only bins with maximum number of entries selected) and figure 7.19 (SCT, all bins with at least the threshold value selected). All distributions are for electron RoIs and based on 101×199 (overlapping) or 101×100 (non-overlapping) bins. The total number of entries in these distributions is equal to the total number of RoIs. The results are summarised in table 7.4. Also the results for jet RoIs are given in this table.

The average number of selected bins in jet RoIs is about 20-30% higher than in electron RoIs. The maximum number of selected bins can be significantly larger for jet RoIs than for electron RoIs (up to a factor 4), especially when only the bins with the maximum number of entries are selected. In the case of overlapping bins the number of selected bins is always about a factor 2 larger compared to non-overlapping bins. The number of selected bins increases significantly (up to a factor 7) when all bins with at least the threshold value are selected instead of only the bins with the maximum number of entries. Due to the lower threshold, the number of selected bins increases significantly (up to a factor 4) when the data from the pixel detectors is not used in the LVL2 trigger. For the worst combination (101×199 bins, SCT only, all bins with at least the threshold value selected), the histogramming algorithm selects up to 10% of all bins.

³ detection layers with at least one space-point in the current road

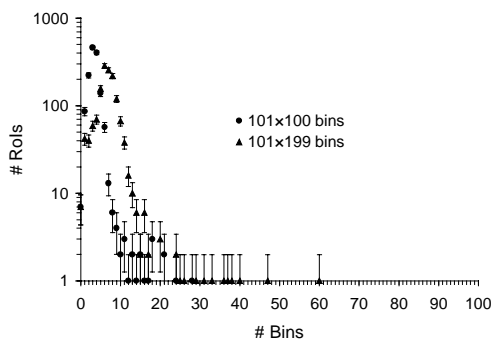


Figure 7.16 Number of histogram bins selected (bins with maximum number of entries), SCT + pixel detectors, electron RoIs.

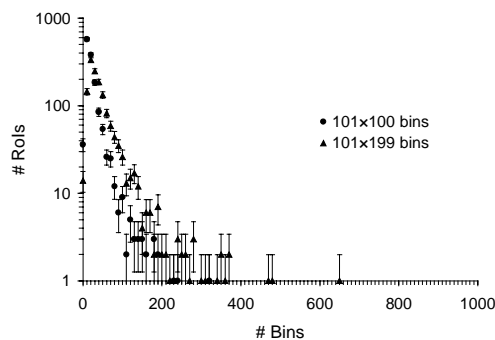


Figure 7.17 Number of histogram bins selected (all bins with at least the threshold value), SCT + pixel detectors, electron RoIs.

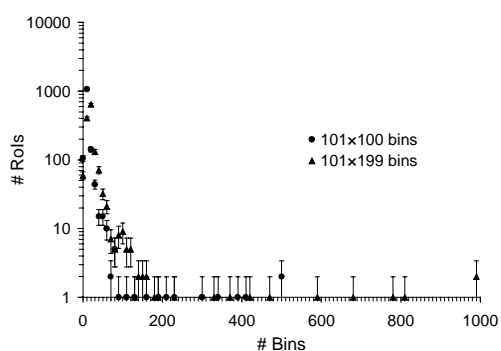


Figure 7.18 Number of histogram bins selected (bins with maximum number of entries), SCT, electron RoIs.

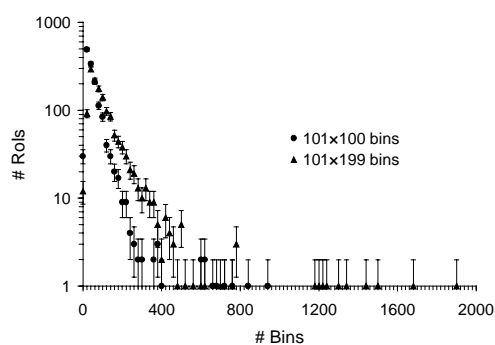


Figure 7.19 Number of histogram bins selected (all bins with at least the threshold value), SCT, electron RoIs.

Table 7.4 Number of histogram bins selected.

	only bins with maximum number of entries selected		all bins with at least the threshold value selected	
	mean	max	mean	max
electron RoIs				
101 × 100 bins, SCT + pixels	3.6	28	23.8	322
101 × 199 bins, SCT + pixels	7.0	60	47.6	653
101 × 100 bins, SCT	13.4	499	59.0	949
101 × 199 bins, SCT	26.4	992	117.8	1894
jet RoIs				
101 × 100 bins, SCT + pixels	4.7	122	26.4	436
101 × 199 bins, SCT + pixels	8.5	247	52.5	890
101 × 100 bins, SCT	17.7	516	70.3	824
101 × 199 bins, SCT	34.3	1032	140.0	1629

7.5.3 Number of fit combinations

Electron RoIs

The distributions for the total number of fit combinations investigated for the four methods to determine the best combination are given in figure 7.20 (SCT + pixel detectors) and figure 7.21 (SCT). All distributions are based on 101×100 bins and electron RoIs. The total number of entries in these distributions is equal to the number of RoIs for which at least one histogram bin with at least the threshold value is found. The results are summarised in table 7.5. Also the results for overlapping bins (101×199 bins) are presented in this table.

The number of combinations tested increases significantly (up to a factor 2) when overlapping bins are used, especially when for a given number of entries in the histogram all combinations are checked. The number of combinations tested reduces significantly (up to a factor 10 in the case of non-overlapping bins) when the search stops as soon as an acceptable combination has been found. For non-overlapping bins the number of combinations increases in general by about 10-20% when all bins with at least the threshold value are selected instead of only the bins with the highest number of entries. This means that in most cases an acceptable least square fit combination is found for one of the selected bins with the maximum number of entries. The number of combinations tested increases significantly (up to a factor 8) when the data from the pixel detectors is not used due to the lower histogram threshold. Especially the maximum number of combination tested can be very large in this case. For the worst combination (101×199 bins, SCT only, all bins with at least the threshold value selected, all combinations for a given number of entries in the histogram are checked) up to 10^5 least square fits are performed.

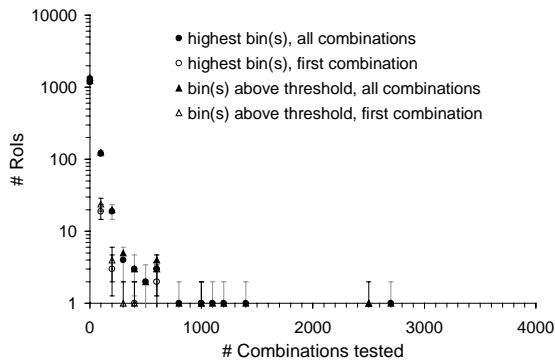


Figure 7.20 Total number of least square fit combinations tested, SCT + pixel detectors, 101×100 bins, electron RoIs.

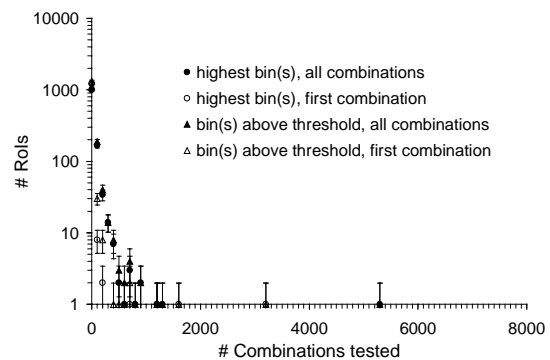


Figure 7.21 Total number of least square fit combinations tested, SCT, 101×100 bins, electron RoIs.

Table 7.5 Number of least square fit combinations tested (electron RoIs).

	highest bin(s), all combinations		highest bin(s), first combination		bin(s) above threshold, all combinations		bin(s) above threshold, first combination	
	mean	max	mean	max	mean	max	mean	max
	101×100 bins, SCT + pixels	32	2.72×10^3	9	1.00×10^3	34	2.72×10^3	12
101×199 bins, SCT + pixels	57	2.59×10^3	9	1.00×10^3	61	5.06×10^3	15	5.06×10^3
101×100 bins, SCT	223	5.37×10^4	20	1.22×10^4	242	5.37×10^4	23	1.22×10^4
101×199 bins, SCT	440	1.03×10^5	19	8.13×10^3	481	1.03×10^5	26	8.13×10^3

Jet RoIs

The corresponding numbers for jet RoIs are given in table 7.6. The numbers are significantly larger than the numbers for electron RoIs. Depending on the option and parameters used, the average number of combinations tested for jet RoIs can be a factor 3 larger than for electron RoIs. The maximum number of combinations tested can even be a factor 5 larger. This is caused by the larger number of histogram bins selected and the larger number of space-points in the bins selected by the histogramming algorithm.

Table 7.6 Number of least square fit combinations tested (jet RoIs).

	highest bin(s), all combinations		highest bin(s), first combination		bin(s) above threshold, all combinations		bin(s) above threshold, first combination	
	mean	max	mean	max	mean	max	mean	max
	101×100 bins, SCT + pixels	92	1.12×10^4	24	3.13×10^3	113	1.12×10^4	57
101×199 bins, SCT + pixels	139	1.25×10^4	27	5.41×10^3	164	1.25×10^4	71	9.00×10^3
101×100 bins, SCT	510	7.68×10^4	59	1.58×10^4	633	7.68×10^4	84	1.58×10^4
101×199 bins, SCT	1.01×10^3	1.55×10^5	95	2.93×10^4	1.27×10^3	1.55×10^5	143	2.93×10^4

7.5.4 Quality of reconstructed tracks

The distributions for the fraction of space-points in the reconstructed track created by the high- p_T electron are given in figure 7.22 (101 × 100 bins), both with and without the data from the pixel detectors. The total number of entries in these distributions is equal to the number of RoIs for which at least one histogram bin with at least the threshold value and one least square fit combination matching (7.18) and (7.19) is found. Only the bins with the highest number of entries are selected, all combinations of space-points are investigated and the best one is selected.

The fraction of reconstructed “fake tracks” and the fraction of reconstructed “best tracks” are given in table 7.7. In this table also the results are presented when the search stops as soon as a combination matching (7.18) and (7.19) is found.

The “fake tracks” are defined as reconstructed tracks where less than 50% of the space-points are created by the high- p_T electron. The fraction of “fake tracks” is at maximum about 1% and reduces significantly when the data from the pixel detectors is used.

The “best tracks” are defined as reconstructed tracks where all the space-points are created by the high- p_T electron. This fraction drops from 95% without using the pixel detector data to 85% when the data from the pixel detector is used. This is due to the higher probability of one of the space-points being created by background when the reconstructed track consists of more space-points.

The fraction of “best tracks” reduces significantly (15-30%) when the search stops as soon as a combination matching (7.18) and (7.19) is found, instead of investigating all combinations and selecting the best one. This indicates that the quality factor Q introduced in section 7.3.7 is an effective quantity to determine the best combination of space-points.

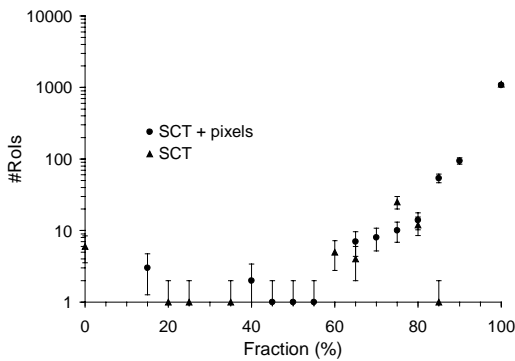


Figure 7.22 Fraction of space-points created by the high- p_T electron in the reconstructed track (101×100 bins, highest bin(s) selected, all combinations tested).

Table 7.7 Reconstruction performance of SCTFEX algorithm for electron RoIs (101×100 bins, highest bin(s) selected).

all combinations tested	“fake tracks” [%]	“best tracks” [%]
SCT + pixels	0.47	84.7
SCT	0.77	95.2
first combination		
SCT + pixels	0.73	55.4
SCT	1.19	81.9

7.5.5 Efficiency results

Electron RoIs

The efficiency values for reconstructing high- p_T electron tracks are given in table 7.8. The first column gives the acceptance ratio (relative to the total sample of RoIs) after the histogramming algorithm. A track is accepted if a bin with at least the threshold value is found, with the α -index of the bin corresponding to a p_T -value above $p_{T,threshold}$. The other columns give the acceptance ratio (relative to the total sample or RoIs) after the least square fit algorithm for the different options. A track is accepted if at least one bin with the threshold value is found in the histogramming algorithm and in the least square fit algorithm a fit combination matching (7.18) and (7.19) is found with a reconstructed p_T above $p_{T,threshold}$.

Using the data from the pixel detectors helps significantly to increase the efficiency after the least square fit algorithm, especially when only the histogram bins with the maximum number of entries are selected. In this case the efficiency increases by about 7% (absolute value). When all bins above threshold are selected, the efficiency increases by about 1-2% (absolute value).

In general the efficiency does not increase significantly or even decreases when overlapping bins are used instead of non-overlapping bins. The only exception is the result for 25 α -bins when the data from the pixel detectors is used. In this case the efficiency increases by 3% (absolute value) when overlapping bins are used.

The efficiency increases significantly when the histogram bins with a lower number of entries are investigated when no acceptable solution is found. When the pixel detector data is used, the increase is about 3% (absolute value). When only the SCT data is used, the increase is much larger (about 8-10% absolute value).

The efficiency slightly increases (up to 1% absolute value) when the combinatorial least square fit algorithm stops as soon as a fit combination matching (7.18) and (7.19) is found. One can expect that the efficiency increases in this case because the selected histogram bins are ordered with respect to their p_T -value. This means that the first fit combination matching (7.18) and (7.19) will in general correspond to the largest p_T -value and has the largest probability to be above threshold.

By including the data from the pixel detectors in the LVL2 trigger, the efficiency reduces when the number of α -bins is reduced from 101 to 25. Without the data from the pixel detectors the efficiency remains almost the same or increases.

The results presented in this table are based on the full sample of space-points. For the data samples studied, the loss of efficiency for high- p_T tracks is only a few percent when only the space-points are used in the track reconstruction algorithm with (ϕ, r, z) co-ordinates inside the (ϕ, r, z) range spanned by the RoI (see section 6.8 of the previous chapter).

Table 7.8 Efficiency of the track reconstruction algorithm implemented in SCTFEX for accepting electron RoIs.

	histogram	highest bin(s), all combina- tions	highest bin(s), first combination	bin(s) above threshold, all combi- nations	bin(s) above threshold, first combi- nation
SCT + pixels	[%]	[%]	[%]	[%]	[%]
101 × 100	98.4	93.8	93.9	97.2	97.3
25 × 100	96.5	90.4	90.5	93.0	93.2
101 × 199	98.5	93.9	94.1	97.3	97.5
25 × 199	97.8	93.7	93.8	96.6	96.8
SCT					
101 × 100	97.3	85.6	86.2	94.9	95.5
25 × 100	97.2	87.6	88.2	94.5	95.1
101 × 199	97.4	85.4	85.8	95.1	95.6
25 × 199	97.2	86.1	86.7	94.8	95.4

Jet RoIs

The (acceptance) efficiency values for jet (background) RoIs are given in table 7.9, together with the corresponding LVL2 acceptance rates for single electromagnetic RoIs (EM30I). The meaning of the columns is the same as for the table with the results for electron RoIs. The rejection performance against jet RoIs of the local precision tracker algorithm alone is not very good. Up to 50% of the RoIs is passed, corresponding to a LVL2 output rate for single electromagnetic RoIs between 5 kHz and 10 kHz. This number however decreases significantly when the features of the precision tracker are combined with the calorimeter, see section 7.7.

In general the fraction of accepted background events (after the least square fit algorithm) increases when the data from the pixel detectors is used in the LVL2 trigger (due to the higher histogram threshold value, the fraction of accepted background events after the histogramming algorithm reduces). This is not true anymore when the features calculated for the

precision tracker and calorimeter are combined. In this case the fraction of accepted background events after the least square fit algorithm reduces when the data from the pixel detectors is used in the LVL2 trigger, see section 7.7.

The fraction of accepted background events increases significantly (up to 10% absolute value) when all bins with at least the threshold value are selected, instead of only the bin(s) with the highest number of entries.

The fraction of accepted background events increases by about 2-3% (absolute value) when the combinatorial least square fit algorithm stops as soon as a fit combination matching (7.18) and (7.19) is found.

In general the fraction of accepted background RoIs reduces when the number of α -bins is reduced from 101 to 25. When the data from the pixel detectors is used, this reduction can be significant (up to 10% absolute value). When only the SCT data is used, the reduction is much less or the accepted fraction even increases.

Table 7.9 Fraction of jet RoIs accepted by the track reconstruction algorithm implemented in SCTFEX and the corresponding LVL2 output rate.

	histogram	highest bin(s), all combinations		highest bin(s), first combination		bin(s) above threshold, all combinations		bin(s) above threshold, first combination	
SCT + pixels	%	%	rate [kHz]	%	rate [kHz]	%	rate [kHz]	%	rate [kHz]
101 × 100	71.0	38.3	6.3	40.4	6.7	47.2	7.8	49.3	8.1
25 × 100	59.6	33.5	5.5	34.9	5.8	38.7	6.4	40.2	6.6
101 × 199	72.1	38.4	6.3	40.5	6.7	48.1	7.9	50.1	8.3
25 × 199	65.5	36.7	6.1	38.4	6.3	43.7	7.2	45.4	7.5
SCT									
101 × 100	80.2	32.3	5.3	34.9	5.8	46.1	7.6	48.6	8.0
25 × 100	76.6	34.0	5.6	36.8	6.1	44.8	7.4	47.6	7.9
101 × 199	81.1	32.1	5.3	34.9	5.8	46.1	7.6	48.9	8.1
25 × 199	77.5	33.6	5.5	36.6	6.0	45.3	7.5	48.3	8.0

7.5.6 Conclusions on strategy

The best performance is achieved when the combinatorial least square fit stops as soon as a combination matching (7.18) and (7.19) is found. In this case the quality of the reconstructed track is worse than in the case where all combinations are investigated and the best one is selected, but this has almost no influence on the efficiency for accepting high- p_T electrons or on the rejection performance against background events. Both the fraction of accepted high- p_T electrons and the fraction of accepted background events increases when the histogram bin(s) with a lower number of entries are investigated when no acceptable solution is found. The use of overlapping bins does in general not significantly increase the physics performance.

7.6 Benchmark results

The calculation time of the track reconstruction algorithm is presented in table 7.10 (non-overlapping bins, highest bin(s) selected) and table 7.11 (non-overlapping bins, all bins above threshold selected) both for electron RoIs and jet RoIs, with and without the data from the

pixel detectors. All timing measurements are done with a Dell PC with a 400 MHz Pentium II processor and 64 MB internal memory.

The results for overlapping bins are not given because it has been shown in the previous section that the use of overlapping bins does not significantly increase the physics performance. However, both the calculation times of the histogramming algorithm and the least square fit algorithm are longer in this case.

Only the results for the version of the algorithm in which the search stops as soon as a combination matching (7.18) and (7.19) is found are presented. In the previous section it has been shown that investigating all possible least square fit combinations and selecting the best one, instead of stopping as soon as an acceptable combination is found, does not significantly increase the physics performance. The calculation time of the least square fit algorithm is however much longer in this case.

In the tables the total calculation time is given together with the calculation time of the histogramming and least square fit algorithm. The results for the total calculation time and the timing results of the histogramming algorithm are based on the full sample of RoIs. The timing results of the least square fit algorithm are based on the fraction of RoIs for which the least square fit algorithm is performed, i.e. the fraction of RoIs for which at least one bin is selected by the histogramming algorithm.

The calculation time increases by more than a factor 2 when the data from the pixel detectors is used in the LVL2 trigger. The number of bins used in the histogram has an important impact on the calculation time. For 101×100 bins, the average calculation time is more than 3 times larger than for 25×100 bins. On average the calculation time of the least square fit is only a relative small fraction of the total calculation time (in general less than 10% for 101×100 bins). The maximum calculation can however be very long, especially for jet events (up to 140 ms), due to the tail in the distribution of the number of combinations tested (section 7.5.3).

The total calculation time and the calculation time of the histogramming algorithm are about 15% larger for jet RoIs than for electron RoIs, mainly due to the larger number of space-points. The time necessary for the least square fit algorithm is significantly larger (up to about a factor 5), mainly due to the higher number of histogram bins selected and the larger number of fit combinations.

The total calculation time and the histogramming calculation time are about 15% larger when all bins above threshold are selected instead of only the bin(s) with the highest number of entries. The time necessary for the least square fit can be up to about a factor 2 larger.

The results presented in these tables are based on the full sample of space-points. For the current data samples, the calculation times can be reduced by roughly a factor 2 when only the space-points are used in the track reconstruction algorithm with a (ϕ, r, z) co-ordinate inside the (ϕ, r, z) range spanned by the RoI (see section 6.8 of the previous chapter).

Table 7.10 Calculation times of track reconstruction algorithm implemented in SCTFEX (highest bins selected, first combination) [ms].

Electron RoIs	SCT + pixel detectors				SCT			
	101 × 100		25 × 100		101 × 100		25 × 100	
	mean	max	mean	max	mean	max	mean	max
histogram	8.64	45.3	1.90	9.97	3.08	39.3	0.722	8.86
least square fit	0.132	8.89	0.093	1.82	0.299	155	0.110	32.0
total	8.77	54.2	1.99	11.8	3.38	194	0.832	40.8
Jet RoIs								
histogram	10.1	35.9	2.14	7.80	3.69	28.2	0.860	6.32
least square fit	0.592	86.0	0.244	12.1	0.671	140	0.243	30.6
total	10.6	109	2.33	19.5	4.31	168	1.08	36.9

Table 7.11 Calculation times of track reconstruction algorithm implemented in SCTFEX (all bins above threshold selected, first combination) [ms].

Electron RoIs	SCT + pixel detectors				SCT			
	101 × 100		25 × 100		101 × 100		25 × 100	
	mean	max	mean	max	mean	max	mean	max
histogram	9.01	48.0	1.96	10.6	3.31	44.7	0.766	9.59
least square fit	0.209	70.2	0.116	15.2	0.344	162	0.128	39.0
total	9.22	118	2.07	25.8	3.66	207	0.894	48.6
Jet RoIs								
histogram	10.6	35.7	2.28	8.26	3.99	33.0	0.910	6.90
least square fit	1.15	85.3	0.411	30.8	1.06	141	0.324	31.2
total	11.6	119	2.59	35.1	4.97	174	1.20	38.1

The corresponding distributions for the total execution times of the algorithms implemented in SCTFEX, are given in figure 7.23 to figure 7.26 for electron RoIs and in figure 7.27 to figure 7.30 for jet RoIs. Only the bins with the highest number of entries are selected, the search stops as soon as a combination matching (7.18) and (7.19) is found. The calculation times can be described approximately by a Gaussian distribution, however with a long tail.

In the distributions the integrated fraction is shown, giving the fraction of the RoIs that fall within a certain calculation time. When the data from the pixel detectors is used, around 96% of the electron RoIs is calculated within 15 ms (101 × 100 bins). This drops significantly when the number of bins is reduced to 25 × 100 bins. In this case 96% of the electron RoIs is calculated within 3.25 ms. When only the data from the SCT is used, 96% of the electron RoIs is calculated within 6 ms (101 × 100 bins). For 25 × 100 bins, 96% of the electron RoIs is calculated within 1.4 ms.

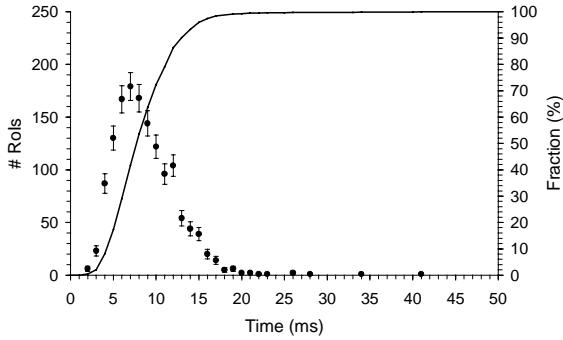


Figure 7.23 Calculation time of algorithm implemented in SCTFEX, electron RoIs, 101×100 bins, SCT + pixel detectors, highest bin(s) selected, first combination.

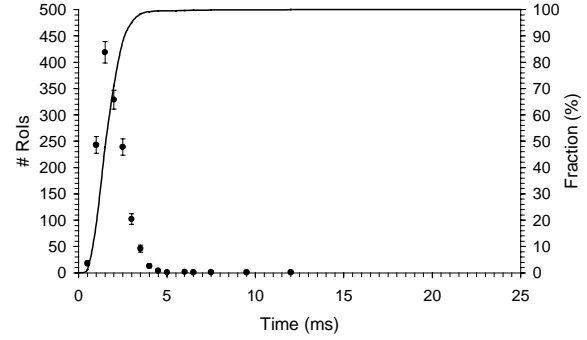


Figure 7.24 Calculation time of algorithm implemented in SCTFEX, electron RoIs, 25×100 bins, SCT + pixel detectors, highest bin(s) selected, first combination.

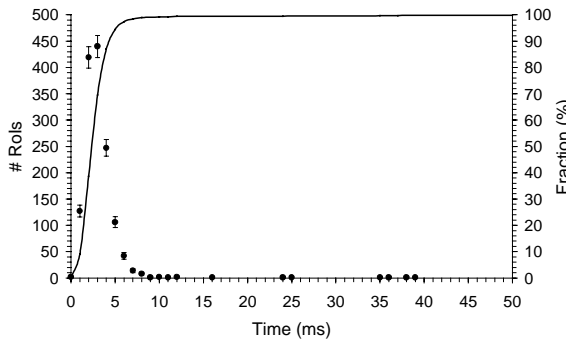


Figure 7.25 Calculation time of algorithm implemented in SCTFEX, electron RoIs, 101×100 bins, SCT, highest bin(s) selected, first combination.

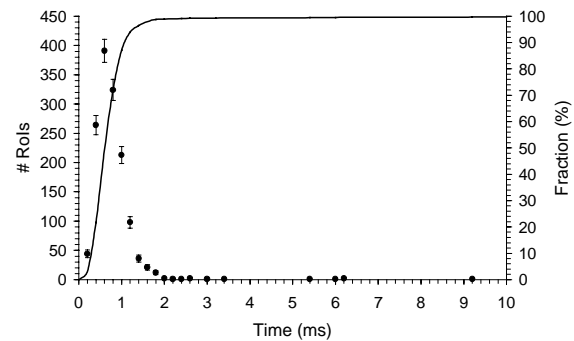


Figure 7.26 Calculation time of algorithm implemented in SCTFEX, electron RoIs, 25×100 bins, SCT, highest bin(s) selected, first combination.

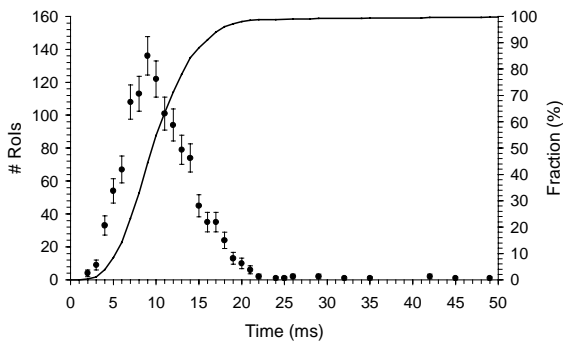


Figure 7.27 Calculation time of algorithm implemented in SCTFEX, jet RoIs, 101×100 bins, SCT + pixel detectors, highest bin(s) selected, first combination.

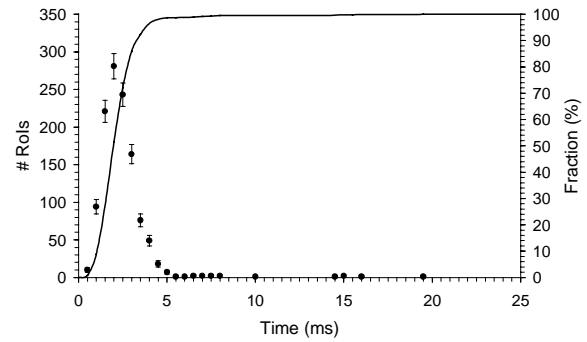


Figure 7.28 Calculation time of algorithm implemented in SCTFEX, electron RoIs, 25×100 bins, SCT + pixel detectors, highest bin(s) selected, first combination.

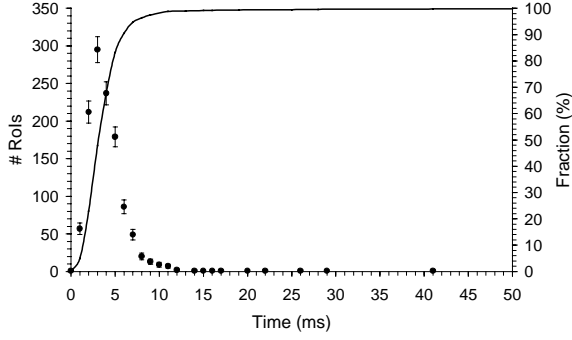


Figure 7.29 Calculation time of algorithm implemented in SCTFEX, jet RoIs, 101×100 bins, SCT, highest bin(s) selected, first combination.

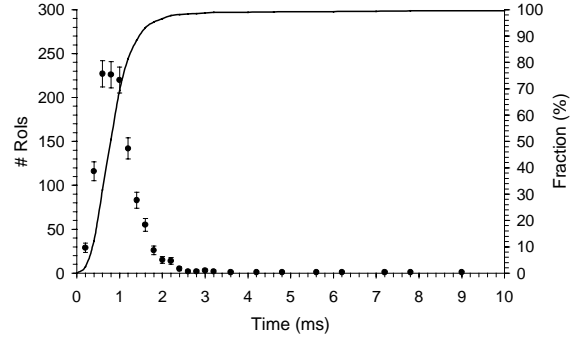


Figure 7.30 Calculation time of algorithm implemented in SCTFEX, jet RoIs, 25×100 bins, SCT, highest bin(s) selected, first combination.

The calculation time of the histogramming algorithm versus the number of space-points for electron RoIs is given in figure 7.31 (SCT + pixel detectors) and figure 7.32 (SCT). The corresponding plots for jet RoIs are given in figure 7.33 and figure 7.34. Only the bins with the highest number of entries are selected, the search stops as soon as a combination matching (7.18) and (7.19) is found. Like for the TRT [6], also for the precision tracker the histogramming algorithm scales approximately with $M \log N$, with N the occupancy⁴.

When the data from the pixel detectors is used in the LVL2 trigger, the calculation time versus the number of space-points for 101×100 bins is approximately given by:

$$\begin{aligned} \text{Time } [\mu\text{s}] &= 15.9 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (electron RoIs)} \\ \text{Time } [\mu\text{s}] &= 16.2 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (jet RoIs)} \end{aligned} \quad (7.23)$$

For 25×100 bins, the calculation time versus the number of space-points is approximately given by:

$$\begin{aligned} \text{Time } [\mu\text{s}] &= 3.36 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (electron RoIs)} \\ \text{Time } [\mu\text{s}] &= 3.44 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (jet RoIs)} \end{aligned} \quad (7.24)$$

When the data from the pixel detectors is not used in the LVL2 trigger, the calculation time versus the number of space-points for 101×100 bins is approximately given by:

$$\begin{aligned} \text{Time } [\mu\text{s}] &= 18.6 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (electron RoIs)} \\ \text{Time } [\mu\text{s}] &= 18.2 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (jet RoIs)} \end{aligned} \quad (7.25)$$

For 25×100 bins, the calculation time versus the number of space-points is approximately given by:

$$\begin{aligned} \text{Time } [\mu\text{s}] &= 4.16 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (electron RoIs)} \\ \text{Time } [\mu\text{s}] &= 4.18 \times (\#\text{Space-points}) \times \log(\#\text{Space-points}) \text{ (jet RoIs)} \end{aligned} \quad (7.26)$$

A simple parameterisation of the calculation time of the least square fit algorithm is not possible because it depends strongly on the number of space-points per least square fit and the number of least square fit combinations.

⁴ The number of space-points is linearly related to the occupancy.

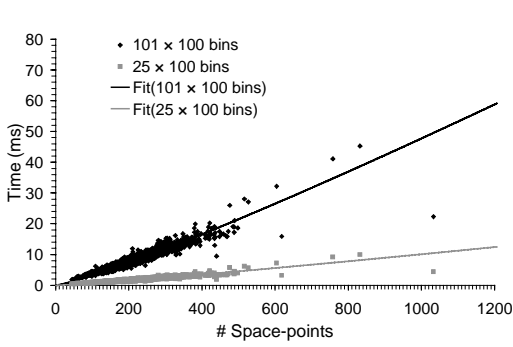


Figure 7.31 Calculation time of histogramming algorithm versus number of space-points, electron RoIs, SCT + pixel detectors.

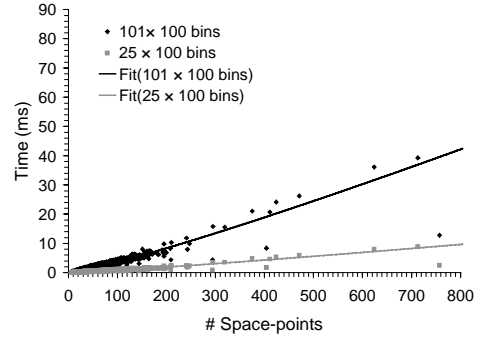


Figure 7.32 Calculation time of histogramming algorithm versus number of space-points, electron RoIs, SCT.

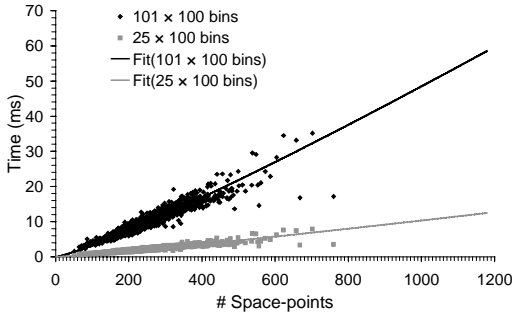


Figure 7.33 Calculation time of histogramming algorithm versus number of space-points, jet RoIs, SCT + pixel detectors.

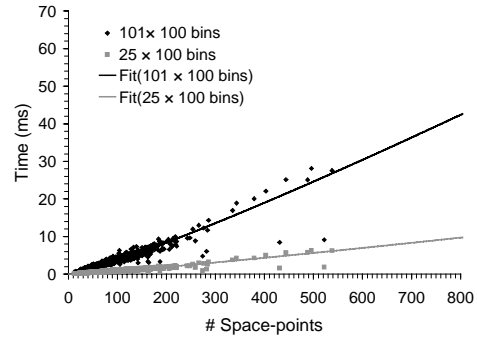


Figure 7.34 Calculation time of histogramming algorithm versus number of space-points, jet RoIs, SCT.

7.7 Global algorithm

7.7.1 Introduction

To further reduce the rate of triggers from the background jets, while maintaining a good efficiency for high- p_T electrons, the track parameters reconstructed with the precision tracker are combined with the cluster information from the calorimeter in the global part of the LVL2 system. A very simple global algorithm is demanding a match between the reconstructed (E_T, η, ϕ) values in the calorimeter and the reconstructed (p_T, η, ϕ) values in the precision tracker. This means that a reconstructed track is only accepted if:

$$\begin{aligned} |\phi_{SI} - \phi_{ECAL}| &< |\phi_{SI} - \phi_{ECAL}|_{max} \\ |\eta_{SI} - \eta_{ECAL}| &< |\eta_{SI} - \eta_{ECAL}|_{max} \\ \left(\frac{E_T}{p_T}\right)_{min} &< \left(\frac{E_T}{p_T}\right) < \left(\frac{E_T}{p_T}\right)_{max} \end{aligned} \quad (7.27)$$

with $(\eta_{ECAL}, \phi_{ECAL})$ the (η, ϕ) values reconstructed by the calorimeter and (η_{SI}, ϕ_{SI}) the (η, ϕ) values reconstructed by the precision tracker. One expects that the parameters for high- p_T electron RoIs match much better than for jet RoIs. For jet RoIs the assumption that the RoI contains only one high- p_T particle is completely wrong and the reconstructed track will in general contain space-points created by several charged (high- p_T) particles.

The values of the parameters used in the global algorithm are described in section 7.7.2. The efficiency results after applying the global algorithm are presented in section 7.7.3.

7.7.2 Determination of parameters of global algorithm

The distributions for $\phi_{SI} - \phi_{ECAL}$ are given in figure 7.35, both for electron RoIs and for jet RoIs. The tracks reconstructed in the precision tracker are based on 101×100 bins. Only the bins with the maximum number of entries are selected, the search stops as soon as a combination matching (7.18) and (7.19) is found. Both data from the SCT and pixel detectors is used. The distributions for $\eta_{SI} - \eta_{ECAL}$ are given in figure 7.36. The distributions for $E_T p_T^{-1}$ are given in figure 7.37. In the case of a perfectly reconstructed track and cluster one expects⁵ $E_T p_T^{-1} = 1$. From the distributions it follows that the reconstructed track matches much better with the reconstructed calorimeter cluster for electron RoIs than for jet RoIs. Based on the distributions presented, five cuts are defined, described in table 7.12. Cut 1 corresponds to the cut used in the TDR [6].

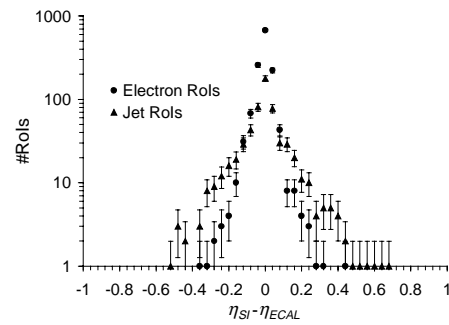
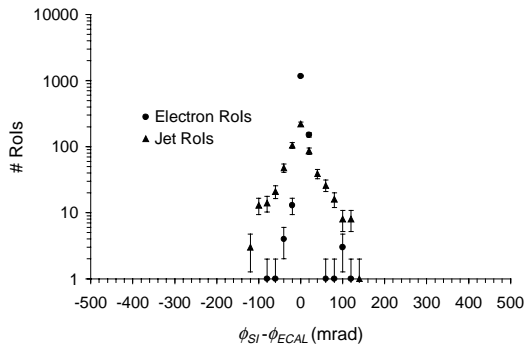


Figure 7.35 $\phi_{SI} - \phi_{ECAL}$, 101×100 bins, highest bin(s), first combination, SCT + pixel detectors. **Figure 7.36** $\eta_{SI} - \eta_{ECAL}$, 101×100 bins, highest bin(s), first combination, SCT + pixel detectors.

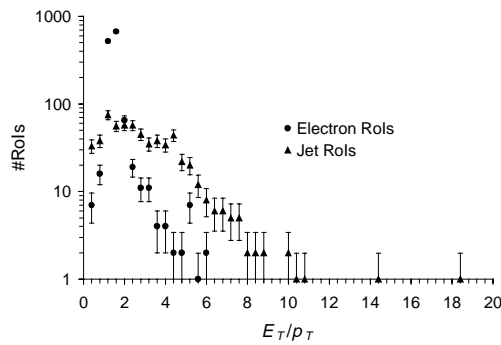


Figure 7.37 $E_T p_T^{-1}$, 101×100 bins, highest bin(s), first combination, SCT + pixel detectors.

⁵ $E_T = p_T \oplus m$, with m the particle mass. This means that for a highly energetic particle ($p_T \gg m$) $E_T \approx p_T$.

Table 7.12 Global algorithm cuts.

	$ \phi_{SI} - \phi_{ECAL} _{max}$	$(E_T p_T^{-1})_{min}$	$(E_T p_T^{-1})_{max}$	$ \eta_{SI} - \eta_{ECAL} _{max}$
cut 1	30	-	4	-
cut 2	30	-	3	-
cut 3	30	0.5	3	-
cut 4	20	0.5	3	-
cut 5	20	0.5	3	0.15

7.7.3 Efficiency results

The efficiency results after applying the global algorithm are presented in table 7.13. For the jet RoIs the corresponding LVL2 acceptance rate for single electromagnetic RoIs is also given. From the values presented it can be concluded that the matching check helps significantly to reduce the background rate, while keeping a good efficiency for isolated high- p_T electrons. When the data from the pixel detectors is used in the LVL2 trigger, the background rate can be reduced by a factor 4, together with about 6% efficiency loss for high- p_T electrons. When the data from the pixel detectors is not used, the efficiency loss for high- p_T electrons is somewhat larger (about 10%). A combination of cuts on $|\phi_{SI} - \phi_{ECAL}|_{max}$, $|\eta_{SI} - \eta_{ECAL}|_{max}$, $(E_T p_T^{-1})_{min}$ and $(E_T p_T^{-1})_{max}$ seems to give the best result. Both the efficiency for high- p_T electrons and the fraction of accepted background events increase when the bins with a lower number of entries are investigated if no acceptable solution is found.

The physics performance increases significantly when the data from the pixel detector is used in the LVL2 trigger. The fraction of accepted high- p_T electrons increases by about 10% (absolute value). The LVL2 acceptance rate reduces by about 10%.

The LVL2 acceptance rates presented are somewhat larger than the acceptance rate mentioned in the TDR. The reason is that the results in the TDR are based on a somewhat different algorithm (see section 7.8) and a different data sample that passed more stringent calorimeter cuts [6]. In the inner detector TDR it is shown that the physics performance can be increased further when the precision tracker algorithm is combined with the TRT requiring both a reconstructed track in the TRT and the precision tracker [6]. In this case the LVL2 output rate for single electromagnetic RoIs (EM30I) at high luminosity can be reduced to 0.7 kHz with an efficiency for high- p_T electrons above 80%

Table 7.13 Fraction of events remaining after applying global algorithm and the corresponding LVL2 output rate (101×100 bins).

	SCT + pixel detectors			SCT		
	electron RoIs	jet RoIs	rate	electron RoIs	jet RoIs	rate
highest bin(s)	%	%	[kHz]	%	%	[kHz]
precision tracker alone	93.9	40.4	6.7	86.2	34.9	5.8
cut 1	90.0	17.6	2.9	83.4	20.0	3.3
cut 2	89.3	15.4	2.5	82.6	16.8	2.8
cut 3	88.3	13.4	2.2	79.3	15.3	2.5
cut 4	88.2	11.7	1.9	79.3	14.2	2.4
cut 5	87.3	10.4	1.7	76.8	11.5	1.9
bin(s) above threshold						
precision tracker alone	97.3	49.3	8.1	95.5	48.6	8.0
cut 1	93.2	21.4	3.5	91.8	26.5	4.4
cut 2	92.5	18.6	3.1	90.7	22.7	3.7
cut 3	91.2	15.9	2.6	86.8	19.7	3.3
cut 4	91.1	13.9	2.3	86.8	18.1	3.0
cut 5	90.1	12.0	2.0	83.8	13.7	2.3

7.8 Comparison between the TDR algorithm and the SCTFEX algorithm

The LVL2 efficiency results presented in the inner detector TDR [6] are based on a somewhat different algorithm (see below). The physics performance of this algorithm is slightly better than for the algorithm implemented in SCTFEX [6]. Some not very detailed benchmarking measurements of this algorithm show that the average calculation time is however much larger (about a factor 30) than the algorithm implemented in SCTFEX. This means that this algorithm is not suitable for an online implementation.

The algorithm is based on the same principle as the algorithm implemented in SCTFEX. The algorithm is divided in a histogramming part and a combinatorial least square fit part. Tracks are assumed to be straight lines in the (r, ϕ) plane, the (z, ϕ) plane and the (z, r) plane, bremsstrahlung energy losses and detector inefficiencies are ignored. There are however several differences in the implementation [11]:

- In the TDR algorithm a histogram of the total number of space-points in the current histogram road is made (instead of the number of detector planes with at least one space-point in the current road).
- In the TDR algorithm all the histogram bins with at least the threshold value and all possible combinations of space-points are investigated in the least square fit algorithm, not only the bin(s) with the highest number of entries. In the SCTFEX algorithm in most cases only a fraction of the histogram bins with at least the threshold value and a fraction of the possible combinations of space-points are investigated, for all options described.

When a bin is selected in the TDR histogramming algorithm, it is not checked whether the same bin was already selected before. Due to the method used for implementation a bin with n entries (space-points in the current histogram road) is selected $1 + n - k$ times,

with k the lower histogram threshold. This means that frequently in the least square fit algorithm exactly the same calculation is repeated several times.

- Space-points are only used in the TDR histogramming algorithm if their ϕ -value is inside the ϕ -range spanned by the RoI:

$$\phi_{RoIMin} \leq \phi_{point} \leq \phi_{RoIMax} \quad (7.28)$$

In the algorithm implemented in SCTFEX, the selection of space-points is regarded as part of the pre-processing algorithm. Space-points can be selected or rejected based on their (ϕ, r, z) co-ordinates, described in more detail in section 6.8 in chapter 6. This selection step is not used for the results presented in this chapter, all space-points are used. The selection based on the ϕ -value described in section 6.8 is less restrictive than the selection used by the TDR algorithm and takes into account that a track does not have a constant ϕ -value due to the influence of the solenoid magnetic field. The effective RoI size depends on the distance from the calorimeter (equation (6.40) in chapter 6).

- For an RoI defined by the end-cap calorimeter, the area spanned by a bin corresponds to the area spanned by a bin for the algorithm implemented in SCTFEX (see also equation (7.9)):

$$\begin{aligned} \phi_{min}(i, j, z) &= \phi_{RoIMin} + i\Delta\phi + \alpha_z(j)(z_{cal} - z) \\ \phi_{max}(i, j, z) &= \phi_{RoIMin} + (i+1)\Delta\phi + \alpha_z(j)(z_{cal} - z) \end{aligned} \quad (7.29)$$

For an RoI defined by the barrel calorimeter, a wrong intercept position is used in the TDR algorithm (z_{cal} instead of r_{cal}). The spanned area is given by:

$$\begin{aligned} \phi_{min}(i, j, r) &= \phi_{RoIMin} + i\Delta\phi + \alpha_r(j)(z_{cal} - r) \\ \phi_{max}(i, j, r) &= \phi_{RoIMin} + (i+1)\Delta\phi + \alpha_r(j)(z_{cal} - r) \end{aligned} \quad (7.30)$$

- The selection of the combination of selected space-points giving the best least square fit is rather complicated in the TDR algorithm. The best combination is defined as the best combination of the best histogram bin. A histogram bin is acceptable if:
 - The number of entries is at least the threshold value
 - The bin contains at least one acceptable least square fit combination

The best bin (of the acceptable bins) is given by:

- The highest number of entries
- A histogram index corresponding to the smallest slope in the (r, ϕ) plane or the (z, ϕ) plane (highest p_T -value)

An acceptable least square fit combination is given by:

$$\sigma_\phi^2 \chi_{r\phi}^2 < 1.5 \text{mrad}^2 \quad \sigma_\phi^2 \chi_{z\phi}^2 < 1.5 \text{mrad}^2 \quad \sigma_r^2 \chi_{zr}^2 < 15 \text{cm}^2 \quad |z_0| < 1000 \text{cm} \quad (7.31)$$

The cut for the allowed $|z_0|$ range is extremely soft. From the acceptable combinations, the best combination (for an individual bin) is the combination with the smallest $\sigma_\phi^2 \chi_{r\phi}^2$ -value. This means that also in the end-cap part, the least square fit in the (r, ϕ) plane mainly determines the best combination. The least square fit in the (z, ϕ) plane should mainly determine the best combination, because this is the plane with the best resolution in the end-cap part.

- The bins selected by the TDR histogramming algorithm are not automatically ordered by number of entries and slope (p_T -value).
- In the TDR algorithm the histogram needs to be reset for each new RoI.
- The TDR algorithm is based on 100×100 bins with a lower histogram threshold of four.

- In the TDR algorithm there is no maximum for the number of least square fit combinations in a bin.

7.9 Comparison between the XKALMAN algorithm and the SCTFEX algorithm

7.9.1 Introduction

An offline track recognition algorithm for charged particle tracks with $p_T > 0.5$ GeV for the ATLAS inner detector is provided by the package XKALMAN. The XKALMAN algorithm is described in section 7.9.2. The differences between XKALMAN and the algorithm implemented in SCTFEX are described in section 7.9.3. The performance of the XKALMAN algorithm is shortly described in section 7.9.4.

7.9.2 XKALMAN algorithm description

The XKALMAN algorithm is divided in an initial step in the TRT, a step in the precision tracker and a final step in the TRT.

Initial TRT step

In the initial TRT step, more or less promising track directions are separated from obviously wrong ones, using a simple and fast procedure. This step is based on a histogramming algorithm in the (r, ϕ) plane in the barrel area and the (z, ϕ) plane in the end-cap area. A bin is accepted as track candidate if the number of hits is above a threshold.

Track reconstruction algorithm in the precision tracker

For each track-candidate found in the TRT, a progressive track-following procedure is applied in the second step of the XKALMAN algorithm to find all possible prolongations of the initial track-candidate in the SCT and the pixel layers. The progressive track following algorithm consists of a filter algorithm, iterated from the outermost to the innermost precision layer and a smoother procedure, iterated in the opposite direction, from the innermost layers to the outermost layers. In the filter algorithm the helix parameters and covariance matrix (see chapter 3) are propagated from layer k to layer $k - 1$. The helix prediction in layer $k - 1$ is compared with the measured space-points. The best matching space-points are added to the current track. In the smoothing procedure old measurement points can be subtracted and new best measurement points can be added. For both filter and smoothing multiple scattering and bremsstrahlung energy losses are taken into account.

All propagations found of sufficient quality can be defined as final track candidates, even when they originate from the same initial track candidate found in the global TRT scan. The quality of any prolongation is determined from its number of precision layer hits and its total χ^2 -value per degree of freedom.

Final TRT step

The accepted helix trajectories are extrapolated back into the TRT, where a narrow road can be defined around the results of the extrapolation. The final step in the XKALMAN algorithm is the optimal assignment of hits in the TRT to all the tracks retained after the XKALMAN filter-smoother procedure through the precision layers. This algorithm consists of a search for the best track prolongation through the TRT, followed by a final track fit through all points.

7.9.3 Differences between the XKALMAN and the SCTFEX algorithm

There are several important differences between the algorithm implemented in SCTFEX and the XKALMAN algorithm. Basically the algorithm implemented in SCTFEX contains some important simplifications:

- In XKALMAN, the TRT defines the roads in an initial histogramming step. In the algorithm implemented in SCTFEX, the precision tracker itself defines the roads in an initial histogramming step. The local LVL2 precision tracker algorithm works completely independent from the TRT and is always restricted to the RoIs. In the SCTFEX algorithm it is assumed that each RoI contains only one high- p_T track.
- The XKALMAN algorithm is based on a progressive track following algorithm in the precision tracker in which the helix parameters and covariance matrix are iteratively propagated. The algorithm implemented in SCTFEX is based on a combinatorial least square fit.
- The XKALMAN algorithm is based on three dimensions (ϕ, r, z). The algorithm implemented in SCTFEX is based on three two-dimensional projections in the (z, ϕ) plane, the (r, ϕ) plane and the (z, r) plane.
- The XKALMAN algorithm is based on the full helix-equations (3.6). The helix parameters and covariance matrix are iteratively propagated. The LVL2 algorithm is based on the straight-line approximations of the helix-equations (7.3).
- Multiple scattering and bremsstrahlung energy losses are taken into account in the XKALMAN algorithm but not in the algorithm implemented in SCTFEX. Especially for electrons, the contribution from bremsstrahlung energy losses can be a problem. For muons the contribution is absent.
- The possibility that a detection layer may not have a space-point from the high- p_T particle due to detector inefficiencies is taken into account by the XKALMAN algorithm but ignored by the algorithm implemented in SCTFEX. In the algorithm implemented in SCTFEX it is assumed that each detection layer selected by the histogramming algorithm, always contains one space-point from the high- p_T track.

7.9.4 Performance of the XKALMAN algorithm

The track reconstruction efficiency of the XKALMAN algorithm is described in more detail in the inner detector TDR [6]. The efficiency presented is significantly better than the efficiency of the algorithm implemented in SCTFEX but is for muon data only. The efficiency for reconstructing an isolated muon with $p_T > 5$ GeV is 97.7% (averaged over η , without minimum bias background). The efficiency for reconstructing an isolated muon with $p_T > 20$ GeV is 98.5% (averaged over η , without minimum bias background). This drops to 97.6% when minimum bias background is included. The fraction of “fake tracks” is less than 0.1%. A “fake track” is defined as a track with less than 50% of the precision hits created by the high- p_T muon. The results for electrons will probably be worse due to bremsstrahlung energy losses. No timing results of the XKALMAN algorithm are presented in the TDR [6].

7.10 Conclusions and future work

Both the physics performance and the calculation time of the track reconstruction algorithm increase significantly when the data from the pixel detectors is used in the LVL2 inner detector trigger.

The best performance is achieved when the combinatorial least square fit algorithm stops as soon as a combination matching (7.18) and (7.19) is found and the histogram bins with a

lower number of entries are investigated if for a given number of entries no acceptable solution is found. The quality of the reconstructed track is somewhat better when all combinations are investigated and the best one is selected (section 7.5.4) but this does not strongly influence the acceptance efficiency for high- p_T electrons and the reduction of background. The calculation time of the least square fit algorithm is much longer in this case.

Both the fraction of accepted high- p_T electrons and the fraction of accepted background events reduces when the number of histogram bins is reduced from 100×101 bins to 100×25 bins. The calculation time decreases significantly. The use of overlapping bins has almost no influence on the physics performance of the algorithm. The calculation time however increases significantly.

The current studies give promising results that with computer technology probably available in 2005 the LVL2 track reconstruction algorithm for the precision tracker can be implemented with an average latency time of a few milliseconds. A good efficiency for high- p_T electrons and a good rejection performance against background events is achieved when the features of the reconstructed tracks are combined with the reconstructed calorimeter clusters. In the TDR [6] it is shown that the physics performance can be increased further when the precision tracker is combined with the TRT in the global part of the LVL2 trigger, requiring both a reconstructed track in the TRT and the precision tracker.

More detailed studies with larger data samples are however necessary. Especially important is a study of the acceptance efficiency as function of the p_T and $|\eta|$ -value of the high- p_T electron. For the data samples currently available the calculation time can be reduced by roughly a factor 2 with only a few percent loss of efficiency for high- p_T electron tracks when only the space-points are used in the track reconstruction algorithm with their (ϕ, r, z) coordinates within the (ϕ, r, z) range spanned by the RoI. More detailed studies with a data sample in which the length of the LHC interaction region along the beam axis is correctly taken into account are necessary. Also the performance of the algorithm for muons has to be investigated in more detail, especially the combined performance when the reconstructed muon track is combined with the track reconstructed in the TRT and the track reconstructed in the muon spectrometer.

The algorithm described works completely independent from the TRT and can both be implemented on a parallel local-global system and a sequential single farm system (see chapter 5). In a local-global system the precision tracker algorithm and TRT algorithm can be performed in parallel. In a single farm system, the precision tracker and TRT algorithms have to be performed sequentially. It has to be investigated in more detail if this is possible within the maximum latency time allowed and if it is really necessary to include the TRT in the high- p_T trigger. A good physics performance is also achieved without TRT.

If the TRT is used in the high- p_T trigger, it can be attractive, especially in a sequential single farm system, to combine the TRT and the precision tracker in an early stage, instead of only combining the calculated features. Because the histogramming part is by far the most time consuming part of the precision tracker algorithm, it can be attractive to use the TRT for defining roads (instead of the precision tracker itself) in which the least square fit algorithm of the precision tracker is performed. This algorithm is quite similar to the XKALMAN offline algorithm although with a much simpler track reconstruction algorithm in the precision tracker based on straight lines.

7.11 References

1. A. Gheorghe and W. Krischer, *Pattern Recognition Algorithms for Triggering with a Silicon Tracker*, ATLAS DAQ-NO-008 (1992).

2. R.J. Hawkings, *Electron Triggering in the Transition Region of the ATLAS Inner Detector*, ATLAS INDET-NO-096 (1995).
3. S. Sivoklov, R.J. Dankers and J.T.M. Baines, *Second Level Triggering in the Forward Region of the ATLAS Inner Tracker*, ATLAS INDET-NO-111 (1995).
4. RD2 and RD11 Collaboration, *A Study of a Second Level Track Trigger for ATLAS*, Nucl. Instr. Meth. A336 (1993) 59-77.
5. R. Hauser and I. LeGrand, *Algorithms in Second-Level Triggers for ATLAS and Benchmark Results*, ATLAS DAQ-NO-27 (1994).
6. ATLAS Inner Detector Community, *Inner Detector Technical Design Report Volume 1*, ATLAS TDR-4, CERN/LHCC 97-16 (1997).
7. R.J. Hawkings and A.R. Weidberg, *Triggering a Choice – Second Level Electron Triggering in the ATLAS Inner Detector*, ATLAS INDET-NO-052 (1994).
8. R.J. Hawkings, *Tracking and Triggering Using Silicon Detectors and a Study of Higgs Physics at the Large Hadron Collider*, RALT-029-95, University of Oxford, 1994.
9. J.T.M. Baines, Private communication.
10. ATLAS Trigger Performance Group, *Trigger Algorithms in ATRIG*, ATLAS DAQ-NO-060 (1996).
11. S. Sivoklov, Source code TDR FEX algorithm.