The STAR-TPC Clusterfinder/Hitfinder - STAR Note 238

M.A. Lisa
Lawrence Berkeley National Laboratory, Berkeley, CA 94720 USA

February 2, 1996

1 Introduction

The STAR-TPC cluster/hitfinder is a software package, implemented in the TPC Analysis Shell (TAS) as the TCL package, designed to extract from the raw data pad-row crossing positions ("hits") for charged particles that traverse the active volume of the TPC. The cluster/hitfinder is the first offline analysis (non-calibration) software that will process the TPC data in the current model of data reduction; almost all other offline software, including physics analysis software, build on the output of the cluster/hitfinder. Therefore, it is important that the general workings of this first software package are understood.

Our current model of data reduction, in which one first finds hits, and then does tracking on these hits, corresponds to the standard method used by most TPC groups. Of course, other models may be envisioned; for example, one may want to try tracking with the pixel information alone. These alternative avenues should be pursued and the results compared with the standard model.

The relationship of the TCL package and its associated input and output structures (tables) to other low-level simulation and reconstruction software is shown in Figure 1. Two analysis modules, TCL and TPH, perform the clusterfinding and hitfinding, respectively.

2 Data Types and Structures

The task of hitfinding is, at some level, one of grouping into structures and summarizing the raw ADC data. Here, we briefly discuss some definitions used in discussing the data, as well as the table structures employed before the track level.

Users who simply want an overview of the hitfinding methods may want to look over the definitions below, and skip the structures altogether.

2.1 Definitions

The relationship between pixels, sequences, clusters, and hits is indicated in Figure 2, where the result of three tracks passing over a padrow is shown.

A **pixel** (a.k.a. **voxel**) for the TPC describes one time bucket on one pad. A **sequence** is a series of consecutive time buckets on a given pad with ADC values above the readout threshold. A **cluster** is a set of sequences on adjacent pads on a given padrow, each sequence of which overlaps in time with at least one other sequence in the cluster on an adjacent pad. A cluster may be thought of as a continuous "land mass" of pixels in a padrow, if altitude is given by ADC value, and threshold is sea level.

Charged particles passing through the TPC cross over the padrows at **padrow crossings**. The ionization left by these tracks is digitized into spatial pixels. To reconstruct the spatial positions of these padrow crossings, the cluster/hitfinder identifies and localizes regions of high charge density (ADC value) called **hits**. Using the above analogy, these are "mountain peaks" in the padrow terrain. If two (or more) tracks are spatially close together as they pass over a padrow, their ionization patterns may overlap, resulting in a single cluster. If the tracks are not too close (see merging distance, below), then more than one hit will be associated with the cluster.

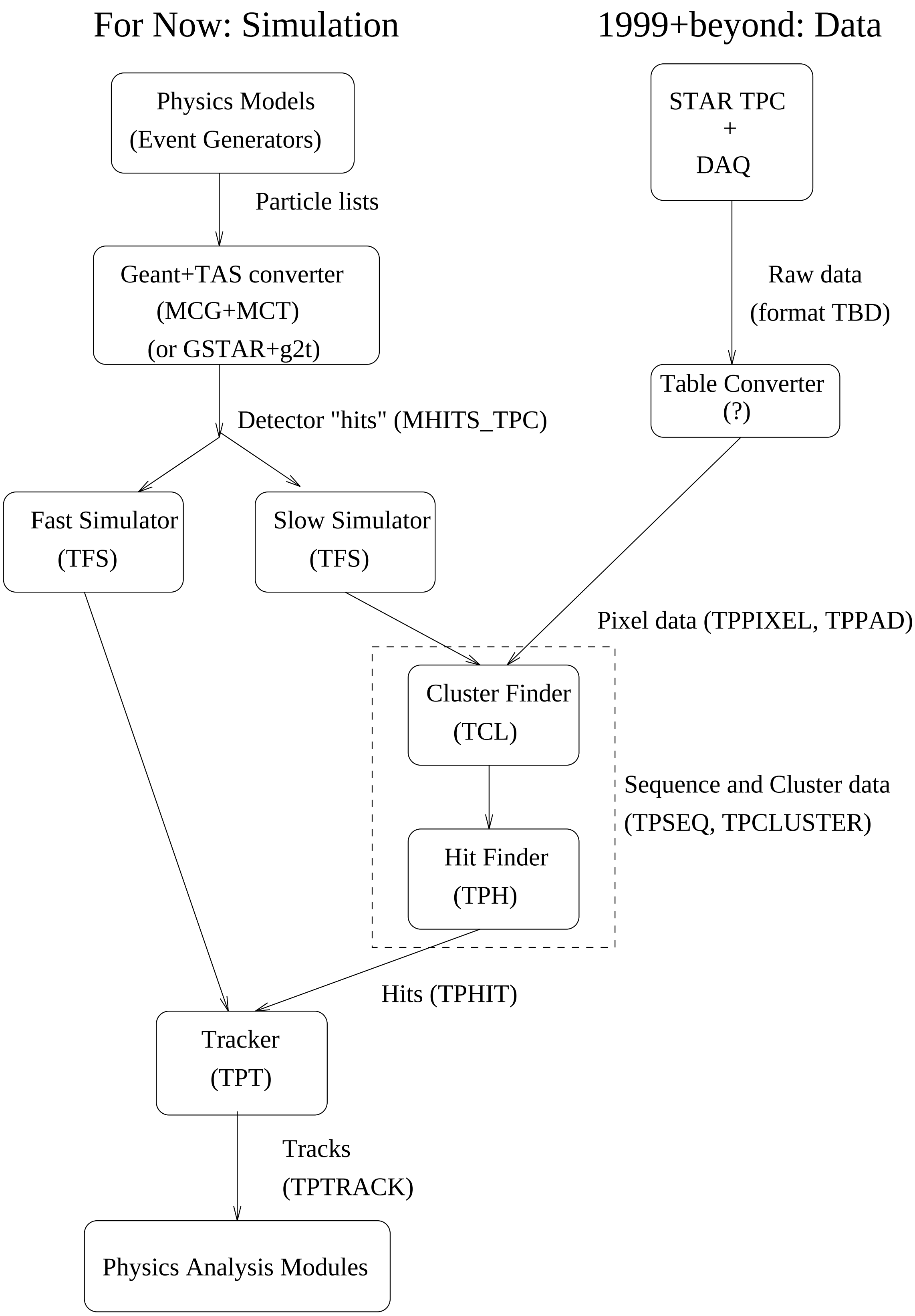

Figure 1: Relationship of low-level software modules.

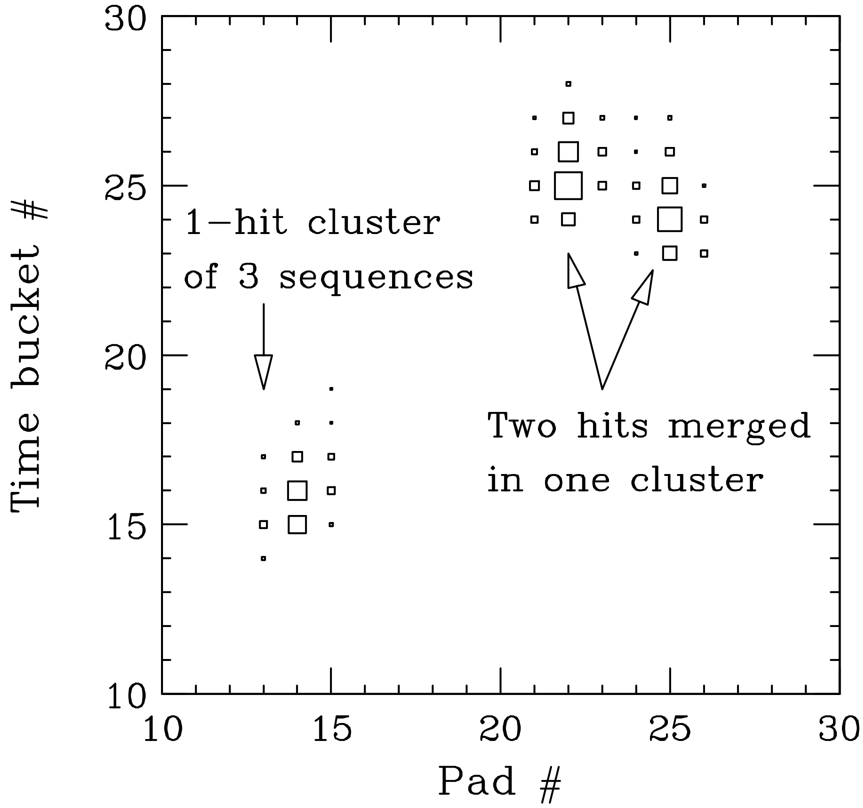

Figure 2: Pixel data for part of a padrow is shown for three tracks crossing the row. The size of a box indicates the magnitude of the ADC value of the pixel.

2.2 Table structures

Because pixel and related data are so voluminous, the table structures used to pass them are designed for compactness, and not neccessarily for transparency. Although the structures will evolve, it is useful to document the current table structures used at the pixel and hit level for those who will be working closely with the data.

2.2.1 Pixel table

The TPC is broken into 70 million pixels. Simulations suggest that a single Au+Au event at RHIC will produce 8%-13% pixel occupancy [1]. To avoid the large overhead with (1) tabulating the pixel data with no zero-suppression, or (2) associating a pixel identification field with each pixel ADC value, a pair of somewhat "intertwined" tables is employed to tabulate the zero-suppressed data. See Figure 3

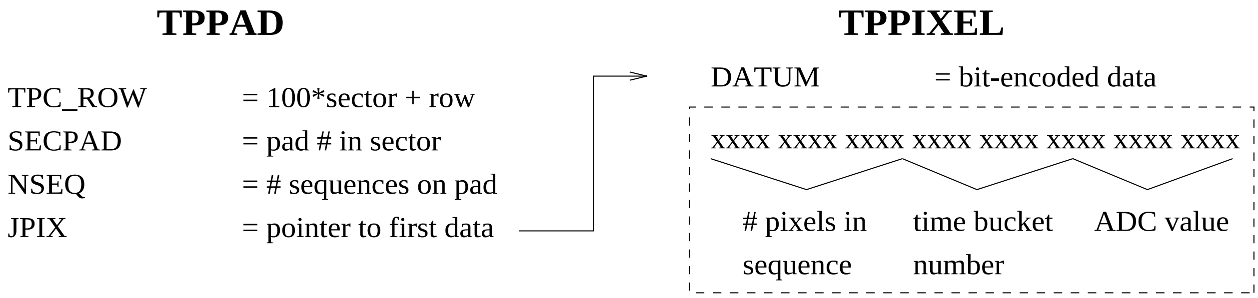

Figure 3: Structures used for tabulating TPC pixel data.

The table TPPAD has one entry for each pad which has at least one pixel above threshold. It contains pad identification information, and points to the start of its data in the TPPIXEL table. The sequences on that pad then follow one another.

The TPPIXEL table has one entry for each above-threshold pixel. Its single field is an I*4 variable bit-encoded with the following information: ADC value, time bucket number (a.k.a. TDC), and the number of buckets in the sequence that the pixel belongs to.

Note that at this hit level, the STAR software standard of using primary keys, and not pointing to the index of a table structure, is violated. Therefore, shuffling the entries in the TPPIXEL table will render the data indecipherable. It was decided to break the SAS standard at this level because of the considerable overhead involved in giving each pixel entry an ID field, and because there was no reason forseen for shuffling the raw pixel data.

Note also that it is assumed that the pixel data is in sequence form, with pointers built to the beginning of data on the pad. This is the current output format of the slow simulator, and it is assumed that the "table converter" (see Figure 1) will perform this function for the real data.

2.2.2 Sequence and Cluster tables

The first level of organization and summary of pixel data is in the form of clusters, which are composed of sequences. At this level, the clusterfinder uses another pair of intertwining tables to store cluster data. These are shown in Figure 4.

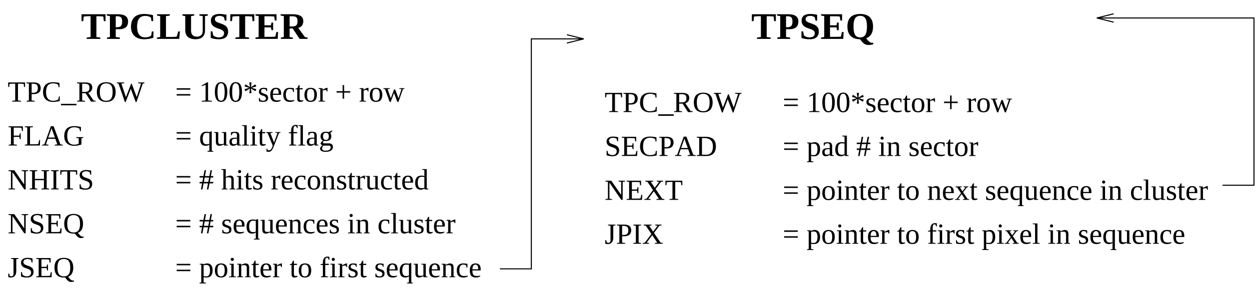

Figure 4: Structures used for tabulating TPC cluster data.

The TPCLUSTER table specifies the padrow of the cluster, its "trustworthiness" (not currently implemented), the number of hits that have been associated with this cluster by the hitfinder, and a path to the first sequence in a linked list of sequences that form the cluster.

The TPSEQ table simply contains some geographical information, a linked list self-pointer, and a pointer for every sequence to the first pixel of data (in the TPPIXEL table) in the sequence.

2.2.3 Hit table

The reconstructed hit data is stored in the TPHIT table, whose entries are listed in Figure 5.

All fields listed are filled by the hitfinder (TPH) module, except for TRACK and ID_GLOBTRK, which are filled by the tracking and global tracking software, respectively.

The FLAG field is set to 0 (1) if the hit was extracted from a single-hit (multi-hit) cluster pattern. See below. This flag should be useful for analysis modules (e.g. particle identification modules based on gas ionization, and some kink-finding modules) whose performance may improve if they only use "clean" hits.

The fields containing crossing angle information, ALPHA and LAMBDA, are filled in the hitfinder based on the shape of the hitpattern (see below), and are currenly overwritten by the tracking software if the hit is associated with a reconstructed track.

The following fields are currently not being filled and may eventually be eliminated from the table: DQ, DALPHA, DLAMBDA.

3 Clusterfinding Module TCL

The clusterfinder module, TCL, is based in part on an algorithm developed by Mike Levine, Michi Botlo, and David Patterson [2] which has been refined to identify general cluster shapes.

TPHIT

ID	= id of hit	ALPHA	= crossing angle of track along row
CLUSTER	= pointer to CLUSTER table	DALPHA	= uncertainty in alpha
FLAG	= quality flag	LAMBDA	= crossing angle of track along time
ID_GLOBTRK	= global track id	DLAMBDA	= uncertainty in lambda
TRACK	= TPC track id	PHI	= orientation of hit pattern w.r.t. padrow
NSEQ	= # sequences in parent cluster	X	= x-position in global coordinates
ROW	= 100*sector + row	DX	= uncertainty in x
PRF	= width of hit pattern along row	Y	= y-position in global coordinates
ZRF	= width of hit pattern along time	DY	= uncertainty in y
Q	= ionization charge of hit	Z	= z-position in global coordinates
DQ	= uncertainty in ionization charge	DZ	= uncertainty in z

Figure 5: Structure used for tabulating TPC hit data.

The algorithm is based on building up a list of time-overlapping sequences (see above) by using each sequence in the list as a "seed" around which to search for overlapping sequences, which are then added to the list. Any sequence which is added to any list is marked as "used."

A schematic of the clustering algorithm is shown in Figure 6. The module identifies and stores clusters of pixels for each padrow by beginning from the "left-bottom" (= low pad number, low time bucket number), and searching until it finds an unused sequence, one which has not been associated with a cluster. This sequence becomes the seed. The algorithm looks in succession to the left and to the right of the seed for unused sequences which overlap with the seed. These are added to the list, later to become seeds. When clustering around the seed is finished, the next sequence on the list becomes the seed. When all sequences have been seed, the cluster list is complete, and the seed for a new cluster is the first unused sequence looking again from bottom-left.

We note that this algorithm finds the most general clusters, as defined above. This includes the three clusters shown in Figure 7. As indicated in the figure, simpler (but faster) algorithms, such as that being proposed for the STAR trigger [2], would break such structures into more than one piece. While there may indeed be reason for breaking such structures into seperate hits, it is the philosophy in the TCL package to perform any hit deconvolution in the hitfinder package, to make the software more modular and maintainable.

The clusterfinder has essentially no tunable parameters. A field in the switch table (see below) specifies the minimum number of sequences for a set of overlapping sequences to be stored as a cluster. The default is 2, simply eliminating noise sequences occurring on individual pads. Reduction to 1 would allow noise into the hitfinding stage, while not increasing the number of good hits. Increasing the switch value to 3 would result in a loss of good two-pad hits.

Currently, sequences in the clusterfinder are defined by the first and last bucket above the threshold set by the front-end electronics (or the slow simulator). One could imagine a second (higher) threshold used by the clusterfinder. The potential utility of such a threshold needs study.

4 Hitfinding Module TPH

Cluster structures found by the clusterfinding module TCL are stored and passed to the hitfinding module TPH. It is the goal of this module to locate padrow crossings of parent tracks, and to some extent to characterize the track (in terms of crossing angles and ionization left by the track).

The hitfinding algorithm is more modular than the clusterfinding algorithm described above, allowing improvements in any component of the process. This is important, as it is forseen that improvements in the overall padrow crossing reconstruction efficiency will come from improvements in TPH (although improvements in speed may come as well in TCL). Here, we briefly discuss the main elements in the algorithm, as well as the current implementation of these elements. See Figure 8 for a schematic overview of the TPH algorithm.

Because of the high track density, there is a good chance that more than one hit can be found in a given

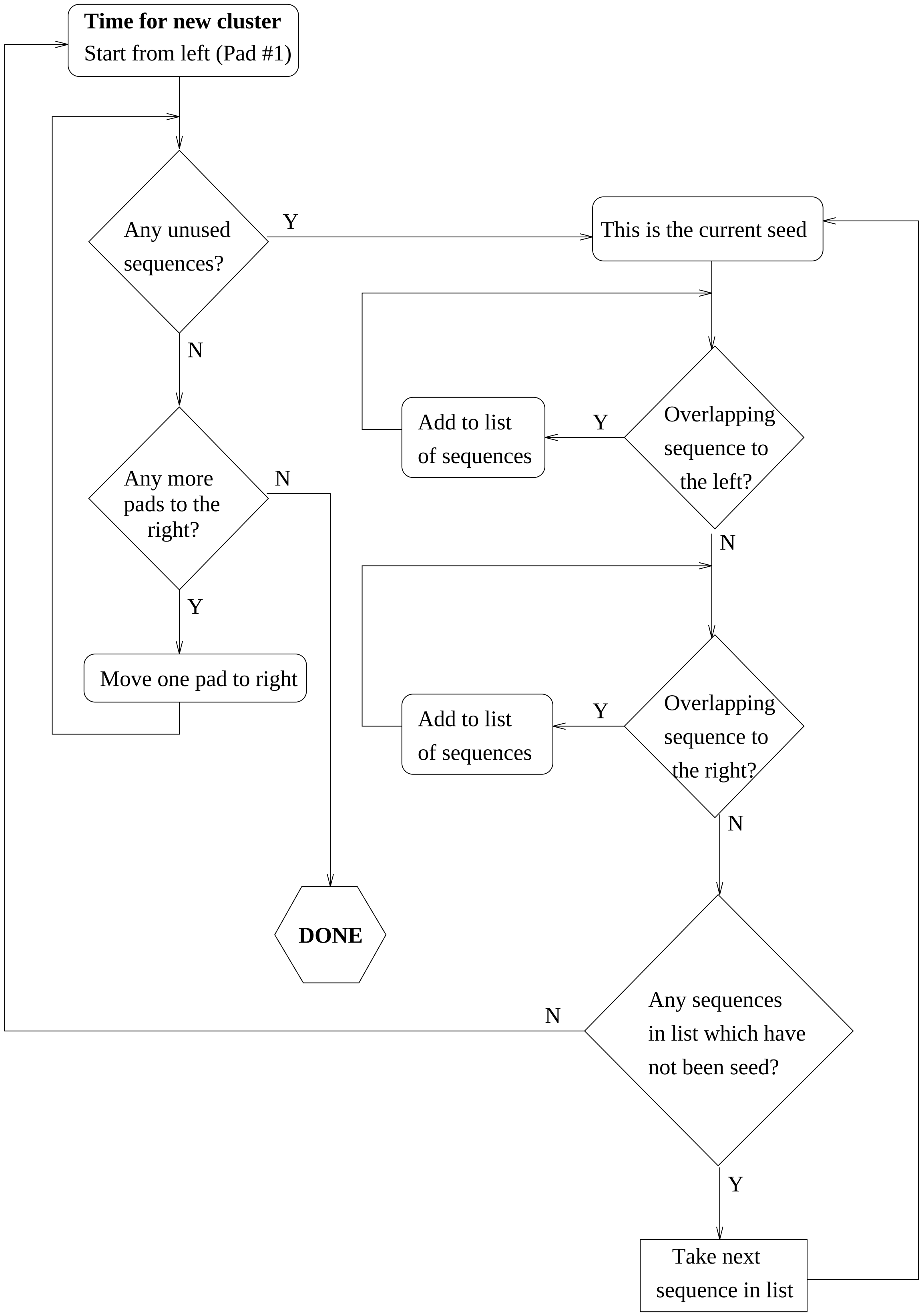

Figure 6: Schematic flow chart for clustering algorithm.

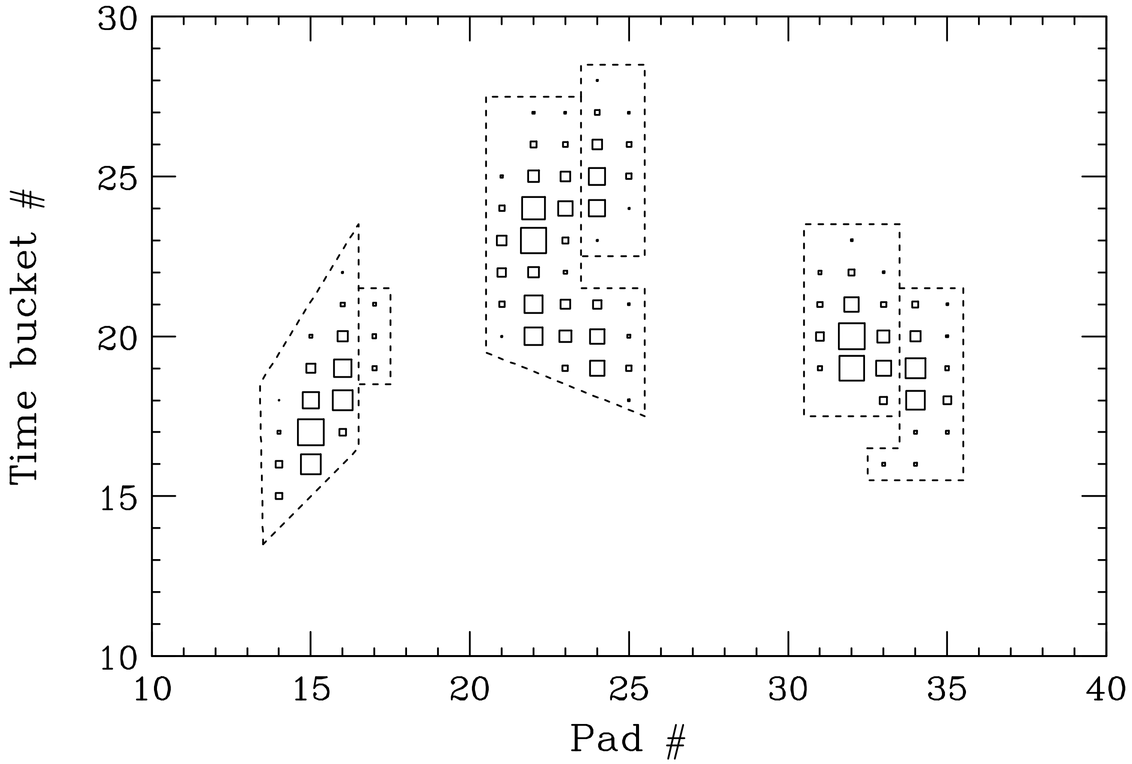

Figure 7: Three clusters that would be identified by the TCL algorithm. A simpler, more cpu-efficient algorithm breaks the clusters as indicated by the dotted lines.

cluster, especially on the inner padrows (see below). The "Mountainfinder" algorithm, described below, can be used to search for local maxima in the pad-TDC-ADC space. However, the search is not cheap in terms of cpu cycles (see below), and it is desirable to decide quickly whether a particular cluster is likely to warrent a search for multiple peaks. This decision is the first step in the TPH algorithm. A correlated plot of the RMS of the cluster in pad number and time bucket, shown in Figure 9, shows a clear division between clusters for which the Mountainfinder algorithm found a single hit, and those for which more than one hit was found. The position and orientation of the cut boundary may change with detector configuration (e.g. gas used, drift velocity) or improvement of the multipeakfinder, and so is specified by switch settings.

After the peak(s) are found in the adc-tdc-pad space for each cluster, local hit position information is extracted from the pixels around each peak. See Section 4.2 for details. These local positions are then translated into global x, y, and z coordinates.

Most tracking software is based upon fitting a set of space points (hits) with a functional form believed to represent a track. Thus uncertainties, as well as positions, are calculated and transformed to global coordinates. In order to obtain accurate and meaningful track fit parameters (e.g. particle momentum), it is important that these uncertainties represent an accurate estimate of the width of the residual distribution. Track characteristics that determine the spatial resolution of a hit have been studied. These characteristics include the hit signal-to-noise ratio and the drift distance, which are available at the hitfinder level of analysis. However, the resolution is also strongly determined by track crossing angle, which is not well known until the tracking level of analysis. Therefore, only partial uncertainty information for each hit is stored in TPHIT.(DX, DY, DZ). These uncertainties need to be modified at tracking time. See Sections 4.3 and 6.3 for details.

A measure of the energy lost by the track segment is given by the magnitude of the hit (in ADC counts), and is useful for particle identification. Stored in TPHIT.Q is the amount of this energy loss in keV:

$$TPHIT.Q = \frac{TSSPAR.AVE\_ION\_POT * TSSPAR.SCALE}{TSSPAR.GAIN} * \sum_i ADC_i, \quad (1)$$

Where ADC_i are the ADC values corresponding to member pixels of the hit. The gas and detector parameters– average ionization potential, electronics gain, and gase gain– which scale energy loss to ADC value are found in

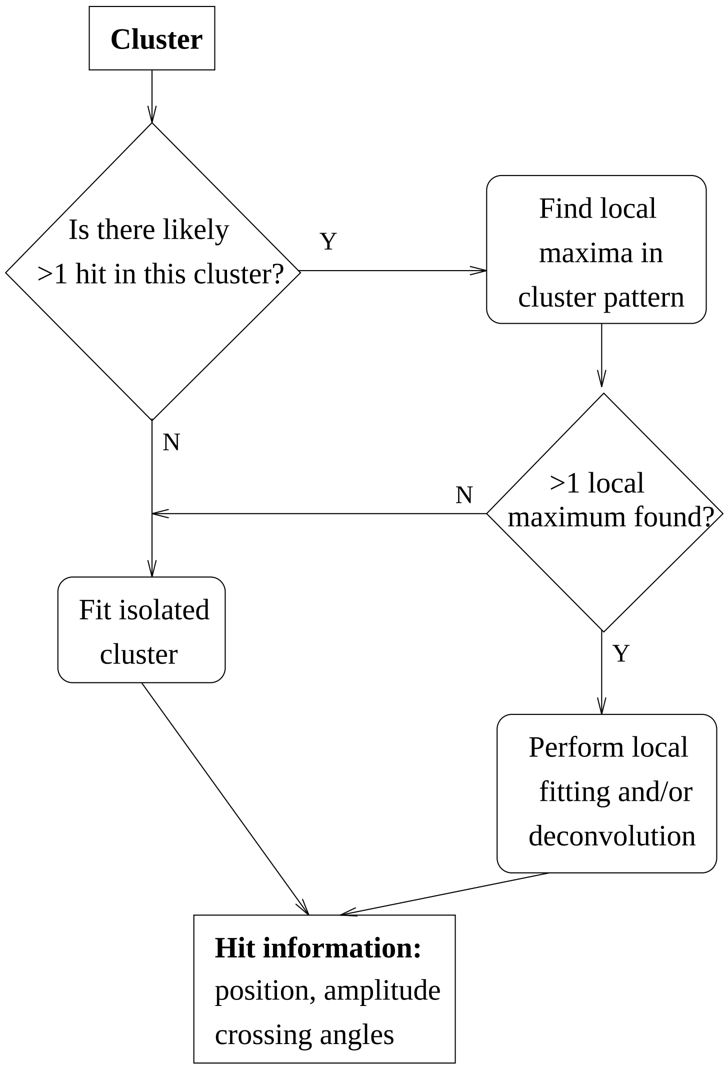

Figure 8: Schematic flow chart for hitfinding algorithm that is applied to each cluster. The default path is shown. Branch points for bypassing elements of the algorithm by non-default switch settings are not indicated.

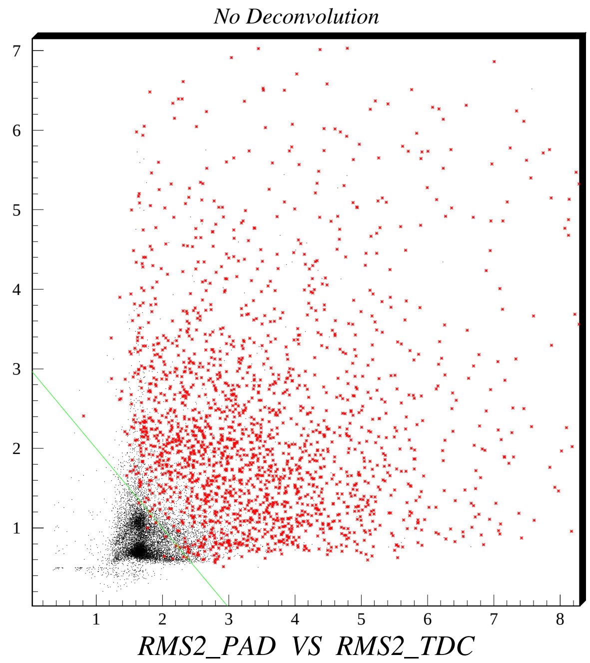

Figure 9: A simple cut on the rms of the cluster pattern in the pad and time bucket variables allows a quick decision whether it is worthwhile to search for many hits in the cluster. The line marks the cut. Multipeak searches on almost all clusters lying to the lower left of the cut resulted in only one peak found.

the slow simulator switches. See [1] for details.

From the shape of the cluster pattern, some information about the crossing angles of the parent track may be deduced. This information may be of use to tracking software in the decision whether to include a given hit in a track. See Sections 4.4 and 6.4 for details.

4.1 Multipeakfinder Moundainfinder

The multipeakfinder, Mountainfinder, is designed to find local peaks in the three-dimensional pad-tdc-adc domain of a cluster pattern, and to implement an on-the-fly peak-to-valley cut to select significant peaks.

The algorithm is diagrammed schematically in Figure 10. The pixels in a cluster pattern are sorted in decreasing order according to their ADC value. The first pixel in the list is then a local (and global) peak in the pattern. The position of this peak is stored and this pixel and its neighbors in the two-dimensional pad-tdc space are marked as "used." A "used" pixel is no longer considered a possible peak. For the first candidate, no peak:valley cut is applied, and the peak is stored.

The pixels are then considered according to the sorted list. If the pixel under consideration has already been marked "used," then its neighbors are now marked "used," and one considers the next pixel on the sorted list. Only a pixel who is "unused" *when his turn in the list comes up* has an ADC value larger than all of its neighbors, and so becomes a peak candidate. Although there is some redundancy in sometimes re-marking a "used" pixel as "used," significant time is saved by not having to compare each pixel's ADC value against its neighbors in each direction.

Even if a pixel is a local maximum, it is often useful to select on the significance of the maximum before designating it a peak. Such a cut may be useful, for example, to eliminate noise or digitization peaks. A peak:valley cut in two dimensions may be implemented in several ways. For example, one could take an average of the ADC values of a candidate pixel's neighbors, and compare it to the ADC value of the candidate. However, in our case, the relevant quantity is the relative height (ADC value) of the peak candidate to pixels between the candidate and other peaks. In Figure 11 is shown a candidate peak in a cluster pattern which contains two larger (and so previously-found) peaks. The relevant valleys with which to compare the candidate peak are defined by the lines connecting the candidate with the larger peaks. The pixel with the lowest ADC value

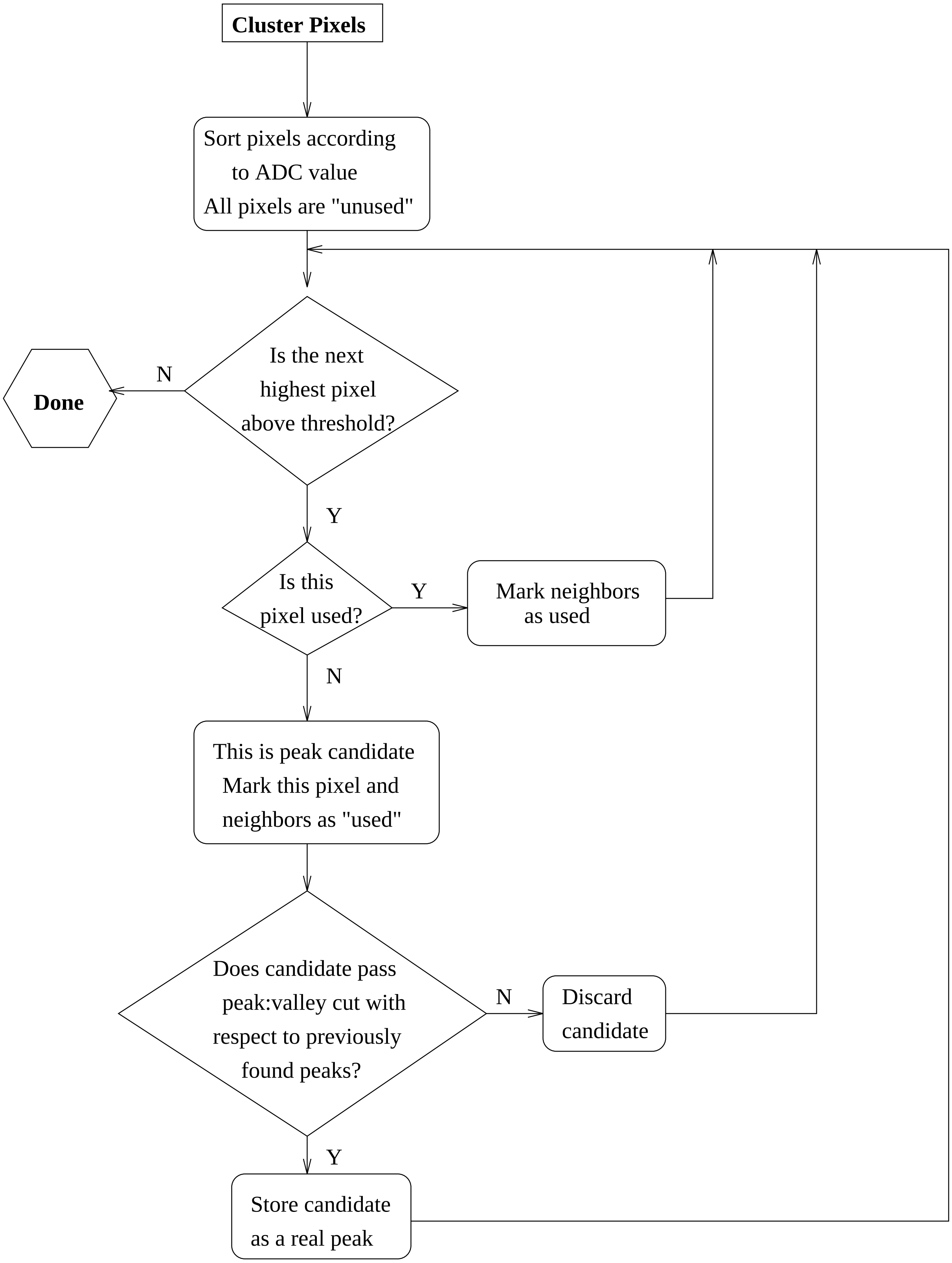

Figure 10: A schematic of the algorithm used to find multiple peaks in a cluster pattern.

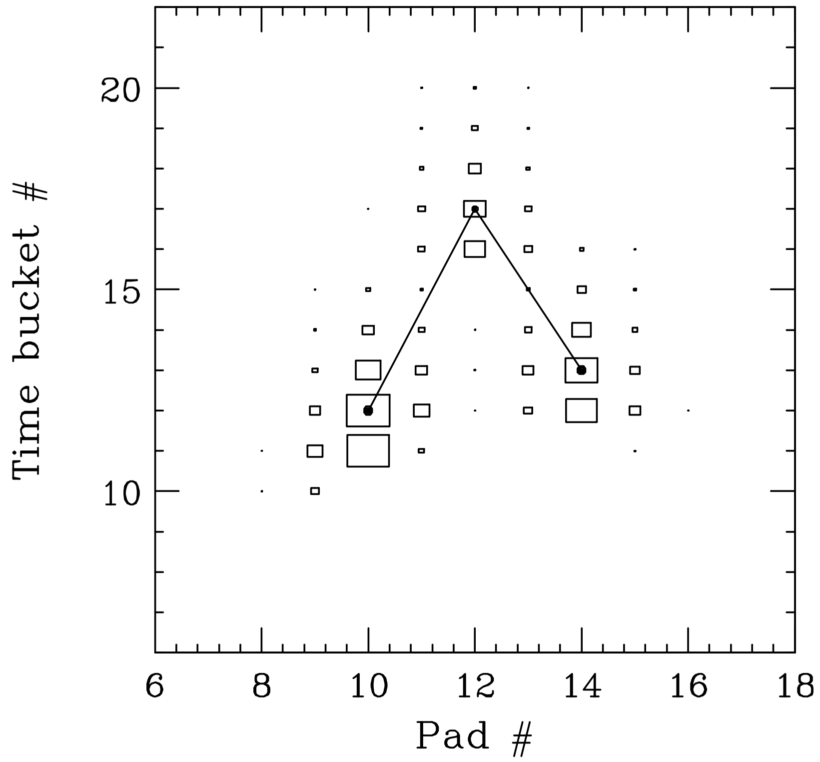

Figure 11: A cluster pattern in which two peaks have been found (larger filled circle) and a third candidate is being considered (small filled circle). Valleys are formed along lines in the pad-tdc space connecting the peak positions with the candidate position. Cuts are performed on the ratio of the height of the candidate peak to the lowest depth of these valleys.

lying along each line defines the valley value. The peak:valley cut is then made on the quantity (ADC value of candidate peak pixel)/(ADC value of valley) for each line (Typically a ratio of ≥ 1.3 is required. See below.) If a candiate passes the peak:valley cut with respect to all previously found peaks, it is considered a peak and stored. Any further peak:valley tests involving that peak will be against smaller candidate peaks. In the case of failure, the smaller candidate peak is discarded, but the larger peak is not. Thus, in this algorithm, if two possible peaks do not have a sufficient dip (valley) between them, the larger possible peak is kept.

Local peaks are sought until no pixels remain above a lower threshold.

Once the peaks in a cluster pattern are found the position and orientation information is associated with each (see above), and an entry is stored in the TPHIT table.

4.2 Local position determination

The hit position in local coordinates is extracted with simple forms using projections of the pixel information; if desired, more sophisticated techniques should be easy to substitute.

In the direction perpendicular to the padrow and parallel to the padplane (i.e. "radial"), the reconstructed hit is positioned at the center of the padrow. The two-dimensional pixel pattern gives no information regarding this direction.

The position along the padrow is determined using a 3-point Gaussian fit [3]:

$$x_0 = x_2 + \frac{\Delta}{2} \frac{\ln y_3/y_1}{\ln y_2^2/y_1 y_3} \tag{2}$$

where Δ is the pad pitch, and x_2 and y_2 are the position and the sum of ADC values on the pad with the largest signal. y_1 and y_3 are sums on the neighboring pads.

In the time direction, an offset weighted mean is used.

$$t_0 = \frac{\sum_i t_i P_i}{\sum_j P_j} - t_{offset} \tag{3}$$

where P_i is the ADC sum in time bucket i, projected onto the time axis.

The STAR shaper amplifier (SAS) has response function [1]

$$g(t) = \Phi(t - t_0)(\frac{t - t_0}{\tau})^2 e^{-\frac{t-t_0}{\tau}} \tag{4}$$

to a TPC signal originating at time t_0. The weighted mean of this function, as well as the convolution of it with a time-symmetric function (like Gaussian longitudinal diffusion), is offset from t_0 by

$$t_{offset} = 3\tau \tag{5}$$

This holds even for non-zero crossing angles λ.

4.3 Position uncertainties

The spatial resolution for hit measurement in the TPC is non-trivial, and so, therefore, are the uncertainties which try to estimate the resolution. Hence, the simple question "What is the spatial resolution?" cannot be answered simply. Here, and in 6.3, we describe how hit resolutions are determined in the cluster/hitfinder. See [3, 6, 7, 8, 9] for further details.

It is believed that such careful evaluation of spatial uncertainties based hit-by-hit on track characteristics represents an improvement over assignment of uncertainty based on signal width and signal:noise alone (perhaps scaled by some *ad hoc* factor). It is hoped that a corresponding improvement is thus achieved in the track parameters; verification of this improvement is a task for the future.

The parametrizations employed in the cluster/hitfinder are based on simulations with current (1/96) "standard" versions of Geant and TPC slow simulator for STAR. Tuning these packages to better reflect reality may require some changes to TCL. These will be reported in future as needed.

4.3.1 Uncertainty along the padrow (x and y directions)

The hit position in the padrow direction is fitted with a 3-point Gaussian parametrization [3]. This fit is performed on the projection of the ADC values onto the padrow; the pad with the largest projection (highest $\sum$ADC), and its two neighbors are used. Neglecting for the moment finite-α and finite electron number effects, the uncertainty of measuring the centroid of the Gaussian shows a linear dependence on the uncertainty of the ADC projections [8]:

$$\begin{aligned} \delta x_0 &= \frac{1}{\ln \frac{y_2^2}{y_1 y_3}} \sqrt{(\frac{\Delta}{2} - x_0)^2 (\frac{\delta_1}{y_1})^2 + (2x_0)^2 (\frac{\delta_2}{y_2})^2 + (\frac{\Delta}{2} + x_0)^2 (\frac{\delta_3}{y_3})^2} \\ &= \frac{\sigma^2}{\Delta} \sqrt{(\frac{1}{2} - \frac{x_0}{\Delta})^2 (\frac{\delta_1}{y_1})^2 + (2\frac{x_0}{\Delta})^2 (\frac{\delta_2}{y_2})^2 + (\frac{1}{2} + \frac{x_0}{\Delta})^2 (\frac{\delta_3}{y_3})^2} \end{aligned} \tag{6}$$

where y_i and δ_i are the pad projections and their uncertainties, respectively, x_0 is the centroid position relative to a pad edge, and Δ is the pad pitch.

What is the uncertainty on the pad projection (δ_i) to use in Equation 6? The noise for the signal on each pixel comes from 3 sources [1]: (1) electronic noise originating before or in the shaper amplifier (SAS), (2) electronic noise in the switched capacitor array (SCA) chip, and (3) digitization noise– here meaning the "noise" associated with rounding a number to integer value. The noise contribution from the second and third terms varies randomly from bucket to bucket, and so is "white" noise. However, the noise signal coming through the SAS is filtered with the repsonse function of the amplifier, and so the noise signal on one bucket is correlated with that on the following bucket; see Figure 12.

If the SAS noise were white ($\tau_{SAS} \to 0$) then the error on a pad projection would be

$$\delta_i = \sqrt{N_i (RMS_{SAS}^2 + RMS_{SCA}^2 + RMS_{digit}^2)}, \tag{7}$$

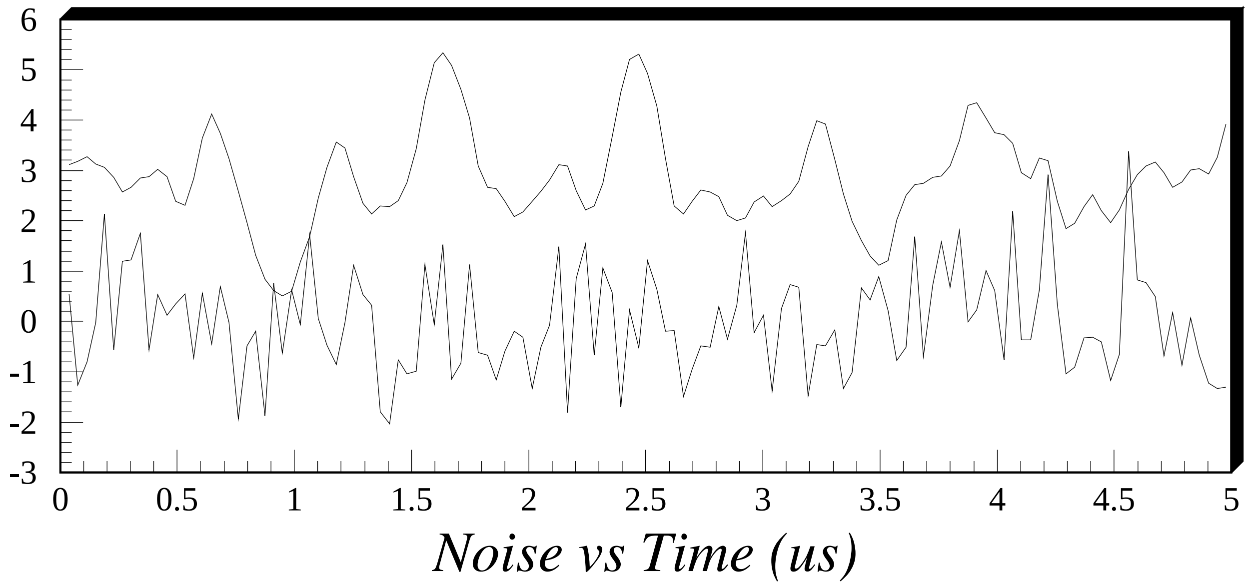

Figure 12: A random ("white')' noise spectrum is compared to a filtered noise spectrum, offset on the y-axis. Time sampling corresponds to using 1024 time buckets on each side of the TPC. Both spectra are normalized to have unit RMS. Noise coming through the SAS will be correlated in time, due to finite bandwidth; therefore, the error on the sum of consecutive time buckets will not add in quadrature.

where $RMS_{digit}=\frac{1}{\sqrt{2}}$ in ADC counts, and N_i is the number of pixels that contribute to the pad projection.

If the SAS had an extremely long time constant ($\tau_{SAS} \rightarrow \infty$), then all pixels contributing to a pad projection would be shifted by the same amount by the SAS, so that

$$\delta_i=\sqrt{N_i^2 RMS_{SAS}^2 + N_i(RMS_{SCA}^2 + RMS_{digit}^2)}. \tag{8}$$

Equations 7 and 8 represent extreme cases, as τ_{SAS} is in reality on the order of the sampling time. Since the SAS contributes heavily to the noise (800 e^- RMS, corresponding to 1.19 ADC counts, as compared to 0.45 and 0.71 ADC counts RMS from SCA and digitization, respectively), it does matter which equation one chooses. Simulations with 512 time samples in P-10 suggest that N$\simeq$9, so that the δ_i differs by a factor of 2.5 between the two approximations. Comparison of uncertainties to residual distributions suggests that Equation 8 is the better approximation for our situation, and that is what is used in TPH. See also Section 6.3.

Calculation of the centroid uncertainty for a 3-point Gaussian fit is straightforward. However, the facts that (1) the signal originated through the creation of a finite number of primary electrons (or electron clusters), and (2) from a track that crosses the padrow at a non-zero angle α, also affect the spatial resolution in the TPC, quickly dominating it, in fact. The position resolution then takes a form [9]

$$\sigma^2=\sigma_{3-ptGauss}^2 + \sigma_{B'}^2 \frac{L \sec\alpha}{dE/dx} + \sigma_{C'}^2 \frac{dE}{dx} \tan^2\alpha \cos\alpha \tag{9}$$

where $\sigma_{3-ptGauss}$ represents the centroid uncertainty from Equation 6, and L is the drift distance for the hit. The second and third terms represent the contributions to the uncertainty from finite electron number and finite crossing angle effects, respectively. In Equation 9, $\sigma_{B'}$ and $\sigma_{C'}$ are constants, depending only on detector geometry and TPC gas [9]. These constants must be determined empirically. See Section 6.3.

Note the competing dependencies in the three terms in Equation 9. For example, the first and second terms decrease with increased signal size, while the third *increases*. This is important, as this third term quickly dominates the resolution. See Figure 13 for calculations of the spatial resolution in the outer sector using P-10 gas in the TPC. It is seen that the so-called "intrinsic resolution," which depends on the signal:noise and is given by Equation 6, rapidly becomes insignificant compared to finite-α effects, especially for larger signals. We note that this behaviour, in particular, may change if delta electron production changes in STAR Geant simulations.

For this reason, the position uncertainty cannot be determined at the hitfinding level. Any attempt at this level to produce an "average error" whose distribution resembles that of the average residual distribution would be an unsatisfactory comprise that overestimates the uncertainty of some hits while underestimating that of others. TPH enters the value given by Equation 6 in TPHIT.DX and TPHIT.DY. From this information, knowlege of the constants $\sigma_{B'}$ and $\sigma_{C'}$, the energy loss of the track (in TPHIT.Q), the drift distance (from TPHIT.Z), and a hypothesis of the crossing angle α, the position uncertainty can be calculated hit-by-hit. Tracking software must then update these uncertainties DX and DY, as information becomes available about the parent track.

4.3.2 Uncertainty in time (z direction)

The uncertainty in the mean, specified in Equation 3, arises from uncorrelated as well as correlated uncertainties in the ADC sums (due again to white and "filtered" noise). These may be expressed as [9]

$$\delta t_0^{uncorr}=\frac{1}{S}\sqrt{\sum_i (t_i - \bar{t})^2 (\delta P_i^{uncorr})^2} \tag{10}$$

$$\delta t_0^{corr}=\frac{1}{S}\sum_i |t_i - \bar{t}| \delta P_i^{corr} \tag{11}$$

where $\bar{t}$ is just $t_0 + t_{offset}$, and $S = \sum_j P_j$. The overall uncertainty is then estimated as

$$\delta t_0=\sqrt{(\delta t_0^{uncorr})^2 + (\delta t_0^{corr})^2} \tag{12}$$

Values of the uncorrelated and correlated noise for use in Equations10 and 11 are estimated as

$$\delta P_i^{uncorr}=\sqrt{N_{pads,i}}\sqrt{RMS_{SCA}^2 + RMS_{digit}^2} \tag{13}$$

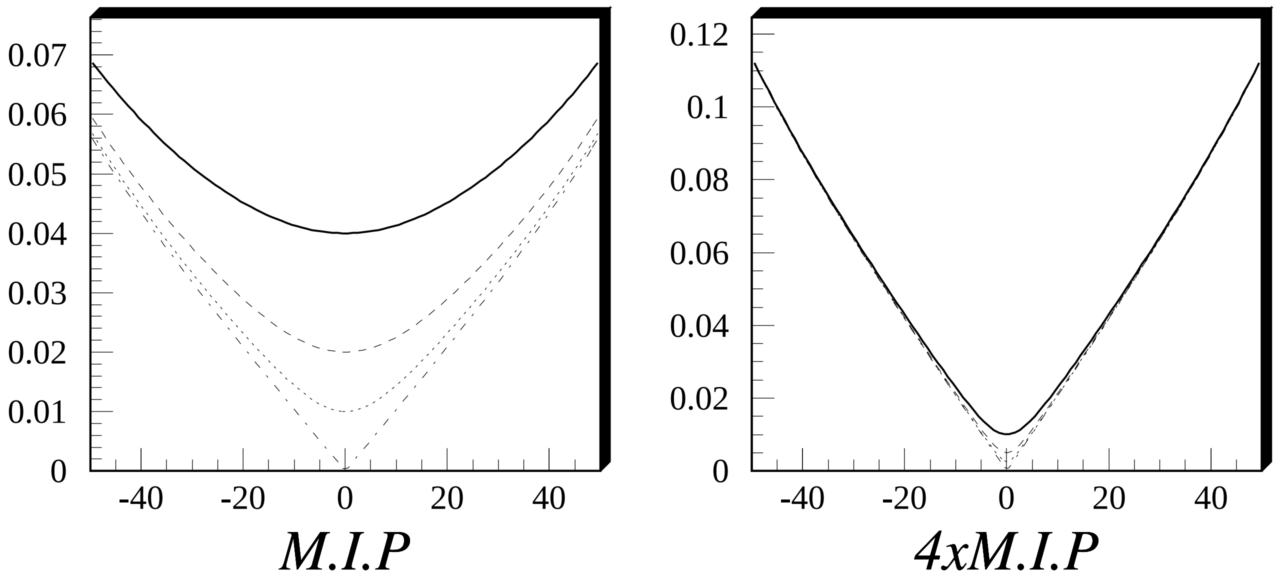

Figure 13: Position resolution as a function of crossing angle α in the outer sector using P-10 gas, according to our parametrization. Resolution for minimum ionizing particle hits (m.i.p) and hits 4 times as large are shown. Curves are shown for zero noise (dotdash lines), the default settings of the slow simulator (dotted curves), twice the default noise in the simulator (dashed curves), and 4x the noise in the simulator (solid curves). This last case corresponds roughly to the situation considered in the STAR CDR, where, in addition, a slightly different pad geometry was assumed.

$$\delta P_i^{corr} = \sqrt{N_{pads,i}} RMS_{SAS} \tag{14}$$

where $N_{pads,i}$ is the number of pads that contributed nonzero ADC values to the i^{th} time bucket of the time projection.

(Of course, since in calculating the mean with finite sums instead of integrals in Equation 3, we have an additional error $\sim (\Delta t g''(t))^2$, where Δt is the width of a time bucket, and the shaper response g(t) is given in 4. We ignore this here, as the effect of time bucket width, when $\geq$ 512 buckets are used, has been seen to be small [11].

As with the uncertainty along the padrow, Equation 12 estimates the uncertainty in time (z-position) only for λ=0 tracks. A non-zero crossing angle will degrade the spatial resolution in z. We follow Equation 9 in parametrizing the time resolution.

$$\sigma^2 = \sigma^2_{signal:noise} + \sigma^2_{F'} \frac{L \sec \lambda}{dE/dx} + \sigma^2_{G'} \frac{dE}{dx} \tan^2 \lambda \cos \lambda \tag{15}$$

As with the resolution along the padrow, the constants $\sigma_{F'}$ and $\sigma_{G'}$ must be determined empirically. See Section 6.3.2.

Again, for the uncertainty in the time (z) direction, only an estimate of the first term in Equation 15 is provided by the cluster/hitfinder and stored in TPHIT.DZ. Tracking software, which has a hypothesis of the crossing angle λ, must update this uncertainty.

4.4 Cluster orienation and crossing angles

The orientation of the pixel pattern corresponding to a hit is also calculated. Shown in Figure 14 is a single-hit cluster pattern which is skewed due to large track crossing angles α and λ. The angle of the pattern with respect to the padrow (indicated by the overlaid line in the figure) is correlated with these crossing angles. This angle is stored in TPHIT.PHI. What should be relevant to tracking algorithms is an estimate of the crossing angles α and λ. As intuitively expected, the angles α, λ, and ϕ are observed to be correlated, so that knowledge of two of them gives the third. For tracks originating from the center of the TPC, $\lambda = \mathrm{Tan}^{-1}(z/r)$, where r is the radial distance of the hit and z is the seperation along the TPC axis between the hit and the central E-field membrane. TPH stores this value into TPHIT.LAMBDA.

Given ϕ and λ, we should be able to estimate the crossing angle α. If we use local coordinates such that x is along the padrow (in the "pad direction"), y is perpendicular to the padrows, and z is along the drift direction, then the two crossing angles are given by

$$\alpha = \frac{\Delta x}{\Delta y} \tag{16}$$

$$\lambda = \frac{\Delta z}{\Delta y} \tag{17}$$

whereas our calculated "orientation angle" ϕ might be expected to be written

$$\phi \stackrel{?}{=} \frac{\Delta z}{\Delta x}. \tag{18}$$

which would lead one to conclude that alpha could be calculated by

$$\alpha_{cal} \stackrel{?}{=} Tan^{-1} \frac{tan(\lambda)}{tan(\phi)} \tag{19}$$

However, the orientation angle ϕ is more complicated, and we find that an empirical relation is needed.

In the simulations we find that there are strong correlations of α with our assumed λ and our calculated ϕ. Two branches in the relation of the true α vs α_{cal} (as given by Equation 19) are seen, and neither one is linear.

Experimenting with different parametrizations leads to a relation

$$\alpha_{cal} = \begin{cases} Tan^{-1}(\frac{1}{2}\frac{tan(\phi)}{tan(\lambda)}) & \text{if } |\phi| > 10^o \\ Tan^{-1}(\frac{tan(\phi)}{tan(\lambda)}) & \text{if } |\phi| < 10^o \text{ and hit is from one-hit cluster} \\ Tan^{-1}(4\frac{tan(\phi)}{tan(\lambda)}) & \text{if } |\phi| < 10^o \text{ and hit is from one-hit cluster} \end{cases}$$

The resolution for reconstructing the crossing angle α is only fair (see Section 6.4 below). However, this hit-by-hit estimate of the track crossing angle, which intrinsically involves the third dimension (perpendicular to the padrow), based on the two-dimensional pad-tdc space, may still be useful to the tracking software in situations in which it is not clear to which track a hit belongs.

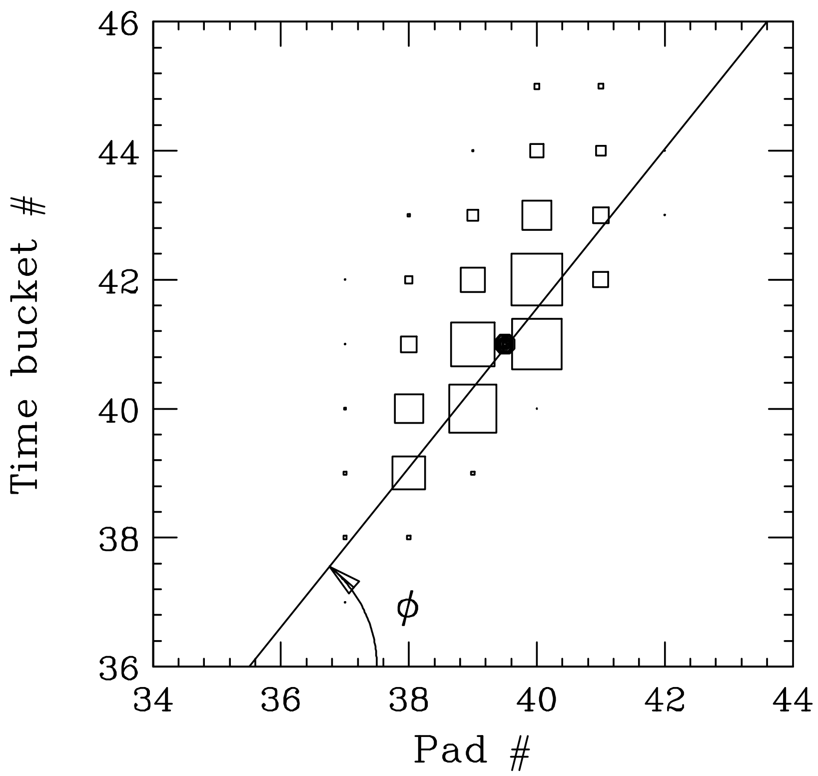

Figure 14: The position and orientation of a hit in a single-hit cluster are shown. The large skew (non-zero orientation) of the pattern arises from large track crossing angles.

5 Switches table

The switch table for the TCL package will likely grow. Listed here are the fields of the table **TCLPAR** in its current form. Currently, no switches affect the clusterfinder TCL. All switches listed are for TPH.

- **DECON**. I*4 =(1/0) to enable/disable multipeak finding. If DECON=0, only one hit is assigned to each cluster. Default is DECON=1.
- **MC**. I*4 =(1/0) to enable/disable assignment of hits to Monte Carlo tracks via the TPMCPIX table, which gives a Monte Carlo track ID for each pixel. This is useful for evaluating the hitfinding efficiency, but is somewhat cpu-costly and should be turned off (MC=0) when estimating the cpu cost of cluster/hitfinding for the real data.
- **MF_CPAD**. I*4 Controls the number of neighbor pixels in the pad direction to set to "used" in the Mountainfinder algorithm (see above). Default is MF_CPAD=1.
- **MF_CTIM**. I*4 Controls the number of neighbor pixels in the time (TDC) direction to set to "used" in the Mountainfinder algorithm (see above). Default is MF_CTIM=1.
- **MF_MIN**. I*4 Sets the lower threshold for Mountainfinder to stop searching for peaks. See above and first decision in Figure 10. Default is MF_MIN=20.
- **MIN_NSEQ**. I*4 Specifies the minimum number of sequences in a cluster in order that the cluster be considered a hit. (see above). Default is MIN_NSEQ=2.
- **TFIT**. I*4 Specifies the method used to fit local pixel information to extract position in time (z). TFIT=0 selects the use of a simple mean. TFIT=1 uses a more realistic fit to the pulse shape. Default is TFIT=0.
- **TRIAGE**. I*4 =(1/0) to enable/disable the use of a quick (triage) cut on a cluster pattern to decide whether to search for more than one peak. See Figure 9. Default is TRIAGE=1.

- **TRIAGE_RMSCUT**. R*4 Sets the value of the triage cut line in the RMS_PAD-RMS_TDC plane. See 9. This switch may have to split into two switches, one for the RMS_PAD direction and one for the RMS_TDC direction, if the appropriate cut line is no longer at -45^o. Default is TRIAGE_RMSCUT=3.0.
- **DXY_FACT**. R*4 Sets the "fudge factor" by which to increase by hand the errors in the x and y global coordinates. Kept for diagnostic study. Default is DXY_FACT=1.0.
- **DZ_FACT**. R*4 Sets the "fudge factor" by which to increase by hand the errors in the z global coordinate. Kept for diagnostic study. Default is DXY_FACT=1.0.
- **MF_PV**. R*4 Sets the value of the peak:valley ratio in the Mountainfinder routine. See above. Default is MF_PV=1.3.

6 Performance

Here we discuss the performance of the cluster/hitfinding software in terms of its hitfinding efficiency and resolution with simulated pixel data, as well as the cpu cost of the package.

6.1 Hitfinding performance

As mentioned in the introduction, the ability of the cluster/hitfinder to identify as many real padrow crossings as psossible (while finding as few false crossings as possible), will affect all TPC and global physics analyses that follow it. Here, some of the main aspects of this ability are discussed.

The HIJING event generator was used to generate p-p and Au-Au events, which were passed through GEANT (and the MCT analysis module) and then through the TPC slow simulator TSS [1]. P10 gas settings with 512 time buckets were used in all simulations, with all other TSS switches set to default values.

For the p-p events, typically 11000 of the 69.9 million (0.16%) pixels were occupied (i.e. above threshold), arising from about 800 padrow crossings of about 40 charged particle tracks.

For the Au-Au events, pixel occupancy was about 7-8%, with about 5300 padrow crossings per sector; this would lead to about 128000 crossings for the full TPC, but we often simulated and analysed only a few sectors at a time, due to machine memory constraints. About 2000-3000 charged particles passed through the active volume of the TPC.

6.1.1 Number of contributors to a cluster pattern

An idea of the task of the cluster/hitfinder is given by looking at the typical number of tracks contributing to a clustr pattern. The slow simulator associates with every occupied pixel the Monte Carlo track ID of one of the tracks that contributed to that pixel. This information is stored in the TPMCPIX table. Then, by counting the number of different track ID's associated with pixels in a cluster pattern, one gets an idea of how many hits a perfect hitfinder may find in the cluster pattern. A distribution of this quantity for clusters found in the inner sector (row# $\leq$ 13) and outer sector (row# $\geq$ 13) is shown in Figure 15. It is seen that single-hit clusters are the most common, but that there is a good deal of overlap as well, especially in the inner rows.

Evaluation of resolution and hitfinding efficiency is based on comparisons of reconstructed hits on a padrow with Monte Carlo padrow crossings. In reality, the current simulation chain gives several (5-9) Monte Carlo "hits" for each track crossing a padrow. In our evaluations that follow, we use the Monte Carlo hit that is closest to the center of the padrow as the padrow crossing.

6.1.2 Hitfinding efficiency

Here, a hit is declared "found" if a reconstructed hit is found within 2 cm of a padrow crossing. If more than one padrow crossing is nearby, then the reconstructed hit is associated with that crossing that has the same Monte Carlo ID as the reconstructed hit. As should be clear from the measurement of spatial resolution (below), using a 1 cm cut instead of 2 cm has little effect on the results.

Two quantities relevant when discussing overall hitfinding performance are the efficiency and the purity, defined as,

$$efficiency = \frac{\#Generated - \#Lost}{\#Generated} \tag{20}$$

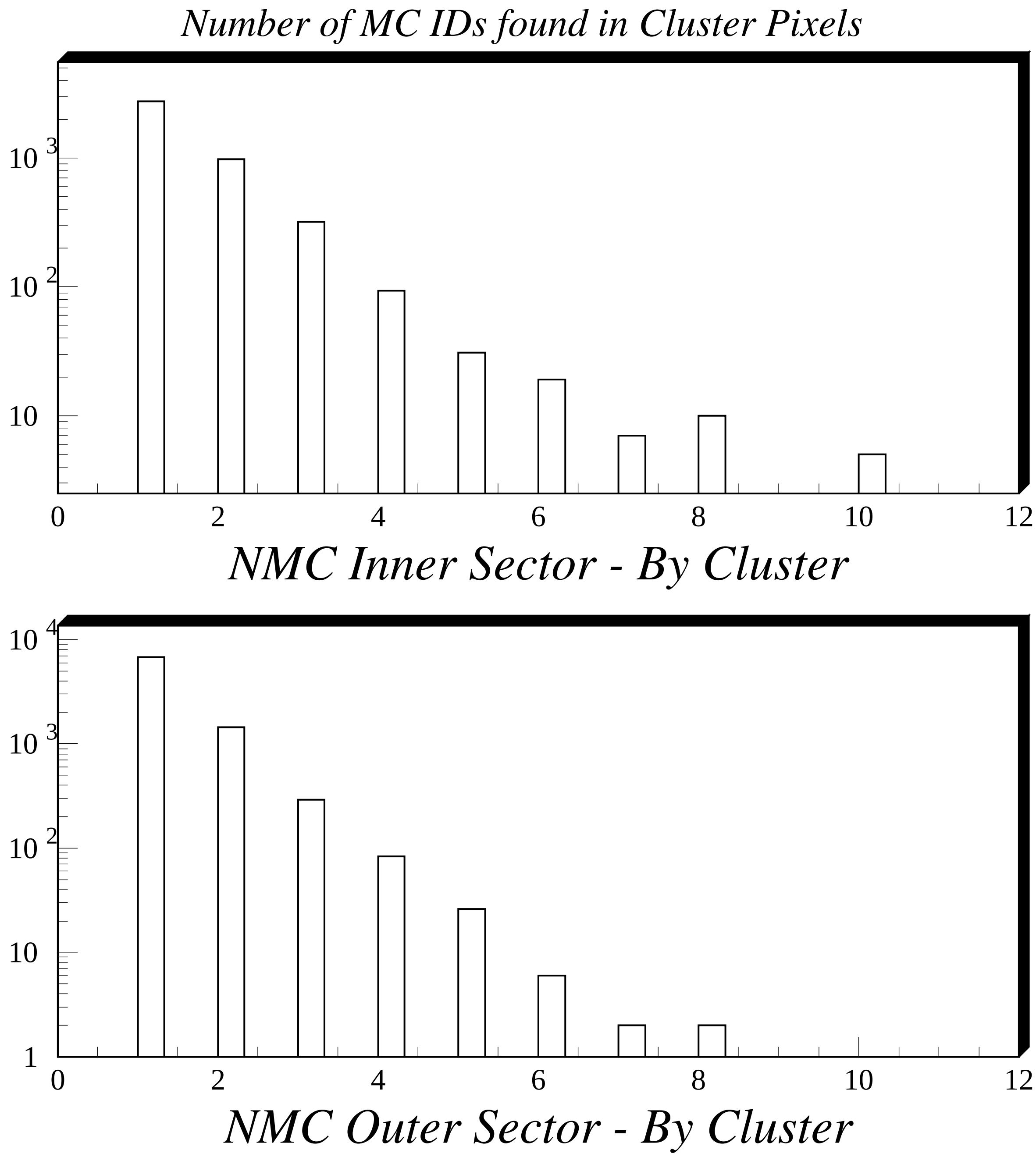

Figure 15: Distributions of the number of different Monte Carlo track ID's associated with pixels of clusters found on the inner and outer sectors from a simulated Au-Au event.

$$purity = \frac{\#Reconstructed - \#Ghost}{\#Reconstructed} \tag{21}$$

where a "lost" padrow crossing is one which is not reconstructed (usually because two crossings were too close to resolve), and a "ghost" hit is a reconstructed hit which cannot be associated with any padrow crossing (usually arising from noise).

Clearly, high efficiency is desirable, but a high hit purity may also be desired if the tracking software is easily confused or overwhelmed by noise hits.

Experimentation with the cluster/hitfinding algorithms with a variety of switch settings indicates that basically only one of the parameters strongly affects the efficiency and purity of the hitfinding, and this is the peak:valley ratio (setting DECON=0 may be seen as equivalent to setting MF_PV to ∞). Figure 16 shows the efficiency and purity averaged for the whole TPC when hitfinding in simulated p-p and Au-Au events. Clearly, for single p-p events, deconvolution is not needed, whereas in Au-Au events, much better efficiency can be obtained with a small price to pay in purity with a peak:valley cut of 1.3.

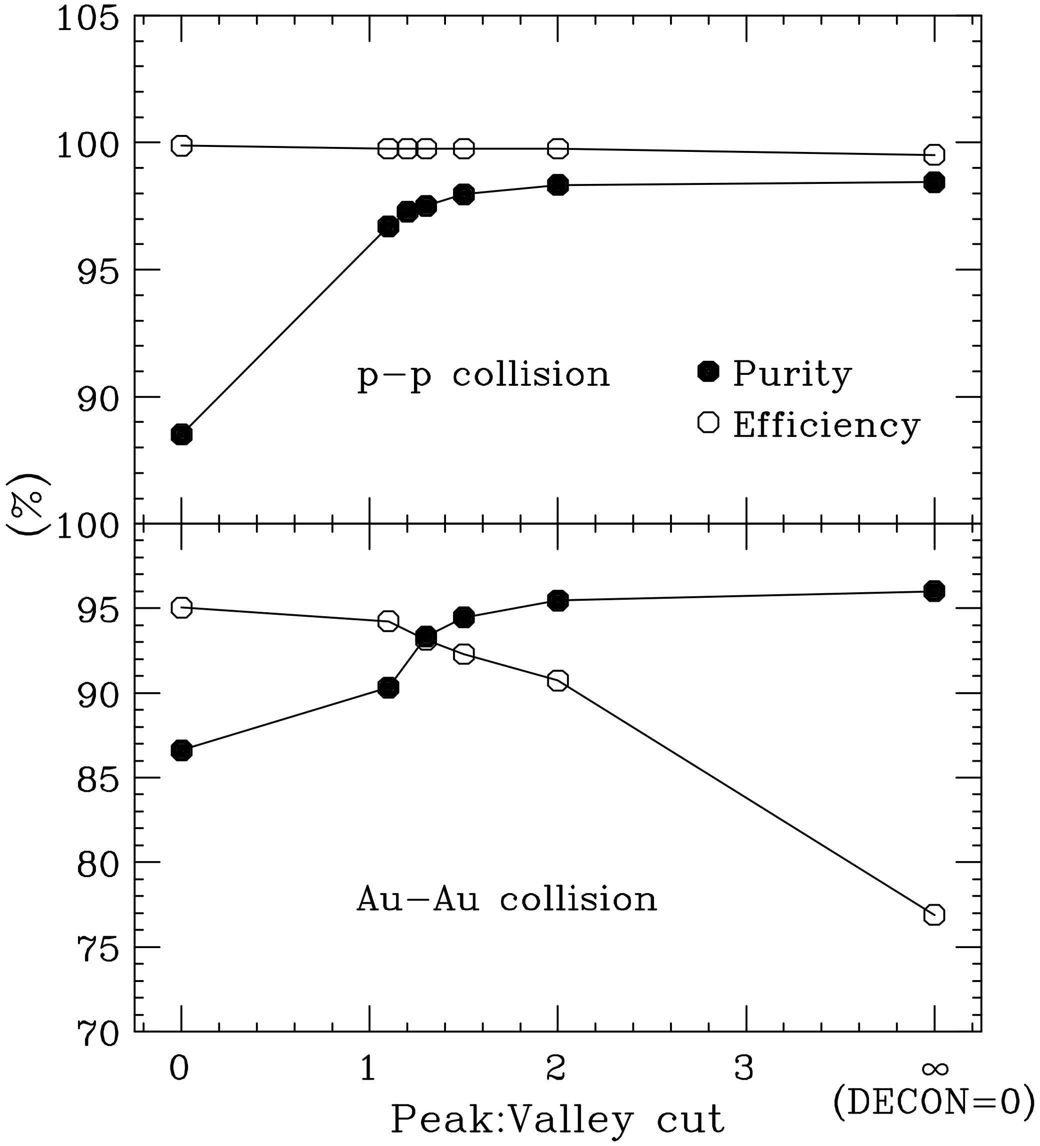

Figure 16: Purity and efficienty as a function of the peak:valley cut used in the multipeak finder for a p-p event and a Au-Au event. Note that also shown is performance with DECON=0, which is essentially an infinite peak:valley cut

As mentioned above, the peak:valley cut basically determines fully the purity and efficiency, so one can study a correlated plot of these two variables. Such plots are shown in Figure 17, where purity vs. efficiency is shown for the inner and outer sector in a Au-Au event. The values corresponding to the default of TCLPAR.MF_PV=1.3 are also indicated. Which value of the peak:valley cut is best for a given tracking algorithm must be explored. Also, it may well be that the ability to seperately set the peak:valley cut for inner and outer sectors would be desirable.

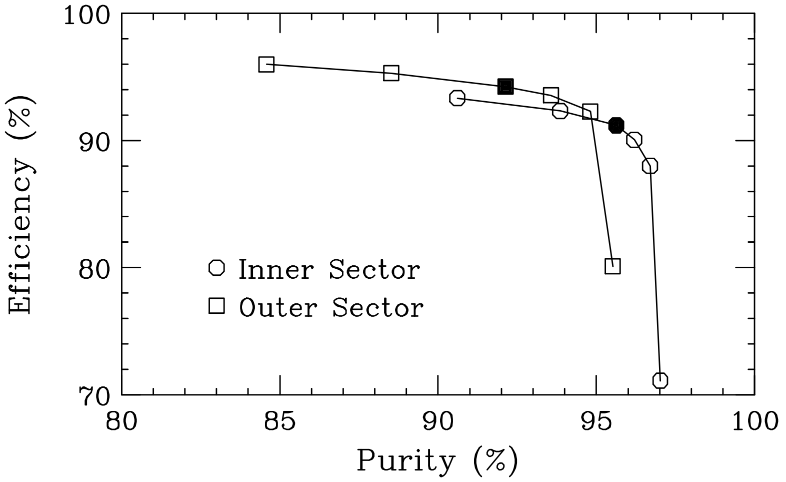

Figure 17: Seperately for the inner and outer sectors, the reconstructed hit purity and efficiency relations obtained for simulated Au-Au events, as the peak:valley cut value is varied. The filled points show performance for the default peak:valley=1.3 setting.

6.2 Two-hit merging distance

An overall hitfinding efficiency is useful, and mapping its effect on the particle-finding efficiency as a function of, e.g. η and p_t, would provide a useful *single particle* efficiency. However, of great importance to HBT studies, kinkfinding studies, and the like, is the two-track resolution, which will be affected by the two-hit merging distance.

In Figure 18 are 2-dimensional histograms of the yield of hits on a padrow as a function of the relative seperation in the pad and tdc direction (although these units have been converted to centimeters). The left panel of Figure 18 is the relative seperation yield of all Monte Carlo padrow crossings, while the right panel shows the relative seperation yield of reconstructed hits. It is clear that padrow crossings that are very close together are not being resolved, and only one hit is found.

Figures 19 and 20 show the ratio of the reconstructed and generated relative seperation yields for padrows in the inner and outer sector, when deconvolution is turned off (TCLPAR.DECON=0) and when standard tcl settings are used. Although the efficiency is not a step function as a funciton of relative distance, the merging distance goes from about 3-4 cm to about 1.5 cm when deconvolution is turned on. Therefore, we would like to emphasize that, although the overall hitfinding efficiency only increases about 20% when deconvolution is turned on (going from 71% to 91% for inner sector), those hits that are recovered with deconvolution are exactly those that are important for, say, HBT studies.

6.3 Uncertainties and resolution

As mentioned above, tracking software usually makes use not only of space points, but of the uncertainty in the position, in the fit. Therefore, it is important that the uncertainty be understood and estimated hit-by-hit. Also mentioned was the fact that the magnitude of the uncertainty can depend on quantities not known at the hitfinder level (i.e. crossing angles), so that uncertainties will have to be "updated" at tracking time.

Any estimate of the position resolution of the cluster/hitfinding software depends strongly on the degree to which the simulations are realistic. In our case, it is known the simulator does not incorporate so-called "finite electron" effects, due to the fact that the number of primary ionization electrons liberated by the track as it passes over the padrow is not infinite. Studies by Howard Wieman have shown that finite electron effects will worsen position resolution by about 20% [4]. This will affect the resolution along the padrow (in the "pad direction") as well as along the time direction. Also, ExB distortions and other field distortions are not taken into account in the simulations. Hence, the resolutions obtained from this study are overly optimistic. Furthermore, we note that the accuracy of the current standard versions of STAR Geant and TPC slow simulator have been questioned, in particular about the handling of delta electrons and energy fluctuations. In order to focus on the cluster/hitfinder, we use these standard versions in the resolution study, with the understanding that updates to the information given here will come as needed in the future.

The residual along the padrow and in the time (z) direction was calculated for each hit that could be associated with a Monte Carlo padrow crossing. These were then transformed to residuals in cm in the x, y, and z directions. These redidual distributions for Au-Au events are shown in Figure 21. Position resolution for hits coming from a one-hit cluster are typically 15% better than those for overlapping hits.

6.3.1 Padrow directions x and y

In the following, we attempt to understand the resolution in the direction along the padrow, so as to correctly estimate the position uncertainty hit-by-hit.

The first term in Equation 9 does not depend on crossing angles and only on the signal:noise and signal shape. To evaluate whether the hitfinder estimates this component of the uncertainty correctly, we generated low-track-density toy events, in which tracks cross the padrow with zero crossing angle. The field was turned off (in this simulation, this did not change the transverse diffusion from its field-on value) so that highly ionizing particles as well as m.i.p's would provide straight tracks. Then, the electronics noise in the simulator was varied and the resolution studied. "100%" noise was the nominal noise: $RMS_{SAS} = 800e^-$, $RMS_{SCA} = 390e^-$. RMS resolutions and uncertainties are shown in Figure 22 as a function of noise. There, it is seen that the hitfinder slightly overestimates the uncertainty from the first term of Equation 9. This is understood to be the consequence of using Equation 8, which is based on an idealized assumption of infinite shaping time, to calculate the uncertainty on pad projections.

Figure 23 shows that, for each noise value, the resolution scales as expected with signal size: resolution increases linearly with the inverse of the signal size, dE^{-1}.

Having established that the resolution due to the first term in Equation 9 is understood and well-estimated by the hitfinder code, we turn to tracks with finite crossing angle. The slow simulator currently does not incorporate finite electron effects, so the second term in Equation 9 is not expected to contribute to the resolution ($\sigma_{B'}$=0) [9]. The third term, which arises from fluctuations in energy loss as a track crosses the padrow at non-zero angle, will increase the resolution strongly. The resolution in the outer sector from a Au-Au event is shown in Figure 24 as a function of crossing angle α.

A series of low-multiplicity events were run through the slow simulator, with varying vertex position (in order to study seperately effects of crossing angle λ and drift length L), and varying electronic noise. The RMS of the residual distribution was calculated as a function of α, dE/dx, L, λ, and noise level. No dependence of the padrow spatial resolution was seen as a function of λ or L, as expected since $\sigma_{B'}$=0 (see above). The dependence of the RMS on α and signal size is shown in the case of zero electronics noise in Figure 25. The worsening of the resolution with crossing angle is clear, as is the fact that the rate of this worsening is greater for large hits (large dE/dx) [9].

Equation 9 contains two constants, $\sigma_{B'}$ and $\sigma_{C'}$, which depend only on the chamber geometry and gas properties [9]. (The primes on the constants are in part to distinguish them from similar "constants" used elsewhere [6, 7, 10] which depend on hit properties.) These constants must be determined empirically. We find using P-10 gas for the outer sector $\sigma_{C'} \approx 0.06\frac{cm}{\sqrt{keV/cm}}$ and for the inner sector $\sigma_{C'} \approx 0.03\frac{cm}{\sqrt{keV/cm}}$. $\sigma_{B'}$ will be determined after TSS is updated. We note that the values for $\sigma_{C'}$ may also change if, for example, the energy loss gets calculated differently.

The tracker, with $\sigma_{B'}$ and $\sigma_{C'}$ in hand, and a hypothesis of α, can accurately estimate the uncertainty of the hit position, as is shown in Figure 26, where the uncertainties estimated by a tracker using the form 9 are compared for the range of hit classes to the rms of the residual distribution for that hit class.

As numerical examples of the resolution in the x- and y- directions, we note that, averaging over all particles in a Au+Au event, the Gaussian widths of the residual distribution $\sigma_{x,y} \simeq 400\mu m$, whereas for those hits whose parent tracks had $\alpha \leq 2^o, \sigma_{x,y} \simeq 60\mu m$. It is this small α resolution that scales as signal:noise.

6.3.2 The time direction z

Here, we follow closely the discussion of the resolution and uncertainty along the padrow given in the previous Section.

The first term in Equation 15 is independent of crossing angle. This is the component of the resolution that the cluster/hitfinder estimates in TPHIT.DZ. Figure 27 shows the average residual and estimated uncertainty as a function of λ for non-overlapping hits in a Au+Au event. The residual approaches the estimated uncertainty for small λ, but crossing angle effects dominate the resolution otherwise.

In Figure 28, it is seen that those hits with large energy loss show the fastest increase with crossing angle, similar to the large-α effect discussed above.

For the tracker to update the uncertainty in the time direction according to Equation 15, the constants $\sigma_{F'}$ and $\sigma_{G'}$ must be determined. As in the case with the resolution along the padrow, the coefficient corresponding the magnitude of finite electron statistics distortions, $\sigma_{F'}$, vanishes with the current slow simulator. Fits of the width of the residual distribution for non-overlapping hits from Au+Au collisions, for various cuts on L, λ, and dE/dx, show best fit for $\sigma_{G'} \approx 0.02 \frac{cm}{\sqrt{keV/cm}}$ for the outer sector and $\sigma_{G'} \approx 0.01 \frac{cm}{\sqrt{keV/cm}}$ for the inner. Again, the values of these numbers may change as the configuration paramters of the simulator and hitfinder change.

As an example, the z-resolution of hits from a Au+Au event, integrating over dE/dx and crossing angles, is RMS=2.4 mm, σ_z=1.5 mm, while selecting hits with crossing angles $\lambda \leq 2^o$ gives σ_z =219 μm.

6.4 Crossing angle α determination

As mentioned above, an orientation angle phi is calculated for each hit found. We also assume the track is primary, and trivially obtain λ. From this, we use an empirical formula to extract the crossing angle α.

We find reasonable correspondance between the calculated crossing angle and the true one, on average, as shown in Figure 29 for a Au+Au event. However, the dispersion about this average is large. The crossing angle α found by the cluster/hitfinder may be useful to the tracking software in some cases, but clearly a better resolution in the determination of α would be desirable. Our understanding of the relationship between the hit pattern shape and the crossing angles is obviously incomplete, and this issue remains a topic for further study.

6.5 CPU Cost

The cluster/hitfinder represents most of the cpu usage for reconstructing particles in the TPC [12]. Here, we present some numerical examples of the cpu cost for TCL in current form. Running the slow simulator and TCL for a subset of the TPC, and for the full TPC indicates that the cpu time scales with the number of sectors considered, as one would naively guess.

Numbers are presented for the software running on two machines; the first numbers are for rsgi00.rhic.bnl.gov, an SGI machine with a 40 MHz clock, and the numbers in parentheses are for rncsv1.lbl.gov, a Sparc-20 equivalent with a 60 MHz clock. A benchmark test has clocked rncsv1 as about 3 times faster for integer operations [13]. For a Au+Au event (generated by HIJING), the clusterfinding (TCL module) takes 2.7 (1.0) cpu sec/sector, or 64.8 (24.0) seconds for the full TPC. With standard switch settings (see Section 5), the hitfinder (TPH module) takes 61.8 (55.5) sec/sector, or 24.7 (22.2) minutes for the full TPC. Running TPH with no deconvolution (TCLPAR.DECON=0) takes 23.5 (11.2) seconds/sector, or 9.4 (4.5) minutes for the full TPC.

6.6 what settings the tracker likes

this must be studied.

7 Work to be done

There is still work to be done on the cluster/hitfinding software. Examples of such work follow.

To move towards more realistic software that can function outside idealized simulations, a way of handling missing or "hot" electronics channels must be implemented. Similarly, a way to correct for field distortions at the hitfinder level should be implemented. Also, a "front end" must be incorporated that will unpack the 8-bit ADC data (compressed from the 10-bit data by ASICs in the front end electronics), and possibly apply offline pedestal, gain, and time-offset corrections. This will likely be implemented as another analysis package that precedes TCL.

It will be worthwhile to compare quantitatively the offline cluster/hitfinder (described here), and the faster one used in the trigger software. Such a comparison is planned.

Currently, there are not too many switches to tune in TCL (a desirable situation), but for those that do affect performance (e.g. TCLPAR.MF_PV), it is neccessary to see which setting works best with the tracking software that follows TCL.

It is often envisioned that, for analyses like HBT where resolution of overlapping hits may be important, a "second pass" at hitfinding may be made, using tracking information obtained in the first pass. Implementation of this mode of running does not exist yet.

Estimating the crossing angle of the parent track from the shape of the cluster pattern is an interesting and possibly useful twist to clusterfinding. As discussed above, our understanding of this estimation could be improved.

8 Acknowledgements

I would like to thank Dr. Wen Gong for getting the TPC slow simulator into shape enough to begin real work with it. Thanks also to Dr. Dhammika Weerasundara for his thorough study of how the various parts of the "slow simulation" chain work together, and how they compare to the fast chain, and for his improvements to the slow chain. Thanks to Dr. Howard Wieman for explanation of all things technical, especially with the slow simulator, and to Dr. Roy Bossingham for warnings of how things will change when more realistic gas properties are implemented in the slow simulator. Finally, thanks to Dr. Iwona Sakrejda for helping me navigate the vagaries of the STAR software system, for technical advice on hitfinding/tracking issues, and for helpful suggestions on the documentation.

References

[1] W. Gong, STAR Note 0197

[2] M.J. LeVine, M. Botlo, and David Patterson, private communication

[3] Gerry Lynch, PEP4 Note TPC-LBL-78-17 (1978)

[4] H.H. Wieman, private communication

[5] D. Weerasundara, private communication

[6] H. Aihara, et al., IEEE Trans. Nucl. Sci. **30**, 76 (1983)

[7] S.R. Amendiola, et al., Nucl. Instr. Meth. **A283**, 573 (1989)

[8] H.H. Wieman and I. Sakrejda, STAR Note 137

[9] M.A. Lisa, For a future STAR Note.

[10] The STAR Collaboration, STAR Conceptual Design Report, (1992)

[11] H.H. Wieman, and STAR Note 151.

[12] S. Kumar, *et al* ROCOCO II Committee report, and upcoming STAR Note.

[13] D. Irmscher, private communication.

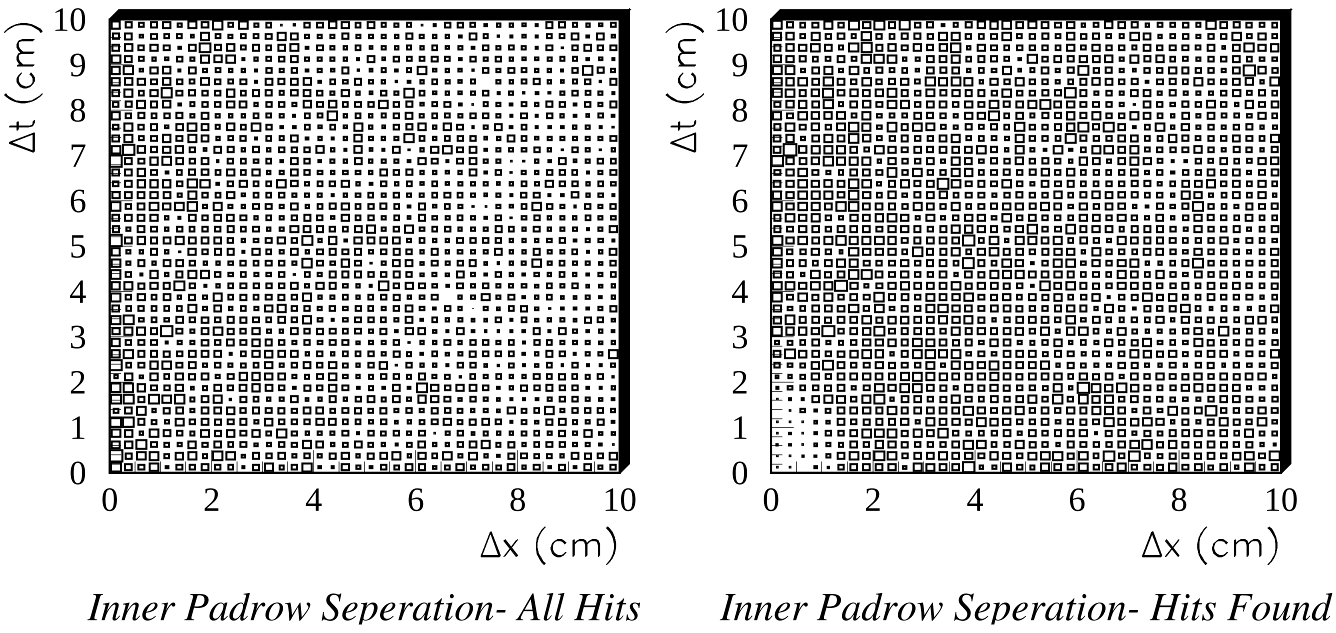

Figure 18: Relative seperation yield in the time and pad directions for Monte Carlo padrow crossings (left) and for reconstructed hits (right).

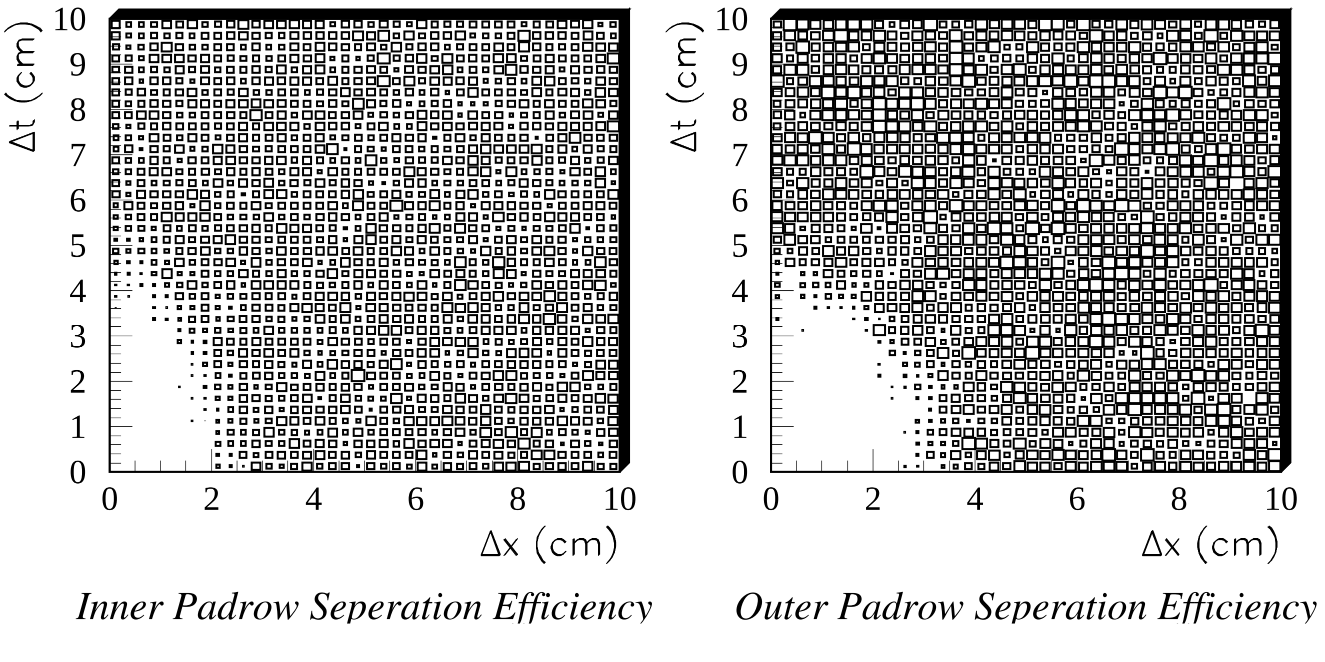

Figure 19: The efficiency of finding a hit as a function of its spatial distance from another hit on the same padrow, when deconvolution is turned off in the hitfinder.

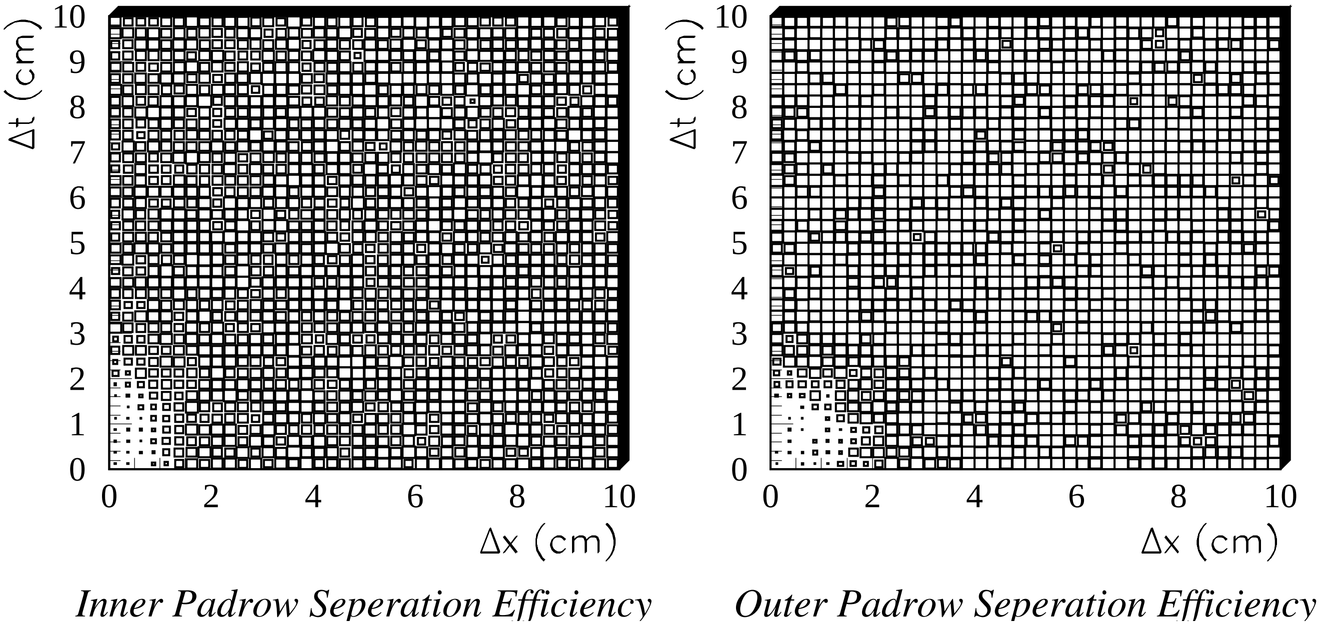

Figure 20: The efficiency of finding a hit as a function of its spatial distance from another hit on the same padrow, for default settings in the cluster/hitfinder.

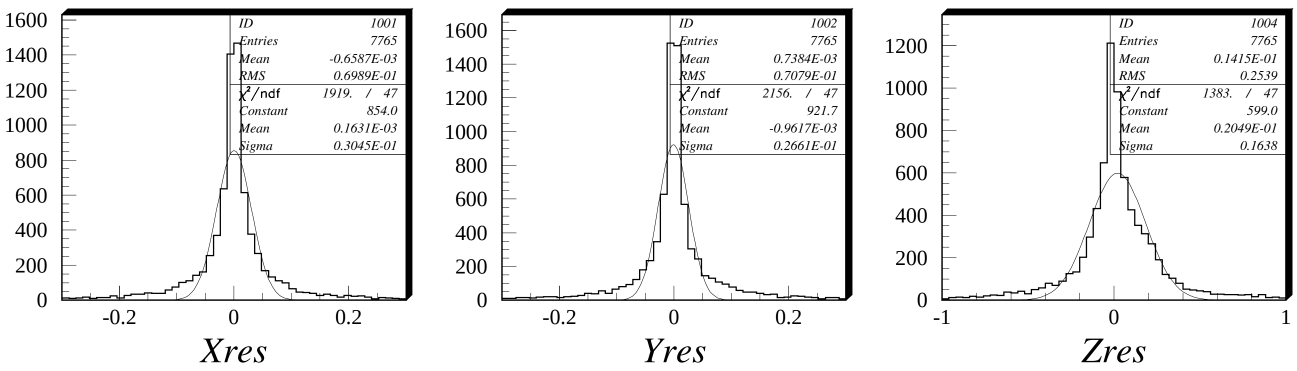

Figure 21: The resolution in the global x-, y-, and z-directions is shown for a simulated Au-Au event. Significant non-Gaussian tails on the distributions lead to a large RMS value of the residual distribution.

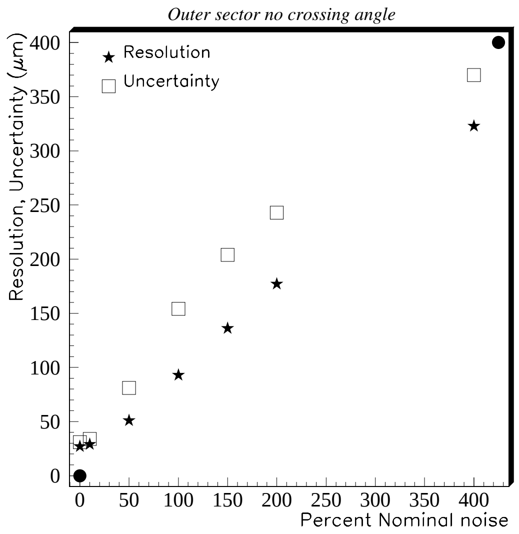

Figure 22: Simulated α=0 tracks passed through the slow simulator with varying levels of noise are used to study resolution and calculated uncertainties. The amount of noise is quantified in terms of the nominal electronics noise in the default simulator. Note that resolution and uncertainty do not vanish at "% noise"=0, because the effective noise introduced by digitization is not removed.

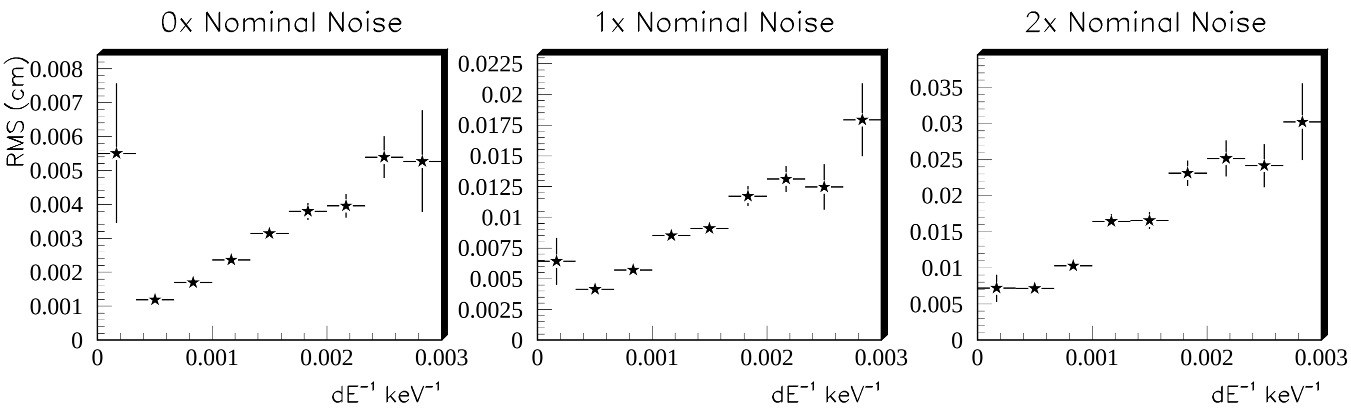

Figure 23: The dependence of the resolution (RMS of the residual distribution) is seen to show a linear dependence on the inverse of the hit size for simulated α=0 tracks, for all values of noise. Note the scale change for resolution as the noise increases.

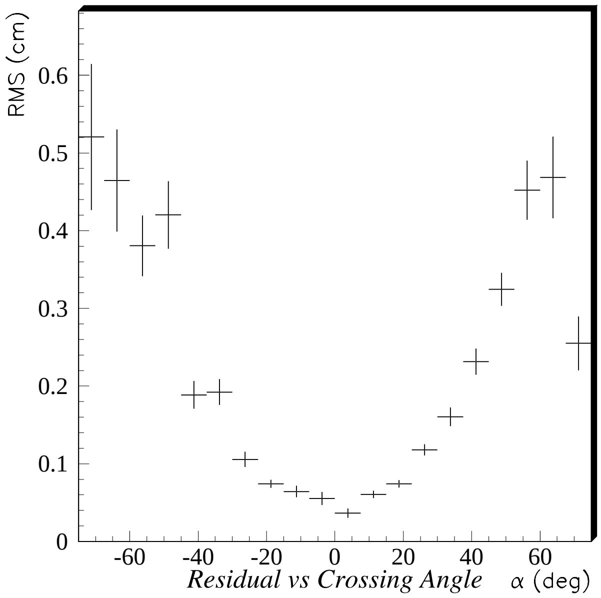

Figure 24: Resolution as a function of padrow crossing angle is shown for the outer sectors in a simulated Au+Au event.

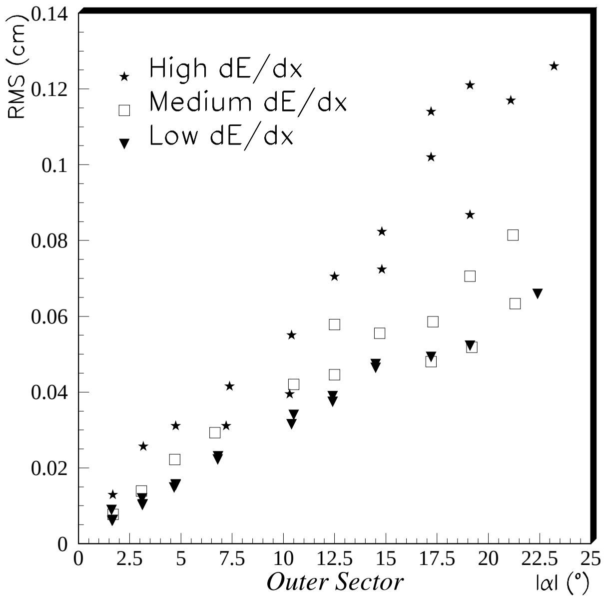

Figure 25: Observed resolution (RMS of residual distribution) for low-multiplicity events with zero electronics noise is shown as a function of the crossing angle alpha, with cuts on dE/dx of the hit.

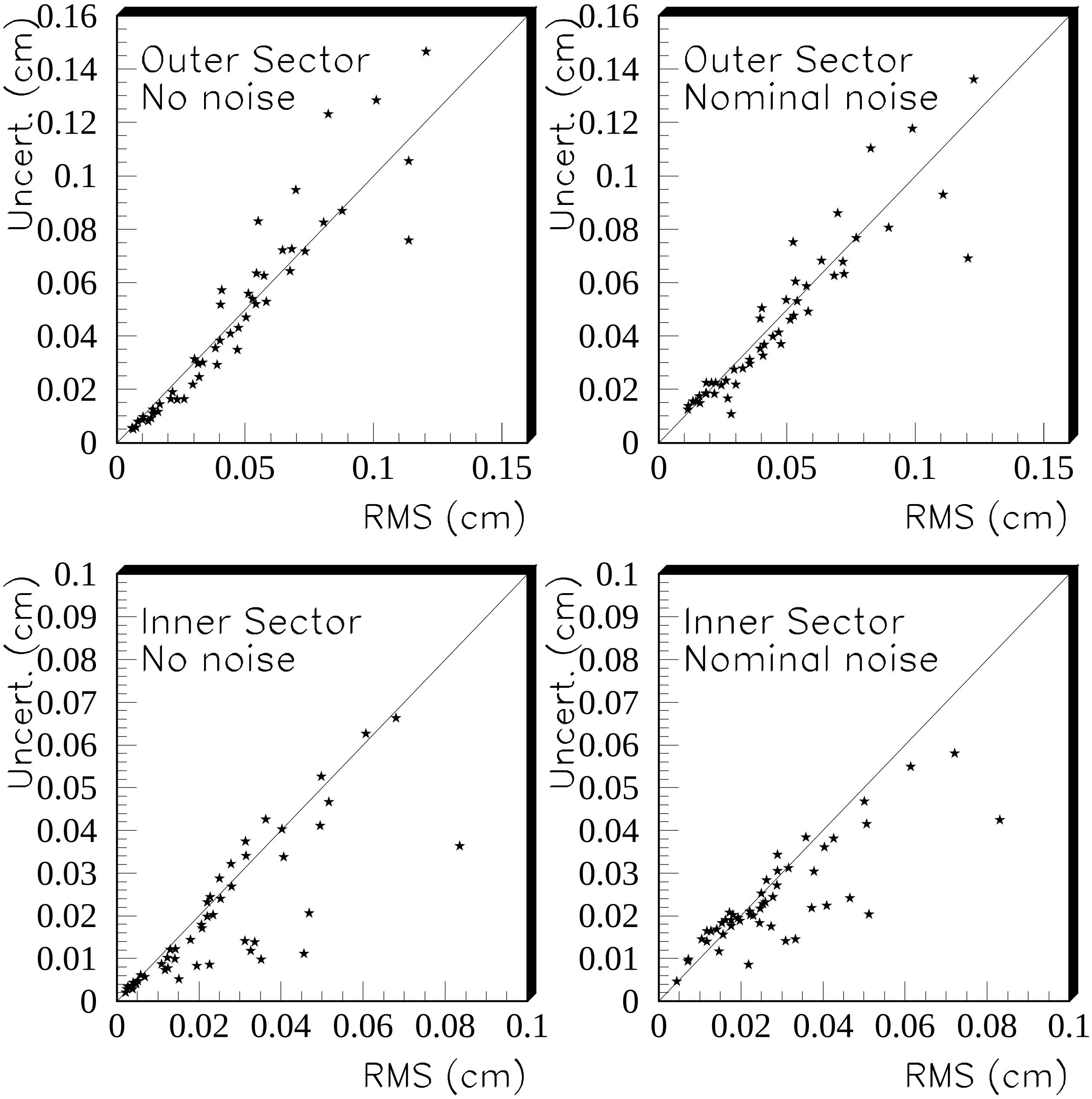

Figure 26: Estimated uncertainty in the position along the padrow that would be calculated by a tracker is compared to the position resolution for inner and outer sector, for two values of electronic noise. 45^o lines are drawn for reference.

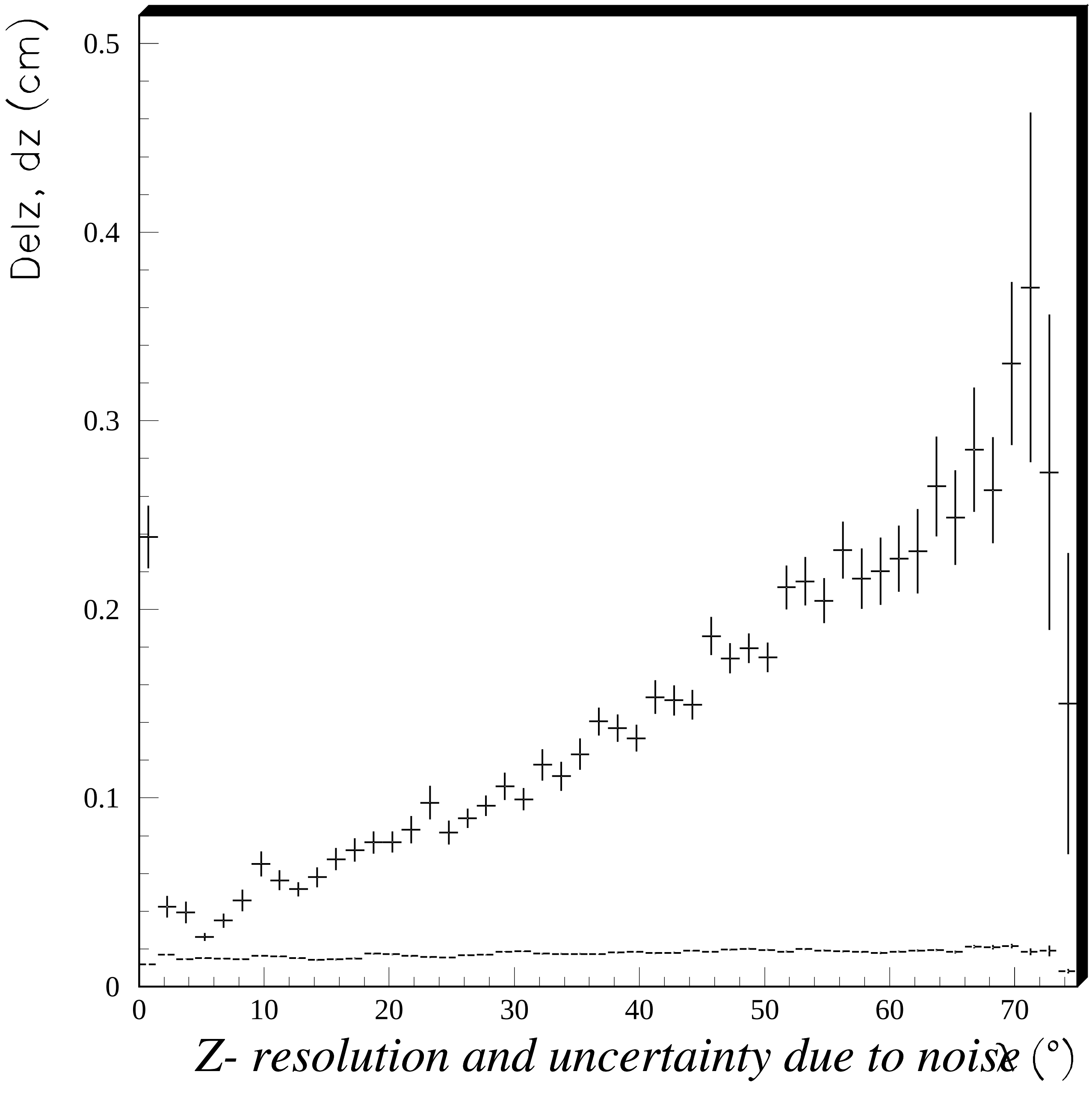

Figure 27: Average value of the residual (solid crosses) and estimated uncertainty calculated by the hifinder (dashed lines) as a function of crossing angle for isolated hits in a Au+Au event. For large λ, crossing angle effects dominate the effects of noise.

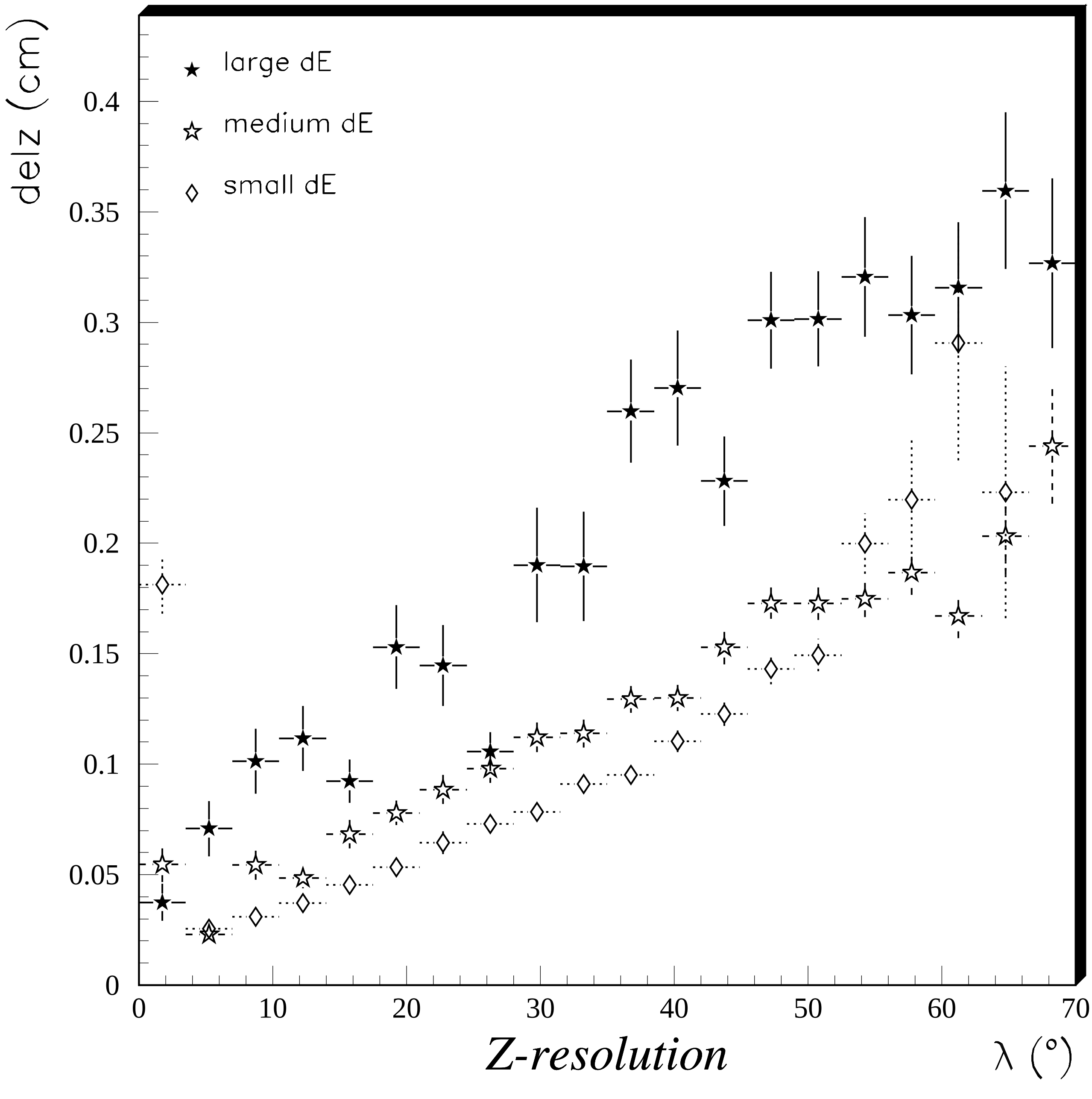

Figure 28: Average value of the residual as a function of crossing angle for isolated hits of varying amplitude in a Au+Au event. Large-amplitude hits show increased degradation in resolution with crossing angle.

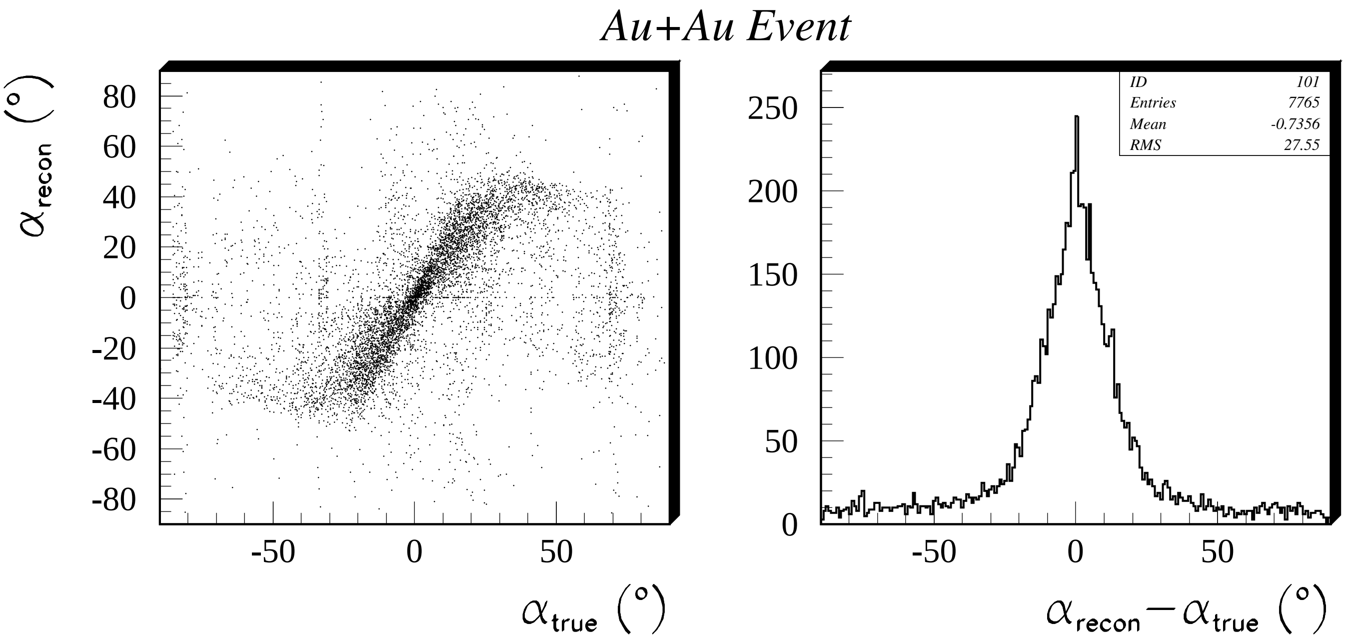

Figure 29: Crossing angle α as calculated by hitfinder based on cluster shape is compared with the true crossing angle generated.