STAR Level 3 Trigger Simulations

by
Eleanor G. Judd

1st December 1995

Introduction

The aim of the STAR collaboration is to investigate particle production in Gold-Gold interactions at RHIC. A central gold-gold interaction will produce 7 MBytes of data on average. With a simple centrality trigger we would expect to record one Petabyte (10^{15} bytes) of data per year. It will not be practicable to analyze this much data looking for rare processes, so more complex triggers will soon be required.

The basic STAR trigger detectors are the Central Trigger Barrel (CTB), the Multi-Wire Chamber (MWC) and the Vertex Position Detector (VPD). These can be used to trigger on gross topological features of an event, but they do not have the granularity to look for very fine detail or the ability to distinguish between different particle species. In order to trigger on these features it will be necessary to use the data from the Time Projection Chamber (TPC) and the Silicon Vertex Detector (SVT).

The TPC data will not become available until it has been read out of the front-end electronics and is being processed by the Data Acquisition System (DAQ).The DAQ i960 processors will analyze the data on a sector-by-sector basis to find clusters. The clusters from certain rows of a given sector will then be passed to a dedicated Level 3 processor which will use a pattern-recognition algorithm to reconstruct tracks from those clusters. These processors will also have access to the Time-of-Flight (TOF) and Electromagnetic Calorimeter (EMC) data corresponding to that sector. Hopefully, that information could be used to identify the tracks. The end results of all this analysis for all 24 sectors would then be passed to one final processor, which would combine all the results and make the trigger decision.

The aim of these simulations was to evaluate the performance of the i960 cluster finding algorithm. This included:

- evaluating the effects of various cuts in removing noise and large merged clusters
- calculating the resolution of the reconstructed clusters

It was also possible to pass these clusters to a pattern recognition program and then evaluate:

- track finding efficiency as a function of number of TPC rows used
- momentum resolution as a function of number of TPC rows used

From these studies is hoped we can start to understand what is and is not possible at Level 3.

GEANT

A sample of 7 central gold-gold events generated by Ron Longacre were used for this analysis. The raw event-generator output can be found on the RHIC cluster at BNL:

/star/trg/evz/rhic_plasma/rhiclnd6.evt
/star/trg/evz/rhic_plasma/rhiclnd7.evt
/star/trg/evz/rhic_plasma/rhiclnd8.evt
/star/trg/evz/rhic_plasma/rhiclnd9.evt
/star/trg/evz/rhic_plasma/rhiclnd10.evt
/star/trg/evz/rhic_plasma/rhiclnd11.evt
/star/trg/evz/rhic_plasma/rhiclnd12.evt

These events were all run through the STAR GEANT simulation. The TPC geometry file used was:

/afs/rhic/star/starlib/ref/sim/mcg/geo/tpc/tpc_tss_het_4mm.geo

This file defines a TPC filled with a Helium-Ethane gas mixture. The following GEANT flags were used:

SETS 'TPC' 'PADS'
RVTX 0
DIAM 0.
BWID 0.0
FIEL 5.
DCAY 1
MULS 1
PFIS 1
MUNU 1
LOSS 1
PHOT 1
COMP 1
PAIR 1
BREM 1
RAYL 1
DRAY 1
ANNI 1
HADR 3

These flags indicate that all the TPC data was written out, a constant magnetic field of 5.0 Tesla was used inside the magnet and all the GEANT physics processes were turned on. Setting the RVTX flag to 0 implies that the event vertex was always at (0.0,0.0,0.0). If it had been set to 1 then the vertex would have been chosen at random within a diamond whose RMS length was DIAM, and width was BWID. For future simulations RVTX should be 1, DIAM should be 1 cm (given by the VPD resolution) and BWID should be 5 mm.

The output from GEANT contains information about each track that crossed at least one TPC sub-volume. There is momentum, mass and vertex information for each track at its point of origin. There is also position and energy-loss information every time a track crosses a TPC sub-volume.

TPC Slow Simulator

The same seven events were then passed through the TPC Slow Simulator package (TSS). TSS converts the GEANT hits to ADC and TDC values on each pad.The charge deposited in each timebin is calculated from the GEANT hit information, taking into account longitudinal and transverse diffusion and gas-gain fluctuations. The signal in each timebin is then the sum of this charge plus the shaper-filtered white noise and the random SCA noise. If this is greater than a threshold value the charge is converted to an ADC value, which is saved. The algorithms used in TSS are described in “STAR Note 197 - The STAR-TPC Slow Simulator” by Wen Gong.

For these simulations the following values of the TSS electronics parameters were used:

Range of TDC values = 1 to 512
Range of ADC values = 0 to 1023
RMS number of electrons in shaper-filtered white noise = 800
RMS number of electrons in SCA random noise = 390
Number of electrons per ADC Count = 670
Threshold number of electrons for producing an ADC value = $4 \times 670 = 2680$

The output from TSS is saved in /star/trg/data/lund. Each file contains 2 TAS tables. One table contains a count of the number of sequences on a given pad, and a pointer to the beginning of that data in the other table. A **sequence** is a consecutive set of timebins on one pad that all have ADC values over threshold. The second table contains a 32-bit word for each timebin over threshold.

Bits 1 - 10 ADC
Bits 11 - 20 TDC
Bits 21 - 30 Number of timebins in this sequence
Bits 31 - 32 not used

Figure 1 shows an example of the TSS output for two rows on one TPC sector for a central Au-Au event. Row 1 is the innermost row and row 45 is the outermost. Each entry in the plots represents a timebin with an ADC value over threshold. Some differences can clearly be seen. The innermost row was crossed by many more tracks than the outermost, so the occupancy there is much higher. This is to be expected since only the high-pt tracks reach the outermost row, and they are a small fraction of the tracks crossing the innermost row. One consequence of the high occupancy is that on the innermost row some clusters have merged to form very large super-clusters. Finally, in both rows, a large number of very small clusters (one timebin) can be seen. These are a result of noise in the electronics.

On the STAR receiver boards the ASICs will extract pointers (TDC values) to the beginning and end of each sequence from the raw data. A full description of the ASIC functionality can be found in “STAR cluster-finder ASIC Functional specification - ver 4.2” by Mike Levine et al. These pointer lists will then be passed to the i960s where they will be used both to zero-suppress the data and to find 2-dimensional clusters. The simulated data is already in this zero-suppressed form, it contains only those sequences of timebins over threshold. This means that the majority of the ASIC functionality is simulated in the TSS package. Therefore a separate ASIC simulator was not written. The results in this report come from simulating the i960s response to the TSS output data.

Cluster Finding

TDH Module

A TAS module was created to run the i960 code developed by Michi Botlo. The module is called TDH and is available in the STAR CVS code repository for any STAR member to use. The aim of this code is to find groups of sequences on consecutive pads of a TPC pad-row that overlap in time. Any such group of sequences is known as a **cluster**. The centroid of a cluster measures the position where a particle crossed a TPC pad-row, the sum of all the ADC values is a measure of the energy deposited by that particle and the cluster width may indicate the error on the reconstructed position.

The input to the TDH module is the TSS output tables. The data is treated in blocks containing approximately 384 consecutive pads. These blocks are the data that would be passed to one i960 for processing. There are 18 i960s per sector, 6 covering the 13 inner rows and 12 covering the 32 outer rows. The allocation of pads to i960s in the outer sectors was taken from Figure 4. of the document “Front-End Electronics ⇔ DAQ Receiver Board, Fiber Optics Interface Specification Version 1.0 ” by Volker Lindenstruth et al. For the inner sectors the 3 innermost rows were given to one i960 and the remaining 5 i960s considered 2 rows each.

The TDH module then considered each block of data in turn and looked for clusters in that block. There was no communication allowed between the blocks, so some clusters were split into two pieces. There are a number of factors that affect the performance of the cluster finder. These include:

minimum sequence length
minimum number of sequences per cluster
maximum cluster size
treatment of clusters that span two i960s

The rest of this section will describe the algorithm used to find clusters and the optimization of the performance of the code.

Cluster Finding Algorithm

The i960 cluster finding algorithm is based on the concept of “linked lists”. The lists that are set up contain the data for all sequences on that i960, there is one element in the list for each sequence. Each element contains:

local pad number
encoded sector and row number

TDC value of start of sequence
TDC value of end of sequence
pointer to previous sequence in the list
pointer to next sequence in the list

Initially the pointers for each sequence link it to the surrounding sequences on the same pad. The sequence occupying the lowest timebins forms the head of the list, and the sequences are then linked together in order of increasing time. The cluster finding process involves re-arranging these pointers to link a sequence to overlapping sequences on the surrounding pads. The i960 loops over all the pads it is considering, working from left to right across each row when viewed from inside the TPC. For each pad it loops over all un-clustered sequences. The current sequence is removed from the pad list and set up as the head of a new cluster list. The code then loops over the remaining pads on this row. When a sequence is found that overlaps with the current one in time it is removed from its pad list and linked to the cluster list. This new sequence then becomes the current one. The process continues until a pad is reached with no sequences overlapping the current one, at which point the cluster terminates.

If the end of the row is reached before a cluster ends then this cluster is discarded. It is assumed that reconstructing only part of a cluster will give a misleading result for the centroid position. This may confuse the track-finding algorithm or distort any tracks reconstructed using these clusters, so it seems safer the reject them. It should be noted that clusters that start right at the beginning of a row, so part of them could be missing too, are kept. This does not seem to be consistent and should be further investigated.

Once a cluster has been found the next stage is to calculate the various moments which correspond to physical quantities. These are listed in Table 1.

Moment	Physical Interpretation	Equation
Zeroth	Energy Deposited	$\sum ADC$
First	Centroid Position	$\frac{\sum pad \times ADC}{\sum ADC}$ $\frac{\sum timebin \times ADC}{\sum ADC}$
Second	Cluster width	$\frac{\sum pad \times pad \times ADC}{\sum ADC}$ $\frac{\sum timebin \times timebin \times ADC}{\sum ADC}$

Table 1: Moments calculated for each cluster

Some results from running this code on the data in Figure 1 are shown in Figures 2 and 3. Figure 2 shows the sequences and cluster centroids found on row 1, the innermost row, Figure 3 shows the results for row 45, the outermost row. It can be seen that all the small noise sequences have been removed. The very large, merged clusters were found, but subsequently discarded, so their centroids have not been calculated. The minimum sequence and cluster sizes were tuned using this data.

Noise Reduction

The majority of very short sequences are produced as a result of electronic noise. This can be seen from Figure 4, which shows the distribution of sequence lengths for sequences produced from GEANT tracks and from electronic noise separately. Sequences that are just one timebin in length are dominated by noise. Two-timebin sequences include a large fraction of noise. The threshold number of timebins per sequence was therefore set to 2, and only sequences longer than this were used.

Width Measurement

The width of a cluster, in both the pad and timebin coordinate, is calculated from:

$$\sqrt{\frac{2ndMoment - 1stMoment^2}{\sum ADC}}$$

It gives a measure of how well defined the centroid is, which can then be related to the error on the reconstructed centroid position. If a cluster is made up of just one sequence the pad value is a constant, so the numerator, and therefore the width, reduces to zero. In this case the width gives no information about the shape of the cluster, leading to an undefined, large error on the centroid position. If the cluster covers just two pads the width still does not give much information. However, 3 pad clusters include pads on either side of the peak, so the width will be well defined and the resolution consequently can be much smaller than the pad width. The threshold number of sequences per cluster was therefore set to two, and only clusters larger than this were kept.

Resolution

Once the clusters have been found, and their moments calculated, the results can be compared to the original GEANT hits. Figure 5 shows the reconstructed x and y coordinates plotted against the generated values. The majority of hits lie close to the line on which the reconstructed coordinate equals the generated one. The vertical line consists of ghost clusters - one segment of clusters that were reconstructed in two separate pieces. The spread in the difference between the reconstructed and generated coordinates is another measure of the error in the reconstructed positions. Figure 6 shows the spread of these distributions as a function of row number. It can be seen that the width is greatest for the innermost rows where the occupancy is highest, and falls off as the row number increase, and occupancy decreases. One reason for this trend is the amount of merging between clusters from close pairs of tracks.

Figure 7 shows plots of the variation in the mean number of Monte Carlo particles that contribute to each cluster. It can be seen that this mean rises as the cluster size increases. This effect is most pronounced on the inner rows of the TPC, close to the primary vertex, before the particles have had time to separate.

If the clusters were Gaussian in shape then their width should represent the errors in the position of the centroid of the cluster. In this case a plot of (reconstructed position - generated position) / (error on position) should be a Gaussian distribution with a sigma of 1.0. These distributions, and Gaussian fits, are shown in Figure 8. While the shapes are roughly Gaussian, the sigmas are not 1.0, indicating that the clusters are not Gaussian in shape, so the second moment is not a very good measure of the error in the centroid position.

Track Finding

Once the clusters have been found a pattern recognition algorithm can be used to reconstruct the original tracks. THE STAR TPT package was used in this study. TPT uses hits from all sectors at once, which will not be possible at the 3rd trigger level. Each level 3 processor will consider just one sector, so tracks will be fragmented and the smaller fragments will have to be removed. TPT should therefore be considered as a “best case” scenario.

Each Level 3 processor will not even have access to a full sector’s data. There is so much data on the inner rows that the i960s would not be able to process all the data in their allotted time. It may be that in order to reconstruct useful tracks, in terms of momentum and position resolution, the trigger processors won’t even need all the outer rows. Three selections of rows were tested:

- 14 - 45 : All 32 rows on the outer sector
- 21 - 45 : The outer 25 rows
- 30 - 45 : The outermost 16 rows.

The reconstructed tracks were not required to originate at the primary vertex.

The results are shown in Figures 9 - 11. The efficiency for reconstructing a track is shown as a function of rapidity and transverse momentum. There is not much difference between the different row selections. The more rows that are used the better the efficiency becomes, but the effect is not large. In all cases there is a region of rapidity, from -1.0 to +1.0, and transverse momentum, above 0.3 GeV/c, where the efficiency is in the range 80-90%.

Large differences between the 3 cases show up however when the momentum resolution is considered. Using all 32 rows in the outer sectors gives a momentum resolution of less than 5% for the vast majority of tracks. Using only 25 rows produces resolutions up to around 8%, but using only 16 rows the resolution is almost 20%.

There are no firm requirements on the momentum and position resolution for tracks found by the Level 3 trigger processors. However, it is very likely that the next stage of the analysis would involve using either the ADC or time-of-flight measurements to identify tracks. It is also hoped to correlate the tracks with the EMC to obtain separate charge and electromagnetic energy depositions. The track resolution would have to be good enough to make this possible. It is unlikely that using the outer 16 rows only will be good enough, at least 25 rows will be required. The determination of the best selection will come from further simulations.

Conclusions

A preliminary study of on-line cluster-finding for the STAR third level trigger has been completed. A cluster finding algorithm has been implemented in TAS, and some initial tuning to remove noise and increase resolution has been done. It was found that the majority of the noise could be removed using a simple cut on the minimum number of timebins per sequence. Clusters consisting of very small number of sequences were also removed because they provide insufficient information to accurately determine their width.

Even with these cuts it was found that the cluster widths, calculated from the second moments, were not really a good measure of the error on the reconstructed cluster centroid. The errors calculated from the widths are too small, by a factor of almost three, when compared to differences between the reconstructed and generated coordinates.

The clusters were then used to reconstruct the original tracks, with the errors increased by this factor of three. Three separate row selections, all on the outer sectors of the TPC, were tested. The inner rows were not used because cluster finding could not be completed on those rows in the allotted time. It was found that the row selection had not much effect on the efficiency, but a very large effect on the momentum resolution. In order to make further use of these tracks the trigger processors will need to use at least the outer 25 rows, and the full outer sector of 32 rows would be preferable.

It would be useful to do another study in which a constant error was applied to all hits on a row. If this gives results that are as good as those obtained from the second moment then the second moment would not need to be calculated. This will reduce the processing time needed by each i960, and may make it possible to use some of the inner rows of the TPC.

The validity of these results depends to a large extent on the quality of the simulated data used in the study. The TSS package does a good simulation of charge diffusion and noise generation in the TPC. However, its simulation of the ASIC functionality has some missing pieces. There are a two features that are not simulated at all. One is the 10 to 8 bit compression. TSS still produces 10-bit ADC values as calculated by the Front-End Electronics (FEE). The ASIC is going to compress the FEE values to an 8-bit range in a non-linear fashion. The other feature is the double threshold system of the ASICs. The ASIC will only produce pointers to sequences where all the ADC values are over a LOW threshold, and some number of the ADC values (not necessarily on consecutive time-bins) are over a HIGH value.

There are a number of issues that still need to be solved. These include:

- is the conversion from row, pad, timebin to x, y, z done in the i960 or the Level 3 CPU
- do the i960s need to calculated the second moments
- can a sample Level 3 CPU complete track finding for one sector within the time budget
- is the track resolution good enough to allow identification of the track
- etc...

Hopefully these questions can be answered in the coming year.