YRHI-98-23

STAR NOTE: 372

Development of an Elastic Tracking Package

Yale Relativistic Heavy Ion Group

Brian Lasiuk, Dan Lyons, Gennadi Ososkov, Thomas Ullrich

Department of Physics
Yale University

Brian.Lasiuk@Yale.edu

Abstract

A prototype package which implements an Elastic Tracking (ET) algorithm is being developed in a fully Object-Oriented framework using C++ as the programming language. This package has capabilities to work on both pre-processed reconstructed hits, or on raw charge distributions that are measured by a detector. It is also being developed with the intent of being able to handle multiple detectors that should remove the need for inter-detector matching of track segments. Although benchmarks on simulated data confirm that the ET method does incur a significant time penalty compared to that of conventional road finders, the time required is strongly dependent on the spatial resolution of the points. The track-finding efficiency remains remarkably high and constant nearly independent of event multiplicity and hit density. The behavior of the ET package will be discussed along with its structure and performance characteristics. Identifiable bottlenecks will be highlighted as well as future plans.

Contributed talk at CHEP'98, Chicago Illinois, 31 August - 4 September, 1998.

1 Introduction

High energy physics experiments confront increasingly complex final states as interactions of increasing center of mass energy at high luminosities are measured. The corresponding escalation in the multiplicity of the measured final state is beginning to stress conventional pattern recognition techniques and different, more powerful methods are required to identify and reconstruct particle trajectories in such environments. This is especially true of ultra-relativistic heavy ion collisions where total multiplicities of central Au-Au at $\sqrt{s}$ =200 AGeV produced at the Relativistic Heavy Ion Collider (RHIC) are estimated to be between 5,000 and 10,000 in the central rapidity region. Furthermore to ensure proper vertexing and reconstruction of short lived particles, detectors will be placed as near as several centimeters to the interaction region. In such an environment, high occupancy and hit density, even with the smallest granularity detectors, provides situations which are at the limit of the capabilities of conventional road finder type trackers. In order to cope with such harsh conditions, alternative tracking methods which have previously been deemed to costly (in time) or unnecessarily complicated are being studied. In particular extensions to the efficient template tracking method such as adaptive template and Elastic Tracking (ET) are being investigated by many experiments because of the extraordinary claims of disentangling closely separated tracks and capability to perform in a densely populated environment with high reconstruction efficiency.

2 Elastic Tracking Developments

The tracking problem in High-Energy Physics (HEP) is to determine the parameters of a particle trajectory from a set of positional measurements determined from a detector, sensitive to ionization produced by a charged particle. The curve is defined by the equations of motion of a charged particle in a magnetic field which can be solved either analytically or numerically.

At low hit and track densities a road finder which locally extrapolates track segments between adjacent active detector layers is quite sufficient, both in speed and efficiency. However when the hit density becomes high, the algorithm can get confused on the basis of simple χ^2 criteria and this can adversely affect track finding efficiency. In high hit density environments, tracks can be split or spliced together in unpredictable ways. Many times the ambiguities at track crossings that arise with local extrapolation algorithms can be overcome with global information of the complete track trajectory. This information can be obtained through the application of various transformation algorithms such as the Hough or Radon transformation, or more simply through the use of templates. Such techniques are more in line with the manner in which humans recognize patterns, rather than local extrapolations. Although the efficiency of these methods is higher, there is a penalty incurred in terms of the amount of time such algorithms require, and few experiments up until this point have needed to turn to such sophisticated methods.

Elastic tracking is a relatively new technique which draws on developments from both neural networks and Hough/Radon transformations. It was proposed in two variants in the early 1990s by two different groups—Harlander and Gyulassy [1] and Ohlsson and Peterson [2]. Although the underlying principles are the same, the differences in implementation and/or philosophies between the two are significant. These differences include the level of parallelism in processing, optimization, and environments that are more suited to each technique. Although these two groups have pioneered the methodology, others have also contributed to developments [3] but it is interesting that advances have dominantly been made by theorists—perhaps indicative of the complex theory that is the basis for the algorithm. There are several implementations for experiments (i.e. Delphi [4] and ATLAS [5]) however they remain outside the main stream of software development within the collaborations.

For our purposes ET can be considered a development of template based track finding methods where a trajectory (i.e. a template in our nomenclature) constrained by the equations of motion is overlaid on the data from the detector, and a set of conditions is applied to determine whether a match exists. This was the basic approach of the NA49 experiment where a set of templates was generated prior to event reconstruction according to the available phase space [6]. The reconstruction phase then consisted of looping over all templates within the set, overlaying each on the raw data. A routine then evaluated whether or not the template matched a track. Although extremely efficient, this approach has the disadvantage that resolution of track parameters is set by number and granularity of the template set generated. Furthermore, this method requires an *a priori* knowledge of the interaction vertex. Although this may not be a difficult parameter to determine by independent means in fixed target experiments, it is not trivial to ascertain in collider environments where the interaction vertex can shift by significant distances—up to 20 cm. Static templates also make it nearly impossible to design efficient methods with which to detect non-vertex tracks, e.g. decay products from long lived neutral particles like Λ^o and K^o. The ET algorithms offer a significant advancement on all these fronts.

First, ET goes beyond a simple binary result for the match evaluation and allows the templates to deform via a well defined set of mathematical equations to match a pattern in the data. This dynamical adaptive fitting approach is much more powerful than neural net algorithms because of the constraints of the equation of motion that govern the template evolution. It also allows developments based on completely parallel (i.e. generation of all templates simultaneously) or sequential (i.e. generation of single templates) algorithms based on the specific application needs and available computing resources.

Although template based track-finding algorithms are typically very expensive both in processing time and memory requirements, it is highly-efficient by its nature and offers the possibility that tracking can occur at the raw data level with no pre-processing to reconstruct individual space points in the detector. Furthermore the post-processing or track re-fitting usually necessary to determine the momentum components becomes redundant because this is done implicitly in the pattern recognition phase. Although it is possible to work with raw data, let us consider the implementation working on pixel data—that is reconstructed space point "hits" in the detector. The procedure and methods can be generalized to raw data once the general algorithm is functional. This will be explained in section 6.

The theory of ET is described in detail elsewhere [1, 2] and we use the prescription of Harlander as a guideline. Mathematically the approach can be modelled on calculating the interaction energy between two charge densities; the template which is attributed a positive charge $\rho_T(r)$ and the charge as measured by the detector which has a negative charge $\rho(r')$. Thus the interaction energy, E, between the two distributions can be calculated as an integral over all points in the detector volume:

$$E = -\int\int dr'\, dr\, \rho_T(r)\, V(r - r')\, \rho(r') \tag{1}$$

where V is a potential that depends on the distance between the template and the point i, as measured in the detector.[1] The energy is minimal when the template exactly overlays the charge distribution (space points) as measured by the detector. Note that because the detector measures discrete space points, $\rho(r')$ can be written as a delta function, and the integral becomes a sum over a set of N space points:

$$E = -\frac{1}{N}\sum_{i}^{N} \Delta r\, \rho_T(r)\, V(r - r')\, \delta(r') \tag{2}$$

Because the energy is determined by the separation distance of the track and template, the energy is implicitly a function of the track parameters and thus may be minimized with respect to these parameters. The strategy of track finding can thus be factorized into two distinct components. First a seed or template must be generated which serves as an initial guess for a track. Second a quantity that is a function of the fit/match quality, as shown in equation 2 is calculated. On the basis of this quantity it is either determined that: (i) the fit can be made better and the track parameters are altered and the processes becomes iterative; (ii) the track is determined to be satisfactory and marked as “found”; (iii) the pattern is deemed not to match any track and is discarded. This process then continues until a criteria is satisfied that no tracks remain.

3 Design and Requirements

The ET package was defined with several requirements clearly in mind. First, a higher track-finding efficiency with a smaller fraction of split or fragmented tracks had to be obtained than with a conventional road finder. If such performance was not foreseen (or achieved), there would be no motivation for adopting such a complex algorithm with the associated increase in processing time. It also had to be able to find tracks independent of the magnitude of the magnetic field. While a non-zero finite value for the magnetic field ensures a helical trajectory, a field free region reduces the trajectory to a straight line. It is highly desirable to retain straight line tracking capabilities without the parallel development of a second package. As such the track parameterization should allow for a collapse of the helix parameterization to a straight line in the absence of a magnetic field in a manner transparent to the user. The code also must be able to handle information from multiple detectors which removes the need for any additional track-segment matching algorithm and it is also necessary for the algorithm to function without any assumption of an event vertex position so that a separate algorithm need not be developed to find non-vertex tracks. Finally it must be able to work on the raw ADC data without preprocessing.

In order to decouple the many different components of the code from one another, the package was factorized into several different categories which closely followed the methodology of the GEANT4 collaboration [7]. This resulted in a much more modular code that

[1] This can also be made to be a function of the pixel amplitude which is important for the implementation on raw data.

allowed greater flexibility in implementing and evaluating different algorithms. As such, the package is divided into 9 separate categories, of which three were not specific to the tracking code. These included global definitions and macros as well as utility classes like three-vectors, four-vectors, and matrices[2] and visualization classes based on OpenGL. Other categories described the geometry of the tracking detectors, as well as defining the interface and management tools necessary to coordinate activity between the classes and allow for variable I/O formats.

The classes specific to tracking centered around the definitions (and containers) for the hits and tracks. The hits category contains not only the pixel structure, but also the container in which the pixels are stored and its own memory management scheme. This is shown schematically in figure 1. In order to meet the conflicting needs of efficient memory

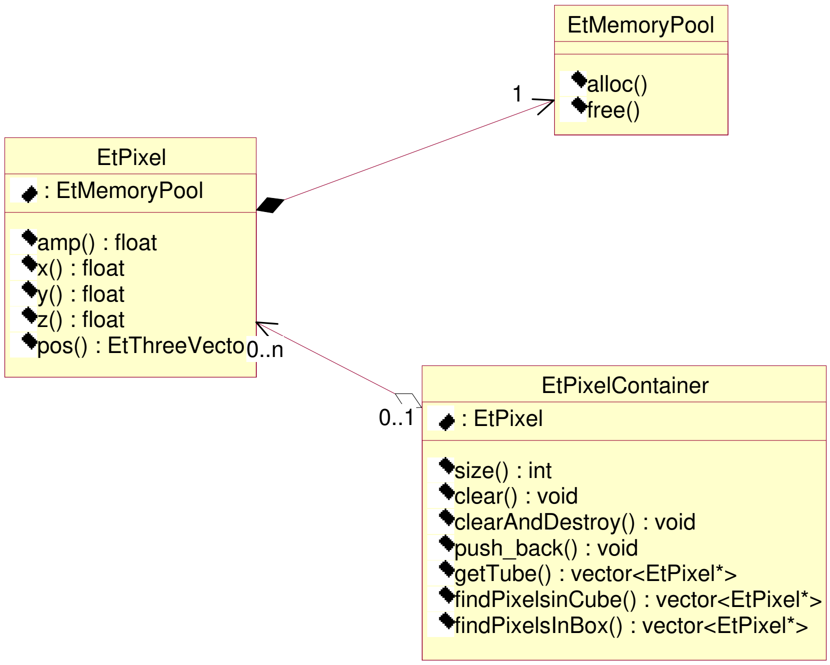

Figure 1: The `Hits` category contains a definition of a pixel, which contains a position and amplitude (either analog or digital). Pixels are contained in a smart container that allows insertion, sorting, and selective access to the pixels. It also manages its own memory allocation.

utilization and fast access, an efficient pixel container (`EtPixelContainer`) was developed. Details of its implementation are described in section 4.

The "`Tracks`" category defines the necessary classes which implement the specific track model. This is shown in figure 2. The basic track model is a parameterization used by Bock [9] which defines a helix in a set of three parametric equations in a purely mathematical

[2] variants and versions from CLHEP and NA45 COOL library were evaluated. It should be noted that this package preceded the development of the STAR Class Library [8].

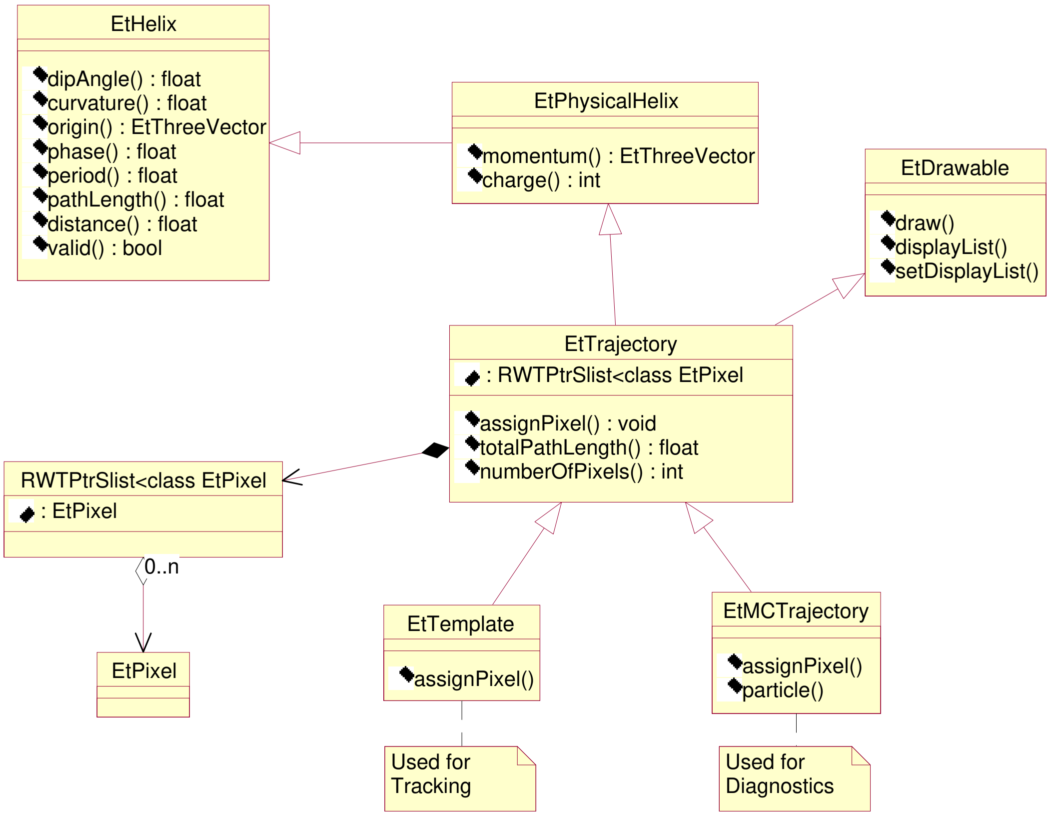

Figure 2: The "Tracks" category which shows the relation from the most abstract to the most specific definition of a helix. Also contained in the category are generalizations of the track which includes a trajectory. The trajectory may have pixels associated with it as well as attributes that make it drawable in the visualization package. Diagnostic information like original Monte Carlo information is contained in further generalizations of the trajectory.

sense—i.e. in terms of a curvature, two angles (dip and phase) and an origin. For tracking (and physics analysis) purposes, it is more useful to have a track defined by more physically meaningful parameters like momentum and particle charge. As such, a `EtPhysicalHelix` is derived from the mathematical helix. From the definition of the helix, it is possible to define an `EtTrajectory` which are the objects which the tracking code manipulates. As such it is possible to associate `EtPixels` with the `EtTrajectory` as well as having the capability to draw it, by inheriting from `EtDrawable` as shown in figure 2. It can also be generalized to an `EtMCTrajectory` such that information from generators (such as momentum and particle type) can be stored. Such information is essential for evaluating efficiency and reconstruction resolution of algorithms. The heart of the package is the actual tracking algorithm which include the minimization algorithm(s), template seed generator(s), match evaluator(s), etc. These are described in more detail in section 4.

4 Implementation

The power of template based track-finding algorithms is its ability to utilize global information throughout the detector. While this provides much more information for decisions to be made, it also imposes managerial difficulties in that much information must be made accessible in all stages of the program. For tracking applications this means access to the pixels in a fast and efficient manner is crucial. In our case where millions of pixels have to be stored in a highly segmented three-dimensional space (70×10^6 pixels) with fractional occupancy, a three-dimensional array is not feasible. On the other hand a compact one-dimensional list of all pixels, although memory efficient, cannot be employed because the time for pixel lookup would become prohibitively long. As such it was deemed necessary to develop a smart "Pixel Container" for the space points.

4.1 The Pixel Container

The pixels are stored as an array of pointers and the pixel container only manipulates these pointers. The container distributes the total number of pixels, N, into N equal sized boxes. For a purely random distribution of points, one pixel would be expected per box. These boxes are represented by an element in a three dimensional array. The array elements are then pointers to the list of pixels that are contained in the box. This requires little memory overhead as seen in the left panel of figure 3. Retrieving pixels within a specific

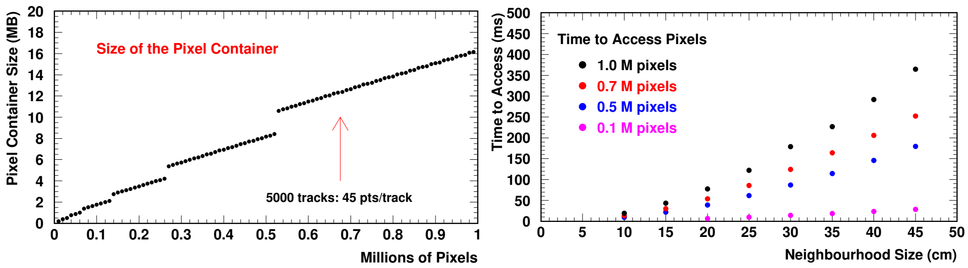

Figure 3: Measured performance of the pixel container. The left panel shows the memory used by the pixel container, not the pixels themselves. The right panel illustrates the average time to access pixels in a given volume as a function of the number of pixels in the container.

volume is thus reduced to searching for boxes within a specified volume. There is an implied assumption that not every pixel should be tested explicitly because this is a computationally expensive process. For boxes that lie completely within the search volume, all pixels within the box are returned. Only for those boxes where only a portion of the volume lie within the search volume is each pixel is tested individually to determine whether it lies within the search volume. The appropriate pixels are then returned in an STL container. The performance of pixel retrieval as a function of the search volume and the pixel container size is shown in the right panel of figure 3.

The implementation of the ET algorithm requires a further specialization and that is to return all pixels within a specified distance from the template candidate. This volume is defined as a tube through the volume of the detector containing the helical template or track.

Again, because calculating the distance of closest approach from a point to a curve (and in our case, a helix) is computationally expensive, each point cannot be tested individually. Therefore a first approximation is to determine which set of boxes lies within the tube radius of the track template, and a rather complicated algorithm is utilized for this purpose. Taking two consecutive points on the template candidate, on adjacent active detector layers, for example, it is possible to define a box centered at the mid-point of a line which adjoins these points. All possible curves between these two points must lie within an ellipsoid with a minor radius the order of the point separation distance. This ellipsoid can then be used to define a cube which in turn can be used to determine which pixel container boxes satisfy the search criteria. Moving to the next point on the curve and repeating the procedure produces a list of boxes of whose contents are the appropriate set of pixels. As before, those boxes that are not completely contained within the specified tube volume must have the points within them checked individually. There is significant freedom for optimization of this algorithm for specific purposes such as the point separations, ellipsoid radii, etc. For ET purposes, the tube radii is maximally 20-40 mm and the order of 100k tube retrievals are expected to be performed on a "normal" event. The current implementation is able to retrieve pixel sets from 5k tubes, generated from Monte Carlo track parameters in 7.7 s. This is a negligible expense when compared to the total CPU time of the ET program.

4.2 The Elastic Tracking Algorithm

The first need of the ET algorithm is to generate a seed for the initial guess. Several methods have been used in the past from generating a set of templates given the constraints of available phase space, to completely random guesses. In our implementation, an analytic local Hough transformation is used as the basis for an initial guess. A pixel is selected from the pixel container and all nearest neighbors within a specified radius (usually $\sim$5 cm) are returned. Using unique combinations of all sub-sets of 3 pixels, helix parameters are calculated analytically since a set of three pixels is sufficient to determine the parameters for a helix—curvature (κ), dip angle (λ) and phase (Φ). A schematic of this calculation is seen in figure 4. Although three points are sufficient to define a helix, the momentum resolution (or curvature resolution) is known to scale with the inverse square root of the number of points (and the inverse of the length squared) because of the measurement error on the position of each pixel. As such three points over a $\sim$5 cm region does not provide a high-quality guess. This is hardly surprising otherwise there would be little motivation for constructing multi-layer detectors. As such, in order to refine the parameters from the local Hough transformation, the curve is extrapolated to the next adjacent pad-rows, much in the spirit of a road-follower, to include six points.[3] A projection of the points is then made into the bend-plane and the initial curvature parameter is recalculated via a fast-fit Ososkov-Chernov algorithm [10]. This improves the quality of the track parameters to a point where the minimization routine is responsible for determining the appropriate points on the track and fine tuning the parameters from the initial guess, as seen in figure 5. It should be noted that the curvature is most sensitive to finite resolution effects and the local extrapolation improves the quality of the initial guess by a factor of 2!

Next the track interaction energy is calculated. In principle one should need only to

[3] The number of points is a tunable parameter which was optimized to 6 points for our application.

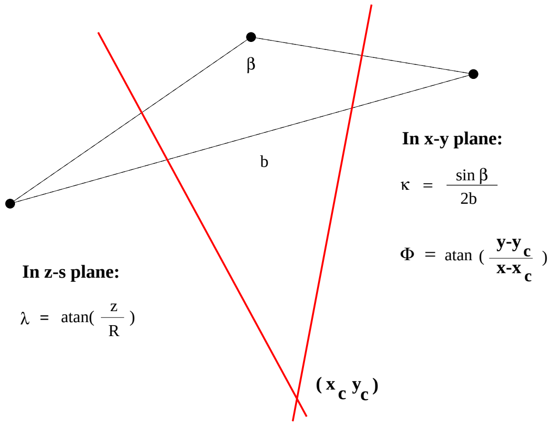

Figure 4: An analytic **local** Hough transformation on three points is sufficient to determine the parameters of a helix.

utilize space points that are within the space point resolution of the template. However, this assumes that the template is already perfect which is not the case. The initial calculation should use points according to the resolution (or quality) of the initial *template*, a crude measure which is shown in figure 5. As the parameters are refined according to the evolution of the fitting algorithm, the volume of the region where points are selected for the fit can be reduced. This variation in the set of points used in the track fitting is a non-linear adaptive fitting procedure which gives this method its power for several reasons. First, by utilizing an extended region around the template to select space-points, the probability that the minimization routine gets trapped in a local minimum is greatly reduced. Second by using a fixed extended region (i.e. the complete detector volume), a substantial increase in the requirement of the number of calculations would render such an algorithm unusable. Therefore, in the initial iterations a large search volume is used and is gradually reduced as the track parameters begin to converge.

Given a subset of the total number of points in the detector, the interaction energy based on equation 2 can be calculated once an appropriate potential is chosen. The incorporation of the adaptive fitting can be modelled by choosing a time or iteration dependent width for the potential. Although any potential of the form $V(|r - r'|)$ can be used, we use a Lorentz potential because of its simple form and well defined behavior as the width approaches zero. Thus equation 2 becomes, in our model:

$$E(\pi, t) = -\frac{1}{N}\sum_{i}^{N} \frac{w^2(t)}{\left(\vec{x_i} - \vec{r}(\pi, \vec{x})\right)^2 + w^2(t)}\ , \qquad \pi = \{\kappa, \lambda, \Phi, \vec{x_o}\} \tag{3}$$

where N is the total number of points to be used, r is the distance from the space point i to the template, and π is the set of track parameters. In order to vary the width of the potential, $w(t)$ is the iteration dependent width of the potential and is taken as:

$$w(t) = b + (a - b)\exp(-t/T) \tag{4}$$

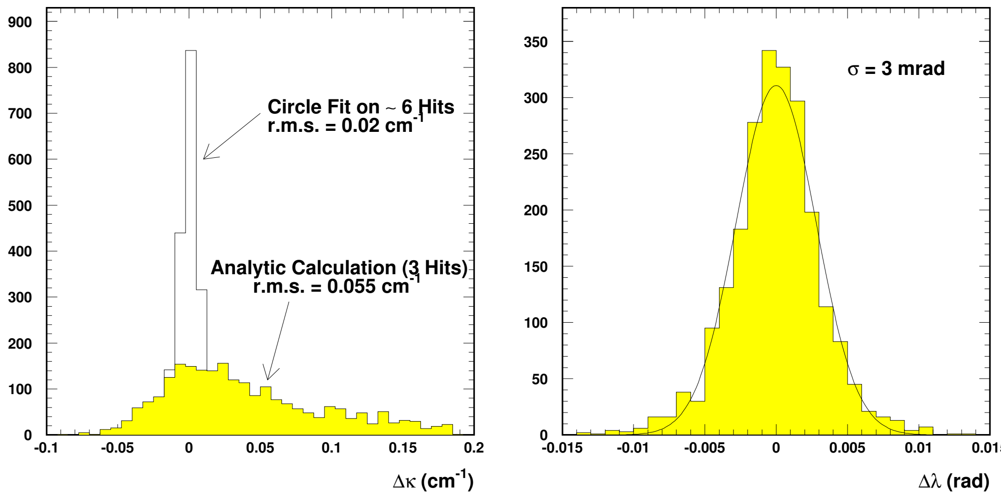

Figure 5: Resolution of the track parameters from the seeds compared to that of the real tracks. The left panel shows the difference between the curvature determined from a 3 and 6 point seed to that of the actual track parameters. The right panel shows the resolution of the dip angle. The points are smeared in the r-ϕ direction by 500 μm and in the z direction by 500 μm to simulate the finite space point resolution of a detector.

where T is a time constant, a is a long range potential (i.e. maximal search radius), and $b \approx \sigma_{\rm res}$, the space point resolution of the detector where $a \gg b$. This gives elasticity to the process and varies the weight assigned to the points as a function of the distance from the template. It should be noted in the limit of an infinite time constant and for $|x - r| \ll a$, the interaction energy can be expanded in a Taylor series to yield:

$$E \approx -\frac{1}{N}\sum_{i}^{N} 1 - \left(\frac{x_i}{a}\right)^2 + \cdots \tag{5}$$

which is very similar to a χ^2 fit to first order and the limit of the conventional road-finder criteria.

The calculation of the interaction energy is dependent on the distance between the template and the space points. The measure of $(\vec{x_i} - \vec{r}(\pi, \vec{x}))$ is the distance of closest approach between the template and the space point in **three** dimensions. This is quite a costly operation but it is necessary. Although it is advocated in previous work [2] that one may project the helix onto a plane and calculate the Euclidean distance between the point and the template in this plane, adding the out-of-plane distance in quadrature, it is not mathematically equivalent to calculating the distance in three dimensions. This simplification as proposed in [2] results in a significant error, especially when the tracks have a large (steep) dip angle and degrades tracking performance significantly in this region of phase space. Once the interaction energy has been calculated, the 5 track parameters are changed according to the gradient of the energy function using a modification of the gradient descent method:

$$\Delta\pi_i(t) = -\eta\nabla_\pi E(\pi, t) + \alpha\Delta\pi_i(t-1) \tag{6}$$

The gradient of the energy function (eq. 3) gives the sign of the change and a "learning rate" (η) scales the numerical value of the gradient to give the actual magnitude of the change to the i^{th} parameter. A second inertia term, proportional to the shift in the parameters in the previous iteration is also added so as to damp the wild oscillations expected as one approaches the minimum. Although this method is slow and faster techniques are perhaps available, it is an extremely safe and robust algorithm that assures convergence.

The final task is to determine when the iteration has converged. Simple tests of magnitude of the interaction energy is not sufficient because of the continuously changing magnitude of the potential width. Complicated measures using the first and second derivatives of the energy function are useful but tend to be dependent on seemingly arbitrary numerical parameters which can be tuned *ad nauseam* with seemingly little physical or statistical meaning. Initially such a criteria was defined along with the history of magnitude of the previous five iterations of the parameters. If the change of the parameters were less than resolution of the detector, convergence was assumed. Currently a much more rigorous definition is utilized using the definition of confidence levels. Every 20^{th} iteration, a χ^2 value (threshold) is calculated from the confidence level and the number of pixels currently *on* the template. A second χ^2 is also calculated from the template pixels and parameters. If this is less than the threshold, the process is terminated, otherwise the process is iterated until either the threshold is eventually achieved or the maximum number of iterations are exceeded. Currently the are no more than 200 iterations allowed for a track candidate.

5 Results

As mentioned several times, the template method is extremely efficient, as shown in figure 6. The efficiency is shown as a function of the number of tracks in a volume of two units of pseudo-rapidity and as a function of the minimum number of points on the track. The tracks are generated from a realistic rapidity and transverse momentum spectrum. A fraction of noise hits, not associated with any track are also added. The finite spatial resolution of the detector is also simulated by smearing the pixel position in the r-ϕ plane with a Gaussian distribution with a width of σ=500-700 μm, and in the z (beam) direction with a Gaussian with a width of σ=500-700 μm. The conditions are taken to closely model the STAR detector at RHIC [11]. The efficiency is defined as the total number of tracks with at least 90% of the correct points associated with it, over the total number of tracks generated. For fast turn-around times, a fraction of the total acceptance in azimuth (i.e. $\Delta\phi = \frac{\pi}{6}$) was evaluated and scaled to the total acceptance. Although this tracking method does not appear to break down at multiplicities even 3–5$\times$ that of the highest estimates from realistic model predictions at RHIC energies, it should be noted that such track densities would correspond to an average detector pixel occupancy of $\sim$50% (in the STAR TPC), and it is certain simple cluster/space-point finding algorithms would break down in this environment. In this study it is assumed all space points are resolved. Such an environment would require track finding to work on the raw data with no intermediate step for space point reconstruction. This is described briefly in section 6. Still, even at these large track densities,

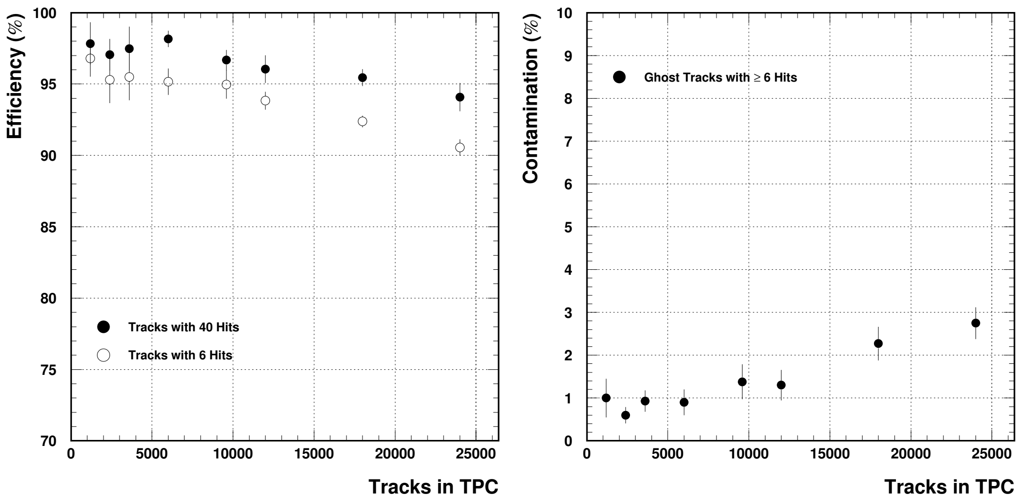

Figure 6: Efficiency and contamination achieved with the ET package. The left panel shows the efficiency for track finding as a function of total multiplicity in the region -2< η <2. The right panel shows the fraction of split or ghost tracks in the same region.

only a slight degradation in performance is seen as a function of total multiplicity—implying that the algorithm is extremely robust. This contrasts to a degradation of track finding efficiency of up to 30% in the high track density environment of the NA49 experiment [12] with a conventional road-finder. In addition to the high efficiency, the contamination of split and misidentified tracks remains remarkably low. This is shown in the right panel of figure 6. This was expected because of the use of global information that removes many local ambiguities where road followers are easily confused and it is very difficult to fragment a track on a global level.

Such plots are encouraging in showing the robustness and high quality of the tracking performance, however the price that is paid is the increase in the amount of time necessary to reconstruct a single event. This is shown in figure 7 where a strong, non-linear growth in time can be attributed to the increase in both the number of track candidates and the number of pixels. It should also be noted that the number of iterations necessary for convergence of track parameters increases dramatically with a degradation in space point resolution and for high momentum (i.e. straight tracks)—the curvature being a parameter especially sensitive to this. The evolution of the energy function for the curvature parameter of a single track as a function of the number of iterations is shown in figure 8. Each iteration requires calculation of the distance from the point to the template which is computationally intensive. Furthermore the gradient of the energy function must also be calculated. It is also noteworthy that a major fraction of the number of iterations are spent in fine tuning the track parameters. This is discussed in more detail in section 6.

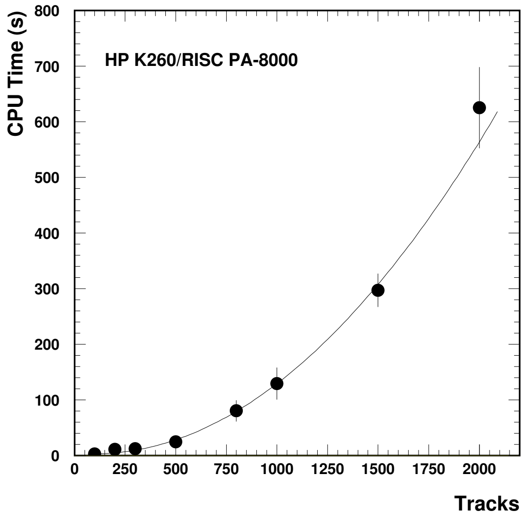

Figure 7: Average CPU time necessary to reconstruct a fraction of a single event as a function of the multiplicity within a $\frac{\pi}{6}$ slice in ϕ. The curve shown in quadratic in time.

6 Future Directions

Figure 8 shows an interesting trend in the evolution of the energy function and that is nearly 50% of the iterations are spent on the final 2% fine tuning of the track parameter, and this indicates a simple way to decrease the processing time. Near iteration #20, it seems obvious that the track has been found. As such the algorithm has succeeded in the first task of pattern recognition, and the remainder of the time is optimizing the track parameters. If at this point, the list of space points was passed to an optimized fitting routine to determine the track parameters, the processing time would be reduced dramatically. Although it was argued earlier that the use of template makes such a "post-processing" momentum fit step redundant, the large computational overhead of computing the energy function and gradient with respect to all the track parameters is unnecessarily spent after a pixel set that comprises a track has been identified. This is currently being investigated.

The package is also flexible enough to deal with multiple pixel containers. As such it is a simple extension that an inner tracking detector can be added. The application of the ET algorithm in this scenario should remove the necessity for stand-alone tracking codes to run in each detector, and then have the track segments linked via a track-matching algorithm.

The next question to address is the reconstruction of tracks in a magnetic field, in three dimensions on raw data—that is, without the intermediate step of space point reconstruction. Such code has never been developed previously, however it may be necessary if pessimistic estimates of >30-40% detector occupancy arises in future detectors. The step necessary to generalize the ET algorithm to work with raw data is to introduce an amplitude dependence into the potential of the energy function. Thus, small charge (ADC) clusters should be repelled away from the centroid of the track while large charge (ADC) clusters

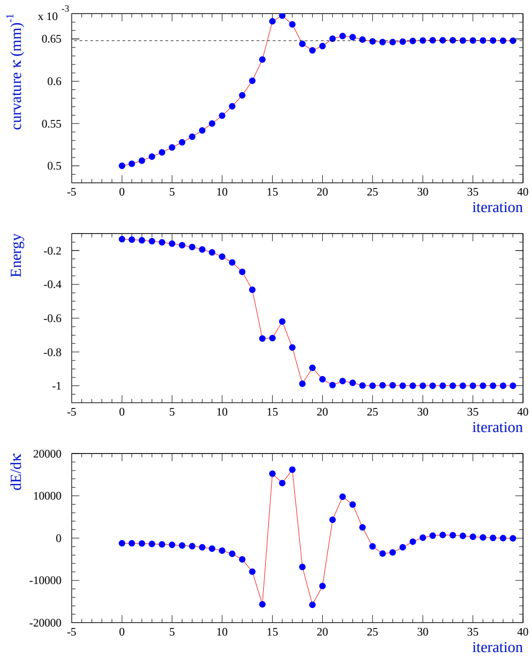

Figure 8: Evolution of the energy function and eventual convergence of the curvature parameter as a function of the number of iterations. The space-point resolution for this track was taken to be 100μm, however the number of iterations increases dramatically with decreasing position resolution.

should be attracted to the centroid of the template. Such a potential has been developed by Ososkov and Chernov in the context of this project. Shown in figure 9 is a “bi-horn” potential which fits this description. At high multiplicities it is also possible to use the Landau-fluctuations in the energy loss spectrum of a charged particle to our advantage. For example, if a large threshold on the amplitude of the raw pixels is set (i.e. 5$\times$ minimum ionizing), only the pixels with the highest energy loss should still be visible. This corresponds to approximately the highest 20% of points in the Landau distribution, which means approximately 20% of all pixels on tracks. These high amplitude pixels should also occur near the centroid of the track trajectory and can be used to generate seeds. As such, this threshold will have the effect of reducing the initial “effective occupancy” of the detector and allow the algorithm to function. Once the seeds are generated, the threshold of the pixels that are used can be gradually decreased such that more pixels are gradually incorporated into the tracking finding routines. Although currently this would be a painfully slow process, should detector occupancies reach levels where space point reconstruction is no longer possible, this type of algorithm may be the only possibility for event reconstruction.

A simple algorithm which utilizes the “bi-horn” potential has been developed in the context of this project. The main problem is the time requirements associated with such processing. Using reconstructed space points, approximately 40-70 distances (from space points to a template) must be calculated in order to quantify the interaction energy as

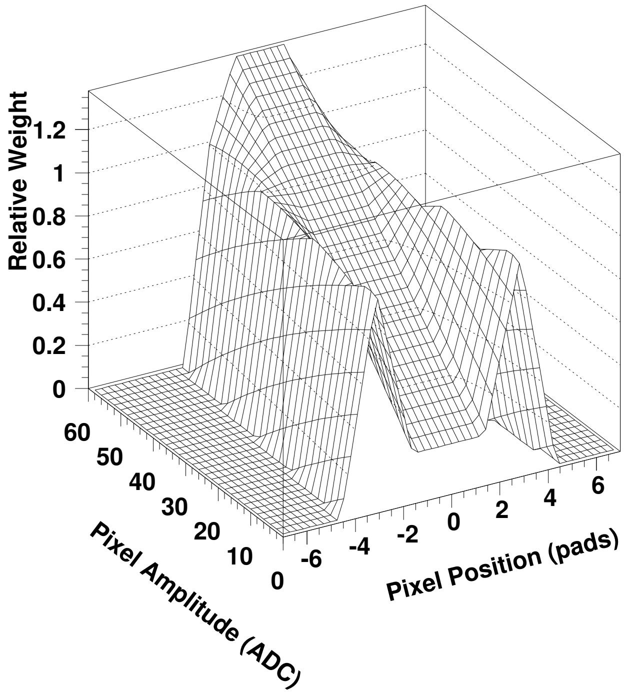

Figure 9: Ososkov-Chernov "bi-horn" potential which gives higher weight to high amplitude pixels near the centroid of a trajectory while repelling low amplitude pixels to greater distances. The potential is infinitely tunable in that the range and magnitude of the minima and maxima can be set independently.

given by equation 3. For a TPC-like detector, a space point is typically composed of 30-40 individual "time-buckets" or detector pixels. As such the number of calculations grows by this factor (i.e. $30\times$) which makes the current implementation very time consuming. As such the development in reconstructing tracks from raw data has been suspended pending further advances in time optimization at the space point level. Nonetheless, single cosmic ray tracks have been reconstructed from the STAR-TPC system test using such a method with no intermediate pre-processing of the raw data.

7 Summary

An Elastic Tracking (ET) package is being developed for use in the high-multiplicity environment of large position sensitive detectors. At this time, functionality to reconstruct tracks from space points in a single detector exists. In a TPC-like detector modelled after STAR, efficiencies beyond 90%, even for the highest multiplicity collisions deemed possible at RHIC energies, have been achieved. While the time to reconstruct an event is significant, the retention of such capabilities are necessary should conventional road-finding techniques fail. The present development is focusing on ways to decrease the time used in event reconstruction and will move to the incorporation of multi-detector systems in the near future. The ultimate goal of the package is to reconstruct tracks without the need for pre-processing of raw data.

References

[1] M. Gyulassy and M. Harlander, Comp. Phys. Comm. **66** (1991) 31, and M. Gyulassy and M. Harlander, NIM **A316** (1992) 238.

[2] M. Ohlsson, C. Peterson, and A. Yuille, Comp. Phys. Comm. **71** (1992) 77.

[3] for example see R. Blankenbeckler, Comp. Phys. Comm. **81** (1994) 318, G. Stimpfl-Abele and L. Garrido, Comp. Phys. Comm. **64** (1991) 46.

[4] M. Ohlsson et al., Comp. Phys. Comm. **71** (1992) 77.

[5] M. Lindström, NIM **A357** (1995) 129.

[6] M. Fuchs, NIM **A367** (1995) 394.

[7] for example see http://wwwcn.cern.ch/pl/geant/geant4.html

[8] see B. Lasiuk, *Development of a C++ Class Library for STAR*, these proceedings.

[9] R. K. Bock et al., "Data Analysis Techniques for High-Energy Physics Experiments", (Cambridge University Press: 1990).

[10] N. Chernov and G. A. Ososkov, Comp. Phys. Comm. **33** (1984) 329.

[11] M. E. Beddo et al., STAR Conceptual Design Report, PUB-5347.

[12] T. Alber et al.,*The NA49 Experimental Apparatus* to be published in NIM.