AdEPT Gamma Ray Detection and Reconstuction Algorithm:

Volatility of Constants, and Future Considerations

Spencer R. Deevy, B.Eng.

Date Last Modified: August 18, 2017

Contents

1 Algorithm Overview

The following is a rough overview of the algorithm's operation, in hopes that it aids future developers with regards to the functionality of the gamma ray reconstruction code, as in its current implementation.

- Select operating mode (streaming data file or images)
- Import XZ and YZ projection images into software
- Remove background noise from projections
- Create data structure containing detector hits
- Find terminating points of full tracks; Used as start points for partial track generation
- Generate partial tracks via the cellular automaton traversal method; Starting at a terminating point, a heading is produced. From this heading a new reference poiint is chosen such that track point coverage is maximized. The heading vector is moved to this point, and the process repeats until no new heading points are found (indicating the end of the partial track)
- Generate partial tracks via the least linear squares traversal method; For any untracked points after using the previous traversal method, find all nearest neighbours within a large search radius and ensure that those points are all untracked. Fit a line to the points via LLS method, and check to ensure that all points are close enough to the line to validate this section as a straight partial track. Continue to perform this until all untracked points are covered.
- Merge partial tracks that share a sufficient number of points. Classify these as full tracks
- Amongst full tracks, check endpoints to determine if two full tracks share a vertex. Following this, check other endpoints of these tracks to determine if the length of the arms are comparable in size and are distanced far enough apart (this prevents situations where overlapping parallel tracks or tracks of disproportionate length are considered gamma ray candidates)
- From these gamma ray candidates in the XZ and YZ projection, compare their vertices. If within a small difference in the z direction, we check the arm endpoints. If the arms in either projection match up within the same small difference in z position, we can say that this is a gamma ray, and add it to the final gamma ray list.

2 Volatility of Constants

2.1 Noise Removal

Constant	noise_filtering_NN_bound
Value	2
Description	Number of nearest neighbours within noise filtering radius to filter out
Volatility	Low
Rationale	Even in high noise environments, the change that 3 or more noise detector hits will be within such close proximity to each other is very low. In case of extremely noisy envirnments, value can be changed to 3 or possibly 4, but will likely never go past this. Otherwise, actual tracks could be filtered out with the noise.

Constant	noise_filtering_radius
Value	4
Description	Radius at which to search for nearest neighbours in order to filter out noise
Volatility	Low
Rationale	For sparse noise, this radius bound could be increased, and for noisy data it could be reduced. Over a range of test images, values between 2 and greater were used which resulted in noise being filtered out correctly. As with the previous constant, very high noise environments require a tighter restriction on this variable, which may in turn lead to track hits being filtered out.

2.2 Terminating Point Detection

Constant	radius_minimum
Value	10
Description	Minimum radius at which to search for terminating point candidate nearest neighbours
Volatility	Low
Rationale	Used in conjunction with *minimum_NN*, the liklihood of this constant being changed is fairly low. This variable is already set at a fairly high value, while by default the minimum number of nearest neighbours needed is 10. Finding such a low number of nearest neighbours for such a large starting search radius is fairly likely.

Constant	radius_maximum
Value	20
Description	Maximum radius at which to search for terminating point candidate nearest neighbours
Volatility	Low
Rationale	For reasons described in *radius_minimum*, the liklihood of this constant being altered is very low. The liklihood of not finding enough nearest neighbours at this value is very low. This constant is put into place as an upper bound to avoid possible edge cases.

Constant	radius_step
Value	1
Description	Radius step at which to increase search radius for terminating point candidate nearest neighbours
Volatility	Low
Rationale	For reasons described in *radius_minimum* and *radius_maximum*, the liklihood of this constant being altered is very low. The liklihood of not finding enough nearest neighbours at this value is very low. This constant is put into place as a means of stepping between minimum and maximum radii.

Constant	accept_angle
Value	30
Description	Acceptance angle for segments similar to heading (+/- degrees)
Volatility	Low-Moderate
Rationale	This value gives a full arc of 60 degrees of acceptance for nearest neighbours within the current heading estimate. For points along a track that are farther away from the reference point, this value works well, however as NN get closer to this point, their angle can become more erratic. This angle in conjunction with the *reject_angle* have worked well for test cases thus far, but this acceptance angle may need to be increased if experimental data has more erratic behaviour.

Constant	reject_angle
Value	120
Description	Rejection angle for segments disimilar to heading (+/- degrees)
Volatility	Low
Rationale	This angle acts as an upper bound for points behaving erratically near a reference point. Such a large angle to be made against a reference heading is an easy method to disqualify NN that do not satisfy the current heading. The liklihood that this value needs to be reduced or increased relies on how erratic the point behaviour is around candidate terminating points, but to a lesser degree than the *accept_angle*.

Constant	minimum_NN
Value	10
Description	Minimum number of nearest neighbours to create accurate heading
Volatility	Low-Moderate
Rationale	This value is used to ensure that enough NN exist near the reference point to create an accurate heading, otherwise a small group of outliers could be considered a terminating point. The only major concern affecting this variable are when small groups of points are clustered. In this situation one or more of these points may be considered a terminating point. This is mostly offset by having fairly strict conditions on the *accept_angle* and *reject_angle* constants. It should also be noted that a track length may past closely to the terminating point of another track, which will skew results. In case of these scenarios this number can be increased to improve the chances that a argest heading group is constructed in the appropriate direction

Constant	terminating_threshold
Value	0.85
Description	Threshold for percentage of NN within heading for point to be terminating
Volatility	Low
Rationale	Given that the requirement currently is to have at least 85% of nearest neighbours within the accept angle of the heading, this value is restrictive enough in it's current state to warrant a low chance of being required to be altered. Again, the only scenarios that would affect this are extremely noisy environments or several tracks coming within close proximity (but not crossing). In these situations this value can be increased to 90% or so to restrict the required number of nearest neighbours within the acceptance angle to an even higher value.

Constant	filtering_distance
Value	10
Description	Distance at which to filter out terminating points near another
Volatility	Low
Rationale	This constant is unlikely to change because the radius at which it is set over is already a fairly large one, and as it is only being used to filter out terminating points so that only one point exists for each track endpoint, the liklihood that another endpoint for the same side of the same track should exist ourtside this radius is incredibly unlikely.

2.3 Partial Track Generation

Constant	radius_minimum
Value	10
Description	Minimum radius at which to search for partial track points' nearest neighbours
Volatility	Low-Moderate
Rationale	The minimum search radius at which to look for partial track points frrom a reference point on the current fathest point in the partial track is moderately variable depending on the partial track, as well as the number and type of tracks crossing the current partial track. Should the search radius include a substantial number of points belonging to a crossing track, the resulting heading will be skewed. This constant is aided by the *bounding_distance* constant, and so has a low - moderate chance of being altered.

Constant	radius_maximum
Value	20
Description	Maximum radius at which to search for partial track points' nearest neighbours
Volatility	Moderate
Rationale	For reasons described in *radius_minimum*, this constant has a moderate probability of being changed in the future. As the search radius approaches this constant, there is a higher chance that more points from a crossing track will be included within this radius, creating a skewed heading. This constant is also aided by the *bounding_distance* constant, but as it's effect can incorporate a larger number of points foreign to the current partial track, it has a higher variability than the *radius_minimum* constant.

Constant	radius_step
Value	1
Description	Radius step at which to increase search radius for terminating point candidate nearest neighbours
Volatility	Low
Rationale	In order to avoid increasing the search radius too quickly, increasing the possibility of including points foreign to the current partial track, this value should be kept low. Thus it is unlikely that this constant should need to be altered.

Constant	accept_angle
Value	30
Description	Acceptance angle for segments similar to heading (+/- degrees)
Volatility	Moderate-High
Rationale	This value gives a full arc of 60 degrees of acceptance for nearest neighbours within the current heading estimate for the first run, and an arc of 30 degrees for subsequent runs (traversing the track). This cone is fairly restrictive, especially for points close to the reference point. The restriction was placed in order to keep the partial track from veering off course when another track crossed it, but as a result of the restriction, lessens the liklihood the points close to the reference will be accepted (as their angles with the reference will vary more). As such, the variability of this constant is fairly high, and may need to be altered substantially to reflect tracks formed through experimental methods.

Constant	reject_angle
Value	120
Description	Rejection angle for segments disimilar to heading (+/- degrees)
Volatility	Low-Moderate
Rationale	This angle acts as an upper bound for points behaving erratically near a reference point. Such a large angle to be made against a reference heading is an easy method to disqualify NN that do not satisfy the current heading. The liklihood that this value needs to be reduced or increased relies on how erratic the point behaviour is around candidate terminating points, but to a lesser degree than the *accept_angle*.

Constant	minimum_NN
Value	10
Description	Minimum number of nearest neighbours to create accurate heading
Volatility	Moderate
Rationale	This value is used to ensure that enough NN exist near the reference point to create an accurate heading, otherwise a small group of outliers could be considered a part of the current partial track. This variable is attempting to balance retrieving enough NN of the actual partial track to create an accurate heading, but not so many NN as to include those from crossing tracks. Depending on the number of tracks crossing, as well as the angle that those tracks make with the current one, this constant may need to be altered

Constant	bounding_distance
Value	7.5
Description	Distance at which to bound next furthest point; Provides better point coverage
Volatility	High
Rationale	Used as a corrective measure for restrictive acceptance angle of partial track points. By restricting the furthest point, we restrict the next reference point's distance from the current one. This means that some points may be covered more than one time, but also allows the track traversal to inch it's way towards a point where another track crosse the current one. This allows the current track to move forward slowly, and promoted the chances that the majority of the NN in the search radius belong to the current track and not the crossing tracks. This value was arbitrarily chosen through testing of simulated data, such that it improved track reliability, but has high variability in results. Will need to be altered for experimental data.

Constant	LLS_distance_bound
Value	25
Description	Maximum distance at which a point can be considered part of a LLS line estimate
Volatility	Low-Moderate
Rationale	This bound is used as a maximum distance points can be away from the linear least squares line equation for the points to still be considered part of the track. As such, the factor effecting this constant is the width of the track as well as the frequency of noise around a track. This will increase point distanc from the line, and may therefore exclude some points from the partial track. That being said, the current value is set to 25 pixels, which is more than accompanying even for noisy tracks, and so is unlikely to change.

Constant	new_weight
Value	0.4
Description	Weighting for new heading estimate
Volatility	High
Rationale	The next heading to be calculated needs to be a combination of the previous heading and the current heading (averaged from the points within heading acceptance arc). This weighting is used on the latter, and remains a highly volatile constant. Balancing this constant and *old_weight* is extremely delicate and will need to be reconfigured. It should be noted that this volatility could be improved or eliminated by using a kalman filter or similar method.

Constant	old_weight
Value	0.6
Description	Weighting for old heading estimate
Volatility	High
Rationale	The next heading to be calculated needs to be a combination of the previous heading and the current heading (averaged from the points within heading acceptance arc). This weighting is used on the former, and remains a highly volatile constant. Balancing this constant and *new_weight* is extremely delicate and will need to be reconfigured. It should be noted that this volatility could be improved or eliminated by using a kalman filter or similar method.

Constant	min_partial_length
Value	5
Description	Minimum partial track length for the track to be considered valid
Volatility	Low
Rationale	This constant is used to ensure that tracks of insufficient or insignificant lenth are left out of the final partial track list, in order to reduce the number of checks upon track merging. Tracks above this length should be considered valid, and this constant is unlikely to deviate largely from it's original value

2.4 Partial Track Merging

Constant	merge_ratio
Value	0.8
Description	Percentage of similarity either track must share to be merged
Volatility	Low-Moderate
Rationale	Partial tracks should be of sufficient length that they overlap to a reasonably high degree. As such this constant has been set fairly high. Should partial tracks be shorter or longer, the minimum and maximum bounds on this constant may change.

2.5 Find Gamma Candidates

Constant	gamma_vertex_distance
Value	1
Description	Maximum distance between the two endpoints making the vertex for it to be valid
Volatility	Low-Moderate
Rationale	Used to ensure that two tracks forming a vertex, have endpoints that are sufficiently close to each other to be considered a vertex. Depending on how full tracks' are traversed, endpoints may be slightly farther away from each other than in simulated examples. In such a situation, this variable can be increased

Constant	gamma_arm_separation
Value	40
Description	Minimum distance between the two endpoints of the gamma ray arms for it to be valid
Volatility	Moderate
Rationale	Used to ensure that two arms of the gamma ray are sufficiently separated to be considered a gamma ray (overlapping tracks should not count). The liklihood for this constant to change is proportional to the accuracy at which a track can be created without veering off, as closer tracks will require this higher precision.

Constant	gamma_arm_distance
Value	40
Description	Maximum difference in length between either arm of the gamma ray
Volatility	Moderate
Rationale	Used to ensure that two arms of the gamma ray are close enough in length to each other to warrant a gamma ray detection. Volatility is set to moderate, as there is no exclusive restriction on gamma ray arms being different lengths, such that they are reasonable experimentally. This value was chosen based on the simulated test images, and was found adequate, but may need to be altered or removed in future versions of this code.

Constant	gamma_vertex_separation
Value	30
Description	Maximum difference between candidate gamma projections' vertices in the z direction
Volatility	Low-Moderate
Rationale	Used to ensure that the gamma ray candidates in either projection have a common vertex within a reasonable z-diffusion difference. Depending on the difference in diffusion and timing of readouts in either projection, this value bay be smaller or larger. It, however has been chosen with a conservative value of 30 pixels, where in reality it could be under 5 pixels difference in the z-direction. Note that this value is also dependent on how the shallow-curve tracks are traversed, as the two tracks may have different endpoints (by a small margin).

3 Future Considerations

3.1 Performance and Reliability

Performance and reliability of the current software implementation are in a volatile state. On the performance side, excluding the time taken to render images of the software process, is fairly decent. Taking a few minuutes to scan a volume, detect any gamma rays, and to reconstruct those gamma rays as objects separate from the rest of the detector data is a decent milestone to have made on a first attempt. Reliability on the other hand is quite volatile for the current build. Linear least squares method for partial track construction was used as a means to alleviate some of this unreliability, but as it stands, partial track generation still needs improvements. Specifically when parallel tracks are very close together, the partial track algorithm may not be able to differentiate the two. In addition to this, to many tracks crossing one track may coss the tracking on that partial track to fail prematurely.

As such, the primary goal for improving the software is to improve the reliability and accuracy of partial track generation, while optimizing the code to improve runtimes. Some suggestions to improve runtimes would be to implement the algorithm in a compilable, low-level language such as C or C++, and the possibility of testing the use of quadtrees instead of the KD trees used to represent the sparse volume data. With regards to improving accuracy and reliability of track generation, I would suggest modifying the cellular automaton partial track traversal method to have a smaller acceptance angle, as well as an overlapping step pattern for the next point being analyzed, so that points near the current reference point are not discluded. It should also be noted that some success has been made using kalman filtering techniques in time projection chamber particles track reconstruction[1], which may aid in this first traversal method. In addition to this, some partial tracks created via the linear least squares method do not sufficiently overlap with neighbouring partial tracks for merging. This issue should be investigated further, as the cause of this method to stop prematurely is not well understood. These modification suggestions are outlined below:

- Modify cellular automaton traversal method to adopt a kalman filter approach to weighting the previous heading's value and the current heading's value more accurately over track traversal
- Decrease search radius, decrease acceptance angle, and force overlapping reference points to allow for more accurate partial track generation, while ignoring crosing tracks
- Investigate performance different between using KD trees and quadtrees to represent the volume projection data
- Port algorithm to C, C++, or other lower level, compilable language to increase runtime performance

3.2 Testing

Testing performed on the current implementation of the code consists of a small subset of galactic cosmic rays and gamma ray images, which were then added pixel-wise together to produce XZ and YZ projections for a gamma ray event scenario. As this process only used a small set of test images, new test images will need to be made. A larger sample set of test images will aid in further developing/optimizing the software, however, it should be noted that up until this point all images have been created through simulations. As portions of the detector are now operational, it would be highly advisable to create test images from the detector to be used with this software, as those images will more accurately reflect real-world conditions.

References

[1] Belikov, Y., Bracinik, J., Ivanov, M. and Safarik, K. (2003). TPC tracking and particle identification in high-density environment. Computing in High Energy and Nuclear Physics, pp.4-8.