2007

A directional algorithm for an electronically-collimated radiation detector (ECRD)

Adam Lackie
Louisiana State University and Agricultural and Mechanical College, adamlackie@gmail.com

Recommended Citation

Lackie, Adam, "A directional algorithm for an electronically-collimated radiation detector (ECRD)" (2007). *LSU Master's Theses*. 1312.
https://digitalcommons.lsu.edu/gradschool_theses/1312

A DIRECTIONAL ALGORITHM FOR THE ELECTRONICALLY COLLIMATED RADIATION DETECTOR

A Thesis

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Master of Science

in

The Department of Physics & Astronomy

by

Adam Wayne Lackie
B.S., Louisiana State University, 2002
May 2007

ACKNOWLEDGEMENTS

I must thank Dr. Kip Matthews for inspiring a passion for medical imaging. I will always be grateful for his skillful guidance in introducing me to the world of research. His boundless patience, understanding and willingness to help at all times will always stay with me. I am indebted.

I also thank the other members of my thesis committee for their input and time: Dr. Wei-Hsung Wang, Dr. Michael Cherry, and Dr. Kenneth Hogstrom. I thank Dr. Hogstrom additionally for transforming the Medical Physics department into a world-class program. His instruction and guidance have been invaluable. I could not have gotten through without him.

For training both didactic and clinical, I thank Dr. John Gibbons whose clear teaching in the classroom and direction in the clinic will be a great asset. Also, his uncanny (at times disconcerting) ability to find the most difficult question to ask during oral exams will, I am sure, prepare me for anything that the world outside of the University can throw at me.

I thank Dr. Erno Sajo for providing a firm educational basis for all of the topics that were later taught in the program. His meticulous teaching style engendered equally meticulous habits which will serve me well in whatever I do.

I thank Dr. Blair Smith for his generous offers of help and guidance, and Will Hill for being there for me to bounce ideas off of. Also, the students that proceeded through the program in the same year as me (John Richert, Laurie Kelly, Ken Bernstein, Michael

Ashenafi, and Jabari Robinson) deserve profound thanks. We all helped each other. They taught me the meaning of teamwork.

Finally, I would like to thank my dedicated partner Levi Denham for patience, support, and the kick in the rump that I needed periodically. He has stuck with me through thick and thin, lending me great strength and keeping me human. And to my parents Steve Lackie and Karyn Placek (not to mention the rest of my extended family) who always believed in me and supported me in every way that they could.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

An electronically-collimated radiation detector (ECRD) is being developed to be used for locating radiation sources, e.g. for intraoperative localization of sentinel lymph nodes, or for public safety applications. The design emphasizes a compact, portable detector with a wide field of view. Typical probes provide either high sensitivity but no directional information when uncollimated, or directional information but poor detection efficiency when collimated. The ECRD design provides high sensitivity to the presence of radiation because it lacks physical collimation, and simultaneously provides directional information using electronic collimation.

Intended to be a hand-held device, the ECRD front end comprises an array of cadmium-zinc-telluride (CZT) detectors. An incident gamma ray scatters in the primary detector; interaction of the scattered photon in a secondary detector is detected in coincidence. For each photon, Compton kinematics specifies a cone on which the source must be located. Localization is achieved by finding the intersection of many Compton-scatter cones. This paper reports on the development and evaluation of two directional algorithms for this device, a modified Compton telescope algorithm and an algorithm based on finding the intersections of rectangles circumscribing the Compton cones.

The methods developed were evaluated using ideal simulated data from a point source as well as data from a Monte Carlo simulation of an ECRD device. The accuracy, precision and convergence of each directional algorithm were evaluated. It was found that for the modified Compton telescope technique, a useful field of view extending 60º

from the forward direction was observed, an angular resolution better than 20º was achieved throughout the field of view, and the method converged to these values around 300 events; the results for the ideal data did not significantly differ from those using the Monte Carlo data. For the circumscribed rectangle technique, the useful field of view covered nearly the entire area in front of the detector for the ideal data but the more realistic physics of the Monte Carlo simulated data shrank the useful field of view to the region within approximately 30º of the forward direction, while the angular resolution was better than 20º and convergence was approached at approximately 50 events.

CHAPTER 1

INTRODUCTION

1.1 Motivation for the ECRD Project

An "Electronically-collimated Radiation Detector" (ECRD) to be used for localization of radiation sources is currently under development. This thesis describes the development of a directional algorithm for such a device. With current methods of radiation detection, direction-sensing capability typically comes only at the expense of sensitivity, and high sensitivity is usually gained at the cost of localization capability. The ECRD aims to provide both good sensitivity and good localization capability. Using solid-state room-temperature detector technology, such as cadmium-zinc-telluride (CZT) detectors, and portable computers, the ECRD will be compact, portable, and highly versatile. The source localization capability could be used for intraoperative detection of sentinel lymph nodes and metastases. Its design also easily lends itself to being used in a variety of areas where there is concern for public or occupational radiation safety, including problems of national security.

The directional algorithm developed for the ECRD maps coincidence-detection events of Compton-scattered gamma rays to determine the direction to the source of the radiation. The kinematics of Compton scattering allows the device to localize objects without physical collimation (which, by its nature, blocks out much of the incoming radiation thereby decreasing sensitivity) and thus allows higher sensitivity than present methods of source localization. Without physical collimation the detector count rate will be higher than that for a similar detector with physical collimation, but each event gathered by the ECRD will contain less localization information because the radiation source can only be confined to the surface of a cone described by the kinematics of Compton scattering (Suave et al, 1999). The primary aspiration of this project was to implement and evaluate a robust and efficient algorithm for analyzing the detector data, to quickly and accurately locate radioactive material.

1.2 Goal and Aims

The goal of this work was to develop two directional algorithms, at least one of which will meet the needs of the ECRD concept.

Hypothesis 1. *Accuracy*: For a stationary source, the algorithm provides source localization with angular error of no more than 10° out to near ±90° of the field of view.

Hypothesis 2. *Precision*: The algorithm achieves an angular resolution of 20° full-width half maximum (FWHM).

Hypothesis 3. *Convergence*: An estimate of the source position, within the limits described above, will be achieved with fewer than 1000 Compton event-pairs. Convergence is defined as the reduction of the rate of change to near zero of a metric appropriate to the method.

The first part of the project was the development of the two algorithms, which individually may provide the desired performance or could be used together. The second part of the project involved evaluating each algorithm's performance to determine if either method meets the desired goals. The specific aims needed to assess the hypotheses were:

1. Development of the directional algorithm method based on identifying intersections of cones using an image reconstruction-based approach.
2. Development of the directional algorithm method that utilizes a simplified calculation identifying the intersections of cones based on the intersections of circumscribed rectangles.
3. Evaluation of algorithm performance using data from computer-simulated models of the ECRD. The specific tasks were:
    a. Measure the size of the useful field of view (that is, the area of the field of view in which the absolute error of the direction estimate is less than 10º);
    b. Measure angular resolution across the field of view;
    c. Assess rate of convergence to the desired performance levels; and
    d. Determine the influence of detector pixel size and energy resolution on algorithm performance.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

2.1 Radiation Detectors

Conventional portable gamma radiation survey meters (e.g. Geiger-Mueller counters and scintillation detectors) detect radiation by counting the number of interactions with radiation that occur within the detector volume. Generally, these devices allow radiation to enter the meter from any direction and therefore give information only about the level of radiation present and not about the direction from which the radiation is coming. Such detectors have high sensitivity to radiation because none is blocked from entering the detector.

Modifications to add direction-sensing capability commonly involve physically collimating the device, restricting the direction through which radiation can enter the detector. This modification reduces the sensitivity of the detector, usually by a substantial amount, and thus makes low-activity sources difficult to observe.

The utilization of Compton kinematics to aid in image formation is by no means a new technique. Astronomers utilize telescopes with Compton-based reconstruction algorithms to map gamma emissions from the cosmos. These telescopes utilize the information obtained from Compton scattered photons whose sources are so far away that they can be assumed to be at infinity. Another application under development is the Compton camera for radioisotope imaging. In this case, the problem under consideration is equivalent to filtered backprojection along the surfaces of cones (rather than along lines as in conventional computed tomography images) (Rogers, 2004). Medical Compton cameras are concerned with precisely mapping the distribution of a radiopharmaceutical in a patient's body. Three-dimensional information, in this case, is very important. The situation faced by the hand-held ECRD has a difficulty level somewhere between the astronomy and radioisotope imaging applications. The sources cannot be assumed to be at infinity, but only two-dimensional information is required to point the user in the direction of the radiation source. A source distribution is not desired, only a direction to the source (without information about the distance to the source.)

2.2 Electronic Collimation

Electronic collimation is a method of extracting information about the direction that a photon entered a detector using information inherent in the radiation physics rather than by blocking or limiting the direction by physical means. Using Compton scatter data is one means to achieve electronic collimation. Compton scattering is a process by which a photon interacts with an electron, deposits some fraction of its energy (as kinetic energy imparted to the recoil electron), and propagates away through an angle proportional to the energy lost. A primary detector, sensitive to both position and energy, records the scattering interaction. Then, the scattered photon interacts with a second detector which is also position-sensitive and possibly energy-sensitive. (The second detector does not need to be energy-sensitive if the initial energy of the source photon is known.) These two interactions occur almost simultaneously and can be selected by coincidence-gating. The recorded information will therefore be the (x,y,z) coordinates and the energies deposited for each interaction pair. Summing the energies gives the original energy of the incoming photon if the second interaction is photoelectric absorption, and the Compton equation (Equation 1) can be solved for the angle through which the photon scattered:

$$\mu = \cos\theta = \frac{(E_2^2 + E_1 E_2 - m_e c^2 E_1)}{(E_1 + E_2)E_2} \tag{1}$$

where E_1 is the energy deposited by the first interaction, E_2 is the energy deposited by the second interaction, and μ is the cosine of the scattering angle θ. The calculated direction angle is degenerate in the azimuthal direction. The information provided is thus a conical surface (Figure 1) that contains the source of the Compton-scattered photon, with thickness due to uncertainties in the energy measurements as well as the position measurements.

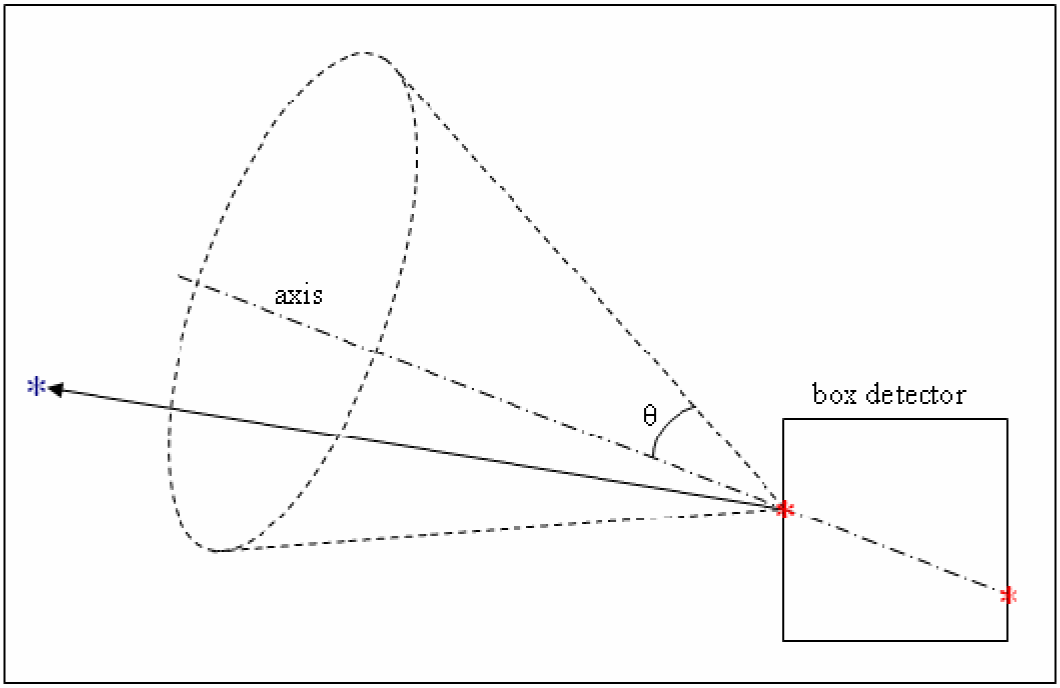


Figure 1: A cone described by the Compton geometry. A vector, along which the source (blue star) lies, is somewhere on the surface of the cone. The cone axis is defined by the two interactions in the detector (red stars).

One coincident pair of interactions, which we will call an event, therefore does not provide the information necessary to localize the gamma-ray source. A number of events, each yielding a conical shell in space, must all intersect at the location of the gamma-ray source (Rogers, 2004). Solving this problem of the intersection of overlapping cones is non-trivial (Miller, 1987). Various approaches have been proposed to solve the problem, which are typically simplifications to the general problem of computing intersections of arbitrary cones. For example, the solutions applied to Compton telescopes assume that the sources lie on a sphere of infinite radius. Compton cameras used in medicine assume that the sources all lie within a finite region surrounded by or near the detectors (Parra, 2000; Suave, 1999; Wilderman, 1998). Most of the Compton camera applications attempt to reconstruct the source distribution in three dimensions which requires an extensive sampling of data space to generate an accurate three-dimensional picture.

2.3 Direct Calculation of Cone Intersections

There are two approaches to directly calculating the intersection curves of overlapping cones. Cones are part of a geometric family called the natural quadrics. An algebraic approach to finding the intersection consists of solving high-order polynomials for each other, i.e., finding the points or curve where the quadrics are equal. The methods based on this algebraic approach are computationally expensive and yield unstable solutions because they are numerical and will involve convergence issues because of thickness of the cones due to measurement errors (Miller, 1987; Du et al, 2001). A geometric approach, on the other hand, looks at the broad attributes of the quadrics (in the case of a cone, these attributes are the vertex, axis vector, and opening angle) and relies on mathematical models based on these few values to provide the intersection curves. In many cases these intersections yield complex non-planar curves (called space-quartics) and will require long computation times or may suffer from convergence issues. These problems limit the effectiveness of this method for the "on-the-fly" calculations in which we are interested (Goldman, 1983; Miller, 1999).

2.4 Formulation of the Reconstruction Problem

2.4.1 The Forward Problem

With a source distribution

$$f'(r,\theta,\phi) = f'(\vec{\Omega}), \tag{2}$$

(renaming $\vec{\Omega}$ as (r,θ,ϕ)) one is faced with finding the set of expected detector events

$$g(\vec{d}_1,\vec{d}_2,E_1,E_2) = g(\vec{X}), \tag{3}$$

where $\vec{d}_n$ are the locations of the interactions and E_n are the energies of the interactions. $\vec{X}$ is the set of cone parameters which are computed using the Compton kinematics described by the locations and energies of the interactions. The expected value of the measurements, $g(\vec{X})$, is related to the source distribution, $f'(\vec{\Omega})$, by

$$g(\vec{X}) = (Hf')(\vec{X}) = \int_{\vec{\Omega}} h(\vec{X},\vec{\Omega}) f'(\vec{\Omega}) d\vec{\Omega}, \tag{4}$$

where *h* is the conditional probability that a photon emitted from a location $\vec{\Omega}$ will result in a measurement of $\vec{X}$, and *H* is the transformation operator. Technically the operator, *H*, is an infinite-dimensional linear operator, but it can be approximated by a matrix, $\boldsymbol{H} = \{h_{db}\}$, which transforms the discrete versions of f' and *g*, which will be called ***f*** and ***g***. The likelihood of observing an arbitrary sequence of events is given (in the discrete case) by

$$f(\tilde{\mathbf{g}}|\mathbf{f}) = \prod_{d=1}^{D} \frac{\left(\sum_{b=1}^{B} h_{db} f_b\right)^{\tilde{g}_d} \exp\left(-\sum_{b=1}^{B} h_{db} f_b\right)}{\tilde{g}_d!} , \qquad (5)$$

where $\tilde{\mathbf{g}} = [\tilde{g}_1, \cdots, \tilde{g}_D]$ are the measured data binned into a total of D detector pixels (Rogers et al., 2005).

2.4.2 The Inverse Problem

From the detector measurements, *g*, we want to recover f'. Inverse filtering convolves the inverse of the transformation operator with the measured data to retrieve the original distribution:

$$H^{-1} \otimes g = \hat{f} , \qquad (6)$$

where $\hat{f}$ is an approximation to the original distribution. Inverse filters are generally unstable because of singularities wherever *H=0*, so other filters must be applied to the problem. However, these solutions can never completely specify f' (Rogers et al., 2005).

2.5 Methods of Image Reconstruction

2.5.1 Analytic Solutions

One can find the inverse of the imaging equation by expanding it in terms of a set of orthogonal functions and finding their inverse. The spherical harmonics, e.g., have been used by Basko et al. (1998) and Parra (2000), among others, to transform the cone projections into plane projections on which the Radon transform can be used (as in typical computed tomography).

This formulation in terms of spherical harmonics considers that each primary detector pixel is the center of a unit sphere (no matter the actual geometry of the detector). This simplifies the inversion problem by allowing the expansion in spherical harmonics. Parra adds to the previous work of Basko et al. by applying weighting functions to account for noise properties. The Radon transform then gives an approximation of the original source distribution.

While such methods may be more elegant than other solutions, they suffer from several limitations (Rogers et al., 2005).

1. Attenuation in the source causes the transformation operator, *H*, to become object dependent.
2. Statistical weighting is better approximated by other alternatives, such as the maximum likelihood (ML) operators (discussed in the next section).
3. Analytic inverse functions are not necessarily computationally efficient.
4. Complete data are often not collected by the detector, i.e. the detector geometry must collect data over 4π radians to produce a complete data set. Incomplete data could be filled using iterative methods, but this would no longer be an analytic method.
5. Solutions are often not constrained to the appropriate function space, e.g. they may produce negative or imaginary values.

2.5.2 Iterative Reconstruction Methods

Iterative reconstruction methods are usually statistically-based, often incorporating source attenuation and other physical factors. They can be implemented for detector geometries that do not collect a complete data set, and can incorporate noise models in a straight-forward manner.

Maximum likelihood (ML) estimation is similar to a weighted least-squares method in that it uses the inverse of the measurement covariance as the weighting function, but here the covariance is estimated by

$$\mathbf{W} = diag^{-1}\left(\mathbf{H}\hat{\mathbf{f}}\right). \tag{8}$$

Thus, the source distribution estimate is

$$\hat{\mathbf{f}} = \left(\mathbf{H}^T \mathbf{W}\mathbf{H}\right)^{-1} \mathbf{H}^T \tilde{\mathbf{g}}\,. \tag{9}$$

Because the covariance is unknown, the estimated function may not completely reproduce the original function (Lehner et al., 2004; Lee and Wehe, 2004).

Using an iterative expectation-maximization (EM) algorithm in conjunction with the ML estimation gets around this particular hurdle. The EM formulation is

$$\hat{f}_b^{(k+1)} = \frac{\hat{f}_b^k}{\sum_d h_{db}} \sum_{d=1}^{D} \frac{g_d h_{db}}{\sum_b h_{db} \hat{f}_b^{(k)}} \tag{10}$$

where k is the iteration number. The main problem with this EM implementation is that it has a slow convergence rate (Rogers et al., 2005).

The list-mode likelihood EM method takes advantage of the case when many detector bins have gathered no data. Here, the formulation is

$$\hat{f}_b^{(k+1)} = \frac{\hat{f}_b^{(k)}}{\sum\limits_d h_{db}} \sum_{\{d:g_d \neq 0\}} \frac{g_d h_{db}}{\sum\limits_b h_{db} \hat{f}_b^{(k)}}\,, \tag{11}$$

where only the non-zero detector bins are used in the summation. The computation time increases when the number of detected events is spread over many bins (Wilderman, 1998; Du et al., 2001).

2.6 Reasons for Developing an Alternative Method

The algorithms described above provide more information than is needed for the ECRD concept. Compton camera algorithms are concerned with precisely defining the three-dimensional distribution of a source in a confined area. They are potentially too computationally expensive and slow, and three-dimensional images are not required. We are not concerned with the source distribution, only the direction to the source. The ECRD needs fast calculations that work well with limited data. Two-dimensional projection images are satisfactory for this implementation.

CHAPTER 3

RECONSTRUCTION METHODS DEVELOPED FOR THE ECRD

Two methods of direction reconstruction have been implemented and evaluated for application to the ECRD. These methods have been chosen over those described in the previous chapter because they provide greater simplicity (and thus allow for easier computation) than other analytical solutions, and they do not suffer from the long computational times required for the iterative methods. The first method is a modification of an imaging algorithm used by Compton gamma-ray telescopes (Gunter, 2005); source localization is determined by filtered backprojection of the Compton-scatter cones' surfaces onto a sphere. The second method locates the source as the smallest common region of overlap of the bases of the cones on the sphere. Other methods of direction reconstruction are possible, such as utilizing matched events (e.g., using events that have the same cone vertex and scattering energy) and direct geometrical or numerical calculation of the arbitrary conic intersections in space; however, these other methods make inefficient use of the available event data or they cannot provide real-time results because of their computational complexity.

The first step in the reconstruction process for both methods is transforming event information into Compton cones directed out from the primary detector. To uniquely define a cone, one must specify three attributes: vertex, opening angle, and axis. The cone vertex is simply the location of the first (Compton scatter) interaction. The opening angle is the scattering angle obtained from the Compton equation. The cone axis is the normalized vector connecting the first and second interactions. This gives all the information needed as input to the two direction reconstruction methods. IDL (Research Systems, Inc.) was the high-level computing language used to develop the algorithms described in this thesis.

3.1 Modified Compton Telescope Technique

The filtered backprojection method collects a set of circles resulting from the backprojection of the Compton cones onto the surface of a sphere (Figure 2) whose radius should be large compared to the size of the detector. See Section 5.1.5 for a discussion of the effect of the sphere size on the reconstructed image.

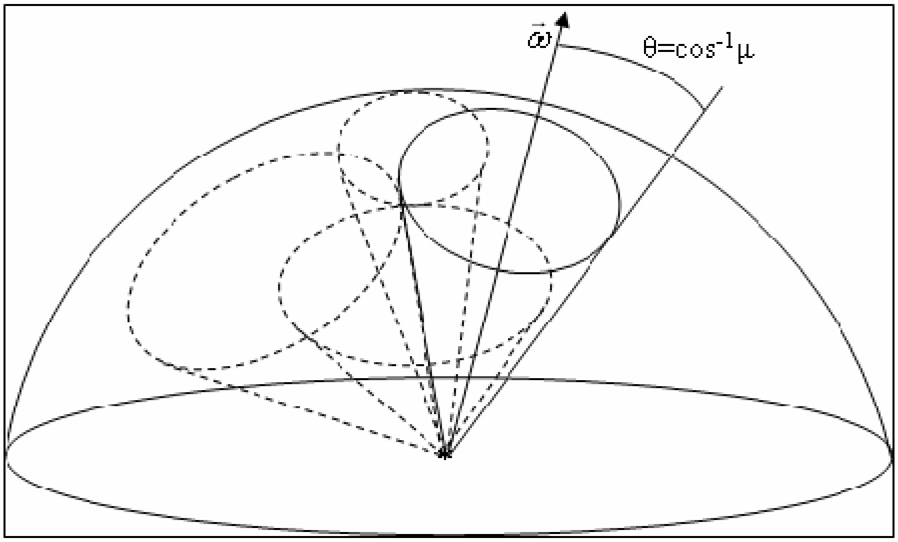


Figure 2. Illustration of backprojection of cones onto a sphere whose center coincides with the vertices of the cones. The backprojections yield circles in this case.

Typical filtered backprojection uses Fourier-space filtering, but Fourier methods are not directly applicable to the sphere surface. Therefore, Gunter (2005) proposed stereographic projection to transform the sphere surface onto a plane. The backprojection on this plane is then ramp filtered in Fourier-space and transformed back from the stereographic projection. The resulting image is the filtered backprojection of the source distribution on the sphere (Gunter, 2005).

In the modified Compton telescope method, the source distribution is described by a function, $f(\vec{\Omega})$. Compton cones, defined by their axes, $\vec{\omega}$, and the cosines of their opening angles, μ, are histogrammed into a function $g(\vec{\omega},\mu)$. The IDL structures that contain the interaction information and the cone parameters are:

```
location = { XYZ_STRUCT, x:0.0, y:0.0, z:0.0, unit:'' }
interaction = { INTERACTION_STRUCT, energy:0.0D,            $
            unit:'keV', location:{ XYZ_STRUCT } }
cone = { CONE_STRUCT, vertex:{ XYZ_STRUCT },                $
            axis_vector:{ VECTOR_STRUCT },                  $
            half_angle:0.0D}
```

Note that the histogram, g, is written in terms of cone parameters instead of detection energies and positions. We assume that the event data have already been converted from energies and positions into cone parameters. The IDL code that achieves this conversion is

```
      ;cone parameters
axis_vect = normalize({VECTOR_STRUCT, x1-x2, y1-y2, z1-z2})
      ;calculate the scatter angle using the Compton equation
cos_alpha=(energy2^2+energy1*energy2-                       $
      511.*energy1)/((energy1+energy2)*energy2)
if abs( cos_alpha ) le 1 then begin             ;error checking
      alpha=acos(cos_alpha)
      cones[j].vertex=events[i,0].location
      cones[j].axis_vector=axis_vect
      cones[j].half_angle=alpha
endif
```

The histogram is related to f by

$$g(\vec{\omega},\mu)=\iint_{|\vec{\Omega}|=1} d^2\vec{\Omega} f(\vec{\Omega})\delta(\mu-\vec{\Omega}\cdot\vec{\omega})\ . \tag{12}$$

The integral is over the sphere. This is the imaging equation developed by Parra (2000) who proposed a reconstruction technique utilizing spherical harmonics. In implementation, a sphere is defined around and centered on the primary detector. The intersections of the cones with the sphere are simply circles (or nearly so, as long as the cone vertex is near the center of the sphere). Backprojection onto the sphere therefore produces a set of circles all intersecting on the source. Scatter angles between 5º and 90º only are chosen for backprojection because angles that fall outside these limits often yield erroneous results (Smith, 2006). The IDL code that achieves this backprojection is

```
      ;find the center of the backprojected circle
r=!RADEG*cones[i].half_angle
x_center=z0*cones[i].axis_vector.i+cones[i].vertex.x
y_center=z0*cones[i].axis_vector.j+cones[i].vertex.y
z_center=z0*cones[i].axis_vector.k+cones[i].vertex.z
th_center=!RADEG*asin(x_center/sqrt(x_center^2+z_center^2))
ph_center=!RADEG*asin(y_center/sqrt(y_center^2+z_center^2))
      ;calculate how many points are needed to create a solid
      ;circle of connected points
step=atan(0.25/r)
num_step=round(360/step)
      ;draw the circle
for p=0l,num_step do begin
      itheta=round((r*cos(step*p)+th_center)*pixel_per_deg)+n/2
      iphi=round((r*sin(step*p)+ph_center)*pixel_per_deg)+n/2
      if itheta+q lt n and itheta+q ge 0 and iphi+q lt n    $
            and iphi+q ge 0 then ftemp[itheta+q,iphi+q]=1
endfor
f=f+ftemp
ftemp[*]=0
```

The image produced, however, has extraneous information from all of the parts of circles that do not intersect, or intersect somewhere other than the source. As in filtered backprojection, Fourier filtering is used to get rid of this blurring. To apply the Fourier transform to a spherical surface, one first applies a stereographic projection $\left(\vec{\Omega} \leftrightarrow \vec{z}\right)$ to map the sphere onto a plane (Figure 3).

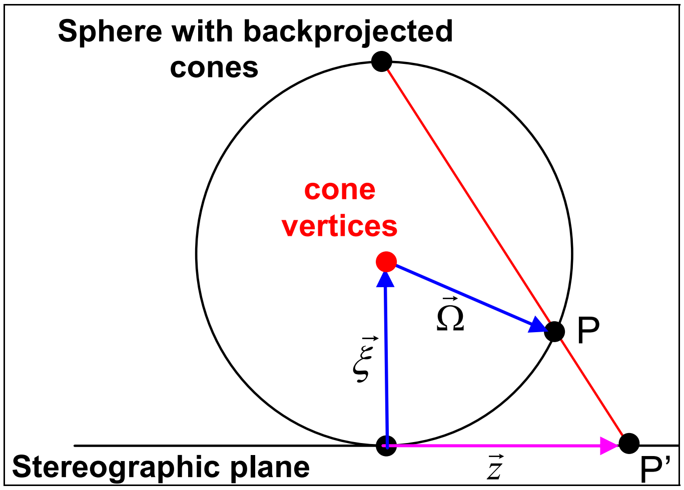


Figure 3. Each point, P, on a sphere is projected from the antipodal point (or center of projection) opposite the sphere's point of tangency onto the plane at a point, P'.

The antipodal vector $\vec{\xi}$ is a unit vector describing the direction to the center of projection. Stereographic projection allows direct filtering of the image without deconstruction into spherical harmonics (Gunter, 2005). This process, however, distorts the image (Figure 4) and resulting artifacts of the reverse transformation may degrade the image quality. One artifact of the process is that a bright spot appears at each of the poles after the inverse stereographic projection. These spots may be brighter than the source spot, so the outer 5° are zeroed out in the final image.

The forward stereographic projection is defined by

$$\vec{z} = \frac{\left\lfloor \vec{\Omega} - \left(\vec{\xi} \cdot \vec{\Omega}\right)\vec{\xi} \right\rfloor}{1 - \left(\vec{\xi} \cdot \vec{\Omega}\right)}, \tag{13}$$

and the inverse stereographic projection is

$$\vec{\Omega} = \frac{\left\lfloor 2\vec{z} + \left(\left|\vec{z}\right|^{2} - 1\right)\vec{\xi} \right\rfloor}{\left|\vec{z}\right|^{2} + 1}, \tag{14}$$

where $\vec{z} \cdot \vec{\xi} = 0$, meaning that these vectors must always be perpendicular to each other, and $|\vec{\xi}| = |\vec{\Omega}| = 1$ because the sphere is defined as a unit sphere. The stereographic projection maps the sphere surface onto a flat plane tangent to the sphere at the point specified by the antipodal vector $\vec{\xi}$. These projection formulas can be separated into components and written as

$$\begin{aligned} x &= k \cos\phi \sin(\theta - \theta_0) \\ y &= k[\cos\phi_1 \sin\phi - \sin\phi_1 \cos\phi \cos(\theta - \theta_0)]' \end{aligned} \tag{15}$$

where θ_0 is the central longitude and ϕ_1 is the central latitude (both of these were chosen to be 0º for this application), x and y are coordinates in the plane and θ and ϕ are coordinates on the sphere, and

$$k = \frac{2R}{1 + \sin\phi_1 \sin\phi + \cos\phi_1 \cos\phi \cos(\theta - \theta_0)}, \tag{16}$$

where R is the radius of the sphere.

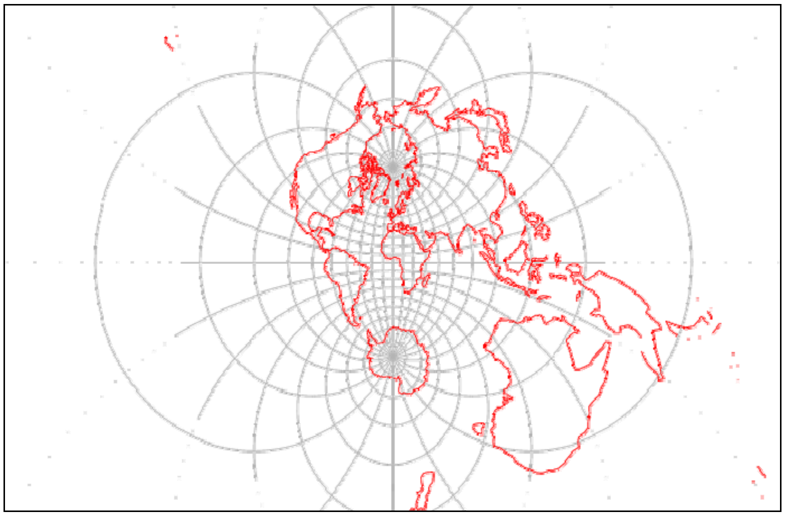


Figure 4. Illustration of the distortion of the globe following the stereographic projection. Note the size of Australia relative to other continents, e.g. Africa (from http://mathworld.com/StereographicProjection.html).

Two-dimensional Fourier filtering with the ramp filter is applied and the inverse stereographic projection then maps the image back onto the sphere. The values in the image are not renormalized after the inverse Fourier filtering (as would be done, for example, when generating computed tomography images) because the specific values are not of particular interest. It is only the relative value of the brightest points in the image that are of concern as these are what the algorithm uses to determine the source direction. The inverse stereographic transforms are

$$\begin{aligned} \phi &= \sin^{-1}\left(\cos c \sin\phi_1 + \frac{y \sin c \cos\phi_1}{\rho}\right) \\ \theta &= \theta_0 + \tan^{-1}\left(\frac{x \sin c}{\rho\cos\phi_1 \cos c - y \sin\phi_1 \sin c}\right) \end{aligned}, \tag{17}$$

where

$$\begin{aligned} \rho &= \sqrt{x^2 + y^2} \\ c &= 2\tan^{-1}\left(\frac{\rho}{2R}\right) \end{aligned}. \tag{18}$$

The direction to the source is picked as either the location of the brightest point in the image (if there is only one point that is brightest) or the average (centroid) of the locations of the brightest points. Figure 5 summarizes this process. Note that this method is capable of mapping a source distribution as its projection on the sphere, although only source location, i.e. the two-dimensional direction to the source, is the primary concern for our application.

3.2 Circumscribed Rectangle Technique

The second method localizes the source by placing circumscribed rectangles (squares) around the intersection of the cones' surfaces with a plane (or sphere); the

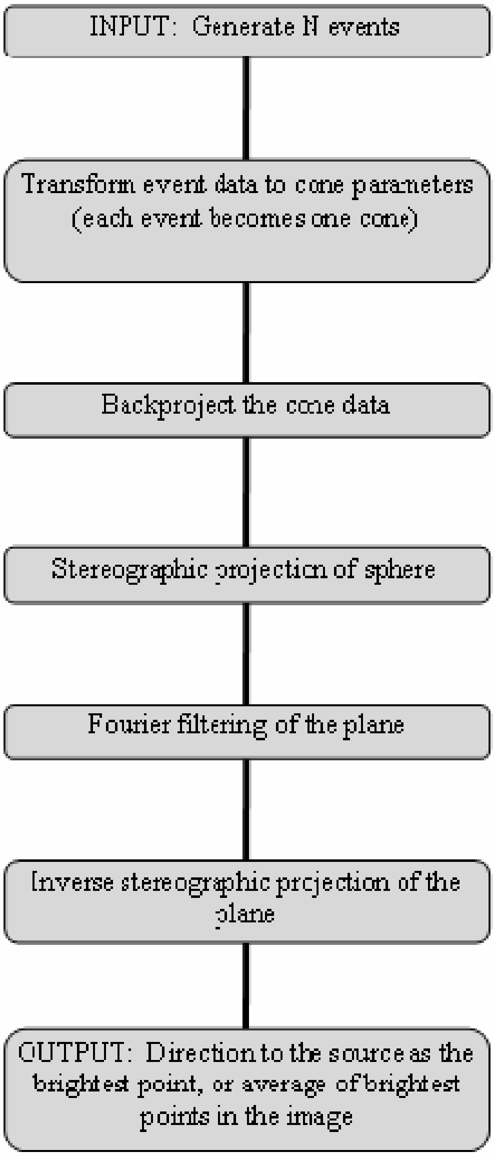


Figure 5. Process describing the modified Compton telescope technique.

intersection (i.e., the logical AND) of the rectangular areas encompasses the source location. This intersection converges on the source location as more cones are added. The conic sections produced when a cone intersects a plane are well-behaved and very easy to compute. By simply comparing a few parameters, the conic section can be determined by geometric method (Miller and Goldman, 1992). One must define only the direction cosine of the cone axis and the sine of the cone opening angle; the relationship of these two angles determines the type of conic section, from which the circumscribed rectangle can be computed. Ideally, the plane would be chosen to pass through the source, but the source location is unknown. This approach has the complication that the curves become highly eccentric when the cone's axis vector approaches ±90°.

Alternatively, the circumscribed rectangle can be defined from the backprojection of the cone on a sphere (cf. modified Compton telescope technique). If the cone vertices are confined to a small region around the sphere's center, the backprojections are nearly circular and the circumscribed rectangle is easily estimated from a cone's axis vector and opening angle (α) – the circumscribed rectangle is a square of side *2*α centered at the direction angles (θ,φ) of the axis vector; an arbitrary padding can be added to the limits to accommodate data issues such as energy resolution. The IDL code that keeps track of the decreasing limits of the area where the source may lie is

```
for i=11,num-1 do begin
      ;replace the maximum theta if it is less than the previous
      if list[i,0] lt overlap[i-1,0] and list[i,0] gt       $
            overlap[i-1,1] then overlap[i,0]=list[i,0] else $
            overlap[i,0]=overlap[i-1,0]
      ;replace the minimum theta if it is greater than the
      ;previous
      if list[i,1] gt overlap[i-1,1] and list[i,1] lt       $
            overlap[i-1,0] then overlap[i,1]=list[i,1] else $
            overlap[i,1]=overlap[i-1,1]
      ;replace the maximum phi if it is less than the previous
      if list[i,2] lt overlap[i-1,2] and list[i,2] gt       $
            overlap[i-1,3] then overlap[i,2]=list[i,2] else $
            overlap[i,2]=overlap[i-1,2]
      ;replace the minimum phi if it is greater than the previous
      if list[i,3] gt overlap[i-1,3] and list[i,3] lt       $
            overlap[i-1,2] then overlap[i,3]=list[i,3] else $
            overlap[i,3]=overlap[i-1,3]
endfor
```

where the array *list* contains the limits of each backprojected circle arranged *[maximum theta, minimum theta, maximum phi, minimum phi]*, and the array *overlap* contains the shrinking limits arranged in the same way that the array *list* is. Figure 6 summarizes this process.

The results may be sensitive to the sphere's size and the source location may fall outside the circumscribed rectangle if the backprojection deviates significantly from circular. Both this behavior and the error in the backprojected cones become more significant when the sphere radius deviates significantly from the distance to the source.

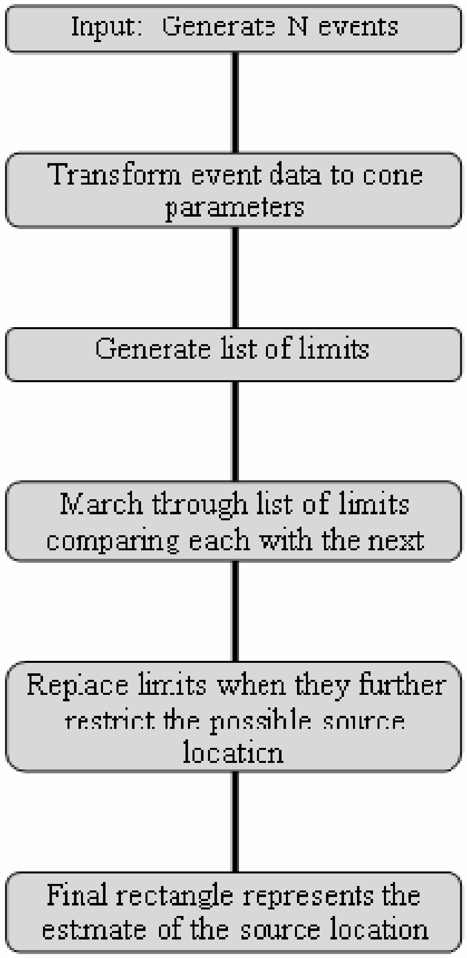


Figure 6. Process describing the circumscribed rectangle technique.

CHAPTER 4

PARAMETERS OF INTEREST IN THE EVALUATION OF THE PERFORMANCE OF THE ALGORITHM

In both methods, errors in the event data limit the accuracy with which the source can be localized. The detector's energy and spatial resolution, Doppler broadening, and other physical factors result in finite angular resolution and limit the detector's field of view (Du et al., 2001; Smith et al, 2006). Also, accuracy limits may be inherent in the algorithms. Investigating these effects on the performance of the directional algorithm was part of this project. For both techniques the following parameters have been evaluated:

- the effect of the distance to the source on direction reconstruction;
- angular resolution as a function of source location;
- useful field of view (that is, the boundary of the region where angular resolution is less than 10º); and
- convergence rates (or the effect of the number of events on the direction reconstruction).

Additionally, the following parameters were evaluated for the modified Compton telescope technique:

- the point-spread functions (PSF) of the resulting filtered images;
- the PSF of the unfiltered image produced by the simple overlapping of the backprojected circles; and

- the signal-to-noise ratio (SNR).

These evaluations were tested for three different cases.

1. Perfect (geometric) data were generated not taking into account detector physics, spatial resolution, energy resolution, or the preferred distribution of scatter angles as a function of source energy.
2. Data from a Monte Carlo simulation of the detector was used to look at the effects on an ideal detector. In this case, the ideal detector had 0.0-mm spatial resolution and only Doppler broadening affected the energy resolution used to generate the test data.
3. The Monte Carlo simulation data were modified to represent a more realistic detector response, including finite spatial resolution and energy resolution.

All evaluations were made at (0°, 0°) and 100 cm away from the front detector face; in some cases, evaluations were also made in the first quadrant with source angles between 0° and 90° to assess the effects of source energy and location on the direction calculation.

Finite spatial resolution was added by binning the interaction location data into discrete pixels. Energy-dependent energy resolution was added by sampling a Gaussian distribution of the desired width. The following code adds energy resolution effects to the interaction energy data.

```
res_factor=E_ref*percent_error/2.35*sqrt(1./E_ref)*    $
        randomn(seed,n,2)
en=(sqrt(1000*ev[i].e1)*res_factor[j,0])+1000*ev[i].e1
```

Here, *E_ref* is the reference energy where energy resolution is defined (in this case, *E_ref* = 0.511 MeV); *percent_error* is the error that is added to the energy measurement (0%, 3%, or 6% FWHM for the current discussion); *randomn(seed,n,2)* produces an n by 2 array of random numbers evenly distributed between 0 and 1; *res_factor* is therefore an $n \times 2$ array of numbers to be added to the energy measurement; *ev* is an array of events, so *ev[i].e1* is the energy of the first scatter of the ith event; and *en* is the modified energy measurement. Tailing due to charge trapping that commonly occurs in CZT detectors is not modeled.

4.1 Geometric Data

The perfect (geometric) data was generated in IDL using the following code.

```
      ; randomly generate sites of scattering interaction, in
      ; centimeters from scatter-detector center
      ; all scatterings occur in scatter-detector plane
events[*,0].location.z = scatdet.p
events[*,0].location.x = 0.1 * scatdet.h.d / scatdet.h.npix $
      * FIX( scatdet.h.npix * RANDOMU( seedx, num_ev ) +    $
      scatdet.h.pmin )
events[*,0].location.y = 0.1 * scatdet.v.d / scatdet.v.npix $
      * FIX( scatdet.v.npix * RANDOMU( seedy, num_ev ) +    $
      scatdet.v.pmin )
      ; randomly generate direction angles to the absorption
      ; interaction
tan_alpha = tan( absdet.h.d / absdet.h.npix                 $
      * FIX( absdet.h.npix * RANDOMU( seeda, num_ev ) +     $
      absdet.h.pmin ) )
tan_beta = tan( absdet.v.d / absdet.v.npix                  $
      * FIX( absdet.v.npix * RANDOMU( seedb, num_ev ) +     $
      absdet.v.pmin ) )
      ; now convert these to x,y,z coordinates on the half-sphere
events[*,1].location.z = -1 * absdet.p / SQRT( tan_alpha^2 + $
      tan_beta^2 + 1 )
events[*,1].location.x = events[*,1].location.z * tan_alpha
events[*,1].location.y = events[*,1].location.z * tan_beta
      ; now calc the scattering energy
      ; First, simple geometry to get the scattering angle:
      ;     a = distance from source to scatter site
      ;     b = distance from scatter site to 2nd interaction
      ;     c = distance from source to 2nd interaction
      ;     theta is the scattering angle
      ; Thus, c^2 = a^2 + b^2 - 2 * a * b * cos( PI - theta )
      ;     or  c^2 = a^2 + b^2 + 2 * a * b * cos( theta )
asquared  = ( src.location.x - events[*,0].location.x )^2
asquared  = asquared+(src.location.y - events[*,0].location.y )^2
asquared  = asquared+(src.location.z - events[*,0].location.z )^2
bsquared  = ( events[*,1].location.x - events[*,0].location.x )^2
bsquared  = bsquared + ( events[*,1].location.y -           $
      events[*,0].location.y )^2
```

```
bsquared  = bsquared + ( events[*,1].location.z -          $
      events[*,0].location.z )^2
csquared  = ( src.location.x - events[*,1].location.x )^2
csquared  = csquared+(src.location.y - events[*,1].location.y )^2
csquared  = csquared+(src.location.z - events[*,1].location.z )^2
costheta  = csquared - asquared - bsquared
costheta  = costheta / (2 * SQRT( asquared ) * SQRT( bsquared ))
      ; Second, Compton kinematics to get the scattered photon
      ; energy
events[*,1].energy = src.energy/(1+src.energy/m_e*(1-costheta ))
      ; The remaining energy was deposited during the scattering
      ; interaction
events[*,0].energy = src.energy - events[*,1].energy
```

This is considered perfect data because the positions are randomly generated and then the scattering energy is calculated so that it corresponds exactly according to Compton kinematics to match the scattering angle. (See Appendix A for a full discussion of the geometric data's production.)

4.2 Monte Carlo Simulation Data

The Monte Carlo simulation data were generated to test the algorithms' performance with more realistic data. GEANT4 was used for simulating the ECRD detector (Smith et al, 2006). The base model was a 6-sided box with one CZT detector module on each end and two modules per side wall. For the purposes of this paper, the simulations, analysis, and discussion assume the module on one end is the primary scatter detector and all other modules are the secondary absorption detectors. In practice, any module could be the primary scatter detector for a given incident gamma ray, with any one of the remaining modules acting as the secondary absorption detector. The primary module operates in coincidence with the other modules. The simulations allowed variations of detector dimensions, pixel size, energy resolution, and source location and energies. The coincident hits were tallied and analyzed for each simulation run. Overall detection efficiency (total number of interactions normalized to total incident flux) and localizing detection efficiency (number of usable coincidence event pairs normalized to incident flux) were tallied. In no case has exact knowledge of the source energy been assumed in the analysis. The pair of energies deposited in the coincident events was used in the analysis. For some parts of the analysis we have assumed the source energy can be determined by energy spectral analysis or some independent means, to evaluate the effect of energy windowing (of the total deposited energy in coincident events) on the performance analysis (Smith, 2006).

CHAPTER 5

EVALUATION OF THE MODIFIED COMPTON TELESCOPE TECHNIQUE

The modified Compton telescope technique reconstructs a projection image of the source distribution; Figure 7 shows a representative image of a point source created by this technique. This image was generated with 1000 events from a simulated point source located 100 cm directly in front of the detector using the geometric data. In all reconstructed images, the θ/longitude coordinate (from -90º to +90º) is horizontal and the φ/latitude coordinate (from -90º to +90º) is vertical on the sphere.

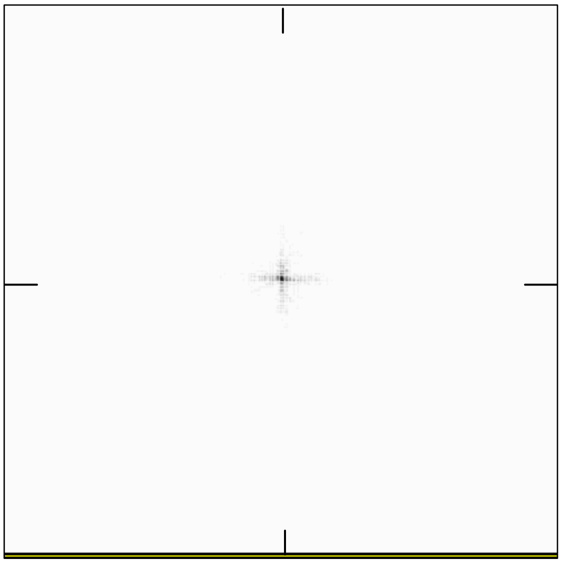


Figure 7. Representative image of a point source at (0º, 0º) recreated by the modified Compton telescope technique.

5.1 Evaluation Using Geometric Data

For a full description of how the geometric data were generated, see Appendix A. Evaluation using geometric data, though not realistic, is important because it shows whether the basic principles of the reconstruction techniques are valid without interference from "real-world" considerations that are taken into account in the Monte Carlo simulations, and eventually in the actual detector system. The source energy need not be specified with the geometric data because the scattering angles are chosen randomly, uniformly over the hemisphere, with no regard to the Klein-Nishina cross-section. Also, in this case, all events comprise a Compton scatter in the first detector and photoelectric absorption in a second detector.

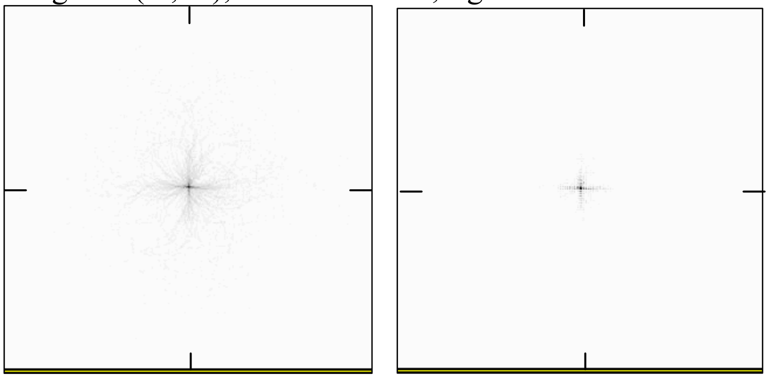
5.1.1 Sample Images

Although the task for the directional algorithm is not an imaging task, the modified Compton telescope technique is inherently an imaging algorithm. Figure 8 compares unfiltered backprojected images to filtered images generated with the modified Compton telescope technique for simulated point sources located along the diagonal in (θ,φ). Each image represents 1000 events generated with the geometric event simulation code.

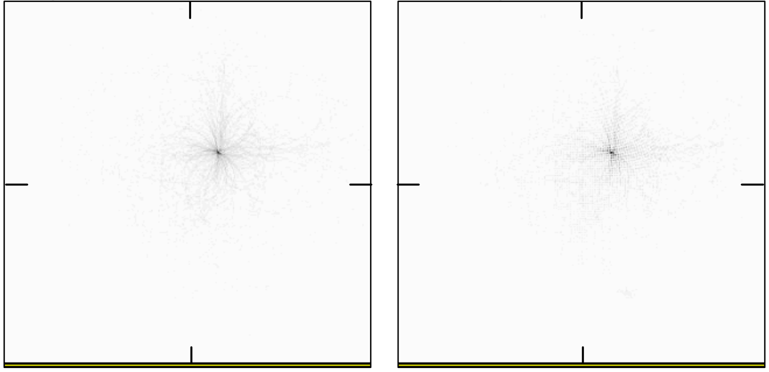
5.1.2 Useful Field of View

The useful field of view was defined as the portion of the field of view where the absolute error in the direction localization estimate is less than 10°. Direction location estimates were made for idealized sources located at 15° intervals throughout the first quadrant and then compared to the actual source positions. The other three quadrants of the field of view are equivalent because the detector is rotationally symmetric. Figure 9 is a contour plot of the measured direction localization error defined as the absolute difference between the direction to the estimated source position and the direction to the actual position; the useful field of view is the area within the 10° error contour.

(a) images at (0°, 0°); left: unfiltered, right: filtered



(b) images at (15°, 15°); left: unfiltered, right: filtered



(c) images at (30°, 30°); left: unfiltered, right: filtered

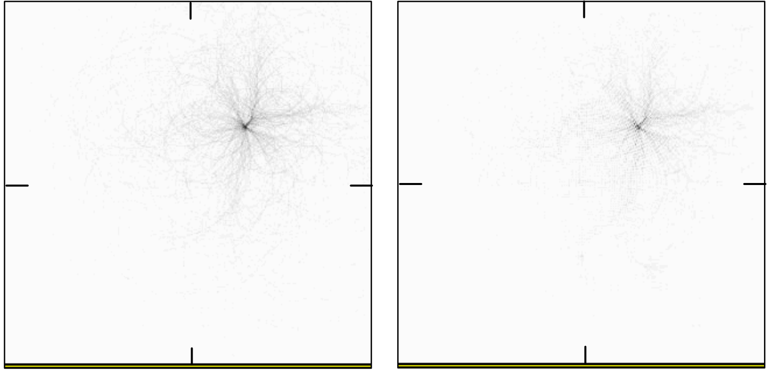
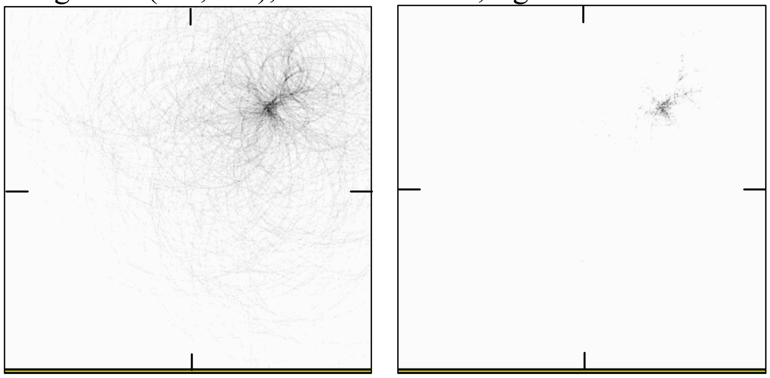
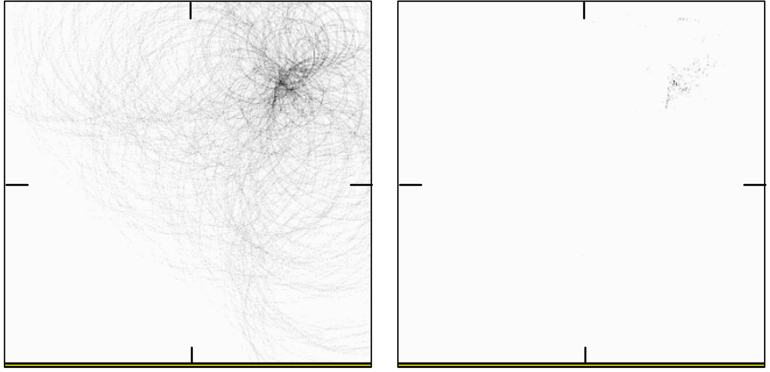


Figure 8 (a-g). Images of idealized point sources located along the radial line from (0°, 0°) to (89°, 89°). At each source location, the unfiltered backprojected image is shown on the left and the filtered image is shown on the right. Notice that the farther the point source is located from the center of the field of view, the less like a point the reconstructed image looks.

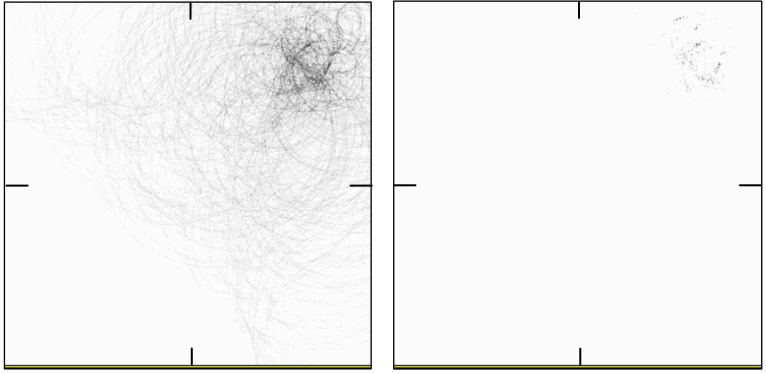
(d) images- at (45º, 45º); left: unfiltered, right: filtered



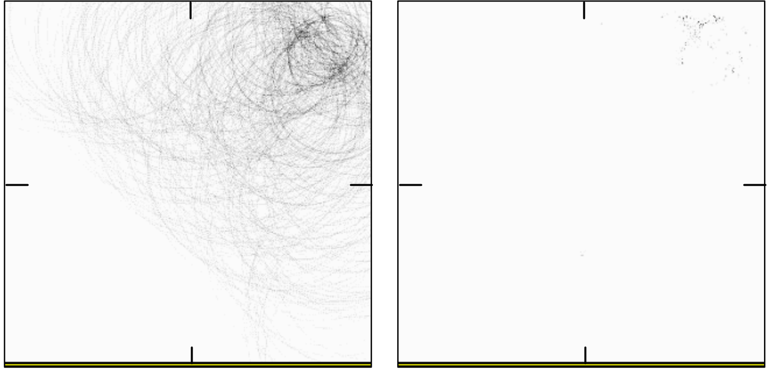
(e) images at (60º, 60º); left: unfiltered, right: filtered



(f) images at (75º, 75º); left: unfiltered, right: filtered



(g) images at (89º, 89º); left: unfiltered, right: filtered



(Fig. 8, continued)

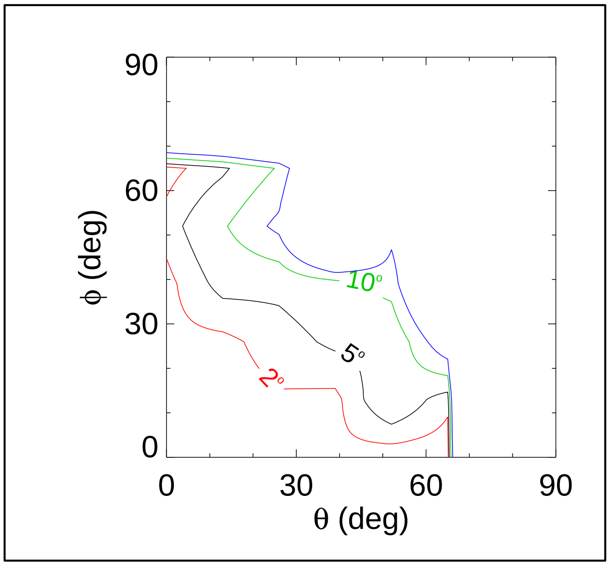


Figure 9. Contour plot of direction localization error in the first quadrant. The useful field of view is the region extending from the origin to the (green) 10º contour line.

5.1.3 Angular Resolution

Angular resolution measures the spread of a point source due to the image reconstruction algorithm. The angular resolution is represented as the full-width at half-maximum (FWHM) of the point spread function (PSF). The PSFs of point source images generated using the modified Compton telescope technique were evaluated at various locations throughout the first quadrant. The FWHM and signal-to-noise ratio (SNR) were measured for each point source image. The PSFs of the filtered images were also compared to the PSFs of the unfiltered images to demonstrate the effect of Fourier filtering on the image data. Figure 10 shows horizontal profiles through the maximum value of each point source image. Horizontal profiles may differ from other profiles (vertical or radial, for example). These horizontal profiles have been chosen because they nonetheless demonstrate the effect of the filtering process. The signal to noise ratio (SNR) was defined as the ratio of the peak height to the background level in the image.

SNR was measured at several locations in the first quadrant. Table 1 illustrates the results. The FWHMs and SNRs of these distributions have been defined where a clear peak is present and are summarized in the following table. The PSFs for (0º, 60º) and (0º, 75º) are too spread out and exhibit no clear peak so the FWHMs and SNRs are not defined for these points. Figure 11 represents PSFs positioned radially out from (15º, 15º) to (75º, 75º). Note that in Figure 11(b) there is a peak at 90º in the unfiltered image which is zeroed out in the final image. The unfiltered PSFs in figure 11 were so blurred that a definite peak could not be identified in most cases. The FWHM could not be determined for these; SNR was defined using the maximum value of the filtered image. FWHMs (where possible) and SNRs are summarized in Table 2.

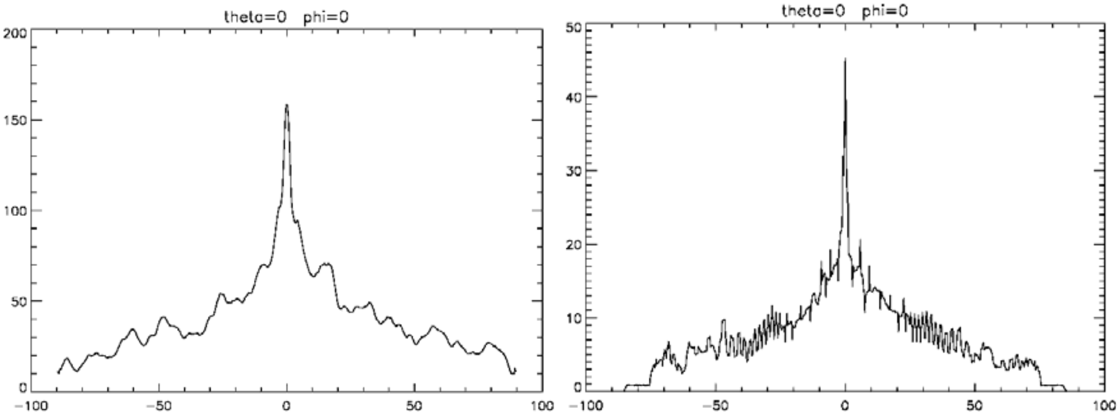
Table 1. Summary of the FWHMs and SNRs of the PSFs defined along the phi-axis.

	FWHM unfiltered	FWHM filtered	SNR filtered
(0º, 0º)	10º	2º	14.0
(0º, 15º)	10º	5º	2.5
(0º, 30º)	30º	8º	6.0
(0º, 45º)	55º	30º	7.5

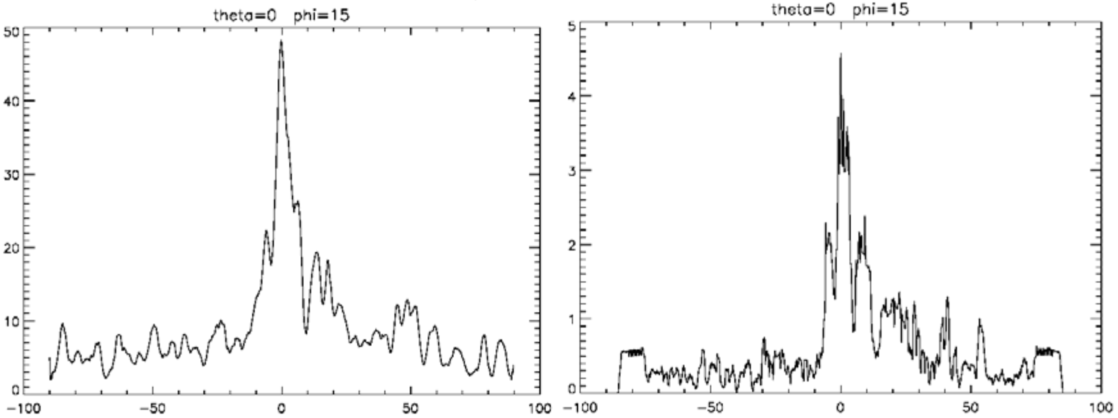
Table 2. Summary of the PSFs defined radially outward from the origin. Xs mark points where the PSF was too spread out to define a peak and therefore the FWHM is undefined.

	FWHM unfiltered	FWHM filtered	SNR filtered
(15º, 15º)	10º	2º	3.0
(30º, 30º)	X	10º	3.0
(45º, 45º)	X	18º	4.0
(60º, 60º)	X	8º	7.0
(75º, 75º)	X	X	3.0

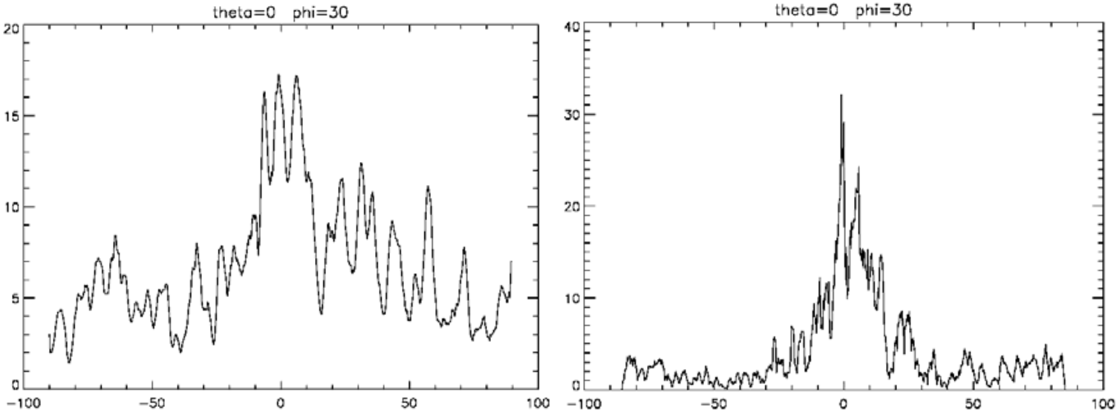
(a) PSF at (0º, 0º); left: unfiltered, right: filtered



(b) PSF at (0º, 15º); left: unfiltered, right: filtered

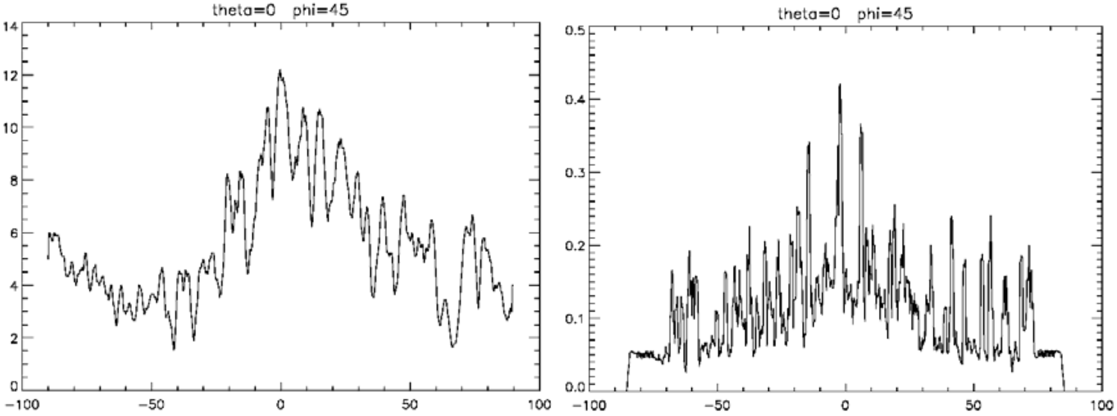


(c) PSF at (0º, 30º); left: unfiltered, right: filtered

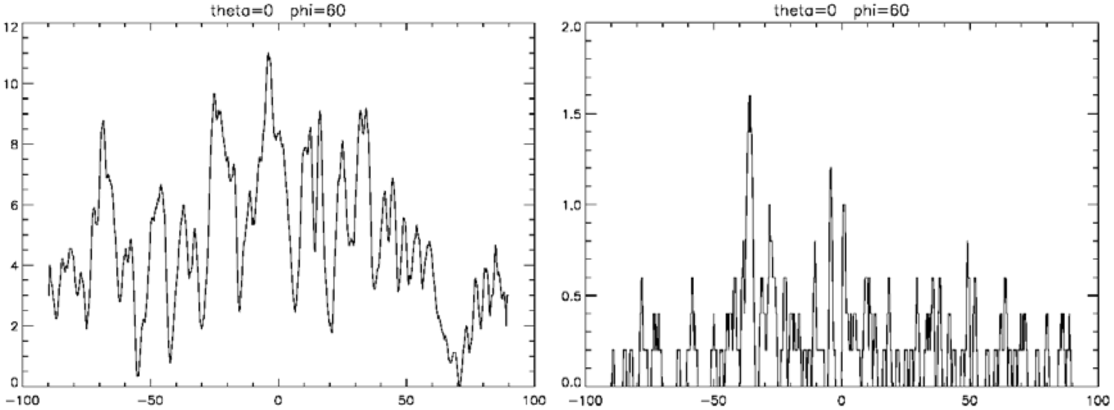


Figures 10 (a-f). PSFs along the axis from (0º, 0º) to (0º, 75º) in 15º increments in the phi-variable. The horizontal axis is the theta-variable and the vertical axis is relative magnitude.

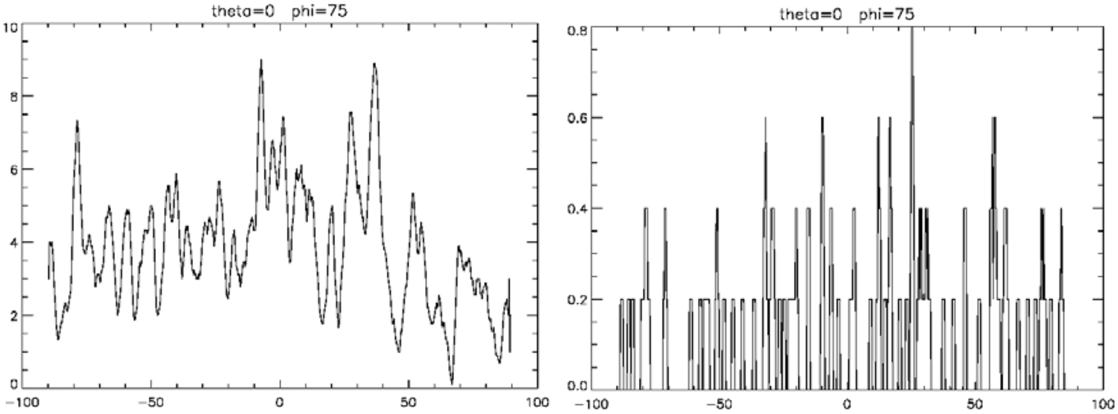
(d) PSF at (0º, 45º); left: unfiltered, right: filtered



(e) PSF at (0º, 60º); left: unfiltered, right: filtered

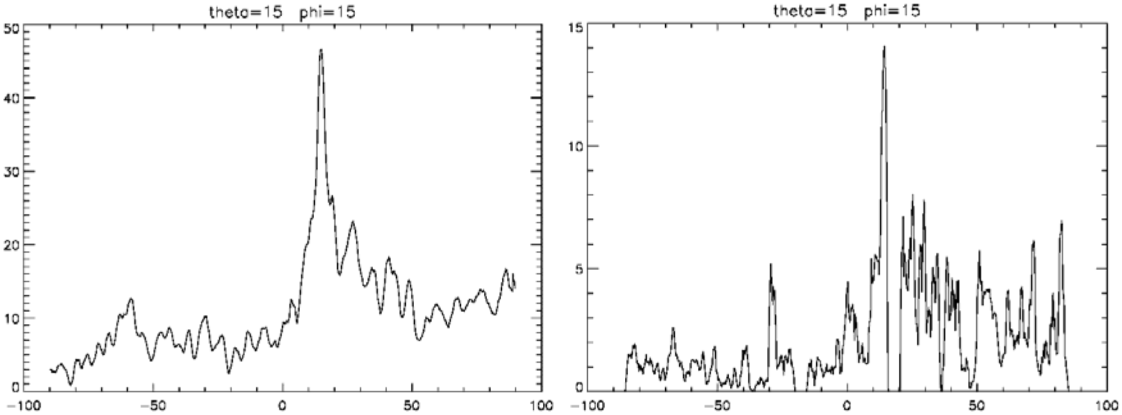


(f) PSF at (0º, 75º); left: unfiltered, right: filtered

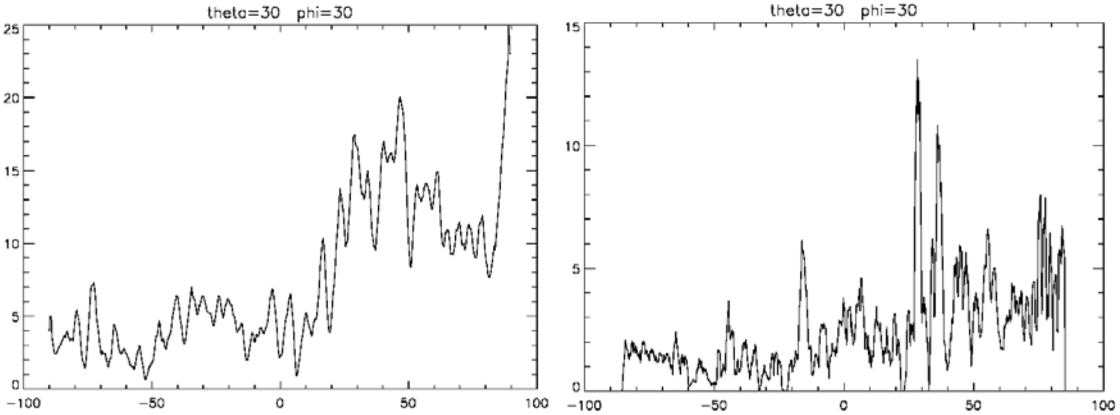


(Fig. 10, continued)

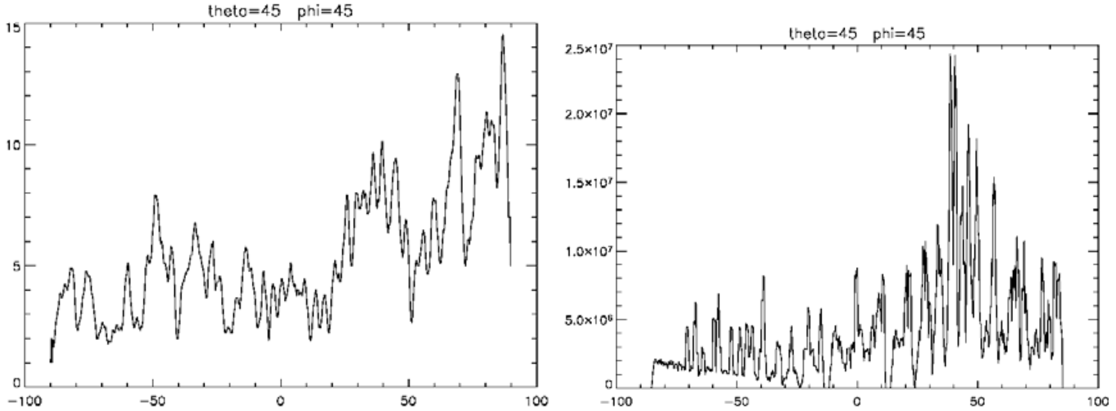
(a) PSF at (15º, 15º); left: unfiltered, right: filtered



(b) PSF at (30º, 30º); left: unfiltered, right: filtered

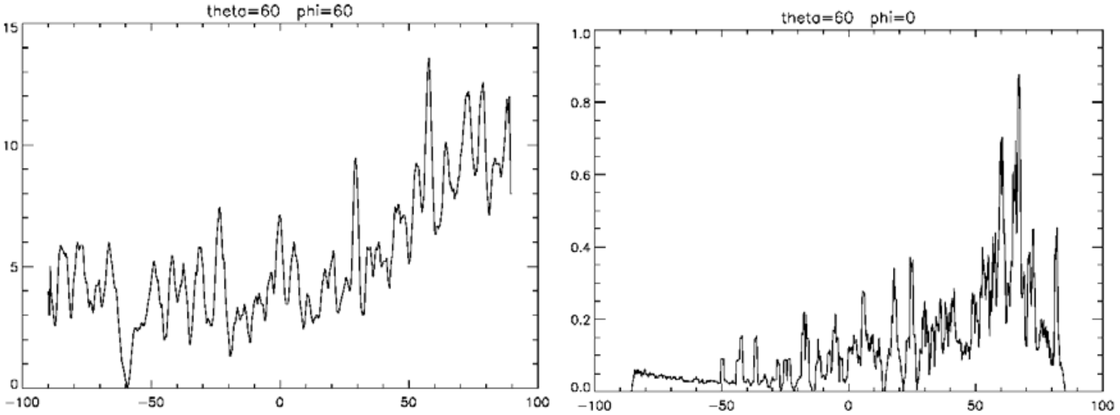


(c) PSF at (45º, 45º); left: unfiltered, right: filtered

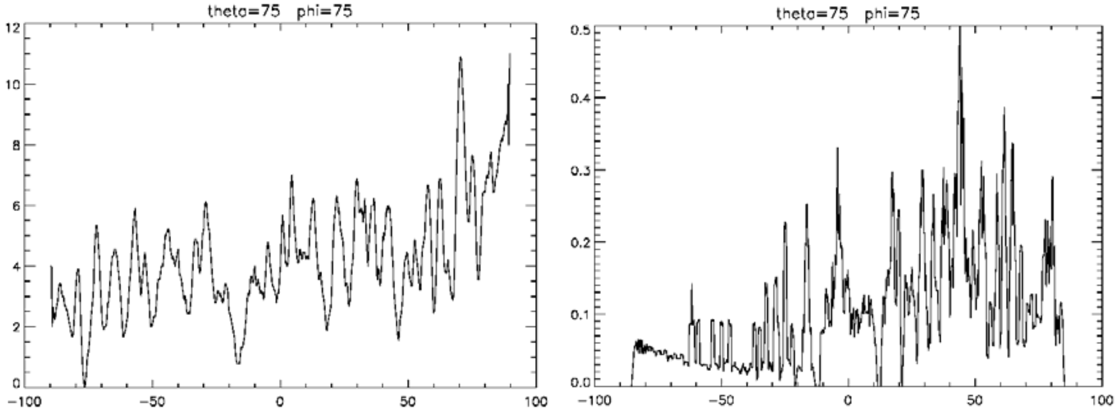


Figures 11 (a-e). PSFs along the radial line from (15º, 15º) to (75º, 75º). The horizontal axis is the theta-variable and the vertical axis is relative magnitude.

(d) PSF at (60º, 60º); left: unfiltered, right: filtered



(e) PSF at (75º, 75º); left: unfiltered, right: filtered



(Fig. 11, continued)

5.1.4 Rate of Convergence

Convergence rate is important for understanding how many events are needed to create an adequate source direction reconstruction. For the modified Compton telescope technique, the convergence rate was calculated from the maximum value of the image versus the number of cones (event pairs) used to generate the image. Convergence rate was measured at (0º, 0º) with geometric data (Figure 12).

5.1.5 Effect of Distance to Source

The modified Compton telescope technique assumes that the source lies on the reconstruction sphere. If the sphere radius is not the same as the source-to-detector distance, the reconstructed image may exhibit additional blurring due to parallax. In the previous sections, a 1-m sphere was used that matched the 1-m distance to the source. Figure 13 illustrates the effect of different source-to-detector distances, relative to a 1-m radius reconstruction sphere. Figure 7 displays an image which was reconstructed with the sphere's radius equal to the source-to-detector distance. Notice that the 50-cm case and the 500-cm case both look similar to the case shown in Figure 7.

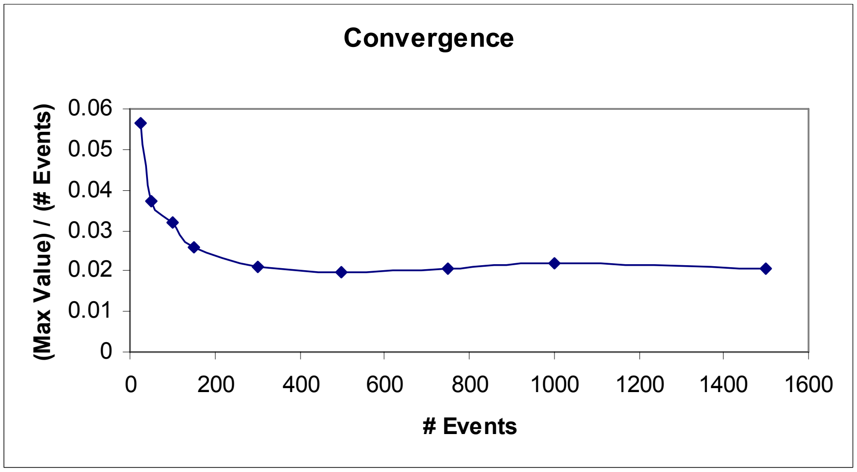


Figure 12. Convergence rate was defined by the number of events needed for the maximum value of the image normalized by the number of events to approach a steady-state value.

5.2 Evaluation Using Monte Carlo Simulated Data

A GEANT4 Monte Carlo simulation of the proposed ECRD geometry was developed by colleagues (further discussed in Appendix B), separate from the work described in this thesis. To evaluate the modified Compton telescope technique, simulation data was generated with perfect spatial resolution and energy resolution that only accounted for Doppler broadening of the energy spectrum. The simulations included accurate Compton scattering physics so that the source energy affects the distribution of scattering angles. Also, all physical types of interactions can occur, so it is possible to have multiple Compton scattering in the detector or to have photons that interact in one of the secondary detectors without first interacting in the primary detector. The simulation models discrete pixels to account for detector pixellation. Charge collection effects on energy resolution are not modeled; energy-resolution effects are added to the data after the simulation runs. These considerations thus make the detector simulation more realistic than the previous geometric data. Unless otherwise noted, the simulations were run for a source energy of 0.511 MeV.

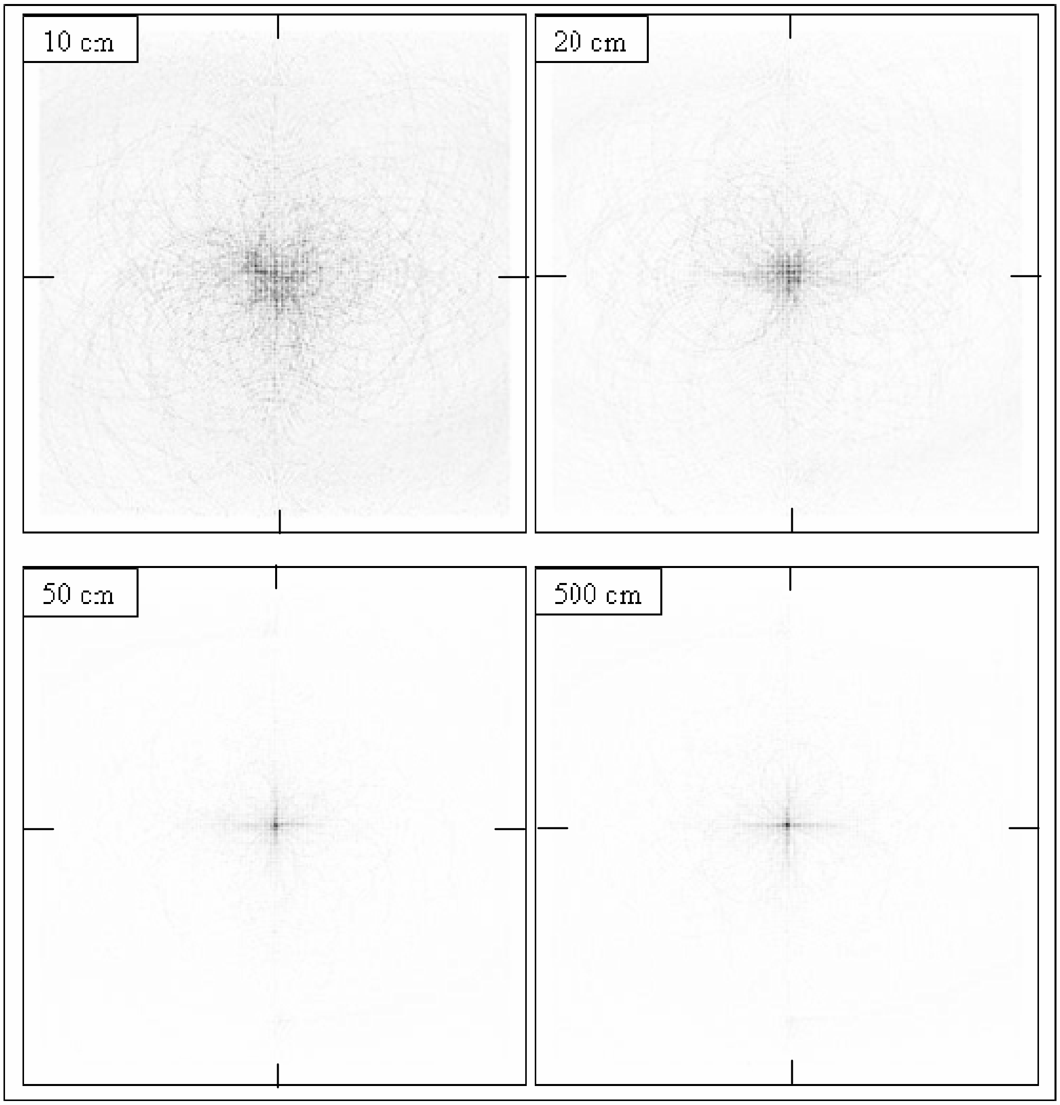


Figure 13. These images have been reconstructed with the source at varying distances from the detector. The source-to-detector distance is displayed in the upper left corner of each image.

5.2.1 Useful Field of View

The useful field of view was assessed using the Monte Carlo simulation data. Figure 14 shows contour plots of the measured direction localization error with varying spatial resolution; the useful field of view (as before) is bounded by the 10º error contour. The contour plot for the perfect energy and spatial resolution (Figure 14 on the left) closely matches the contour plot generated by the geometric data (Figure 9).

The effect of energy resolution was tested for three cases, 0%, 3%, and 6% FWHM resolution at 0.511 MeV. Energy resolution was simulated as a Gaussian distribution. These results are displayed in Figure 15. Finally, energy and spatial resolution are both modeled at the same time. Figure 16 represents the useful field of view with 1-mm spatial resolution and 3% energy resolution at 0.511 MeV. These values for the resolutions were chosen because they represent the most probable values a real detector would have.

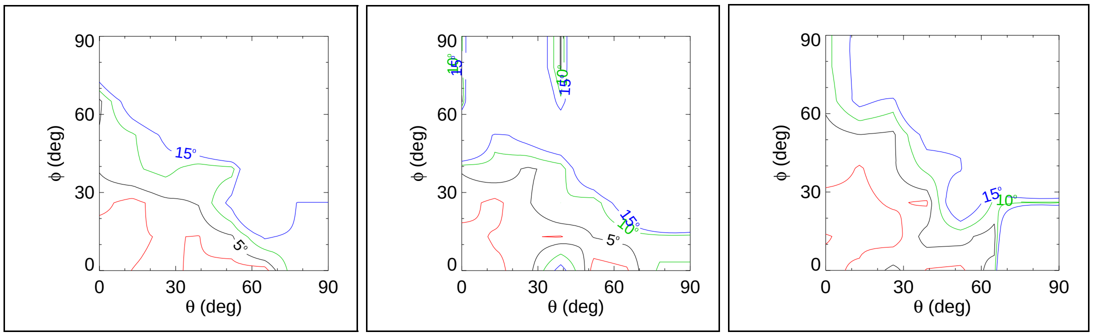


Figure 14. Contour plot of the direction localization error in the first quadrant for Monte Carlo data with perfect energy resolution. The useful field of view is the region extending from the origin to the (green) 10º contour line. The plots represent (left) perfect spatial resolution, (center) 1-mm spatial resolution, and (right) 2-mm spatial resolution.

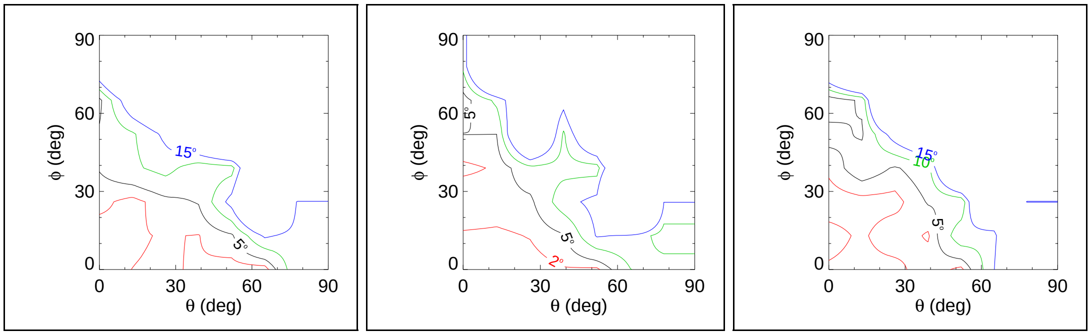


Figure 15. Contour plot of the direction localization error in the first quadrant for Monte Carlo data with perfect spatial resolution. The useful field of view is the region extending from the origin to the (green) 10º contour line. The plots represent (left) perfect energy resolution, (center) 3% energy resolution at 0.511 MeV, and (right) 6% energy resolution at 0.511 MeV.

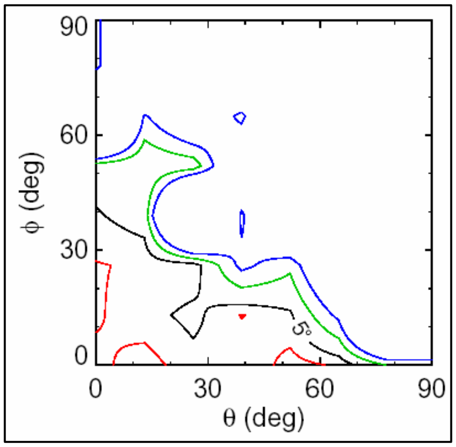


Figure 16. Contour plot of the direction localization error in the first quadrant for Monte Carlo data with 1-mm spatial resolution and 3% energy resolution at 0.511 MeV. The useful field of view is the region extending from the origin to the (green) 10º contour line.

CHAPTER 6

EVALUATION OF THE CIRCUMSCRIBED RECTANGLE TECHNIQUE

The circumscribed rectangles technique is not, by its nature, an image reconstruction technique. Figure 17 illustrates this method, locating a point source at 100 cm in front of the detector. Only 25 events were used to generate this figure

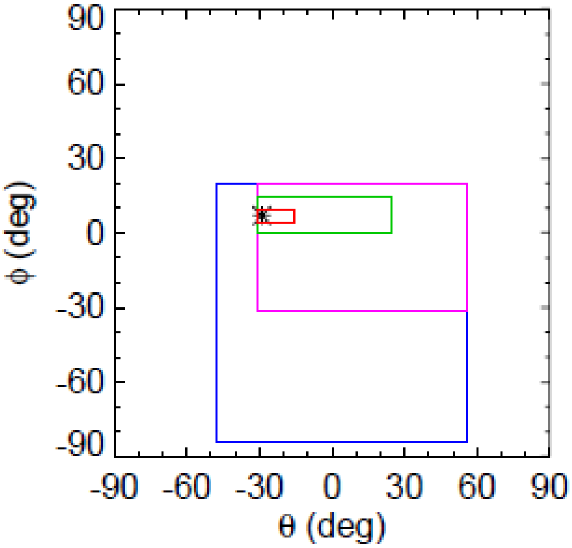


Figure 17. Illustration of the circumscribed rectangles technique converging on a source at (-29°,7°). The blue, purple, green and red boxes represent the localization based on 1, 5, 10, and 25 cones; the asterisk represents the true source position.

6.1 Evaluation Using Geometric Data

Geometric data were generated (see Chapter 4 or Appendix A) in an analogous manner to the data used in Chapter 5. Again, this evaluation is important because, though it does not take into account "real-world" physics, it shows whether or not the basic principles of the reconstruction technique are valid.

6.1.1 Useful Field of View

The useful field of view was defined as the portion of the field of view where the absolute error in the direction localization estimate is less than 10º. This measurement was made by taking estimates every 15º using the geometric data and comparing them to the actual source position. This is done only over the first quadrant, as the other three quadrants follow from symmetry. The angular error is illustrated in Figure 18.

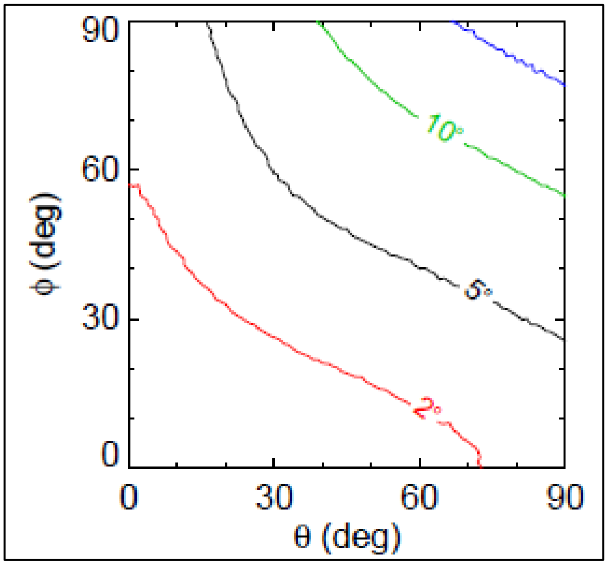


Figure 18. Contour plot of direction localization error in the first quadrant. The useful field of view is the region extending from the origin to the (green) 10º contour line.

6.1.2 Angular Resolution

Angular resolution for the circumscribed rectangle technique was defined as the area of the final circumscribed rectangle that the method converges to. Figure 19 illustrates convergence of the method showing that the method converges to a rectangle that is approximately 100 square-pixels corresponding to 6.25 degrees^2 (or 2.5° x 2.5°) because each pixel is defined as 0.25° × 0.25°.

6.1.3 Rate of Convergence

The convergence rate is important for understanding how many events are needed to create an adequate source direction reconstruction. For the circumscribed rectangle technique, the convergence rate was defined as the area of the circumscribed box versus the number of cones (event pairs) required to generate the box. Convergence was measured for a point source at (0°, 0°) with geometric data (Figure 19).

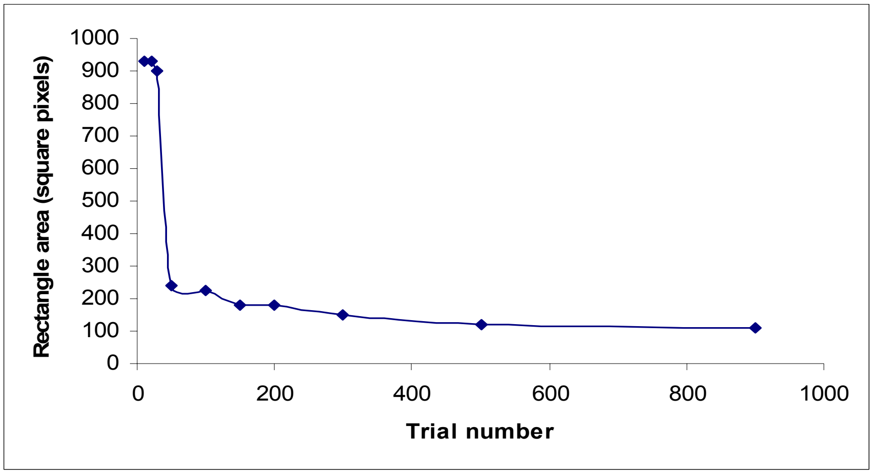


Figure 19. The convergence rate has been defined by the number of events needed for the area of the circumscribed rectangle to become close to constant. The area of the circumscribed rectangle is defined in terms of square pixels; each pixel is 0.25° squared in area.

6.2 Evaluation Using Monte Carlo Simulated Data

The same simulation data were used for this evaluation as those which were used in section 5.2 for the modified Compton telescope technique. Unless otherwise noted, all of the following simulations were run using a source energy of 0.511 MeV.

6.2.1 Useful Field of View

Without adding spatial or energy resolution to the Monte Carlo simulation of the detector, the effect of the physics added by the simulation on the useful field of view was quantified for perfect energy resolution and spatial resolution. The useful field of view in the first quadrant is shown in Figure 20. The useful field of view for 1-mm spatial resolution looks almost exactly the same as that for the ideal case. For 2-mm spatial resolution, the field of view again looks similar to the others except that the error at the origin rises above 10º. For both of these cases, the useful field of view extends out to only (15º, 15º) radially and about twice that along the axes.

The effect of energy resolution has been tested for three cases, 0%, 3%, and 6% FWHM resolution at 0.511 MeV. Energy resolution was simulated as a Gaussian distribution (as in Section 5.2.1). These results are displayed in Figure 21. For the 3% case, the field of view was again similar to the previous cases. Adding 6% energy resolution, though, has a substantial effect on the field of view. The configuration of 3% energy resolution and 1-mm spatial resolution is displayed (in Figure 22.)

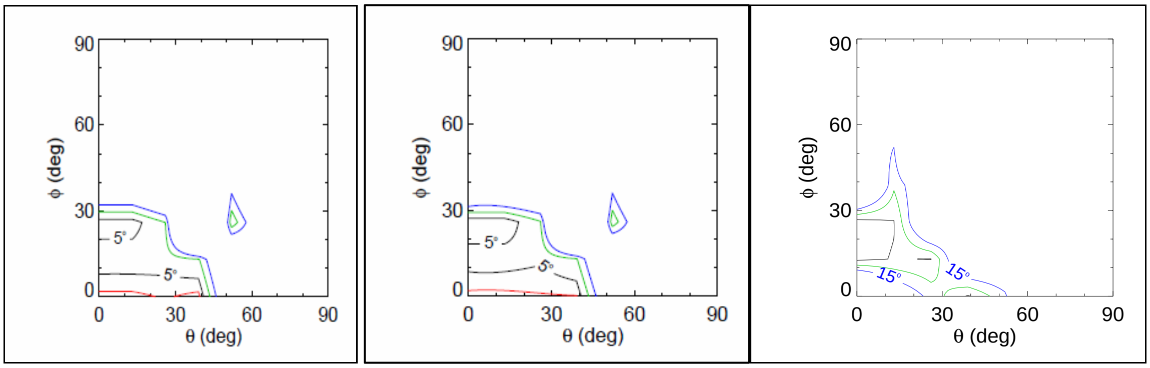


Figure 20. Contour plot of the direction localization error in the first quadrant for Monte Carlo data with perfect energy resolution. The useful field of view is the region extending from the origin to the (green) 10º contour line. The plots represent (left) perfect spatial resolution, (center) 1-mm spatial resolution, and (right) 2-mm spatial resolution.

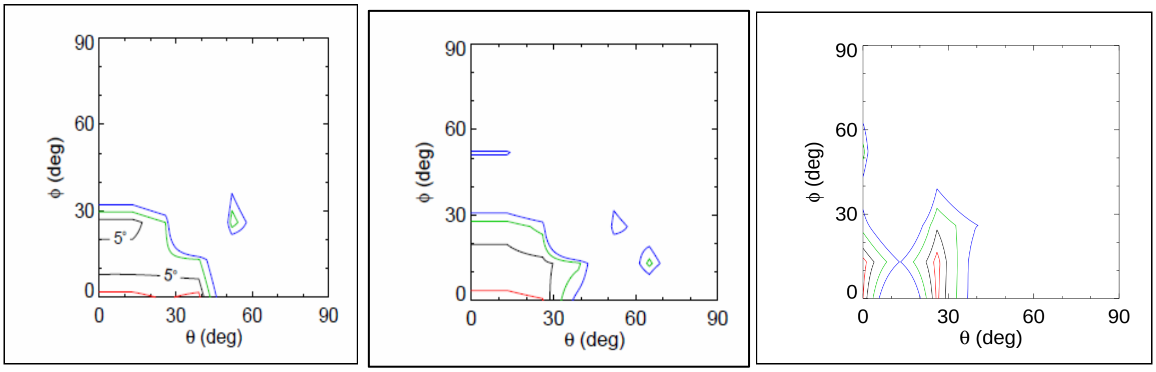


Figure 21. Contour plot of the direction localization error in the first quadrant for Monte Carlo data with perfect spatial resolution. The useful field of view is the region extending from the origin to the (green) 10º contour line. The plots represent (left) perfect energy resolution, (center) 3% energy resolution at 0.511 MeV, and (right) 6% energy resolution at 0.511 MeV.

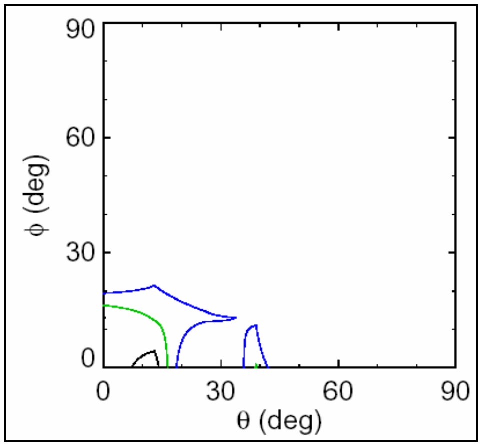


Figure 22. Contour plot of the direction localization error in the first quadrant for Monte Carlo data with 1-mm spatial resolution and 3% energy resolution at 0.511 MeV. The useful field of view is the region extending from the origin to the (green) 10º contour line.

CHAPTER 7

DISCUSSION AND CONCLUSIONS

The data presented here were evaluated in the first quadrant only, because the performance of the algorithm will follow the same patterns over the whole hemisphere (in the other three quadrants.) During the algorithm development, individual components of code were tested for source locations with both positive and negative values of theta and phi, but the data presented here for one quadrant of the hemisphere is sufficient because of the symmetry of the detector system.

7.1 Modified Compton Telescope Technique

7.1.1 Results Using Geometric Input Data

The size of the useful field of view defined in terms of absolute error was demonstrated in Figure 9; the error remains below 10º out to approximately (45º, 45º) radially and past 60º along the axes. This did not meet our expectations that the algorithm would be able to reconstruct point sources over the entire field of view. This behavior is due to the significant blurring that occurs the farther from (0º, 0º) the source gets. Blurring, in part, is due to the deviations from circularity of the backprojected cones, due to the assumption that the cones all intersect the sphere as circles. We ignore the fact that the vertices are not all at the center of the sphere. This effect becomes more pronounced the further away from the forward direction the source lies. Also, for sources that lie at large angles most of the recorded events will be the result of large-angle scatter and are therefore prone to greater error (Smith, 2006).

Angular resolution was demonstrated for source positions along the theta-axis. The FWHMs of the PSFs get larger the further away from straight in front of the detector the point source is. It is clear from the images in Figure 8 that the point source becomes more spread out the further away from (0º, 0º) the point source gets. Most of the data at the more extreme angles come from scattering angles that are also extreme. Because of this, the circles do not overlap in a point but rather congregate around a point. For some of the distributions that are far from (0º, 0º), the reconstructed images are so spread out that no discernible peak is found. Although the image gets blurred out by the algorithm in reconstruction, the algorithm can still pick a maximum value which is approximately the center of the smeared-out area. Alternatively, the algorithm calculates a centroid of the largest values if a single maximum point is not found. Within the useful field of view, though, the FWHM of the point spread function remains below 10º after filtering.

These results show that the method is useful, although the field of view is not as large as desired. Also, the calculation time of the process could be speeded up by removing the filtering process. This method, then, would only consist of backprojecting the cones. If angular resolution is not as much of a concern, we can do this because the filtering only makes the images look better visually and decreases the angular resolution a bit. The peak that the algorithm picks out will not change after filtering.

Convergence, measured as the number of events required to significantly decrease the rate of change of the maximum value of the image normalized by the number of events, occurs in 200-300 events. If we can expect a source strength adequate to generate one good event per second, then this corresponds to a collection time of around 5 minutes. This may not be feasible in field applications (for homeland security) but longer collection times may be acceptable in an intraoperative setting.

7.1.2 Results Using Monte Carlo Simulated Data

Using the simulation without spatial or energy resolution yields a field of view that is similar to the one generated with the geometric data both along the radial line and along the axes. This suggests that the GEANT4 simulation (although adding Doppler broadening, detector geometry, and the physics of the photon interactions that are likely to take place) matches well with the field of view generated without regard to these processes and validates that the reconstruction process remains sound as it deals with slightly more realistic data.

The plots generated with spatial resolution and energy resolution added only show a slight decrease in the size of the effective fields of view. This demonstrates that the addition of unsound data has little effect on this method. This behavior is due to the fact that each good event adds to the peak value, while events only slightly add to the background noise.

Convergence and angular resolution were not tested with Monte Carlo simulated data because they are expected to follow the same pattern as that of the geometric data, being properties of the method rather than the data.

7.2 Circumscribed Rectangle Technique

7.2.1 Results Using Geometric Input Data

The useful field of view was defined in terms of the absolute error of the calculated position versus the actual position. In this case, the area within which the absolute error is under 10º covers more of the total field of view extending further towards (90º, 90º) than the modified Compton telescope technique does. Also, the error is under 15º for almost the entire field of view (Figure 18.) This is a very promising result. The circumscribed rectangle method was chosen for its speed and simplicity; for the geometric data, its performance appears to be better than the modified Compton telescope technique.

For the circumscribed rectangles technique, the rate of convergence is much quicker than the modified Compton telescope technique, leveling off around 50 events. Angular resolution is basically a meaningless quantity for this algorithm. The size of the final circumscribed box is the same over the entire field of view, given enough events for convergence, because it is defined by the error factors included in the algorithm (e.g., in the data presented in Chapter 6 a padding of 2.5º (10 pixels) was added to the limits defined for the circumscribed box to account for rounding errors in the calculations and errors in the data resulting in a minimum circumscribed box of 2.5º x 2.5º). In other words, this method would always converge on a point if the data had no errors and no padding was used. The more important metric is the angular error described in the previous paragraph.

7.2.2 Results Using Monte Carlo Simulated Data

With ideal Monte Carlo simulated data (that is, using the simulation without adding spatial or energy resolution), the field of view for the circumscribed rectangle technique is greatly reduced compared to the geometric data. In this case, adding the realistic physics of the simulation noticeably affects the direction reconstruction method. The circumscribed rectangle method is very sensitive to spurious events, especially when they occur near the beginning of the list of data that is being analyzed.

Energy resolution was added to the simulated data after the data had been generated. Gaussian noise (3% and 6% FWHM at 0.511 MeV) was added to simulate this effect. At 6% energy resolution, the circumscribed rectangle technique has limited utility as it is implemented now. There were only a few areas where the algorithm came close to the correct source position. Overall, detector energy resolution had a greater effect on the size of the useful field of view than the detector pixel size. This is the same pattern that was seen for the Compton telescope technique.

7.3 Conclusions for the Modified Compton Telescope Technique

Hypothesis 1. *Accuracy*: For a stationary source, the algorithm provides source localization with angular error of no more than 10º out to near ±90º of the field of view.

FALSE: This hypothesis did not prove true for any of the cases studied. Still, the useful field of view demonstrated for the modified Compton telescope technique may be adequately large to prove useful for the applications of the ECRD that have been discussed.

Hypothesis 2. *Precision*: The algorithm achieves an angular resolution of 20º full-width half maximum (FWHM).

TRUE: The angular resolution of the modified Compton telescope technique remains below 20º for those cases where a peak is discernible.

Hypothesis 3. *Convergence*: An estimate of the source position, within the limits described above, will be achieved with fewer than 1000 Compton event-pairs.

TRUE: It takes approximately 200-300 good events for the modified Compton telescope technique to converge on its estimate of the source location.

7.4 Conclusions for the Circumscribed Rectangle Technique

Hypothesis 1. *Accuracy*: For a stationary source, the algorithm provides source localization with angular error of no more than 10º out to near ±90º of the field of view.

FALSE: This hypothesis was reasonable correct for the circumscribed rectangle technique only for the case using "perfect" data. The FOV for this method when using realistic data is even smaller than the field of view for the modified Compton telescope technique.

Hypothesis 2. *Precision*: The algorithm achieves an angular resolution of 20º full-width half maximum (FWHM).

TRUE: The angular resolution for the circumscribed rectangle technique is defined as the size of the final circumscribed box. This quantity is rather meaningless, though, as the final box size is an arbitrary magnitude chosen to accommodate energy resolution or other limitations in the data and calculations.

Hypothesis 3. *Convergence*: An estimate of the source position, within the limits described above, will be achieved with fewer than 1000 Compton event-pairs.

TRUE: The circumscribed rectangle technique converges in approximately 50 Compton event-pairs which is much faster than the modified Compton telescope technique.

7.5 Future Work

Both of the reconstruction methods show promise for application to the ECRD. Each method has its issues, though. (See Appendices A and C for all of the IDL code needed to run these two algorithms.) For the modified Compton telescope technique, a different filter may help to decrease the blurring inherent in the method, improving angular resolution. The primary objective of this algorithm, though, is not to provide an image of the source. The directional algorithm's sole purpose is to provide the direction to the source. Blurring, therefore, is a secondary concern. In fact, the Fourier filtering of the stereographic projection may be a superfluous step because the algorithm chooses the direction to the source as the direction to the point with the highest value (or the centroid of the points with the highest value if there are more than one.)

The circumscribed rectangle technique is very sensitive to bad data. Modification of this method to weed out the spurious events should make this method more robust. This could be achieved by adding an iterative loop to check each event making sure that it overlaps other events and is not simply a data point that is completely off. This method is particularly sensitive to bad events when they occur early in the list of Compton event-pairs because later spurious events can be discarded as inconsistent. Originally, iterative methods were dismissed as possible directional algorithms because they are computationally expensive. The circumscribed rectangles method was developed to be simple and fast, but implementing this iterative functionality may be necessary to achieve adequate utility of the method. However, the impact on computational speed may not be too severe, because only the first few events need to be screened carefully since when the algorithm goes awry at the beginning the method cannot recover. Events that come later in the list are determined to be good or bad compared to these first few events.

Other methods of determining the direction to a source or speeding up the calculation may be investigated. Such methods may include using penetrating events (i.e., those events that interact in a secondary detector only and penetrate through the primary detector) to rule out areas of space in front of the detector as possible source locations. In this case, the ECRD housing acts as simple physical collimation. Another possibility is using events that Compton-scatter in the same primary-detector pixel. These events will generate concentric rings on the rear detectors. The vector connecting the center of these concentric rings with the vertex will point to the source location.

Finally, the evaluation of the algorithms performed here has focused on stationary, single point sources. Multiple sources, moving sources and extended sources need to be studied.

REFERENCES

1. Basko, R., Gengsheng, L.Z., Grant, T.G. Application of Spherical Harmonics to Image Reconstruction for the Compton Camera. **Physics in Medicine and Biology, 43**, 887-894, 1998.

2. Bloser, P.F., Ryan, J.M., McConnell, et al. MEGA: A Medium-Energy Gamma-Ray Astronomy Mission Concept.

3. Boggs, S.E., Jean, P. Event Reconstruction in High Resolution Compton Telescopes. **Astronomy and Astrophysics Supplement Series, 145**, 311-321, 2000.

4. Brechner, R.R., Singh, M. Iterative Reconstruction of Electronically Collimated SPECT Images. **IEEE Transactions on Nuclear Science, 37**(3), 1328-1332, 1990.

5. Du, Y.F., He, Z., Knoll, D.K., Wehe, W.L. Evaluation of a Compton Scattering Camera Using 3-D Position Sensitive CdZnTe Detectors. **Nuclear Instruments and Methods in Physics Research A, 457**, 203-211, 2001.

6. Gunter, Donald L. Filtered Backprojection Algorithms for Compton Cameras in Nuclear Medicine. **Manuscript**, 2003.

7. Gunter, D.L., Mihailescu, L., Burks, M., Vetter, K. A Filtered Backprojection Algorithm for Compton Telescopes. **IEEE NSS-MIC 2005**, paper J03-47

8. Krane, Kenneth S. **Introductory Nuclear Physics 2nd Edition.** John Wiley & Sons Inc., New York, 1998.

9. Lee, W., Wehe, W. 3D Position of Radiation Sources Using an Automated Gamma Camera and ML Algorithm With Energy-Dependent Response Functions. **Nuclear Instruments and Methods in Physics Research A, 531**, 270-275, 2004.

10. Lehner, C.E., Zhong, H., Zhang, F. 4π Compton Imaging Using a 3-D Position Sensitive CdZnTe Detector Via Weighted List-Mode Maximum Likelihood. **IEEE Transactions on Nuclear Science, 51**(4), 1618-1624, 2004.

11. Miller, James R. Geometric Approaches to Nonplanar Quadric Surface Intersection Curves. **ACM Transactions on Graphics, 6**(4), 274-307, 1987.

12. Miller, J.R., Goldman, R.N. Using Tangent Balls to Find Plane Sections of Natural Quadrics. **IEEE Computer Graphics and Applications**, 68-82, 1992.

13. Miller, J.R., Goldman, R.N. Geometric Algorithms for Detecting All Conic Sections in the Intersection of Any Two Natural Quadric Surfaces. **Graphical Models and Image Processing, 57**(1), 55-66, 1995.

14. O'Connor. Natural Quadrics: Projections and Intersectons. **IBM Journal of Research and Development, 33**(4), 417-446, 1989.

15. Parra, Lucas C. Reconstruction of Cone-Beam Projections From Compton Scattered Data. **IEEE Transactions on Nuclear Science, 47**(4), 1543-1550, 2000.

16. Rogers, W.L., Clinthorne, N.H., Bolozdynya, A. Compton Cameras for Nuclear Medical Imaging. In: **Emission Tomography, The Fundamentals of PET and SPECT.** Wernick, M.N. and Aarsvold, J.N. (eds). Elsevier Academic Press, San Diego, 383-419, 2004.

17. Singh, Manbir. An Electronically Collimated Gamma Camera for Single Photon Emission Computed Tomography. Part I: Theoretical Considerations and Design Criteria. **Medical Physics, 10**(4), 421-427, 1983.

18. Singh, M., Doria, D. An Electronically Collimated Gamma Camera for Single Photon Emission Computed Tomography. Part II: Image Reconstruction and Preliminary Experimental Measurements. **Medical Physics, 10**(4), 428-435, 1983.

19. Smith, B., Matthews, K., Lackie, A., et al. An Electronically-collimated Gamma-ray Detector for Intraoperative Localization of Radiation Sources. **Medical Physics, 33**(6), 2277, 2006.

20. Suave, A., Hero III, A.O., Rogers, W.L., Wilderman, S.J., Clinthorne, N.H. 3D Image Reconstruction For a Compton SPECT Camera Model. **IEEE Transactions on Nuclear Science, 46**(6), 2075-2084, 1999.

21. Weisstein, E.W. Cone. **Mathworld—A Wolfram Web Resource**. http://mathworld.wolfram.com/Cone.html

22. Weisstein, E.W. Stereographic Projection. **Mathworld—A Wolfram Web Resource**. http://mathworld.wolfram.com/StereographicProjection.html

23. Wilderman, S.J., Rogers, W.L., Knoll, G.F., Engdahl, J.C. Fast Algorithm for List Mode Backprojection of Compton Scatter Camera Data. **IEEE Transactions on Nuclear Science, 45**(3), 957-962, 1998.

24. Zoglauer, A., Andritschke, R., Kanbach, G. Data analysis for the MEGA prototype. **New Astronomy Reviews**, **48**(1-4), 231-235, 2004.

APPENDIX A

GEOMETRIC (OR IDEAL) DATA GENERATION

A.1 Event Generation Procedure

For the geometry shown in Figure A.1, a gamma ray from the source scatters at a point in the first detector; the scattered photon subsequently deposits its energy in the second (planar) detector. The relationship between the primary photon energy E_0, the scattered photon energy E_s, and the scatter angle θ is

$$E_s = \frac{E_0}{1 + \frac{E_0}{m_e c^2}\left(1 - \cos\theta\right)}. \tag{A.1}$$

The scattering angle is also related to the relative positions of the source, first scatter interaction, and second absorption interaction by (derived from the Law of Cosines)

$$c^2 = a^2 + b^2 + 2ab\cos\theta. \tag{A.2}$$

To implement, the source energy and location are specified, as are the dimensions and locations of the (first) scatter detector plane and the second (absorption) detector. Then for each event, the interaction coordinates in the two detectors are chosen as two pairs of random numbers. The scattering angle θ then is calculated from Equation A.2. This result is fed into Equation A.1 to determine the energy E_s of the scattered photon that must occur for a source photon to have scattered from the first detector at this scattering angle. The process repeats itself to generate the desired number of events.

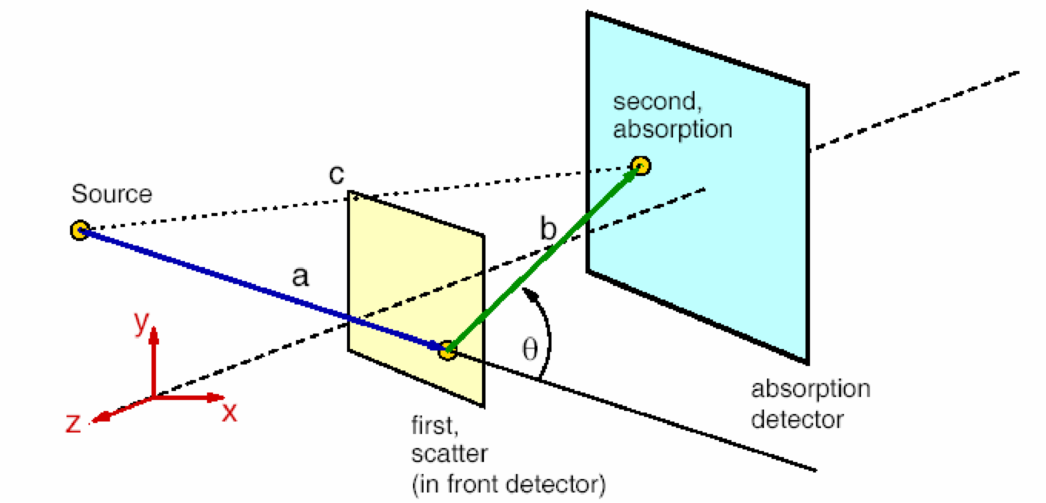


Figure A.1: Illustration of geometry used to generate event data. Gamma rays emitted from the source (blue line) interact in the first scatter detector (yellow shading). The second, absorption interaction (green line) occurs in second detector plane (light blue shading).

Common parameters for generating data:

- Event locations are specified in centimeters (plus or minus relative to the detector's center. The event data records the site (x,y,z)-coordinate and deposited energy for each interaction.
- Energy is specified in keV.

The data for each Run are recorded in an IDL "*SAVE*" file. To recover the data from a file, within IDL, use the command "*restore, <filename>*". Restoration creates a variable called *events*. *events* is a 2D array of structures; *events[i,0]* contains the energy and location of a Compton scattering interaction in the (point or distributed) front detector, while *events[i,1]* contains the energy and location of the matching absorption interaction in the planar (or hemispherical) second detector. Each interaction is stored as an IDL structure, with the format

```
{ INTERACTION_STRUCT,                 $
     energy : 0.0                     ; interaction energy
     units : ''                       ; string, e.g., 'keV'
```

```
                location : { XYZ_STRUCT,       $
                      x : 0.0                  ; x coordinate
                      y : 0.0                  ; y coordinate
                      z : 0.0                  ; z coordinate
                      units : ''               ; string, e.g., 'cm'
                      }
          }
```

The pairs of energies and coordinates for each event are the data that we will have to work with data from the actual ECRD device.

A.2 IDL Code for Event Generation

The first function (***run_generate_events***) is required to call the event generation function. It serves to set up all the specifics about the detector geometry, and the source energy and location. While the second function (***generate_events***) creates a list of events to be used by the two direction reconstruction algorithms.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function run_generate_events,loc

      ;structure definitions
interaction = {INTERACTION_STRUCT,energy:0.0,unit:'',location:    $
            {XYZ_STRUCT,x:0.,y:0.,z:0.,unit:''}}
      ;For a planar detector, p is the z-coord of the plane and h,v
      ;     are the horizontal and vertical dimensions in the plane.
      ;For a hemispherical detector, p is the radius of the
      ;     hemisphere. The center of the hemisphere is at the origin
      ;     of the coordinate system the flat face of the half-shell
      ;     lies in the x-y plane. h,v are angular dimensions of the
      ;     half-shell with "h" specifying the longitude (in the x-z
      ;     plane) and "v" specifying latitude (in the y-z plane).
      ;     Angles in radians; range limits: [-PI/2,PI/2].
      ;Within DET_DIMS_ST, d is the size in physical units (mm for
      ;     planar detector, radians for hemisphere); npix is the # of
      ;     pixels, and pmin is trailing edge of the sensitive volume
      ;     (in pixels relative to the z-axis).
detector = {DETECTOR_STRUCT,p:0.0,unit:'',h:{DET_DIMS_ST,         $
            d:0.0,npix:0UL,pmin:0L},v:{DET_DIMS_ST}}
num_ev = 1000L
;window, xsize=1200, ysize=750
;!p.multi=[0,3,2]
source = {INTERACTION_STRUCT,662.,'keV',                          $
            {XYZ_STRUCT,loc[0],loc[1],loc[2],'cm'}}
      ;hemispherical absorption detector, with dimensions expressed
```

```
      ;    in radians and pixel index ranges
absorption_detector = { DETECTOR_STRUCT, 10., 'cm',               $
      { DET_DIMS_ST, !PI, 1000, -500 },         $;an oblong 2nd det.
      { DET_DIMS_ST, !PI, 1000, -500 } }
!p.multi[0] = 0   ; or 6
      ;point scatter detector
;scatter_detector = { DETECTOR_STRUCT, 0.0, 'cm',                 $
;     { DET_DIMS_ST, 0.1, 1, 0 },                                 $
;     { DET_DIMS_ST, 0.1, 1, 0 } }
;events = generate_events(num_ev, source,                         $
;           scatter_detector, absorption_detector, /DISPLAY)
;!p.multi[0] = 3
      ;area scatter detector
scatter_detector = { DETECTOR_STRUCT, 0.0, 'cm',                  $
      { DET_DIMS_ST, 38.0, 380, -190 },                           $
      { DET_DIMS_ST, 38.0, 380, -190 } }
events = generate_events(num_ev,source,scatter_detector,          $
            absorption_detector,/DISPLAY)
;     ;point source at (8,-19,100)
;source = { INTERACTION_STRUCT, 662., 'keV',                      $
;           { XYZ_STRUCT, 50.5, -85, 15., 'cm' } }
;!p.multi[0] = 5
;     ;point scatter detector
;scatter_detector = { DETECTOR_STRUCT, 0.0, 'cm',                 $
;     { DET_DIMS_ST, 0.1, 1, 0 },                                 $
;     { DET_DIMS_ST, 0.1, 1, 0 } }
;events = generate_events(num_ev, source,                         $
;           scatter_detector, absorption_detector, /DISPLAY)
;!p.multi[0] = 2
;     ;area scatter detector
;scatter_detector = { DETECTOR_STRUCT, 0.0, 'cm',                 $
;     { DET_DIMS_ST, 38.0, 380, -190 },                           $
;     { DET_DIMS_ST, 38.0, 380, -190 } }
;events = generate_events(num_ev, source,                         $
;           scatter_detector, absorption_detector, /DISPLAY)
;     ;point source at (-35,25,100)
;source = { INTERACTION_STRUCT, 662., 'keV',                      $
;           { XYZ_STRUCT, 57.735, 57.735, 57.735, 'cm' } }
;!p.multi[0] = 4
;     ;point scatter detector
;scatter_detector = { DETECTOR_STRUCT, 0.0, 'cm',                 $
;     { DET_DIMS_ST, 0.1, 1, 0 },                                 $
;     { DET_DIMS_ST, 0.1, 1, 0 } }
;events = generate_events(num_ev, source,                         $
;     scatter_detector, absorption_detector, /DISPLAY)
;!p.multi[0] = 1
;     ;area scatter detector
;scatter_detector = { DETECTOR_STRUCT, 0.0, 'cm',                 $
;     { DET_DIMS_ST, 38.0, 380, -190 },                           $
;     { DET_DIMS_ST, 38.0, 380, -190 } }
;events = generate_events(num_ev, source,                         $
;     scatter_detector, absorption_detector, /DISPLAY)
;;save, events, file = 'my_distributed_detector_data.save'
return, events
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

The previous function (***run_generate_events***) calls ***generate_events*** which follows. This function creates a list of events each consisting of two energies and two positions in (x,y,z) coordinates. These correspond to the two readings (primary Compton scatter and secondary photoelectric absorption) that make up each event. This function does not generate the Compton cone information. That calculation is taken care of by another function (see Appendix C.1)

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;=======================================================================
;=======================================================================
;=====================                          ========================
;=====================     EVENT GENERATOR      ========================
;=====================                          ========================
;=======================================================================
;=======================================================================
;======The scatter detector is assumed to be a planar detector.========
;==The absorption detector is assumed to be a hemispherical detector.==
;=======================================================================
;=======================================================================
function generate_events, num_ev, src, scatdet, absdet, DISPLAY=display

      ;elo, ehi
      ;definitions
m_e = 511   ;keV = electron mass
      ;structure definitions
junk = { XYZ_STRUCT, x:0., y:0., z:0., unit:'' }
junk = { INTERACTION_STRUCT, energy:0.0, unit:'',                  $
            location:{ XYZ_STRUCT } }
      ;These detector structures should have already been defined -
      ;     the detector input parameters use these structures
      ;For a planar detector, p is the z-coord of the plane and h,v
      ;     are the horizontal and vertical dimensions (in
      ;     millimeters!) in the plane.
      ;For a hemispherical detector, p is the radius of the
      ;     hemisphere.  The center of the hemisphere is at the origin
      ;     of the coordinate system; the flat face of the half-shell
      ;     lies in the x-y plane. h,v are angular dimensions of the
      ;     half-shell with "h" specifying the longitude (in the x-z
      ;     plane) and "v" specifying latitude (in the y-z plane).
      ;     Angles in radians; range limits: [-PI/2,PI/2]
junk = { DET_DIMS_ST, d:0.0, npix:0UL, pmin:0L }
junk = { DETECTOR_STRUCT, p:0.0, unit:'', h:{ DET_DIMS_ST },       $
            v:{ DET_DIMS_ST } }
      ;events[?,0] is the scattering interaction,
```

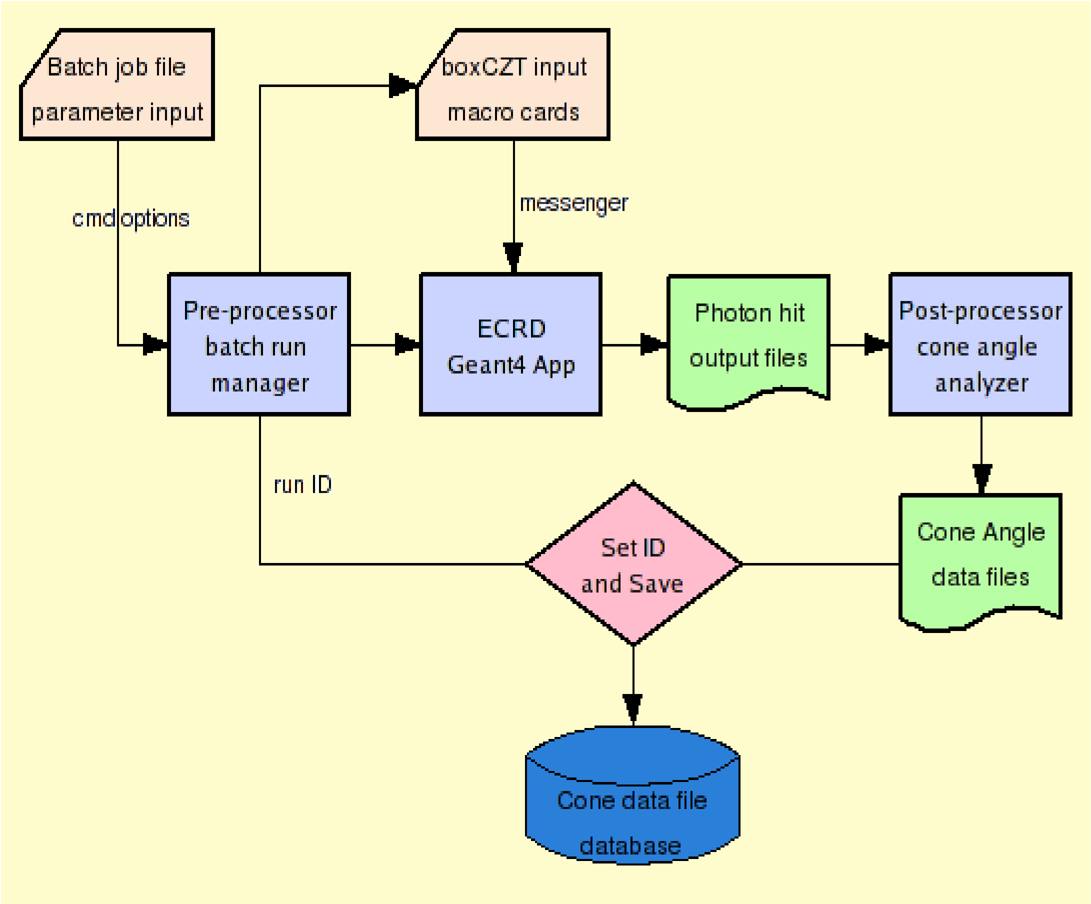
```
      ;events[?,1] is the second interaction
events = REPLICATE( { INTERACTION_STRUCT }, num_ev, 2 )
      ;initialize events structure elements for units
events[*,*] = { INTERACTION_STRUCT, 0.0, 'keV',                   $
            { XYZ_STRUCT, 0,0,0, 'cm' } }
      ;specific seeds are for debugging (delete for normal operation)
;seedx = 100 & seedy = 101
;seeda = 102 & seedb = 103
      ;randomly generate sites of scattering interaction, in
      ;centimeters from scatter-detector center all scatterings occur
      ;in scatter-detector plane
events[*,0].location.z = scatdet.p
events[*,0].location.x = 0.1 * scatdet.h.d / scatdet.h.npix       $
      * FIX(scatdet.h.npix * RANDOMU( seedx, num_ev ) + scatdet.h.pmin)
events[*,0].location.y = 0.1 * scatdet.v.d / scatdet.v.npix       $
      * FIX(scatdet.v.npix * RANDOMU( seedy, num_ev ) + scatdet.v.pmin)
      ;randomly generate direction angles to the absorption interaction
tan_alpha = tan( absdet.h.d / absdet.h.npix                       $
      * FIX(absdet.h.npix * RANDOMU( seeda, num_ev ) + absdet.h.pmin ))
tan_beta = tan( absdet.v.d / absdet.v.npix                        $
      * FIX(absdet.v.npix * RANDOMU( seedb, num_ev ) + absdet.v.pmin ))
      ;now convert these to x,y,z coordinates on the half-shell
events[*,1].location.z = -1*absdet.p/SQRT(tan_alpha^2 + tan_beta^2 + 1)
events[*,1].location.x = events[*,1].location.z * tan_alpha
events[*,1].location.y = events[*,1].location.z * tan_beta
      ;display generated coordinates, for debugging
;IF KEYWORD_SET( display ) THEN BEGIN
;     xrange = absdet.p * [-1,1]
;     yrange = absdet.p * [-1,1]
;     PLOT, [0,0], XSTYLE=3, YSTYLE=3, /NODATA, TITLE='Interaction$
;           sites', XTITLE='absorption detector, x (cm)',         $
;           YTITLE='absorption detector, y (cm)',                 $
;           SUBTITLE='scatter=blue, absorption=green,red/yellow=  $
;           e_window', XRANGE=xrange, YRANGE=yrange, XTICKLEN=0.01$
;           ,YTICKLEN=0.01
;     PLOTS, events[*,1].location.x, events[*,1].location.y,      $
;           PSYM=3, COLOR=255*256L
;     PLOTS, events[*,0].location.x, events[*,0].location.y,      $
;           PSYM=3, COLOR=255*256L^2
;ENDIF
      ;now calc the scattering energy
      ;First, simple geometry to get the scattering angle:
      ;     a = distance from source to scatter site
      ;     b = distance from scatter site to 2nd interaction
      ;     c = distance from source to 2nd interaction
      ;     theta is the scattering angle
      ;Thus, c^2 = a^2 + b^2 - 2 * a * b * cos( PI - theta )
      ;     or  c^2 = a^2 + b^2 + 2 * a * b * cos( theta )
asquared  = ( src.location.x - events[*,0].location.x )^2
asquared  = asquared + ( src.location.y - events[*,0].location.y )^2
asquared  = asquared + ( src.location.z - events[*,0].location.z )^2
bsquared  = ( events[*,1].location.x - events[*,0].location.x )^2
bsquared  = bsquared + ( events[*,1].location.y -                 $
            events[*,0].location.y )^2
bsquared  = bsquared + ( events[*,1].location.z -                 $
            events[*,0].location.z )^2
csquared  = ( src.location.x - events[*,1].location.x )^2
```

```
csquared  = csquared + ( src.location.y - events[*,1].location.y )^2
csquared  = csquared + ( src.location.z - events[*,1].location.z )^2
costheta  = csquared - asquared - bsquared
costheta  = costheta / (2 * SQRT( asquared ) * SQRT( bsquared ))
      ;Second, Compton kinematics to get the scattered photon energy
events[*,1].energy = src.energy/(1 + src.energy / m_e * (1 - costheta))
      ;The remaining energy was deposited during the scattering
      ;     interaction
events[*,0].energy = src.energy - events[*,1].energy
;IF KEYWORD_SET( display ) THEN BEGIN
;     elo = 450 & ehi = 500
;;    READ, elo, ehi, PROMPT='Enter elo, ehi > '
;     w = WHERE( ( events[*,1].energy GE elo )                    $
;           AND ( events[*,1].energy LT ehi ), cnt )
;     IF cnt GT 0 THEN BEGIN
;           plotdata = events[w,1].location
;           PLOTS, plotdata.x, plotdata.y, PSYM=3, COLOR=255
;           plotdata = events[w,0].location
;           PLOTS, plotdata.x, plotdata.y, PSYM=3, COLOR=255+255*256L
;     ENDIF
;ENDIF
RETURN, events
END
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

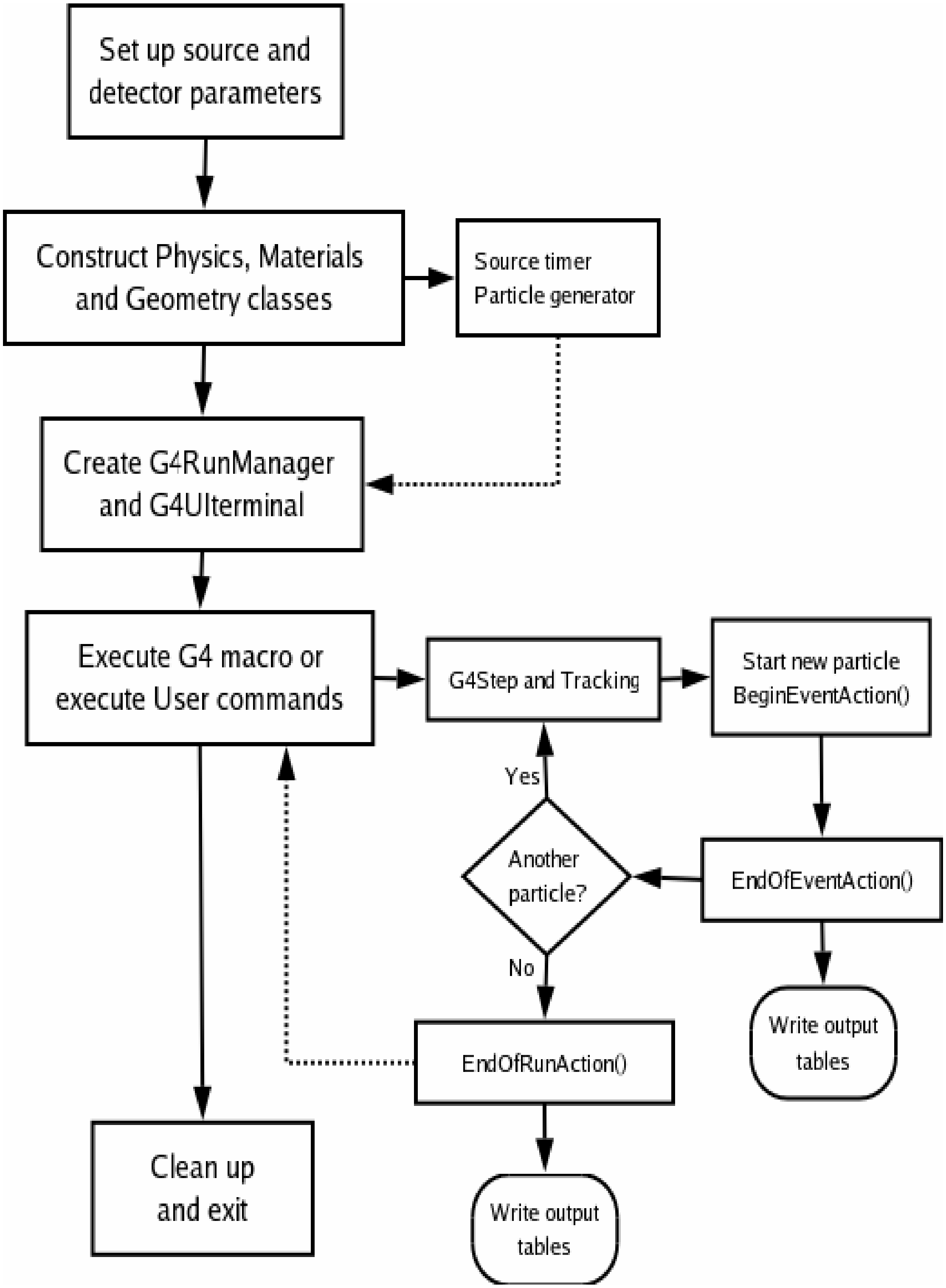
APPENDIX B

MONTE CARLO SIMULATION DETAILS

A Monte Carlo simulation of an ECRD system has been programmed using GEANT4. This task has been developed by Dr. Blair Smith. The flowchart on the following pages (Figure B.1) illustrates the details and functionality of the simulation process.



(Fig. B.1, continued on next page)



(Fig. B.1, continued on next page)

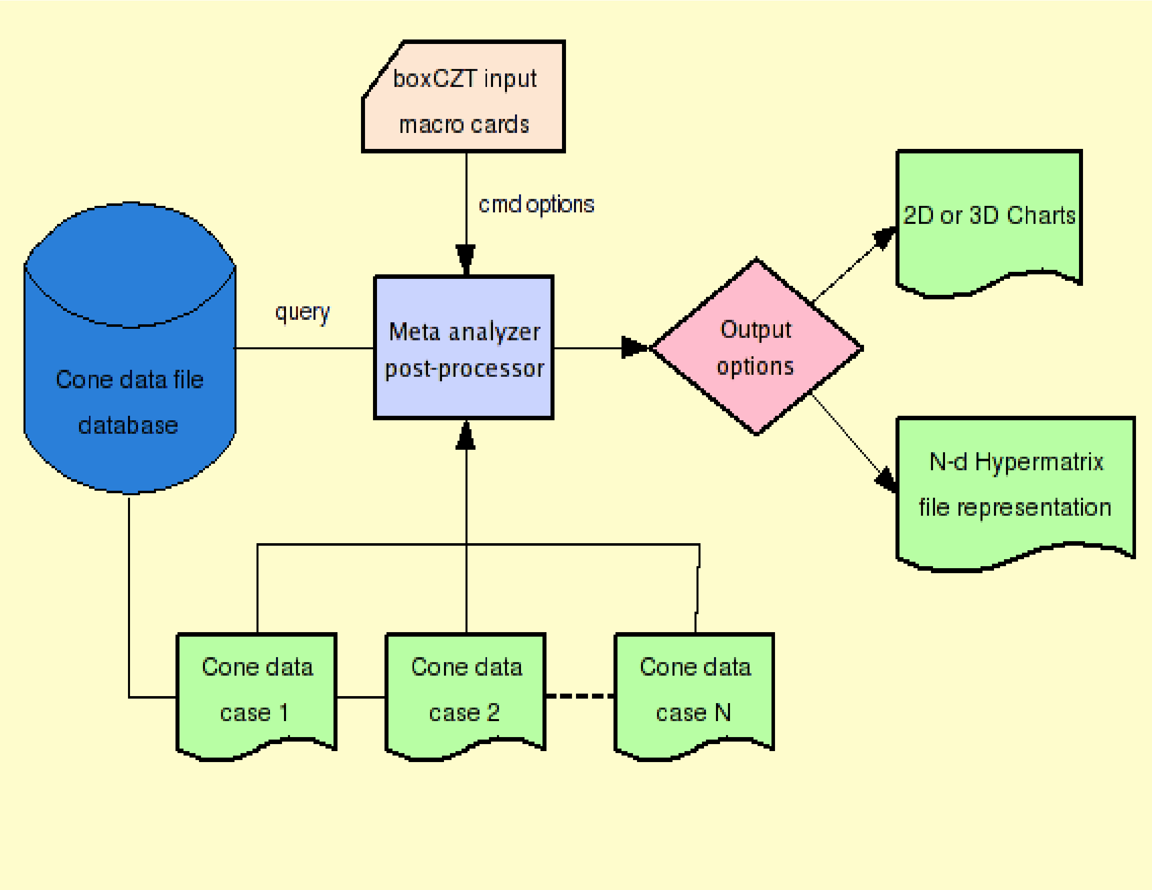


Figure B.1: Three flowcharts illustrating the functionality of the GEANT4 Monte Carlo simulation of an ECRD system.

APPENDIX C

IDL CODE FOR BOTH DIRECTION RECONSTRUCTION METHODS

C.1 Common Functions

This section contains those functions that are used by both the modified Compton telescope technique and the circumscribed rectangles technique. Common functions include ***run_generate_events***, and ***generate_events***, which are listed in Appendix A.2. The only other function used by both methods is ***event_to_cone_info*** whose input is the list of events generated by the preceding two functions and creates a list of the Compton coned that these events correspond to. This function is listed below.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function event_to_cone_info, events

;+
;NAME:
;   event_to_cone_info
;
;PURPOSE:
;   This function returns an array of back-projected cones given an
;     array of Compton interaction pairs.
;
;CALLING SEQUENCE:
;   event_to_cone_info (events,z0)
;
;INPUT:
;   events: An array of INTERACTION_STRUCTs with the dimensions [number
;           of events, 2].
;     *{INTERACTION_STRUCT,energy:0.0,unit:'',location:{XYZ_STRUCT}}
;     *{XYZ_STRUCT, x:0.0, y:0.0, z:0.0, unit:''}
;   where events[i,0] is the primary Compton scattering interaction,
;   and events[i,1] is the secondary event.
;
;OUTPUT:
```

```
;   cones:  An array of cones structures
;     *{CONE_STRUCT,vertex:{XYZ_STRUCT},axis_vector:{VECTOR_STRUCT}$
;           ,half_angle:0.0}
;     *vertex: {XYZ_STRUCT,x:0.0,y:0.0,z:0.0,unit:''}, cone origin
;     *axis_vector: {VECTOR_STRUCT,i:0.0,j:0.0,k:0.0}, orientation
;     *half_angle:  the cone opening angle (in radians)
;
;MODIFICATION HISTORY:
;   Written by: Adam Lackie, Nov, 2005.
;-
      ;structure definitions
vector = { VECTOR_STRUCT, i:0.0D, j:0.0D, k:0.0D }
point = { XYZ_STRUCT, x:0.0D, y:0.0D, z:0.0D, unit:'cm' }
interaction={INTERACTION_STRUCT,energy:0.0D,unit:'keV',location:  $
            {XYZ_STRUCT}}
cone={CONE_STRUCT,vertex:{XYZ_STRUCT},axis_vector:{VECTOR_STRUCT},$
            half_angle:0.0D}
num_events = n_elements(events)/2
cones = REPLICATE( { CONE_STRUCT }, num_events)
j=0uL
for i=0l,num_events-1 do begin
      point1 = events[i,0].location
      point2 = events[i,1].location
      energy1= events[i,0].energy
      energy2= events[i,1].energy
      x1=(point1.x)
      y1=(point1.y)
      z1=(point1.z)
      x2=(point2.x)
      y2=(point2.y)
      z2=(point2.z)
            ;cone parameters
      axis_vect = normalize({VECTOR_STRUCT, x1-x2, y1-y2, z1-z2})
            ;calculate intersection point of axis with sphere and back-
            ;projected circle parameters
      cos_alpha=(energy2^2+energy1*energy2-511.*energy1)/          $
                  ((energy1+energy2)*energy2)
      if abs( cos_alpha ) le 1 then begin
            alpha=acos(cos_alpha)         ;calculate the scatter angle
            cones[j].vertex=events[i,0].location
            cones[j].axis_vector=axis_vect
            cones[j].half_angle=alpha
            j=j+1
      endif
endfor
      ;only return the j good events
return, cones[0:j-1]
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

C.2 Modified Compton Telescope Technique IDL Code

In addition to the common functions listed in Appendices A.2 and C.1 there are four functions required for the calculation of the filtered backprojected images. These are: ***backproject***, ***ster_proj***, ***filter***, and ***inv_ster_proj***. The first of these to be considered will be the ***backproject*** function. This function has a list of cones and the radius of the reconstruction sphere as input. It calculates and draws the circles that are generated by the intersection of the cones and sphere. Its output is therefore an array which represents the surface of the sphere with the intersection circles drawn on.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function backproject, cones,z0

;+
;NAME:
;   backproject
;
;PURPOSE:
;   This function creates an image of the cone intersections with the
;     surface of a sphere.  The cones are assumed to have their vertex
;     at the center of the sphere, so the intersections are always
;     circles.
;
;CALLING SEQUENCE:
;   rectangles (cones,z0)
;
;INPUT:
;   cones:  An array of structures of the form
;     {CONE_STRUCT,vertex:{XYZ_STRUCT},axis_vector:{VECTOR_STRUCT},
;     half_angle:0.0}
;     * vertex is the point of origin of the cone with the form
;           { XYZ_STRUCT, x:0.0, y:0.0, z:0.0, unit:'' }
;     * axis_vector is the direction the cone points of the form
;           { VECTOR_STRUCT, i:0.0, j:0.0, k:0.0},
;     * and half_angle is the opening angle of the cone.
;   z0:     The radius of the sphere surface.
;
;OUTPUT:
;   f:      An array[n,n] which displays the back-projected circles on
;           the surface of the sphere.
;
;MODIFICATION HISTORY:
;   Written by: Adam Lackie, Nov, 2005.
;
;-
```

```
      ;structure definitions
vector = { VECTOR_STRUCT, i:0.0, j:0.0, k:0.0 }
point = { XYZ_STRUCT, x:0.0, y:0.0, z:0.0, unit:'' }
cone = {CONE_STRUCT,vertex:{XYZ_STRUCT},axis_vector:{VECTOR_STRUCT}$
           ,half_angle:0.0}
num_cones = n_elements(cones)
deg=180
pixel_per_deg=4
n=deg*pixel_per_deg
pixel_per_dist=n/(4*z0)
f=fltarr(n,n)
ftemp=f
for i=0l,num_cones-1 do begin
      if !RADEG*cones[i].half_angle ge 5 and                      $
         !RADEG*cones[i].half_angle le 90 then begin
            r=!RADEG*cones[i].half_angle
            x_center=z0*cones[i].axis_vector.i+cones[i].vertex.x
            y_center=z0*cones[i].axis_vector.j+cones[i].vertex.y
            z_center=z0*cones[i].axis_vector.k+cones[i].vertex.z
            th_center=!RADEG*asin(x_center/sqrt(x_center^2+z_center^2))
            ph_center=!RADEG*asin(y_center/sqrt(y_center^2+z_center^2))
            step=atan(0.25/r)
            num_step=round(360/step)
            for p=0l,num_step do begin
                  itheta=round((r*cos(step*p)+th_center)          $
                        *pixel_per_deg)+n/2
                  iphi=round((r*sin(step*p)+ph_center)            $
                        *pixel_per_deg)+n/2
                  for q=-2,2 do begin
                        if itheta+q lt n and itheta+q ge 0        $
                              and iphi+q lt n and iphi+q          $
                              ge 0 then                           $
                              ftemp[itheta+q,iphi+q]=1
                  endfor
            endfor
            f=f+ftemp
            ftemp[*]=0
      endif
endfor
return, f
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

The next function utilized for the modified Compton telescope technique performs the stereographic projection on the sphere surface mapping it onto a flat plane. The input to this function, **`ster_proj`**, is the backprojected array from the previous function and the sphere radius. The output is another array in which each element has been moved according to the rules of the stereographic projection.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function ster_proj, b, z0

;+
;NAME:
;   ster_proj
;
;PURPOSE:
;   This function creates an image in (x,y)-coordinates by transforming
;     an image in (theta,phi)-coordinates using the stereographic
;     transformation.
;
;CALLING SEQUENCE:
;   ster_proj(b,z0)
;
;INPUT:
;   b:      An image in polar coordinates.
;   z0:     The radius of the spherical surface of b
;
;OUTPUT:
;   b_cap:  the stereographically transformed image in
;           (x,y)-coordinates
;
;MODIFICATION HISTORY:
;   Written by: Adam Lackie, Nov, 2005.
;-
deg=180
pixel_per_deg=4
n=deg*pixel_per_deg
pixel_per_dist=n/(4*z0)
th0=0.
ph1=0.
th_arr=(!PI/180.)*(findgen(n)/pixel_per_deg-deg/2)
ph_arr=(!PI/180.)*(findgen(n)/pixel_per_deg-deg/2)
b_cap=fltarr(n,n)
for l=0,n-1 do begin
      for m=0,n-1 do begin
            th=th_arr[l]
            ph=ph_arr[m]
            k=2*z0/(1+((sin(ph1)*sin(ph))+(cos(ph1)*cos(ph)*      $
                  cos(th-th0))))
            x=k*cos(ph)*sin(th-th0)
            y=k*((cos(ph1)*sin(ph))-(sin(ph1)*cos(ph)*cos(th-th0)))
            ix = round(x*pixel_per_dist)+n/2
            jy = round(y*pixel_per_dist)+n/2
            if (ix ge 0) and (ix lt n) and (jy ge 0) and          $
                  (jy lt n) then b_cap(ix,jy)=b_cap(ix,jy)+b(l,m)
      endfor
endfor
return, b_cap
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

After the stereographic projection, the array can be filtered according to the Fourier methods normally applied to computed tomography images. That is, the image is first transformed into phase space using the Fourier transform, a ramp filter is applied to the image and then the array is inverse-Fourier transformed. These three steps are accomplished by the ***filter*** function, which follows.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function filter, b_cap,z0,num

;+
;NAME:
;   filter
;
;PURPOSE:
;   This function creates an image that has been ramp-filtered in
;     Fourier-space and then transformed back into coordinate-space.
;
;CALLING SEQUENCE:
;   filter (b_cap, z0, num)
;
;INPUT:
;     b_cap: An image in (x,y)-coordinates.
;     z0:    radius of reconstruction sphere.
;     num:   The number of events processed.
;
;OUTPUT:
;     b:    An image in (x,y)-coordinates
;
;MODIFICATION HISTORY:
;   Written by: Adam Lackie, Nov, 2005.
;-
deg=180
pixel_per_deg=4
n=deg*pixel_per_deg
np=1080
pixel_per_dist=np/(4*z0)
ramp=dist(np)/np
tot=total(b_cap)
origin=b_cap[np/2,np/2]
b_trans=fft(shift(b_cap,np/2,np/2))
if num ge 100 then begin
      tot_trans=total(b_trans)      ;should equal b_cap at the origin
      if origin ne 0 then begin
            factor=tot_trans/origin
      endif else factor=1
      b_trans=b_trans/factor
endif
b_trans=ramp*b_trans
```

```
b=shift((fft(b_trans,/inverse)),np/2,np/2)
return, b
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

Finally, the inverse stereographic transformation remaps the plane (output from the ***filter*** algorithm) back onto the sphere. The function, ***inv_ster_proj***, which achieves this is listed below.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function inv_ster_proj, b_cap, z0

;+
;NAME:
;   inv_ster_proj
;
;PURPOSE:
;   This function creates an image in (theta,phi)-coordinates by
;     transforming an image in (x,y)-coordinates using the inverse
;     stereographic transformation.
;
;CALLING SEQUENCE:
;   inv_ster_proj(b_cap,z0)
;
;INPUT:
;     b_cap: An image in (x,y)-coordinates.
;     z0:    The radius of the spherical surface of b
;
;OUTPUT:
;     b:     the the inverse stereographically transformed image in
;            polar coordinates
;
;MODIFICATION HISTORY:
;   Written by: Adam Lackie, Nov, 2005.
;-

deg=180
pixel_per_deg=4
n=deg*pixel_per_deg
pixel_per_dist=n/(4*z0)
th0=0.
ph1=0.

x_arr=(findgen(n)-n/2)/pixel_per_dist
y_arr=(findgen(n)-n/2)/pixel_per_dist
b=fltarr(n,n)

for l=0,n-1 do begin
```

```
      for m=0,n-1 do begin
            x1=x_arr[l]
            y1=y_arr[m]
            if x1 eq 0 and y1 eq 0 then begin
                  theta=0.
                  phi=0.
                  itheta = n/2
                  jphi = n/2
            endif else begin
                  rho=sqrt(x1*x1 + y1*y1)
                  c=2.*atan(rho,(z0*2.))
                  theta=th0+atan(x1*sin(c),(rho*cos(ph1)*cos(c)    $
                        -y1*sin(ph1)*sin(c)))
                  phi=asin((cos(c)*sin(ph1))+((y1*sin(c)*          $
                        cos(ph1))/rho))
                  itheta = fix(!RADEG*theta*pixel_per_deg)+n/2
                  jphi = fix(!RADEG*phi*pixel_per_deg)+n/2
            endelse
            if (itheta ge 0) and (itheta lt n) and                 $
                  (jphi ge 0) and (jphi lt n) then                 $
                  b(itheta,jphi)=b(itheta,jphi)+b_cap(l,m)
      endfor
endfor
return, b
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

C.3 Circumscribed Rectangles Technique IDL Code

The functions needed for the circumscribed rectangles algorithm are more straightforward and simpler than those required for the modified Compton telescope technique. There are three functions needed in addition to the common functions. They are titled ***rectangles***, ***generate_lim_array***, and ***overlap_rectangles***. The function titled ***rectangles*** is the first function that accepts the array of cones from ***event_to_cone_info*** and sends it to the other functions for processing. This function returns the final estimate of the source location.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function rectangles, cones

limits_list = fltarr(n_elements( cones ),4)
```

```
limits = fltarr(n_elements( cones ),4)
limits_list = generate_lim_array( cones ) ;returns a list of the limits
                                          ; of each circle (in degrees)
limits = overlap_rectangles(limits_list)  ;returns list of the
                                          ;decreasing limits defined by
                                          ;the ovelaps (in degrees)
return, limits
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

The previous function, ***rectangles***, first sends the array of cones to the function ***generate_lim_array***. This function creates a list of the limits (maximum and minimum coordinated on the sphere) for each cone intersection.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function generate_lim_array, cones

r = 1.1*!RADEG*cones.half_angle   ;add error to r for rounding, eg, 10%
x_center = cones.axis_vector.i
y_center = cones.axis_vector.j
z_center = cones.axis_vector.k
th_center = !RADEG*asin(x_center/sqrt(x_center^2+z_center^2))
ph_center = !RADEG*asin(y_center)
return, [[th_center+r],[th_center-r],[ph_center+r],[ph_center-r]]
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

Finally, the function ***rectangles*** sends the array of limits generated by ***generate_lim_array*** to ***overlap_rectangles***. This function marches through the list of limits and as the new limits further restrict the area defined by the previous limits, a new, more restrictive value replaces the old limit.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
function overlap_rectangles, list

num = (n_elements( list ))/4.
deg=180
overlap = fltarr( num,4 )
overlap[0,*] = [90,-90,90,-90]
```

```
for i=11,num-1 do begin
      if list[i,0] lt overlap[i-1,0] and list[i,0] gt           $
            overlap[i-1,1] then overlap[i,0]=list[i,0] else       $
            overlap[i,0]=overlap[i-1,0]
      if list[i,1] gt overlap[i-1,1] and list[i,1] lt           $
            overlap[i-1,0] then overlap[i,1]=list[i,1] else       $
            overlap[i,1]=overlap[i-1,1]
      if list[i,2] lt overlap[i-1,2] and list[i,2] gt           $
            overlap[i-1,3] then overlap[i,2]=list[i,2] else       $
            overlap[i,2]=overlap[i-1,2]
      if list[i,3] gt overlap[i-1,3] and list[i,3] lt           $
            overlap[i-1,2] then overlap[i,3]=list[i,3] else       $
            overlap[i,3]=overlap[i-1,3]
endfor

return, overlap
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

C.4 IDL Programs to Run the Two Methods

The following programs were designed to call all of the necessary functions (listed previously) to operate the algorithms. First listed is the program that drives the modified Compton telescope technique, ***test_cone_recon***. It calls all of the functions from Appendices A.2, C.1, and C.2.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
pro test_cone_recon

;filename='C:\input_file.dat'
window, xsize=360, ysize=360
z0=100.
deg=180
pixel_per_deg=4
n=deg*pixel_per_deg
pixel_per_dist=n/(4*z0)
map=fltarr(n,n)
error=fltarr(7,7)
error_th=fltarr(90,90)
error_ph=fltarr(90,90)
background=fltarr(20,20)
n_psf=1000
psfs=fltarr(n_psf,2)
deg_var=[0.,15.,30.,45.,60.,75.,89.]
deg_var2=['0','0.26','0.52','0.78','1.04','1.30','1.57']
deg_name=['0','15','30','45','60','75','89']
for i=0,6 do begin
```

```
      for j=0,6 do begin
            z=sqrt(z0^2/(1+tan(!DTOR*deg_var[i])^2+               $
                        tan(!DTOR*deg_var[j])^2))
            x=z*tan(!DTOR*deg_var[i])
            y=z*tan(!DTOR*deg_var[j])
            loc=[x,y,z]
            paired_ev=run_generate_events(loc)
            cones=event_to_cone_info(paired_ev);,z0)
            ;file='C:\input_file.txt'
            ;file='C:\input_file.txt'
            ;cones=read_cc_table(file)
            ;cones=read_real_data(filename)
            map1=backproject(cones,z0)
            map2=ster_proj(map1,z0)
            fmap=shift(filter(map2,z0,n_elements(cones)),360,360)
            image=inv_ster_proj(fmap,z0)
                  ;the following is to zero out the N and S poles
                  ;(which is an artifact of stereographic projection)
            image[0:20,0:719]=0
            image[699:719,0:719]=0
            image[0:719,0:20]=0
            image[0:719,699:719]=0
            image=(abs(image)>0)
;           tvscl,rebin(image,360,360)>0;alog10(n_elements(cones))/4
;           tvscl,rebin(map1,360,360)>0;alog10(n_elements(cones))/4
;           write_jpeg,'C:\output_image.jpg',tvrd(true=1),true=1
;           write_jpeg,'C:\output_image.jpg',tvrd(true=1),true=1
            ind=where(max(image))
            result=where(image eq max(image))
            ph_loc=1.*(fix(result/n)-n/2)/pixel_per_deg
            th_loc=1.*((result mod n)-n/2)/pixel_per_deg
            indth=fix(result/n)
            indph=result mod n
            if n_elements(ph_loc) gt 1 then begin    ;find the centroid
                  avg=moment(ph_loc)
                  ph_loc=avg[0]
                  avg=moment(th_loc)
                  th_loc=avg[0]
            endif
;;==================================================
;;;;;;;;;;;;;;;Max and SNR calcs;;;;;;;;;;;;;;
;;          b_th = indth-50
;;          b_ph = indph-50
;;          for l = 0,19 do begin
;;                for m = 0,19 do begin
;;                      background[l,m] = image[b_th-l/2,b_ph-m/2]
;;                endfor
;;          endfor
;;          b_rate = total(background)/100
;;          SNR = max(image)/b_rate
;;          print, 'max value= ',max(image),'   SNR= ',SNR
;;==================================================
;;;;;;;;;;;;;;;percent error;;;;;;;;;;;;;;;;;;
;;          error[i,j]=abs(sqrt(deg_var[i]^2+deg_var[j]^2+z^2)-   $
;;                sqrt(th_loc^2+ph_loc^2+z^2))/                   $
;;                (0.5*(sqrt(deg_var[i]^2+deg_var[j]^2+z^2)+      $
;;                sqrt(th_loc^2+ph_loc^2+z^2)))
```

```
;;          error_th[i,j]=(abs(deg_var[i]-th_loc)/                $
;;                (0.5*(deg_var[i]+th_loc)))
;;          error_ph[i,j]=(abs(deg_var[j]-ph_loc)/                $
;;                (0.5*(deg_var[j]+ph_loc)))
;;==================================================
;;;;;;;;;;;;;;;;abs error;;;;;;;;;;;;;;;;;;;;;
;;          error[i,j]=sqrt((deg_var[i]-th_loc)^2+                $
;;                (deg_var[j]-ph_loc)^2)
;;          error_th[i,j]=abs(i-th_loc)
;;          error_ph[i,j]=abs(j-ph_loc)
;;==================================================
;;;;;;;;;;;;;;;;;;for PSF;;;;;;;;;;;;;;;;;;;;;
            window, xsize=720, ysize=360
            x=findgen(720)/4-90
            set_plot, 'ps'
            device, file = 'C:\output_PSF.ps'
            psf = map1[*,th_loc+n/2]
            plot,x,smooth(psf,8),title='theta='                   $
                  +deg_name[i]+'  phi='+deg_name[j]+              $
                  ''device,/close
            set_plot,'win'
            window, xsize=360, ysize=360
            tvscl,rebin(map1,360,360)>0
            write_jpeg,'C:\output_unfiltered.jpg',tvrd(true=1),true=1
            tvscl,rebin(image,360,360)>alog10(n_elements(cones))/4
            write_jpeg,'C:\output_filtered.jpg',tvrd(true=1),true=1
;;==================================================
      endfor
endfor
;;==================================================
;;;;;;;;;;;;;;;for FOV display;;;;;;;;;;;;;;;;;;;
;contour, rebin(error,91,91), levels=[2,5,10,15],                 &
;     c_color=colors, xrange=[90,90], /xsty,                      $
;     yrange=[0,90],/ysty, xtitle='theta(deg)',ytitle='phi(deg)', $
;     xticks=3,xminor=3, yticks=3,yminor=3,                       $
;     c_labels=[1,1,1,1],c_annot=                                 $
;     ['2!Eo!N','5!Eo!N','10!Eo!N','15!Eo!N'], font=-1;0
;write_jpeg,'C:\output_FOV.jpg',tvrd(true=1),true=1
;; ==================================================
;;;;;;;;;;;;;;;to display image;;;;;;;;;;;;;;;;
;;tvscl,rebin(map1,360,360)>alog10(n_elements(cones))/4
;;write_jpeg,'C:\output_unfilt_image.jpg',tvrd(true=1),true=1
;;tvscl,rebin(image,360,360)>alog10(n_elements(cones))/4
;;write_jpeg,'C:\output_filt_image.jpg',tvrd(true=1),true=1
;;==================================================
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

The previous program has sections commented out so that only those evaluations that are desired will be processed when the program is run. By commenting and uncommenting each of the sections, different values of interest can be analyzed. The following program, titled **`test_rect`**, drives the circumscribed rectangles technique. It operates much the same as the previous program.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
pro test_rect

z0=100.
pixel_per_deg = 4.
deg_var=[0.,15.,30.,45.,60.,75.,89.]
deg_var2=['0','0.26','0.52','0.78','1.04','1.30','1.57']
deg_name=['0','15','30','45','60','75','89']
error=fltarr(7,7)
error_th=error
error_ph=error
for i=0,6 do begin
      for j=0,6 do begin
            th_real= deg_var[i]
            ph_real= deg_var[j]
            z=sqrt(z0^2/(1+tan(!DTOR*th_real)^2+tan(!DTOR*ph_real)^2))
            x=z*tan(!DTOR*th_real)
            y=z*tan(!DTOR*ph_real)
            loc=[x,y,z]
;           ev=run_generate_events(loc)
;           cones=event_to_cone_info(ev);,z0)
            file='C:\input_file.txt'
            cones=read_cc_table(file)
            n=n_elements(cones)
            limits=rectangles(cones)
            avg_th=(limits[n-1,0]+limits[n-1,1])/2
            avg_ph=(limits[n-1,2]+limits[n-1,3])/2
            error[i,j]=sqrt((avg_th-deg_var[i])^2+                $
                  (avg_ph-deg_var[j])^2)
            error_th[i,j]=abs(avg_th-th_real)
            error_ph[i,j]=abs(avg_ph-deg_var[j])
      endfor
endfor
;;==================================================
;;;;;;;;;;;;;;;for convergence;;;;;;;;;;;;;
;ind=[1,5,15,25,50,75,100,200,300,499,9999]
;for i=0,10 do begin
;     area=(limits[i,0]-limits[i,1])*(limits[i,2]-limits[i,3])
;     print, 'ind= ', i, '    area= ', area
;endfor
;;==================================================
;;;;;;;;;;;;;;for FOV display;;;;;;;;;;;;;;;;;;;
;contour, rebin(error,91,91),levels=[5,10,15],                    $
;     c_colors=[255,255+255*256L+255*256L^2,255L*256L^2],         $
;     xstyle=1,ystyle=1,xticks=9,yticks=9
contour, rebin(error,91,91), levels=[2,5,10,15],                  $
c_color=colors, xrange=[0,90], /xsty,                             $
      yrange=[0,90],/ysty, xtitle='!Mq!3 (deg)',                  $
      ytitle='!Mf!3 (deg)', xticks=3,xminor=3, yticks=3,yminor=3, $
```

```
      c_labels=[1,1,1,1],c_annot=['2!Eo!N','5!Eo!N','10!Eo!N',    $
            '15!Eo!N'],font=0
;write_jpeg,'C:\output_FOV.jpg',tvrd(true=1),true=1
;;==================================================
end
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

Data from either ***`run_generate_events`*** or from an outside source were used as input for these two programs. The data from ***`run_generate_events`*** is output in an array that can be used directly. Other sources will not necessarily output the data in such a straightforward manner. The need arises in this case to write another function (called ***`read_cc_table`*** in these two programs) to read and sort the data. The input is only the filename to be analyzed. This function will not be included as part of this Appendix because it is completely dependent on the format of the data that is output (either by a Monte Carlo simulation or by a prototype detector.) This is the only omission. Otherwise, the code presented in Appendices A and C provide a complete version of the two direction-reconstruction methods.

VITA

Adam Lackie was born in Baton Rouge, Louisiana, but moved to Lafayette, an hour away, at age three. Having grown up in the Heart of Acadiana, he considers himself a proud Cajun, if only by association. He moved back to Baton Rouge in August of 1994 to begin undergraduate studies at Louisiana State University. After testing a number of majors ranging from music to pre-med he finally settled on physics knowing at the time only that his profound love for the subject (a love engendered as a child by a father who could always come up with an answer to the question, “Why?”) should lead him somewhere worthwhile. Supporting himself through college with jobs such as banker, department store cologne salesman, and concession salesman at a cinema, he succeeded in earning a Bachelor of Science with honors in physics. A traveling speaker came during his undergraduate studies to the Physics department to talk about the field of medical physics. Something clicked. It seemed the perfect blend of altruism and science, and still does. Adam will soon be moving to Los Angeles to begin his career. He is the proud partner of Levi Denham and the son of Steve Lackie and Karyn Placek, all of whom share the triumph of this graduation as they have all played their vital parts.