To the Graduate Council:

I am submitting herewith a thesis written by Dean Alan Huff entitled "A Focus of Attention EM Algorithm for PET Image Reconstruction." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Jens Gregor

Dr. Jens Gregor, Major Professor

We have read this thesis and
recommend its acceptance:

Michael G Thomason

Michael W. Berry

Accepted for the Council:

C. W. Minkel

Associate Vice Chancellor and
Dean of The Graduate School

A Focus of Attention EM Algorithm for PET Image Reconstruction

A Thesis

Presented for the

Master of Science Degree

The University of Tennessee, Knoxville

Dean Alan Huff

August 1995

Acknowledgments

Sincere thanks to my major professor, Jens Gregor, for his advice and confidence. Also, thanks to the faculty committee members Michael Thomason and Michael Berry.

Abstract

Positron Emission Tomography (PET) image reconstruction can be done using the Expectation Maximization (EM) algorithm to optimize a likelihood function based on emissions. The EM algorithm utilizes a large probability matrix and thus can be computationally expensive. Consequently, reducing the size (number of nonzeros) of this matrix can make the algorithm significantly more efficient.

We use a parallel implementation of the EM algorithm in which masking is used to reduce computation. A mask indicating relevant emissions is obtained through an algorithm to determine areas of background level emissions. Another mask indicating the relevant portion of the image space is then determined from the first mask and the scanner geometry. The reduced probability matrix used by the EM algorithm then represents the correlation between emissions and the relevant portion of the image space.

The overhead required to eliminate nonzeros with masking is minimal. However, a limited probability matrix can be constructed using fewer resources and can result in faster EM iterations. Considering all factors, this focus of attention technique can reduce required time and memory by approximately fifty percent without loss in the quality of reconstructed images.

Contents

1 Introduction **1**

1.1 Objective 1

1.2 Notation and Definitions 1

1.2.1 Background 1

1.2.2 Gantry Geometry 2

1.2.3 Emission Data 4

1.2.4 Image Geometry 4

1.2.5 Tube-Pixel Mapping 5

1.3 EM Algorithm 6

1.3.1 Algorithm 7

1.3.2 Complexity 7

1.4 Reconstruction Pipeline 8

2 D_{mask} Creation **10**

2.1 Sinogram Description 10

2.2 n^* Filtering 11

2.2.1 Moving Average 12

2.3 Consistency of the D_{mask} 13

2.3.1 Convex Support 13

2.3.2 Optimization 15
2.4 Recapitulation 18

3 B_{mask} Creation **19**
3.1 B_{mask} Description 19
3.2 D_{mask} Conversion 20

4 *P matrix* Construction **23**
4.1 *P matrix* Notation 23
4.2 Area Calculation 23

5 n^* Edge Reduction **27**
5.1 Edge Packing 27
5.2 Count Reduction 28
5.2.1 Alternate methods 29

6 Experimental Work **30**
6.1 Implementation 30
6.2 Load Balancing 30
6.3 Test Cases 31
6.3.1 D_{mask} Results 32
6.3.2 B_{mask} Results 36
6.3.3 P matrix Results 37
6.3.4 Resulting Images 38
6.3.5 Memory Consumption 41
6.3.6 Timings 43

7 Concluding Remarks **47**

7.1 Benefits 47
7.2 Further Possibilities 47

Bibliography **49**

Vita **51**

List of Tables

6.1 Test Cases 32

6.2 Convexity Optimization Iterations 35

6.3 Array sizes without masking 42

6.4 Array sizes (1% threshold) 42

6.5 Array sizes (2% threshold) 42

6.6 Execution CPU Time without masking 43

6.7 Execution CPU Time (1% threshold) 44

6.8 Execution CPU Time (2% threshold) 44

List of Figures

1.1 Gantry 3
1.2 Half Angle 3
1.3 Image Space 5
1.4 Tube-Pixel Correspondence 6
1.5 Original EM Pipeline 9
1.6 Masked EM Pipeline 9

2.1 Object Profile 11
2.2 Moving Average 12
2.3 Support Lines 14

3.1 B_{mask} Pixels 20
3.2 B_{mask} Selection 21

4.1 Tube-Pixel Intersection 24
4.2 Pixel Types 25
4.3 Automaton Transitions 26

5.1 Edge Tube 28

6.1 Test Case Sinograms 32

6.2 Brain D_{mask}, Cardiac D_{mask} 33
6.3 Pancreas D_{mask}, Phantom D_{mask} 33
6.4 Brain $h^T C$ 34
6.5 Cardiac $h^T C$ 35
6.6 Pancreas $h^T C$ 35
6.7 Phantom $h^T C$ 36
6.8 Brain B_{mask}, Cardiac B_{mask} 36
6.9 Pancreas B_{mask}, Phantom B_{mask} 37
6.10 Brain Histogram, Cardiac Histogram 38
6.11 Pancreas Histogram, Phantom Histogram 38
6.12 Brain Images 39
6.13 Cardiac Images 39
6.14 Pancreas Images 40
6.15 Phantom Images 40
6.16 Pancreas Images (128 Iterations) 41
6.17 P matrix Time Comparison (all cases) 45
6.18 Iteration Time Comparison (pancreas case) 46

Chapter 1

Introduction

1.1 Objective

This thesis is based on a parallel implementation of the Expectation Maximization (EM) algorithm for image reconstruction. Typically only a portion of an image is relevant so we would like to only compute for that portion of the image. Masking techniques can be used to limit required calculation and thus reduce CPU time and memory consumption.

1.2 Notation and Definitions

1.2.1 Background

Positron Emission Tomography (PET) is a method for non-invasively studying the physiology of the human body. In a typical scenario a radio-isotope is injected into a patient. The isotope collects in tissue and emits photons in proportion to isotope concentration. After injection the subject is placed in a gantry which consists of a ring of photon detectors. As the isotope decays photons are emitted

and detected by the detectors. The task is to reconstruct an image that reflects the internal isotope concentration on basis of the external distribution of photon emissions [4].

1.2.2 Gantry Geometry

Gantry detectors are arranged in a circle around the subject. Each pair of detectors determines a *tube*, denoted by d. Each tube is given by an angle θ and an offset δ (distance from the center of the gantry). Because the gantry has a finite number of detectors, tube angles and offsets can be considered discrete indexes. Thus each tube corresponds to a point in a two dimensional discrete space (one dimension angle, the other offset). Let this space of tubes be denoted D.

The space to be described is actually the plane of the ring of detectors. This is a continuous plane whos origin is an arbitrary point fixed relative to the gantry. Let this space be denoted G.

In G a tube is the space between the lines connecting the outside edges of its two detectors. Thus it can be completely described by a slope and two y-intercepts. Since the gantry geometry is fixed it is simple to determine slope and y-intercepts given a tube's angle and offset. An illustration of the gantry and a tube is given in Fig. 1.1.

With an even number of detectors some portions of the image space may not be covered by tubes at any full angle. For example, the exact center of the gantry lies between two tubes for all angles. This difficulty is overcome by including half angles. Half angle tubes are detector pairs in which one detector varies one position from that of a full angle tube. The difference between half angles and full angles is illustrated in Fig. 1.2.

Note that the width of the tubes will decrease with the distance from the center

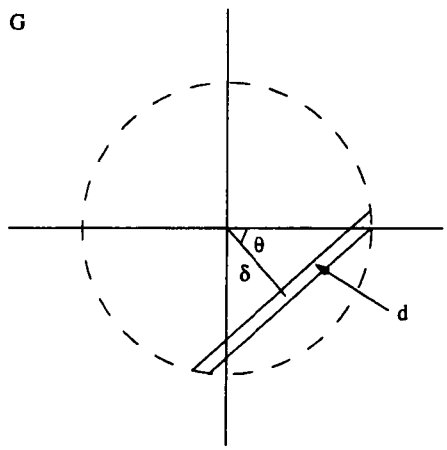


Figure 1.1: Gantry

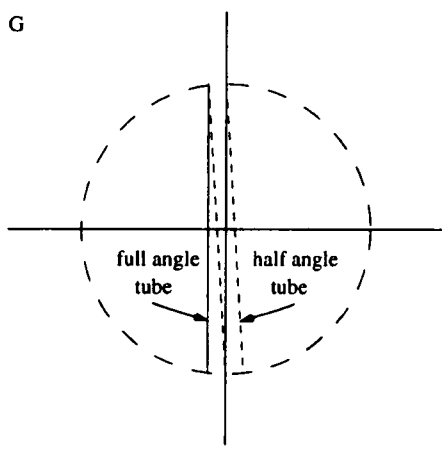


Figure 1.2: Half Angle

of the gantry. Also, for conceptual reasons, it is assumed that there is no distance between tubes.

1.2.3 Emission Data

As the isotope decays, two coincident photons are emitted along a line in opposing directions. Hence if two detectors simultaneously detect a pair of photons, then it is assumed that the emission came from somewhere between the detectors. Thus, the emission can be contributed to the tube determined by those two detectors. During detection the subject is placed in the gantry and emissions are counted for all tubes. After a specified amount of time each tube will have an associated number of emissions. Denote this emission count for tube d by n_d^*.

The n_d^* counts for all tubes can be represented as a *sinogram*. This term comes from the fact that a point in gantry space, G, maps to a sine wave in tube space, D. A sinogram is a representation such that n_d^* values for tubes are arranged by angle and offset. Each row of emission data corresponds to a tube angle, and each column corresponds to a projection offset. More simply, a sinogram is an image in D space indicating emission levels for all angles and offsets.

1.2.4 Image Geometry

The image space consists of a quantization of the plane of the gantry G. Let this space be denoted B, and let a pixel in B be denoted b. B corresponds to a square region of G as shown in Fig. 1.3. The image to be constructed from the emission data is a function in B and hence a picture of emissions as they occurred in G. During reconstruction each pixel b is associated with a Poisson parameter, λ_b, indicating its mean rate of emission.

As illustrated in Fig. 1.3, a pixel is a square region in the plane of the gantry

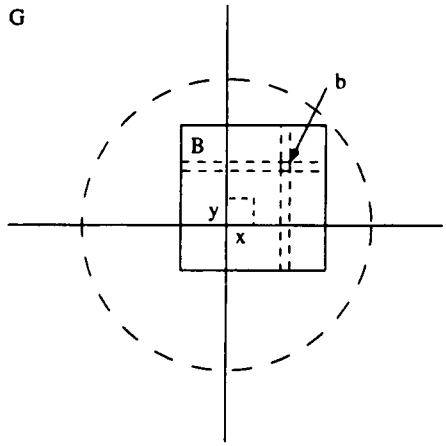


Figure 1.3: Image Space

G. Thus it can be completely described by a minimum and maximum x-value as well as a minimum and maximum y-value. The location and size of a pixel, relative to G, are determined by locational offset and zoom factor parameters.

1.2.5 Tube-Pixel Mapping

With respect to the D domain, a tube is a point given by its angle and offset. With respect to the G domain, however, a tube is the space between the two lines connecting the edges of the corresponding detectors. To relate the notion of a tube to the B domain it is relevant to consider the intersection of tubes and pixels in G. Basically, a tube corresponds to the set of all pixels which have any area in common with the tube in G space.

With respect to the B domain, a pixel is a point given by its row and column (actually a square region in the image space). With respect to the D domain, a pixel roughly corresponds to a sine wave including all the tubes which have any

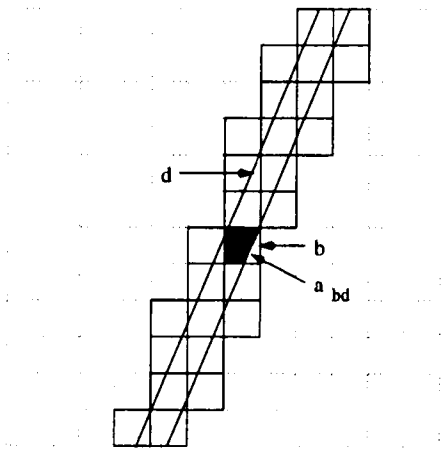


Figure 1.4: Tube-Pixel Correspondence

area in common with the pixel in G space.

Since a tube is a section of G defined by two parallel lines, and a pixel is a square in G, each tube will intersect a limited set of pixels. Fig. 1.4 illustrates the correspondence between a tube and the pixels which it intersects. The area of intersection (in G) of pixel b and tube d is denoted a_{bd}. In the figure, one such area of intersection is shown as the shaded region.

1.3 EM Algorithm

To determine λ_b (for all b) from n_d^* (for all d), a relation between D and B is needed. This relation is p_{bd}, the probability of detection by tube d given an emission from pixel b. The general problem is thus to solve $P\lambda = n^*$ for image λ given probability matrix P and emission data n^*.

1.3.1 Algorithm

The Expectation Maximization algorithm is an iterative method which continually improves the likelihood [5]:

$$L(\lambda) = p(n^* \mid \lambda) = \prod_d e^{-\lambda_d^*} \frac{\lambda_d^{*n_d^*}}{n_d^{*!}}$$

In actuality the log likelihood, $l(\lambda) = \log L(\lambda)$, a concave function, is maximized. The EM iteration is given by:

$$\lambda_b^{k+1} = \lambda_b^k \sum_d \frac{n_d^* p_{bd}}{\lambda_d^{*k}}$$

where

$$\lambda_d^{*k} = \sum_b \lambda_b^k p_{bd}$$

is the expected emission count for tube d given λ_b^k. The sum on the right hand side of the iteration equation can be thought of as a multiplier,

$$M_b^k = \sum_d \frac{n_d^* p_{bd}}{\lambda_x^{*k}}$$

so the EM iteration can be written as follows:

$$\lambda_b^{k+1} = \lambda_b^k M_b^k$$

1.3.2 Complexity

Calculation of λ_d^{*k} for one tube, d, involves one multiplication (and one addition) for every pixel b intersected by d. Calculation of M_b^k also requires one multiplication (and one addition) for every pixel b intersected by each d. In addition, M_b^k

requires one division for each d.

Let N_P denote the number of entries in the P *matrix*, i.e., the number of bd pairs such that $d \in D$ and $b \in B$ have area in common. Let N_D denote the number of tubes $d \in D$. The following represent the computational complexity of an iteration of the EM algorithm:

- Complexity with respect to multiplication = $O(2N_P)$
- Complexity with respect to addition = $O(2N_P)$
- Complexity with respect to division = $O(N_D)$

The P *matrix* is a sparse matrix since each tube only intersects a relatively small number of pixels. Still, reducing the number of tubes and pixels will reduce the number of non-zero entries. This reduction is achieved through the use of masks which indicate the tubes and pixels used in computation. Thus masking can provide significant improvement in CPU time and memory consumption.

1.4 Reconstruction Pipeline

The object of PET image reconstruction is to create an image (in B) using the emission data, n^*. Our reconstruction method is segmented into some well-defined stages between the initial emission data and the final generated image. Firstly a mask of relevant tubes is made via filtering of n^*. From the tube mask, a pixel mask (of relevant pixels) is created. Using both the tube mask and pixel mask a probability matrix of p_{bd} values is generated. Some slight modification of n^* data can then done (if needed) to prevent edge distortion. Finally, the actual EM algorithm iteratively determines a resulting image.

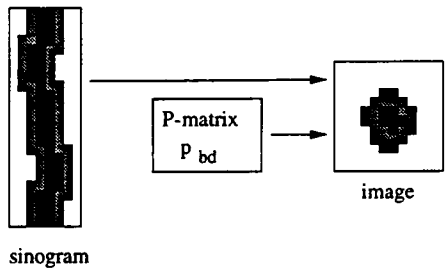


Figure 1.5: Original EM Pipeline

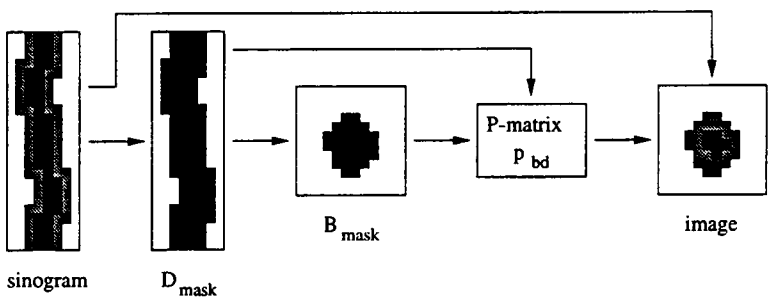


Figure 1.6: Masked EM Pipeline

A simple representation of EM image reconstruction is given in Fig. 1.5. This pipeline involves calculation using all pixels in B and all tubes in D. With the addition of tube and pixel masks, the pipeline becomes that given in Fig. 1.6.

Chapter 2

D_{mask} Creation

The purpose of creating a D_{mask} is to reduce the number of tubes for which computation must be done. The D_{mask} is also used in creating a B_{mask} to reduce the number of pixels for which computation must be done.

2.1 Sinogram Description

As mentioned, a sinogram is a function in D containing the observed emission counts, n_d^*, for all tubes d. Each row of the sinogram thus corresponds to emissions observed at a given angle. In other words, each row of the sinogram represents a profile of the observed object from one angle.

We know that the object being observed is completely contained within the gantry. This means that the profile of the observed object for a given angle will generally not extend to the rightmost/leftmost tubes for that angle. Hence, some tubes near the edges of sinogram rows will not intersect the observed object. The n_d^* values for these tubes will only reflect noise and background radiation and thus will generally be lower than n_d^* for other tubes. This tendency is illustrated by an

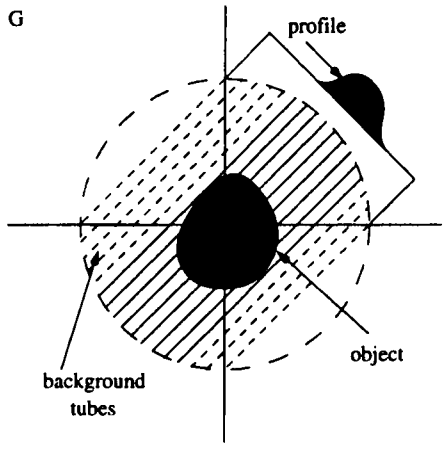


Figure 2.1: Object Profile

example profile in Fig. 2.1. Each row of a sinogram is a discrete profile similar to the continuous one shown in the figure.

2.2 n^* Filtering

Due to the nature of the profiles of an emitter in G, the tubes to be removed from computation are those on the edges of each row of the sinogram. We can use the n^* data to determine which tubes correspond to background portions of G (those tubes that do not intersect the object of interest). Sinogram rows correspond to separate profiles so each row of a sinogram can be processed separately.

Note that all the information in a profile between two extremes is relevant to the corresponding object (information outside the two extremes is considered background). In terms of a sinogram this means that only two values are needed to limit the tubes of each row. A tube mask D_{mask} will therefore consist of a closed range of offsets for each angle.

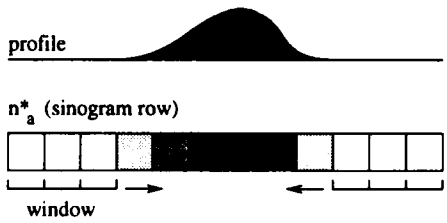


Figure 2.2: Moving Average

2.2.1 Moving Average

Because of noise in the sinogram data, profile boundaries can be uncertain. This uncertainty can be overcome by looking for a general trends in n_d^* values for each angle. Considering a tube d', if nearby tubes have background level n_d^* values then it is likely that tube d' is not relevant. Similarly, if nearby tubes have n_d^* values above background levels then it is likely that d' is relevant and should be included in the D_{mask}.

One method of looking for general trends is to simply average n_d^* values for tubes d within a radius of an offset. This method involves moving a window across tube offsets until the average of tubes within the window is large enough to be considered non-background. More specifically, the window is moved from extreme tubes towards the center of the sinogram. Bounding offsets are chosen when the average count value for tubes within the window reaches a threshold. This concept is illustrated by Fig. 2.2.

There remains the question of the width for the moving window. This radius should correspond to the amount of emission noise relative to the overall emission level, n^*. For data which is well-defined (e.g., a brain scan) and well-segregated

only a small window radius is needed. For data which is more diffuse (e.g., a full torso cross section scan) a larger window radius might yield a better estimate of profile boundaries.

The other required parameter is the threshold n^* value. This is the cutoff value for moving averages that determines the object boundary. Generally a low value relative to the overall n^* should be used. The suggested threshold value is one percent of the maximum n_d^* value.

In the case of tubes near the edge of the sinogram the window might extend beyond the edge of the sinogram. For the purpose of calculating an average of tubes in the window, offsets beyond the edge are assumed to have the same n_d^* as the outermost tube in the sinogram. Alternatively the sinogram can be assumed to be cyclic (i.e., have wrap-around left/right edges).

2.3 Consistency of the D_{mask}

2.3.1 Convex Support

A *support line* is a line tangent to the boundary of a convex hull. Given a convex hull, there will be a support line for any angle. Thus a *support function*, $h(\theta)$, can be defined as the distance from the origin to the support line for angle θ. For a discrete mapping in which there are only finitely many (m) angles, the support function is described by a *support vector* (with m entries) [3].

Tubes on the edge of D_{mask} define support lines for the region of interest in the gantry space, G. Thus, D_{mask} yields a support vector. Due to the heuristic nature of n^* filtering, the support vector given by the D_{mask} may be inconsistent. In other words, for some angles the lines given by the initial D_{mask} might not actually be supporting lines for the convex object given by the D_{mask}.

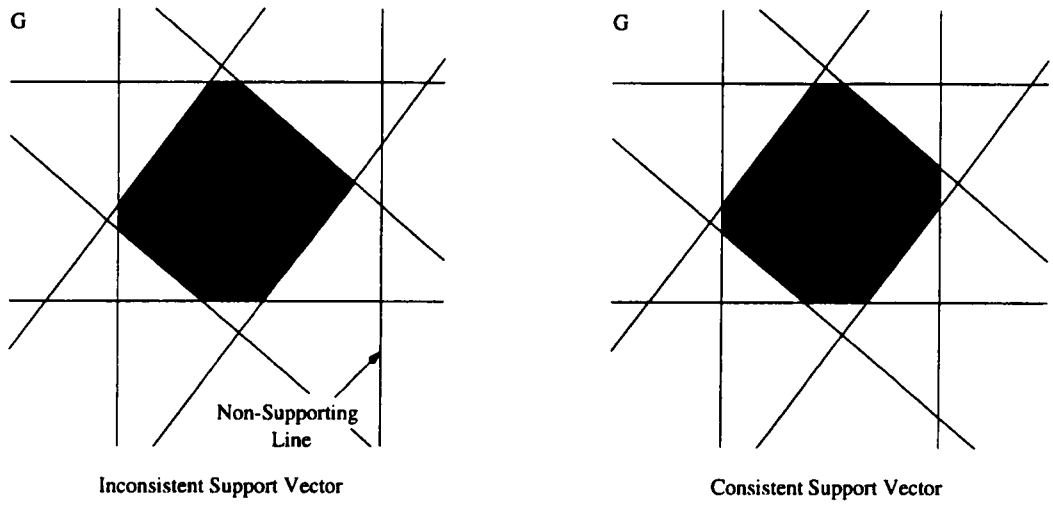


Figure 2.3: Support Lines

From the moving average filter this inconsistency arises because the sinogram is processed one row at a time. Each row of D_{mask} is created out of context, so adjacent support lines may be uncorrelated. This can lead to inconsistencies in the corresponding support vector.

Fig. 2.3 shows a simplified example of a set of lines defining a convex region. In the left illustration, the labeled vertical line does not actually touch the convex set and is hence not a support line. The right illustration shows a set of support lines corresponding to a consistent support vector. It is likely that heuristic sinogram filtering will yield a tube mask which defines some non-supporting lines like the one in the figure.

A consistent D_{mask} is needed to ensure that pixels are not unnecessarily removed from or added to B_{mask} due to noise in the initial creation of the D_{mask} (moving average). A consistent mask ensures that all pixels in the B_{mask} are intersected by at least one tube for each angle (this depends on how the B_{mask} is

created). Otherwise some tubes may have counts from portions of G which are not included in B_{mask}. Such excessive counts may result in bright pixels in the resulting image (on the edge of B_{mask}). Hence, optimization of the support vector defined by D_{mask} is required.

2.3.2 Optimization

The *support theorem* [3] states that a vector $h \in R^m$ is a support vector if and only if

$$h^T C \leq [0 \cdots 0]$$

where

$$C = \begin{bmatrix} 1 & -k & 0 & \cdots & 0 & -k \\ -k & 1 & -k & 0 & \cdots & 0 \\ 0 & -k & 1 & & & \vdots \\ \vdots & 0 & & \ddots & & 0 \\ 0 & \vdots & & & 1 & -k \\ -k & 0 & \cdots & 0 & -k & 1 \end{bmatrix}$$

is an m by m Toeplitz matrix and

$$k = \frac{1}{2\cos\frac{2\pi}{m}}$$

so $k \approx 0.5$ for m sufficiently large. The structure of C is such that the entire matrix can be represented by a vector. Thus the constraint for support is simply

$$h_j - 0.5(h_{j+1} + h_{j-1}) \leq 0 \; for \; 1 \leq j \leq m$$

where indexes are calculated modulo m. This simple calculation compares each support line with the two adjacent support lines.

Prince and Willsky [3] give a *Closest Algorithm* for estimation of a support vector given noisy data as follows:

$$\hat{h}_C = \arg \max_{h:h^T C \leq 0} -\frac{1}{2}(y-h)^T(y-h)$$

This yields a quadratic programming problem:

$$\min_h \frac{1}{2}h^T h - y^T h \;\; s.t.\; h^T C \leq 0$$

Gauss-Seidel optimization will yield a new h which fully supports the convex set. To avoid cutting into the emitter we wish to ensure that the optimized h is not smaller is not smaller than our original estimate. To this end we add the constraint that $h \geq y$ (i.e., $-h \leq -y$). The problem can then be reformulated as follows:

$$\min_h \frac{1}{2}x^T x - b^T x \;\; s.t.\; Ax \leq c$$

where

$$x = h,\; Q = I,\; b = y,\; A = \begin{bmatrix} C \\ -I \end{bmatrix},\; and\; c = \begin{bmatrix} 0 \\ -y \end{bmatrix}$$

The dual problem [2] is thus:

$$\min_u \frac{1}{2}u^T P u + r^T u \;\; s.t.\; u \geq 0$$

where

$$P = AQ^{-1}A^T = \begin{bmatrix} CC^T & -C \\ -C & I \end{bmatrix}$$

and

$$r = C - AQ^{-1}b = \begin{bmatrix} -Cb \\ 0 \end{bmatrix}$$

For this formulation, an iteration of the Gauss-Seidel solution is given by:

$$\tilde{u}_j = u_j - \frac{1}{p_{jj}}(r_j + \sum_{k=1}^{2m} p_{jk}u_k)$$

Vector u can be broken into two vectors for simpler computation:

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

such that both u_1 and u_2 are of size m. Rewriting the G-S iteration we then obtain a final form of the iteration:

$$\tilde{u}_{j:1\leq j\leq m} = u_j - \frac{1}{(CC^T)_{jj}}(r_j + C_1u_1 - C_2u_2)$$

$$\tilde{u}_{j:m+1\leq j\leq 2m} = u_j - (r_j - C_2u_1 + u_j) = C_2u_1$$

where C_1 is the jth row of CC^T and C_2 is the jth row of C. Due to the Toeplitz nonzero structure, CC^T is nearly as simple to compute with as with C itself. For a vector h of size m:

$$(CC^Th)_{j:1\leq j\leq m} = (1 + 2k^2)h_j - 2k(h_{j-1} + h_{j+1}) + k^2(h_{j-2} + h_{j+2})$$

(indexes are modulo m).

Since there are finitely many tube offsets for each angle the corresponding support lines are quantized. The possible offset distances for half angles will vary from those for full angles by half the width of a tube. Consequently it may not

be possible to achieve the ideal constraint of $h^T C \leq 0$. Instead, the constraint of $h^T C \leq c$ is used, where c is equal to half of the maximum width of a tube:

$$c = r_g \frac{\sin(\frac{2\pi}{m})}{2}$$

(r_g is the radius of the gantry and m is the number of angles)

2.4 Recapitulation

From the initial D_{mask} generated from a moving average filter we can extract a possibly noisy support vector. Determining h values from tube offsets is simply a matter of looking up the distance given by the gantry geometry. The distance used is the distance to the center of the tube. Alternatively distance to the inside or outside edge of the tube could be used (with a large number of angles this makes very little difference as long as all tube distance are consistent). From the given h we can use G-S optimization to determine a new h which is a support vector. This then can be mapped back to D as a consistent D_{mask}.

Chapter 3

B_{mask} Creation

The purpose in creating a B_{mask} is to reduce the number of pixels for which computation must be done.

3.1 B_{mask} Description

A consistent D_{mask} in D corresponds to a support vector in G. This support vector defines a convex hull which surrounds the area of interest. From this area in G a set of relevant pixels in B can be determined. This set of pixels is the B_{mask}. Pixels in the B_{mask} are exactly those which are used in the image reconstruction.

It is obvious that pixels completely enclosed in the convex hull defined by the D_{mask} should be included in the B_{mask}. It is also obvious that pixels which have no area in common with the convex hull should not be included. The B_{mask} should contain the minimal set of pixels which completely covers the area of the previously mentioned convex hull. Thus all pixels which have any area in common with the hull will be included. An illustration is given in Fig. 3.1. Notice that this results in some pixels around the edge which only partially intersect the hull.

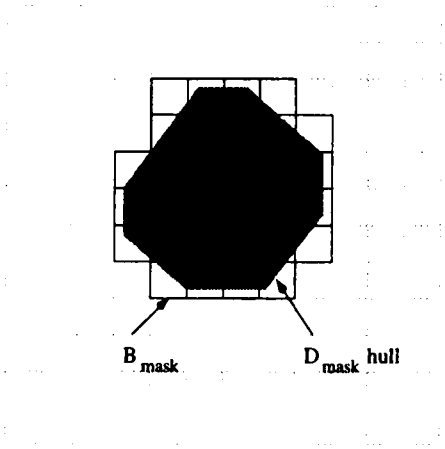


Figure 3.1: B_{mask} Pixels

An alternative choice of B_{mask} might be to compromise near the edge and only include pixels if they have at least a certain area of intersection with the hull. This would result in portions of the hull not covered by any pixels. Such a condition is undesirable since all emissions from within the hull are accounted for in some tube $d \in D_{mask}$. If not all areas of the hull were accounted for by some pixel then some emissions from the uncovered areas would have to be accumulated elsewhere. In the resulting image this could appear as some excessively bright pixels around the edges.

3.2 D_{mask} Conversion

The conversion from D_{mask} to B_{mask} can be accomplished by sequentially removing pixels one angle at a time. As mentioned in the previous chapter, the D_{mask} provides a support vector h for the object of interest in G. This support lines of this vector are the outermost edges of tubes. Given the support vector it is

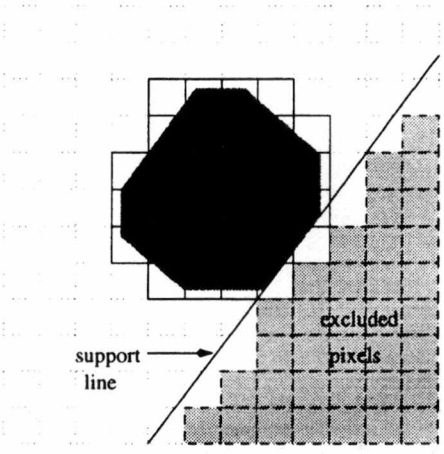


Figure 3.2: B_{mask} Selection

possible to mark pixels which are known not to be in B_{mask} for each row in B. Fig. 3.2 shows an example of pixels excluded from the $Bmask$ for one support line. Once pixels are marked for each support line there remains an unmarked region in the interior which contains exactly the pixels in the B_{mask}.

As previously mentioned, a finite number of tube offsets results in a certain inaccuracy in the convexity of the hull defined by D_{mask}. This is caused by the difference between half angle and full angle offsets. More explicitly, a given support line can be in error up to a distance of half the width of a tube. This distance depends on the offset since tube widths vary, so the largest tube width is used for generalization.

To compensate for this discrepancy, support lines should be moved outwards when generating the B_{mask}. The distance which these lines should be moved outwards depends on the distance of inconsistencies in the D_{mask}. More explicitly,

lines should be moved outwards a distance of

$$cos(\frac{2\pi}{m})r_g\frac{sin(\frac{2\pi}{m})}{2}$$

where r_g is the gantry radius and m is the number of angles. This will yield a slightly larger hull which is at least touched by the outer edge of all tubes in D_{mask}. The larger hull estimate helps to ensure that no part of the emitter object is cut off in the process of mask creation.

Chapter 4

P matrix Construction

4.1 *P matrix* Notation

The EM reconstruction algorithm converts emission count data from D to B. For this it is necessary to have a probability measure relating tubes and pixels. This measure, p_{bd}, is the probability that an emission was detected in tube d given that the emission came from pixel b.

This probability depends upon the gantry geometry, more precisely the areas of intersection between tubes and pixels. Given a tube d and a pixel b their area of intersection is a_{bd}. From these areas the probability value is given by the following:

$$p_{bd} = \frac{a_{bd}}{\sum_{d'} a_{bd'}}$$

4.2 Area Calculation

In order to determine p-matrix values, areas of intersection between tubes and pixels must first be calculated. This means computing a_{bd} for all combinations $b \in B$ and $d \in D$ such that pixel b is intersected by tube d. These area calculations

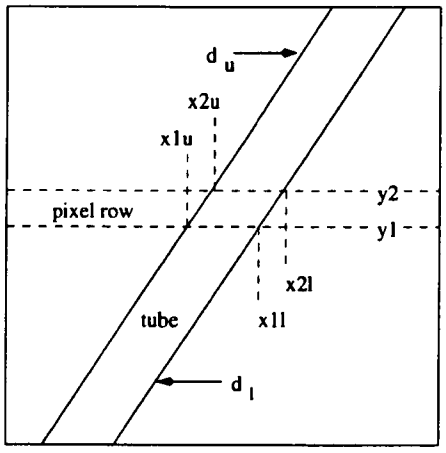


Figure 4.1: Tube-Pixel Intersection

can be done with an automaton as described below.

First it is necessary to describe some aspects of the relation between tubes and pixels (see Fig. 4.1 for an illustration). Each tube is defined by two lines (a slope and two y-intercepts) in G, call them d_u and d_l. Similarly, each row of pixels is given by two horizontal lines, or two y-values, $y1$ and $y2$. Thus, the intersection of a pixel row and a non-horizontal tube gives four x-values, $x1u$, $x2u$, $x1l$, and $x2l$. Two of these values, depending on the sign of the slope, are used to determine the leftmost and rightmost pixel in a row intersected by a tube.

Individual pixels can be classified with respect to the lines of a tube. A pixel's classification depends on the slope of the tube and the location of the pixel's left and right x values relative to $x1u$, $x2u$, $x1l$, and $x2l$. There are five pixel classifications for tubes with positive or negative slope. Horizontal or vertical tubes yield only two pixel classifications. See Fig. 4.2 for an illustration of these pixel types.

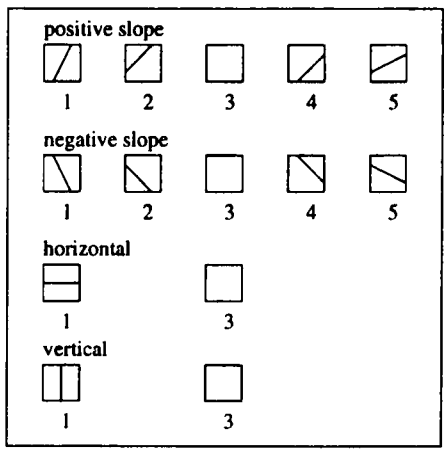


Figure 4.2: Pixel Types

The automaton for area calculation keeps track of pixel types with two state variables, *ncu* (next case upper) and *ncl* (next case lower). The values of these variables correspond to how pixels are intersected by a tube's upper and lower defining lines, as shown in Fig. 4.2. State transitions, shown in Fig. 4.3, correspond to moving to the next pixel in a row. Nondeterminism in the diagram is resolved by comparison of a pixel's x-values, $x1p$ (left) and $x2p$ (right), with $x1u$, $x2u$, $x1l$, and $x2l$. For example, if *ncu* is positive slope type 4 and $x2u < x2p + d_{pixel}$ then *ncu* will make the transition to positive slope type 2. Otherwise, $x2u > x2p+d_{pixel}$, *ncu* becomes positive slope type 5.

Note that the automaton starts each row with the first pixel intersected by d_u. Similarly, it finishes a row after the last pixel intersected by d_l. Thus there is a slight difference in transitions for *ncu* and those for *ncl*. E.g., different transitions to and from state 3 are allowed for *ncu* and *ncl*.

For a tube d, let the area of a pixel below d_u be denoted a_u and the area

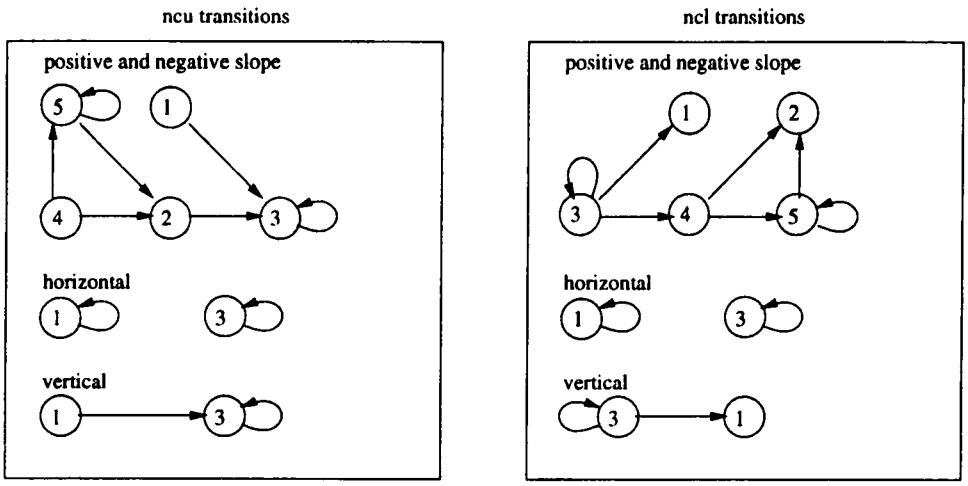


Figure 4.3: Automaton Transitions

below d_l be denoted a_l. Then a pixel's area of intersection with a tube is simply $a_u - a_l$. Calculation of a_u and a_l is simple if the pixel's classification is known. For example, if *ncl* is negative slope type 2 then a_l is the area of a triangle. So when the automaton is in a particular state a simple lookup determines the appropriate area calculations for the corresponding pixel.

In summary, the area calculation process is as follows. Given a tube the automaton is used to calculate area's of intersection for each row of pixels in B_{mask}. For a given row of pixels, the first and last pixels intersected by the tube are first determined. The initial values for *ncu* and *ncl* correspond to the classification of the first pixel (in that row) which is intersected by the tube of interest. Next, the areas of intersection for all relevant pixels for that row can be sequentially calculated. Areas for each tube are stored in an array which is indexed by tube (a_{bd} values). Then areas for each pixel (within the mask) are accumulated ($\sum_{d'} a_{bd'}$) so that p_{bd} values can be calculated.

Chapter 5

n^* Edge Reduction

5.1 Edge Packing

Recorded tube counts, n_d^*, correspond to the entire region of the gantry intersected by the tubes. Using a B_{mask} will generally reduce the area of the gantry which is involved in image reconstruction. This means that there may be a discrepancy between the area from which emissions were recorded and the area to which emissions are attributed by the reconstruction algorithm. If no emissions occurred outside of the area of B_{mask} there is no problem. However, this is most likely not the case due to noise and background radiation.

Tubes near the center of the sinogram (with small offsets) will typically intersect a large enough portion of the B_{mask} that the area discrepancy will be relatively small. Tubes near the edge of the D_{mask} (with higher offsets) will typically only intersect pixels on the periphery of the B_{mask} so that the area discrepancy is much larger.

Fig. 5.1 shows an example of how a tube, d, on the edge of the D_{mask} intersects the B_{mask}. The dark shaded region shows the area, $\sum_{b \in B_{mask}} a_{bd}$, to which emis-

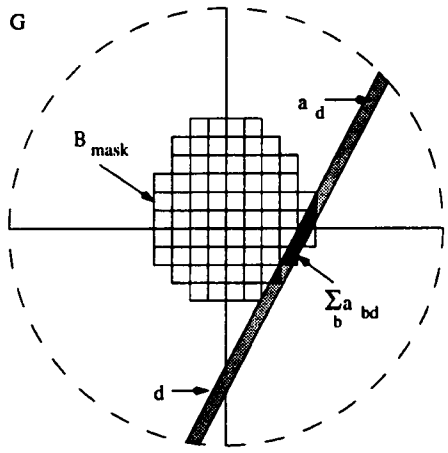


Figure 5.1: Edge Tube

sions are attributed in the reconstruction algorithm. The lighter shaded region shows the area, a_d, from which emissions were actually detected.

Edge tubes tend to correspond to areas with background emissions. Thus, emissions from an edge tube are likely uniformly distributed across the tube's area. Since this area is significantly reduced (by the B_{mask}) the pixels in B_{mask} which intersect edge tubes will have to have disproportionately large λ_b values to compensate. This is the case for tubes which intersect very few pixels if their n_d^* value remains as originally detected.

5.2 Count Reduction

To deal with the edge packing problem, one can reduce emission counts, n_d^*, for tubes on the edge of D_{mask}. Values should be reduced in proportion to $\frac{\sum_b a_{bd}}{a_d}$ where a_d is the total area in G intersected by d. (Alternatively, the total area in B intersected by d is used as a_d since it is easier to compute.) This proportion was

chosen considering that emissions are generally distributed uniformly across tubes on the edge of D_{mask} (background tubes). Such tubes generally only intersect pixels with background levels, so this is a reasonable assumption.

Emission counts for interior tubes remain unchanged by this method. This means that any count attributable to background radiation will be spread across the remaining pixels intersected by a tube. Interior pixels tend to have a high count relative to the background so the percentage change in these pixels should be small (as long as the distribution is uniform).

5.2.1 Alternate methods

Another method of dealing with the edge packing problem is to expand the B_{mask} so that there are more pixels in which to distribute background radiation. The size of B_{mask} relative to D_{mask} affects the ratio $\frac{\sum_{b \in B_{mask}} a_{bd}}{a_d}$ for edge tubes. Expanding B_{mask} will create a ring of pixels containing background radiation around the periphery of the object. This method will slightly affect the computation required, but it does not require modification of n_d^* values for any tubes.

Chapter 6

Experimental Work

6.1 Implementation

The EM algorithm used is implemented in C for a CM-5 connection machine (TMC, Cambridge, MA) operating as a 32 processor MIMD architecture. The algorithm is parallelized so that each node computes for an equal number of tubes. Communication between nodes is achieved by array passing (using combiner operators, as described in the CMMD manual [1]).

Scan data for our implementation is obtained by an ECAT 921 PET scanner (CTI/Siemens, Knoxville, TN). The gantry contains 384 detectors of which the innermost 80 pairs from any given angle are used. Hence, sinograms contain data for 384 angles (including half angles) and 80 offsets (i.e., D space is 384 by 80 tubes). The reconstructed image space, B, is 128 by 128 pixels.

6.2 Load Balancing

The main work in the problem is that of the EM iterations. For this stage, the load on each node is proportional to the number of P *matrix* nonzeros stored on

the node. The work is divided such that each node handles an equal portion of tube space D. Without masking this means that all nodes will compute for equal numbers of tubes, and that the average number of pixels intersected by a tube is roughly equal for all nodes.

Since masks are used to limit the number of tubes and pixels which are relevant in the algorithm, the distribution of tube-pixel pairs is in question. After observing the results of several test cases, it was determined that nodes must compute for a roughly equal number of tube-pixel pairs.

6.3 Test Cases

Several test cases were used to test the stages of the pipeline and obtain results on performance. These cases correspond to real emission data, n_d^*. Table 6.1 shows the test cases with the corresponding parameters required for each run. (Unless otherwise stated results for the brain test case were generated with a zoom factor of 2. Also, unless stated, results for all cases correspond to a threshold of 320.)

Image offset is measured in gantry units (centimeters). All emission maps except for the brain are already centered in the gantry, so the brain is the only case with a nonzero offset.

Pixel diameter for a zoom factor of 1 is 4.55 mm, so for zoom 2 pixel diameter is 2.275 mm.

The threshold parameter relates to the moving average sinogram filtering. A threshold of 320 corresponds to approximately 1% of the maximum n_d^* level. The ideal threshold is somewhat dependent on the emission data, so a threshold of 640 (2% of max) is also considered. For all cases, the moving average uses a window width of 7 offsets.

Table 6.1: Test Cases

Test Case	*Sinogram*	*Offset (x,y)*	*Zoom*	D_{mask} *Threshold*
Brain	rb_half.scn	(0,-5)	1/2	320/640
Cardiac	rh_half.scn	(0,0)	1	320/640
Pancreas	mil_half.scn	(0,0)	1	320/640
Phantom (synthetic)	phantom_half.scn	(0,0)	1	320/640

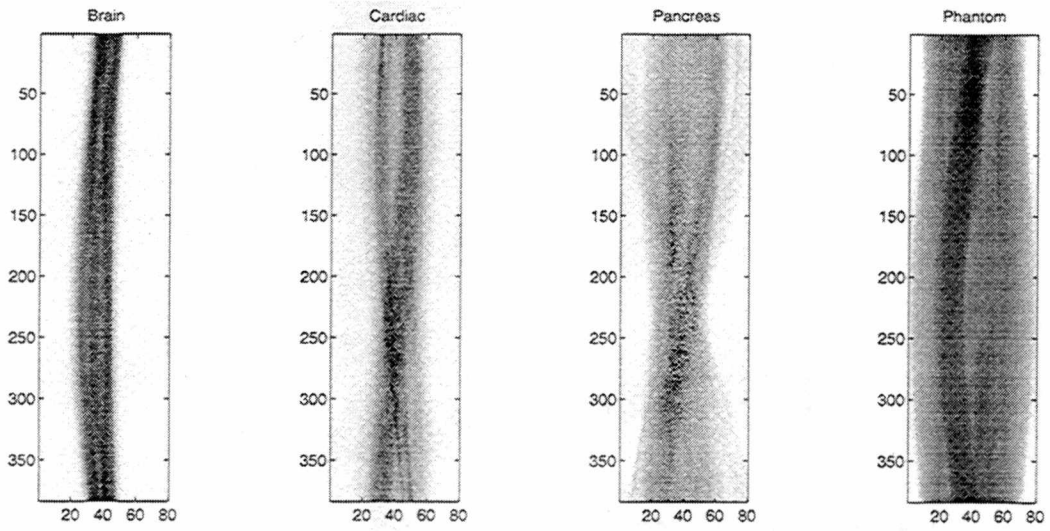


Figure 6.1: Test Case Sinograms

Figure 6.1 shows sinograms for each of the test cases.

6.3.1 D_{mask} Results

Figures 6.2-6.3 are plots representing D_{mask} for each test case. The shaded region of each plot indicates all tubes in the optimized D_{mask}. To the right of each optimized mask is a comparison to the original mask obtained from sinogram filtering. Tubes shown are those added to the original mask by D_{mask} optimization.

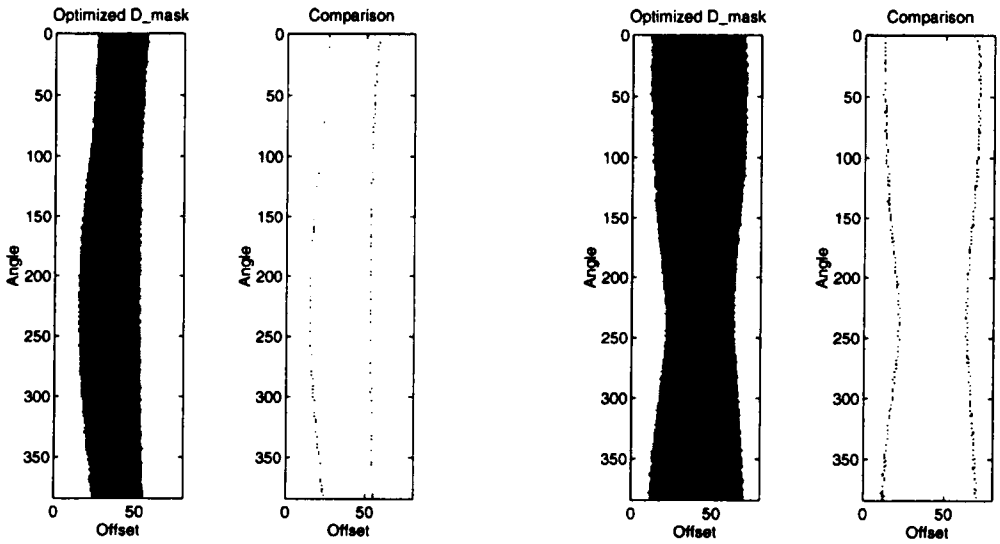


Figure 6.2: Brain D_{mask}, Cardiac D_{mask}

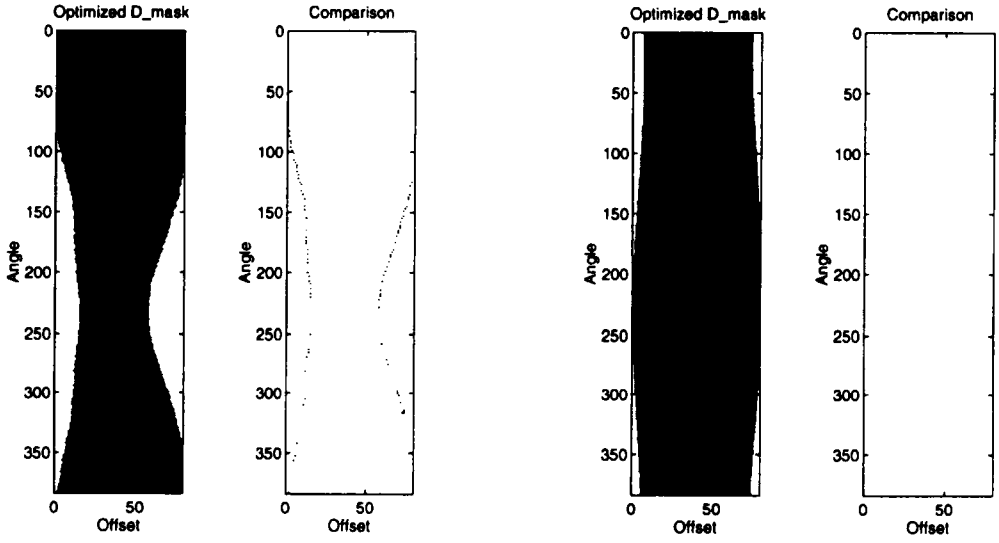


Figure 6.3: Pancreas D_{mask}, Phantom D_{mask}

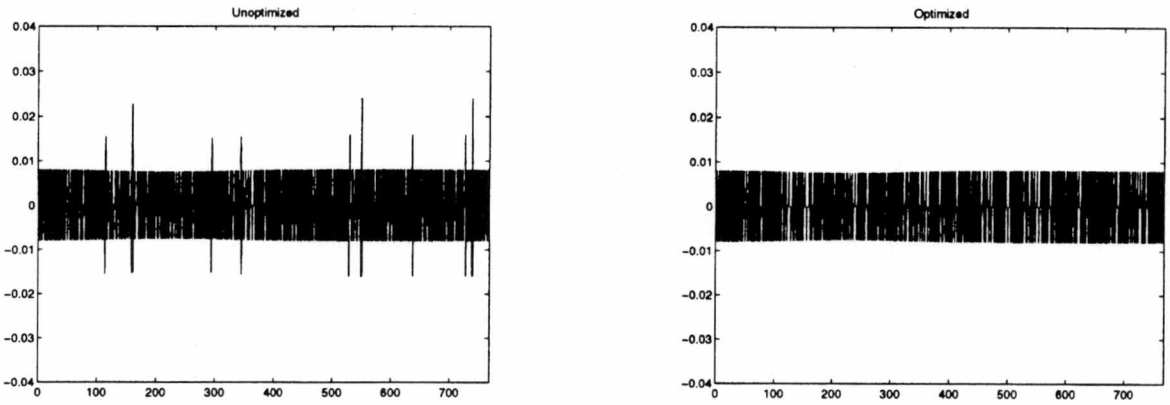


Figure 6.4: Brain $h^T C$

Convex Support

From the support theorem [3] (see Chapter 2) we know the conditions under which a vector h is a support vector, i.e, $h^T C \leq [0 \cdots 0]$. A comparison of $h^T C$ for h values given by D_{mask} illustrates the effect of convexity optimization.

Figures 6.4-6.7 show plots of $h^T C$ for each test case. Both unoptimized and optimized D_{mask}'s are represented. Due to the quantized nature of the tube mask, some *digital noise* is unavoidable. The jagged oscillation between positive and negative values is due to the interlacing of half angles and full angles. The range of this oscillation is approximately equal to half the width of a tube and thus varies slightly with offset. Peaks in $h^T C$ represent inconsistent support lines and are only observed before optimization.

Relatively few iterations of D_{mask} optimization are required, as shown in Table 6.2. In most cases, the moving average sinogram filter produces a fairly smooth mask. Thus only a few tubes must be added to the edge to make the mask consistent.

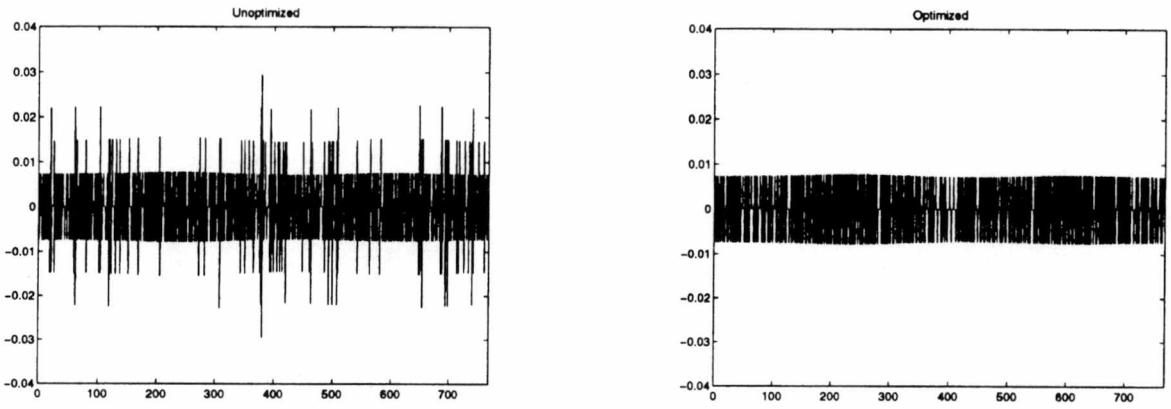


Figure 6.5: Cardiac $h^T C$

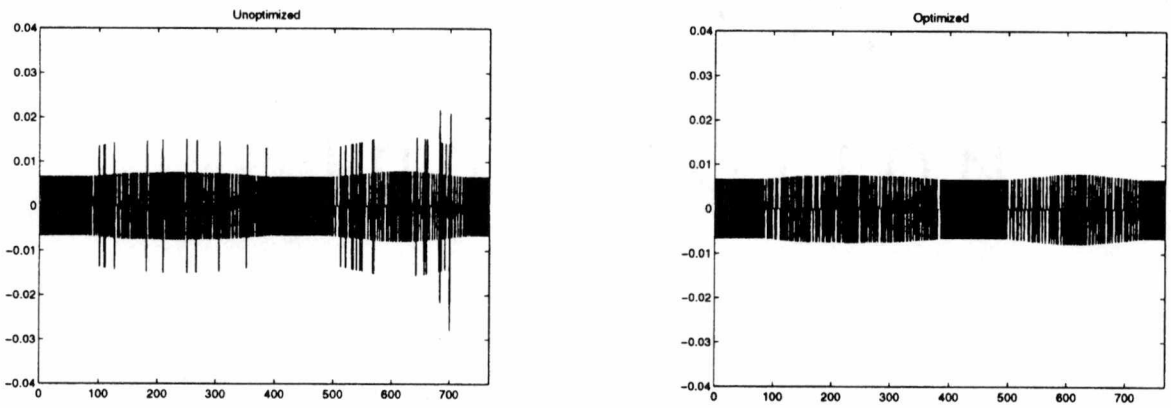


Figure 6.6: Pancreas $h^T C$

Table 6.2: Convexity Optimization Iterations

Test Case	*Number of Iterations*
Brain	2
Cardiac	5
Pancreas	2
Phantom	0

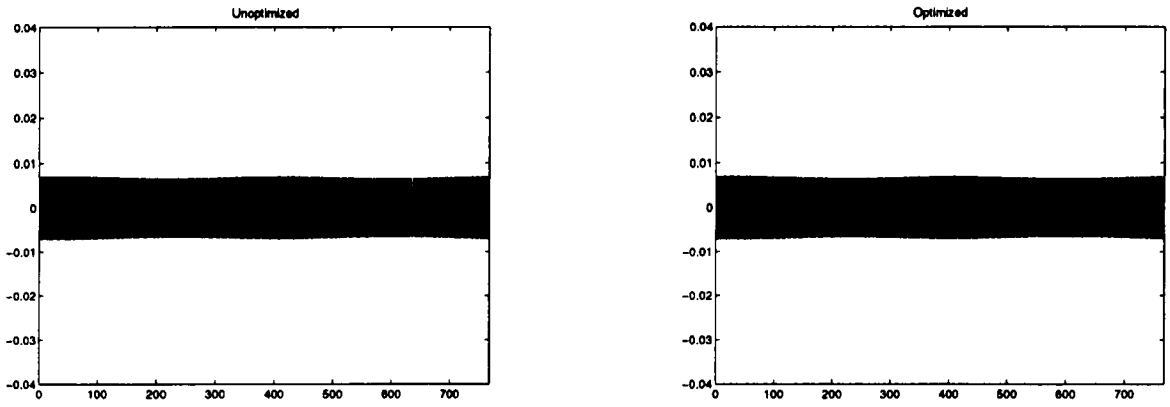


Figure 6.7: Phantom $h^T C$

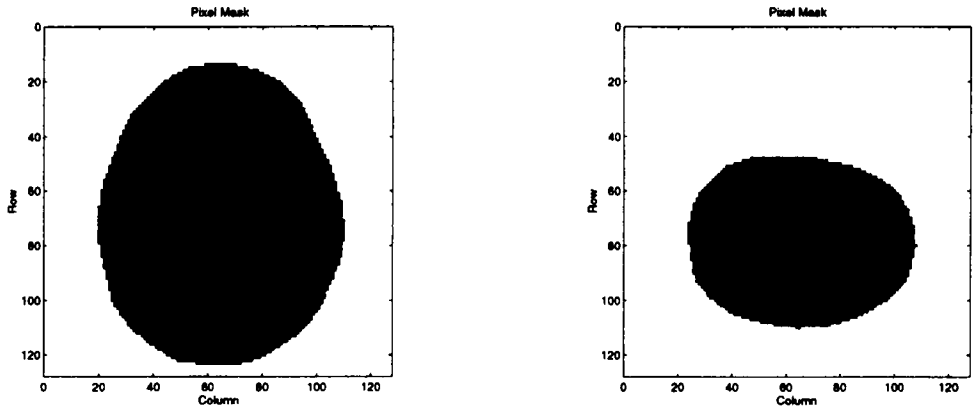


Figure 6.8: Brain B_{mask}, Cardiac B_{mask}

6.3.2 B_{mask} Results

Figures 6.8-6.9 are plots representing B_{mask} for each test case. The shaded region of each plot includes all pixels in B_{mask}.

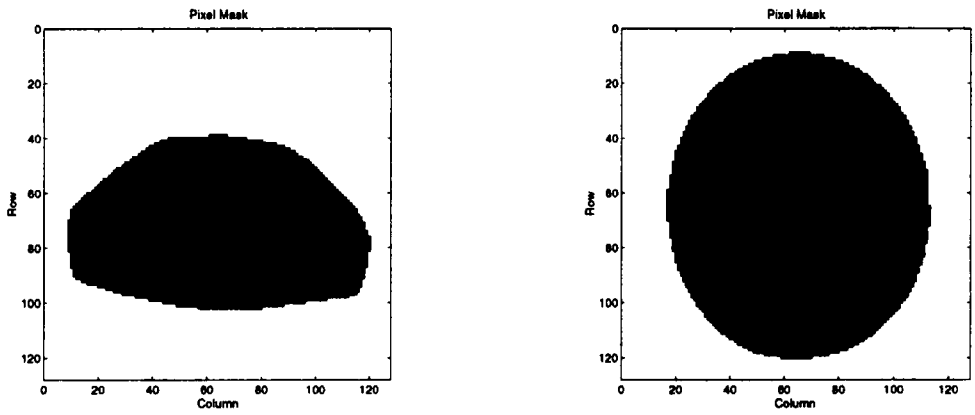


Figure 6.9: Pancreas B_{mask}, Phantom B_{mask}

6.3.3 P matrix Results

Let $p_{ratio} = \frac{p_{bd}}{\sum_{b'} p_{b'd}}$, the proportion of n_d^* contributed by pixel b. If p_{ratio} is close to 1 then tube d will get most of its count from pixel b. Figures 6.10-6.11 show histograms of p_{ratio} values for a chosen pixel on the edge of B_{mask} for all cases. The pixels represented are among the brightest edge pixels. These histograms indicate how many tubes have high p_{ratio} values relative to the given pixel, i.e., the number of tubes which attribute most of their count to the pixel.

Histograms for the brain and cardiac scans show that most tubes intersecting these pixels have low p_{ratio}. For the case of the generated phantom, the values of this ratio are similarly small. More specifically, no tubes here have a value of p_{ratio} greater than 0.04. This means that all tubes through these pixels intersect at least 25 pixels in the corresponding B_{mask}.

The case of the pancreas shows some tubes with slightly higher values of p_{ratio}. One tube through the pixel intersects as few as 8 pixels in B_{mask}, with a p_{ratio} of just over 0.12. Still, these values are generally small due to the size of B_{mask} relative to D_{mask}. Were B_{mask} smaller these ratios would tend to go up, as more

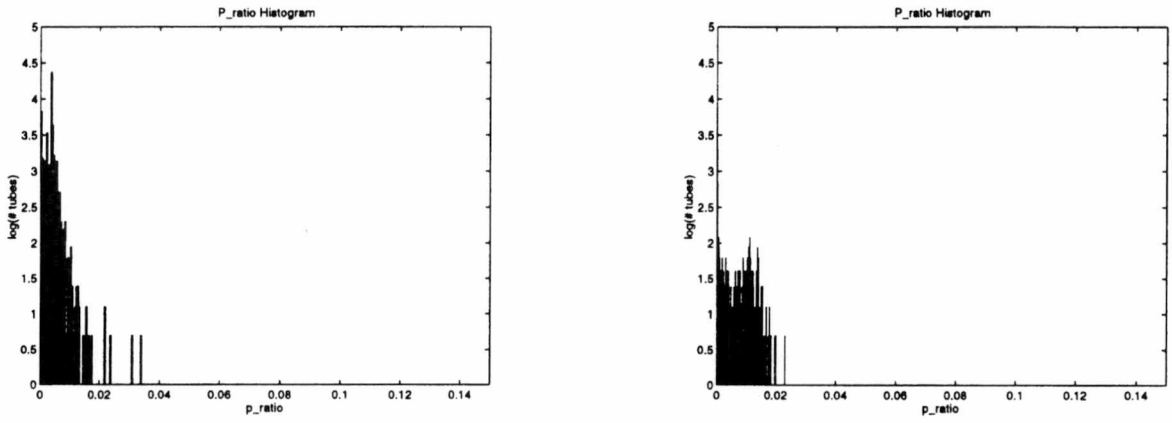


Figure 6.10: Brain Histogram, Cardiac Histogram

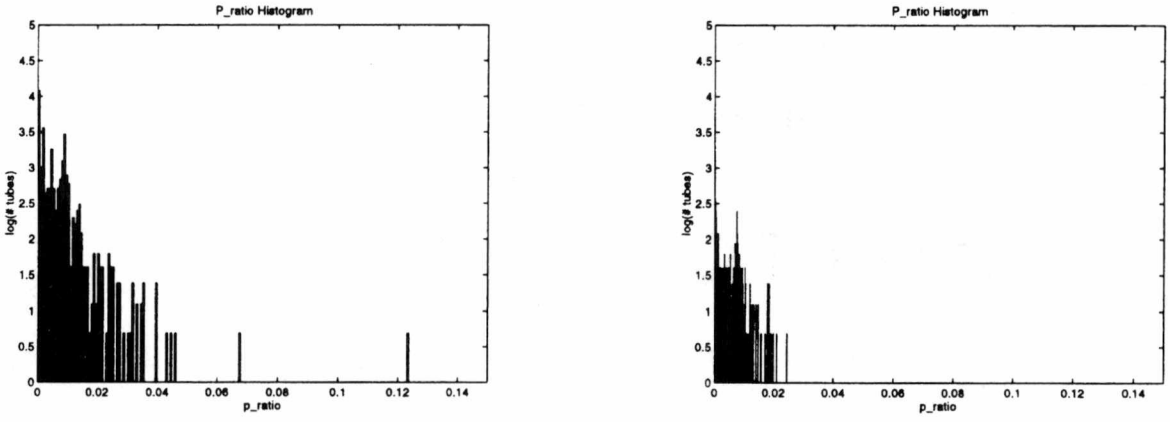


Figure 6.11: Pancreas Histogram, Phantom Histogram

pixels intersecting edge tubes are removed.

6.3.4 Resulting Images

Images generated by 8 iterations of the EM algorithm are given in Figures 6.12-6.15. Masked, edge reduced, and non-masked images are given from left to right for each test case. Although image quality is somewhat subjective, the masked image evidently provides an equally accurate, if not cleaner, picture of the emission map.

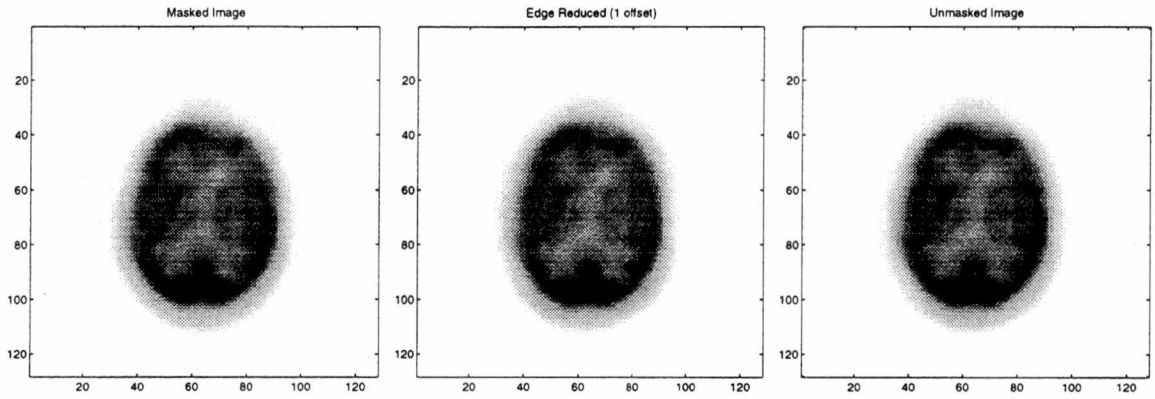


Figure 6.12: Brain Images

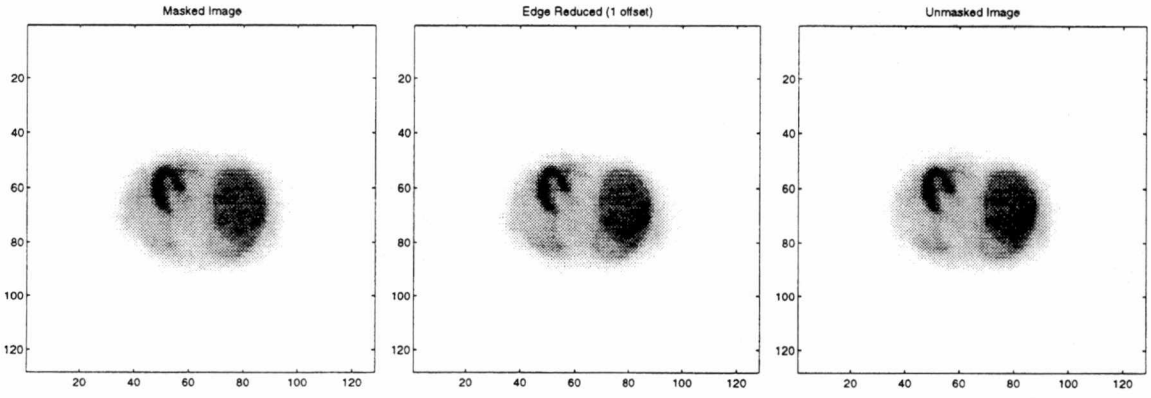


Figure 6.13: Cardiac Images

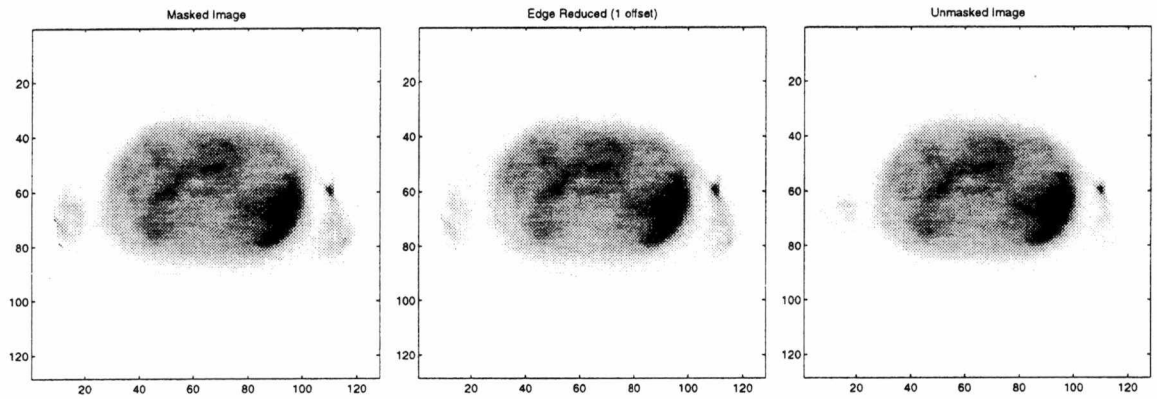


Figure 6.14: Pancreas Images

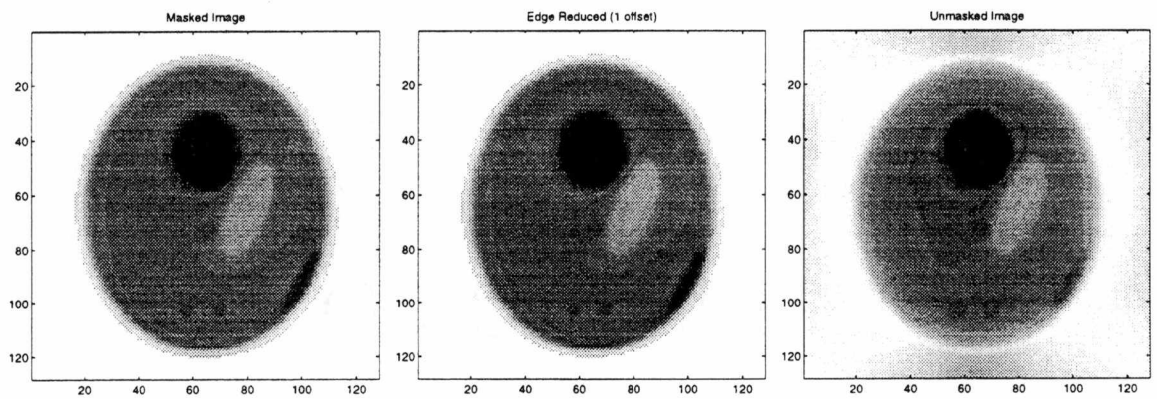


Figure 6.15: Phantom Images

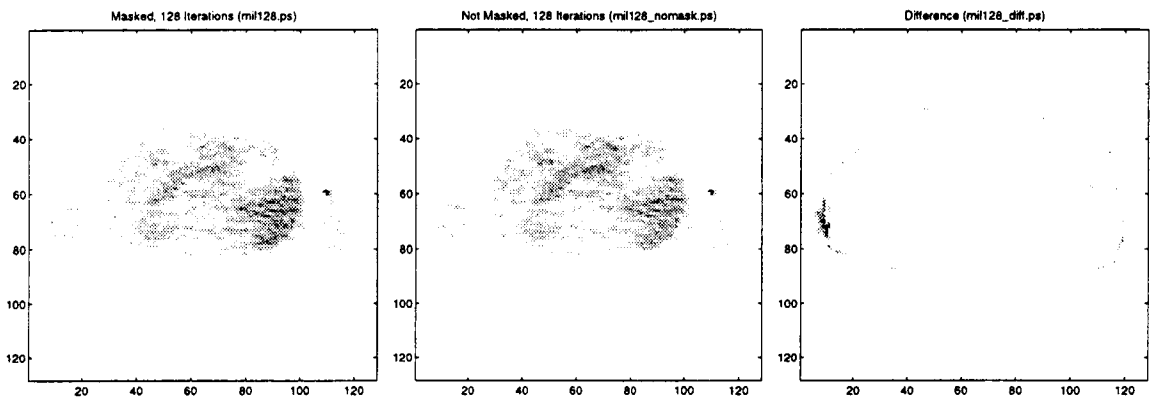


Figure 6.16: Pancreas Images (128 Iterations)

Figure 6.16 compares the pancreas test case after 128 EM iterations. Along with the masked and non-masked cases, the difference (absolute value) between them are given. The only apparent discrepancy is just around the edge of the mask.

6.3.5 Memory Consumption

The benefits of masking can be seen in a comparison of array sizes relating to memory consumption, i.e., the number of tubes, pixels, and entries in the *P matrix*. Both D_{mask} and B_{mask} can be represented by a fixed size array, but the number of *P matrix* entries directly reflects storage space required.

As a benchmark, Table 6.3 shows array size without masking based on zero offset and a zoom factor of 1. The number of entries in the *P matrix* corresponds to the relation between tubes and pixels and can vary with image offset and zoom factor.

Tables 6.4-6.5 show array size for all test cases with a D_{mask} threshold of 1% and 2% (of maximum emission rate) respectively. Beside each size, the percentages

Table 6.3: Array sizes without masking

Data	*Without Masking*
D_{mask} (tubes)	30,720
B_{mask} (pixels)	16,384
P matrix (nonzeros)	10,260,792

Table 6.4: Array sizes (1% threshold)

Data	*Brain*		*Cardiac*
	Zoom 1	*Zoom 2*	
D_{mask} (tubes)	13,164 (43%)	13,164 (43%)	19,685 (64%)
B_{mask} (pixels)	2,005 (12%)	7,805 (48%)	4,277 (26%)
P matrix (nonzeros)	1,431,344 (14%)	4,288,967 (42%)	3,077,101 (30%)
Data	*Pancreas*	*Phantom*	
D_{mask} (tubes)	24,863 (81%)	28,421 (93%)	
B_{mask} (pixels)	5,633 (34%)	8,537 (52%)	
P matrix (nonzeros)	4,078,263 (40%)	6,208,722 (61%)	

relative to corresponding non-masked size is given. For the masks, these are percentages of tubes or pixels relative to the total tube or pixel space. For the *P matrix*, these are percentages of nonzeros relative to the unmasked *P matrix*. In most cases, the number of nonzeros is reduced by over 50% through masking.

Table 6.5: Array sizes (2% threshold)

Data	*Brain*		*Cardiac*
	Zoom 1	*Zoom 2*	
D_{mask} (tubes)	12,438 (40%)	12,438 (40%)	18,141 (59%)
B_{mask} (pixels)	1,813 (11%)	7,038 (43%)	3,685 (22%)
P matrix (nonzeros)	1,290,371 (13%)	3,860,851 (38%)	2,645,928 (26%)
Data	*Pancreas*	*Phantom*	
D_{mask} (tubes)	24,002 (78%)	27,114 (88%)	
B_{mask} (pixels)	5,282 (32%)	7,882 (48%)	
P matrix (nonzeros)	3,821,036 (37%)	5,720,816 (56%)	

Table 6.6: Execution CPU Time without masking

Pipeline Stage	*Time*
D_{mask} Creation	n/a
D_{mask} Optimization	n/a
B_{mask} Creation	n/a
P matrix Creation	4.7940
Edge Reduction (1 offset)	n/a
EM Iterations (8)	7.8512
Total Time (seconds)	12.6452

6.3.6 Timings

Computation varies based on the number of entries in the *P matrix* so execution times will also. Table 6.6 gives user CPU times (in seconds) for execution of 8 iterations of the EM algorithm without masking. Times are averaged over 10 runs with no offset and a zoom factor of 1. Compared to times for masked runs, significantly more time is required for non-masked execution.

Tables 6.7-6.8 give the timings for various test cases (1% and 2% threshold respectively). For each case, times reflect 8 iterations of the EM algorithm averaged over 10 runs. Total times given are calculated from runs without reduction of n_d^* edge values. Percentages of time relative to total unmasked time are given for comparison.

A comparison of execution time of the EM algorithm versus *P matrix* size validates their correspondence. Figure 6.17 shows a plot of this comparison for all data points obtained from test case runs (arranged in increasing order).

In addition to enhanced performance of the EM algorithm, time required for creation of the *P matrix* is reduced through masking. Creation of this matrix is divided first by tubes, then by pixels. Because of this implementation, the time required for this stage of the pipeline depends on the size of the D_{mask} and the number of pixels intersected per tube.

Table 6.7: Execution CPU Time (1% threshold)

Pipeline Stage	*Brain*		*Cardiac*	*Pancreas*	*Phantom*
	Zoom 1	*Zoom 2*			
D_{mask} Creation	0.0023	0.0023	0.0021	0.0021	0.0017
D_{mask} Optimization	0.1791	0.1644	0.2331	0.1722	0.1510
B_{mask} Creation	0.1686	0.1519	0.1519	0.1423	0.1568
P *matrix* Creation	1.2517	1.9045	2.2495	3.0022	3.4155
Edge Reduction (1 offset)	0.1636	0.1565	0.2061	0.1550	0.1678
EM Iterations (8)	1.5072	3.1891	2.4132	2.9477	4.2552
Total Time (seconds)	3.1088	5.4122	5.0499	6.2665	7.9802
(% of unmasked time)	(25%)	(43%)	(40%)	(50%)	(63%)

Table 6.8: Execution CPU Time (2% threshold)

Pipeline Stage	*Brain*		*Cardiac*	*Pancreas*	*Phantom*
	Zoom 1	*Zoom 2*			
D_{mask} Creation	0.0023	0.0023	0.0022	0.0021	0.0018
D_{mask} Optimization	0.1529	0.1560	0.2427	0.2130	0.1265
B_{mask} Creation	0.1698	0.1500	0.1653	0.1420	0.1590
P *matrix* Creation	1.1494	1.7691	2.0481	2.8953	3.2025
Edge Reduction (1 offset)	0.1821	0.1803	0.1739	0.1761	0.1686
EM Iterations (8)	1.4331	2.9358	2.1615	2.7888	3.9216
Total Time (seconds)	2.9074	5.0132	4.6199	6.0412	7.4114
(% of unmasked time)	(23%)	(40%)	(37%)	(48%)	(59%)

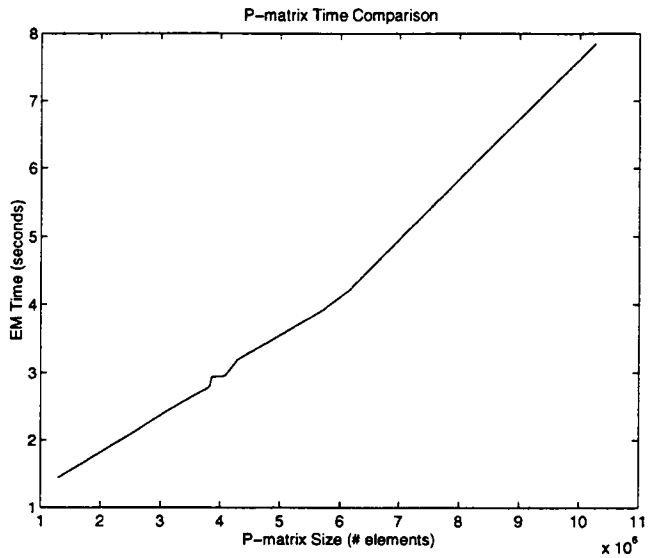


Figure 6.17: P matrix Time Comparison (all cases)

To further illustrate the advantage of reducing the *P matrix* size through masking, Figure 6.18 shows a comparison using the pancreas test case. The graph compares data from runs of 2 to 128 iterations (including all powers of 2). Each iteration takes approximately the same amount of time, so the time advantage of masking increases linearly with the number of iterations. This figure shows that CPU time has been cut in half with this focus of attention technique.

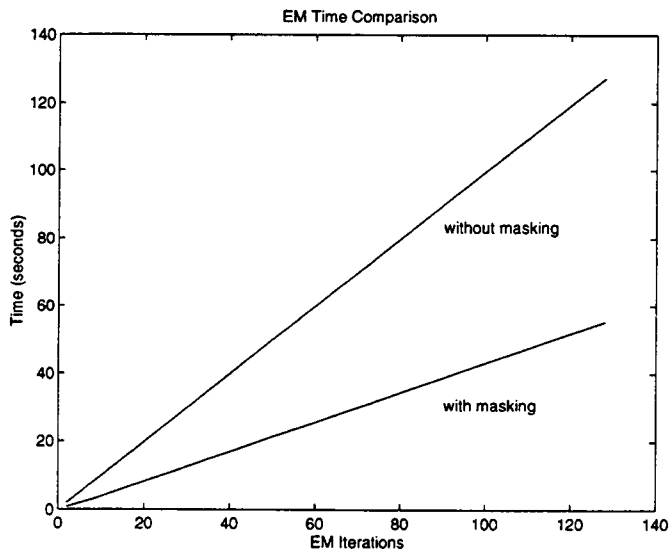


Figure 6.18: Iteration Time Comparison (pancreas case)

Chapter 7

Concluding Remarks

7.1 Benefits

The addition of masking to the EM reconstruction reduces the size of the problem. Computational requirements for the added components of the pipeline are small compared to the savings they provide (at least an order of magnitude smaller). Thus the net effect of masking is to reduce the time and space required for the EM algorithm. On average, CPU time and memory consumption can be reduced by 50% each as compared to the unmasked algorithm. We would also argue that the resulting image quality is certainly not degraded and perhaps even improved in the process.

7.2 Further Possibilities

Some areas of the implementation of the masked EM pipeline could perhaps be improved. Improvements might help expand the benefits achieved through the focus of attention technique by masking.

Another algorithm for sinogram filtering to produce a D_{mask} could be devel-

oped. One difficulty with the moving average filter is that it requires a threshold parameter. Using a globally selected value for this parameter might not yield maximum benefits for all cases. A *moving histogram* filter similar to the moving average might be a possibility. Such a filter would examine the relative distribution of emission levels within a window for each profile. It is possible that this distribution could be used in a filter which is independent of sinogram data.

The edge packing problem could perhaps also benefit from more sophisticated techniques. As mentioned, the important factor seems to be the relation between pixels and tubes (i.e., the size of D_{mask} relative to B_{mask}). An enhanced solution of this problem could perhaps improve overall image quality.

Additional improvement could possibly be achieved through knowledge of noise sources leading to background emissions. A better model of background emissions might include statistics about background tubes. In this way these tubes could be more efficiently isolated and removed from computation. The same statistics could also be used to deal with problems like edge packing.

Bibliography

Bibliography

[1] *CMMD Reference Manual.* Thinking Machines Inc., Cambridge, MA, May 1993.

[2] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and Distributed Computation.* Prentice Hall, Englewood Cliffs, NJ 07632, 1989.

[3] J. L. Prince and A. S. Willsky. Reconstructing convex sets from support line measurements. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(4):377–389, 1990.

[4] L.A. Shepp and Y. Vardi. Maximum likelihood reconstruction for emission tomography. *IEEE Transactions on Medical Imaging*, MI-1(2), 1982.

[5] L. A. Shepp Y. Vardi and L. Kaufman. A statistical model for positron emission tomography. *Journal of the American Statistical Association*, 80(389):8–20, 1985.

Vita

Dean A. Huff received the B.S. degree for Math and Computer Science from the University of Wisconsin, Madison, in 1993. From 1993 to 1995, while working on his M.S. in Computer Science at the University of Tennessee, Knoxville, he was employed at Oak Ridge National Laboratory (Environmental and Economic Modeling Section). There he assisted in developing and programming a genetic algorithm based model for long term technology adoption.