

The Radon Transform

Theory and Implementation



Peter Toft

Ph.D. Thesis

Department of Mathematical Modelling
Section for Digital Signal Processing
Technical University of Denmark

©Peter Toft 1996

Abstract

The subject of this Ph.D. thesis is the mathematical Radon transform, which is well suited for curve detection in digital images, and for reconstruction of tomography images. The thesis is divided into two main parts.

Part I describes the Radon- and the Hough-transform and especially their discrete approximations with respect to curve parameter detection in digital images. The sampling relationships of the Radon transform is reviewed from a digital signal processing point of view. The discrete Radon transform is investigated for detection of curves, and aspects regarding the performance of the Radon transform assuming various types of noise is covered. Furthermore, a new fast scheme for estimating curve parameters is presented.

Part II of the thesis describes the inverse Radon transform in 2D and 3D with focus on reconstruction of tomography images. Some of the direct reconstruction schemes are analyzed, including their discrete implementation. Furthermore, several iterative reconstruction schemes based on linear algebra are reviewed and applied for reconstruction of Positron Emission Tomography (PET) images. A new and very fast implementation of 2D iterative reconstruction methods is devised. In a more practical oriented chapter, the noise in PET images is modelled from a very large number of measurements.

Several packages for Radon- and Hough-transform based curve detection and direct/iterative 2D and 3D reconstruction have been developed and provided for free.

Resume på dansk (Abstract in Danish)

Emnet for denne Ph.D. afhandling er den matematiske Radontransformation, der er velegnet til detektion af kurver i digitale billeder og til rekonstruktion af tomografiske billeder. Afhandlingen er opbygget i to dele.

Del I beskriver Radon- og Hough-transformationen og specielt deres diskrete approximationer med henblik på estimation af kurve parametre i digitale billeder. Der er beskrevet samplingsrelationer for Radontransformationen ud fra et digital signalbehandlingssynspunkt. Den diskrete Radontransformation er undersøgt med henblik på detektion af kurver, og der er behandlet aspekter vedrørende metodens anvendelighed under antagelse af forskellige typer støj. Desuden er præsenteret en ny og hurtig metode for estimation af kurve parametre.

Del II af afhandlingen beskriver den inverse Radontransformation i 2D og 3D med fokus på rekonstruktion af tomografiske billeder. Flere af de direkte rekonstruktionsmetoder er analyseret inklusiv deres diskrete implementering. Desuden er der gennemgået en række lineær algebra baserede iterative rekonstruktionsmetoder, og de er anvendt til rekonstruktion af Positron Emission Tomografi (PET) billeder. En ny og meget hurtig implementering af 2D iterative rekonstruktionsmetoder er foreslægt. I et mere praktisk orienteret kapitel er støj i PET billeder modelleret ud fra et stort antal målinger.

Et sæt programpakker er blevet udviklet til Radon- og Hough-transformation baseret detektion af kurve parametre og til direkte/iterativ 2D og 3D rekonstruktion, og de bliver stillet gratis til rådighed.

Preface

The Ph.D. project has been carried out from March 1, 1993 to May 31, 1996 at the Department of Mathematical Modelling (before January 1, 1996 Electronics Institute), Technical University of Denmark with supervisors John Aasted Sørensen and Peter Koefoed Møller.

The image on the front page shows the surface of a brain generated by PET scanning. The measured sinograms have been reconstructed and the brain volume has been shown using a 3D visualization package.

Contents

This Ph.D. thesis entitled *The Radon Transform - Theory and Implementation* is divided into two main parts. Part I consists of Chapters 1 to 5 and Part II of Chapters 6 to 11. Appendices are collected in Part III, and finally Part IV contains the papers submitted to journals and conferences.

In Chapter 1, the Radon transform is presented in the form used within seismics. Discrete approximations are derived, and it is shown that the Radon transform is well suited for curve parameter estimation, and in this chapter a new way of analyzing sampling relationships is introduced. Several properties are presented along with a set of examples using discrete Radon transformation. Optimization strategies for implementation of the discrete Radon transform are given, and some of the limitations concerning the allowed interval of slopes are also presented. A way to circumvent this restriction is also given.

Another way of defining the Radon transform (using normal parameters) is used in Chapter 2, and sampling relationships are derived. It is shown how this form of the Radon transform is related to the form analyzed in Chapter 1, and that the two definitions mainly cover different types of images. In Chapter 2, the images are assumed quadratic and the lines can have arbitrary orientation.

A very popular Radon-like transform is the Hough transform, which is described in Chapter 3. Possibilities, limitations, and an optimization strategy are given along with a set of examples. Here it is also shown that the discrete Hough transform is identical to the discrete Radon transform, if some of the sampling parameters are restricted.

The Radon and the Hough transforms are generalized in order to handle more general parameterized curve types. The properties of the two transforms are then exploited in the FCE-algorithm [1, 2], which is proposed for fast curve parameter estimation. The potential of the algorithm is demonstrated in two examples.

One of the very strong features of the discrete Radon transform regards noise suppression, which is covered theoretically in Chapter 5. A novel analysis of the influence of both additive noise [3] and uncertainty on the line samples [4] is presented.

In Chapter 6 the thesis changes its aim and describes computerized tomography with respects to reconstruction of PET- and CT-images. A simplified description of the fundamental physics is given and it is motivated why the inverse Radon transform can be used for reconstruction of the measured sinograms.

Several of the common direct inverse Radon schemes are derived in Chapter 7. First using normal parameter, and later in this chapter similar inversion schemes are derived for slant stacking.

The implementation of the Radon based reconstruction methods impose the use of several different elements which are reviewed from a digital signal processing point of view in Chapter 8, but only for the Radon transform using normal parameters. Chapter 8 also includes a set of examples, made with a developed software package. This and other developed software packages are provided for free.

A very different approach for developing reconstruction algorithms is based on linear algebra and statistics. In Chapter 9 the basis of these methods are shown, and the relationship with that the direct reconstruction methods is reviewed. This chapter illustrates that a broad field of research areas have contributed directly or indirectly to the field of reconstruction methods. Iterative reconstruction methods are reviewed and a very fast implementation of 2D iterative reconstruction algorithms [5, 6] is proposed. A set of examples are included, where PET-images (or PET-like images) are reconstructed from noisy sinograms and the performance of the 2D fast iterative reconstruction package is reviewed.

Next Chapter 10, goes into reconstruction of volumes using 3D PET scanners. Some of the Radon transform based reconstruction methods are derived and some of the implementation aspects are reviewed. A software package has been developed, where Radon based and iterative reconstruction methods have been implemented. It is shown that most of the methods can be implemented efficiently on a parallel computer, and a few examples are presented.

The final chapter in the main thesis is Chapter 11, which is of a more practical nature. From a huge set of measurements on phantoms and humans the noise in reconstructed PET images has been modelled and model parameters have been estimated [7, 8].

It should be mentioned that a part of the work done in this project is far better presented using the World Wide Web tools of movies and virtual reality objects. It has been chosen to avoid color images in the thesis, even though that colors normally will enhance the visual impression. MPEG movies and 3D virtual objects can be found at the Human Brain Project WWW-server [9]. This thesis is available as a Postscript file, which can be down-loaded from [10].

Collaborations

Chapter 4 present work made in collaboration with Kim Vejlby Hansen. The work has been carried out as a joint venture project between Ødegaard & Danneskiold-Samsøe and Department of Mathematical Modelling (before January 1, 1996 Electronics Institute), Technical University of Denmark. The ideas and results have been presented at the *EUSIPCO Conference 94* in Edinburgh, Scotland, and at the *Interdisciplinary Inversion Summer School 94* in Mønsted, Denmark, and published in [1, 2]. These papers are shown in Appendices G and H. Kim Vejlby Hansen and I had a very long and good collaboration. He and Peter Koefod Møller are thanked for getting me into the area of the Radon transform in the first place.

Chapters 6, 7, and 8 are inspired by my stay at the National University Hospital in Copenhagen (Rigshospitalet) and by two masters projects carried out at that time. Software packages have been made for 2D direct reconstruction of PET images, and one for analytical generation of sinograms from a set of primitives. These packages can be used for quantifying the quality of reconstruction algorithms. I will like to thank Claus Svarer and Karin Stahr for making our stay

very good, and especially Søren Holm with whom I have had a long and fruitful collaboration. I have learned much about PET and tomography in general from him. My former students Peter Philipsen and Jesper James Jensen are acknowledged for their collaboration and hard work. We have spent many joyful hours together, and their efforts have meant much to me.

In Chapter 9 the fundamentals of linear algebra based reconstruction methods are covered with special focus on the implementation of iterative reconstruction methods. For this work I had the pleasure of working with Jesper James Jensen. We developed a very fast technique for implementing most 2D iterative reconstruction methods. This work has been submitted in [5] and [6], shown in Appendix L and M, respectively.

Some of the functions in the 3D reconstruction package presented in Chapter 10 has been made by Peter Philipsen and he helped by adding the compiler options needed to speed up the program on the Onyx-computer from Silicon Graphics (SGI).

Chapter 11 covers recent work made together with Søren Holm, where noise in PET reconstructed images has been modelled and the model parameters have been estimated from a huge set of measurements. The results have been presented at the *IEEE Medical Imaging Conference 95* in San Francisco USA, and published in a short version [7], shown in Appendix J, and submitted in a longer version in [8], shown in Appendix K.

In Appendix N the published papers [11, 12] are shown, where mean field techniques have been used to improve image quality by using strong priors in the restoration of PET reconstructed images. This work was made with Lars Kai Hansen and Peter Philipsen. It has been presented at the *Fourth Danish Conference for Pattern Recognition and Image Analysis 95* in Copenhagen, Denmark and at the *Interdisciplinary Inversion Conference 95* in Århus, Denmark.

Acknowledgments

I thank my two supervisors, Peter Koefoed Møller and John Aasted Sørensen for getting me into the project, their support during the years, and giving me very free limits, which I have enjoyed.

I am very grateful to Lars Kai Hansen for an inspiring collaboration and for his commitment to create a pleasant and dynamical environment at the department. He got me interested in medical imaging and laid many bricks along the way.

My room mates Søren Kragh Jespersen, Cyril Goutte, and Peter S. K. Hansen are acknowledged for their proofreading and support.

My wife Katja has directly and indirectly contributed to my work, and without her loving support I could never have managed getting this work done.

Thanks to the many WWW users, who have responded on my home page. Also thank you to all the programmers contributing to the Linux project.

Finally thanks to the other current and earlier Ph.D. students and staff at the Electronics Institute and Department of Mathematical Modelling, especially Jan Larsen, with whom I have had many hard and good discussions, Torsten Lehmann for introducing and helping me to use Linux in the early days, Mogens Dyrdal for support, and Simon Boel Pedersen for his friendly attitude and fantastic lectures in Digital Signal Processing.

Papers

During the project a total of 13 reports have been made, mainly for teaching purposes, and additionally several parts of this thesis has been submitted to conferences or journals:

- **P. A. Toft** and K. V. Hansen: “*Fast Radon Transform for Detection of Seismic Reflections*”, Signal Processing VII - Theories and Applications. Proceedings of EUSIPCO 94, pages 229-232. Shown in Appendix G.
- K. V. Hansen and **P. A. Toft**: “*Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform*”. Accepted for publication in IEEE Transactions on Image Processing. Shown in Appendix H.
- **Peter Toft**: “*Using the Generalized Radon Transform for Detection of Curves in Noisy Images*”. Proceedings of the IEEE ICASSP 1996, pages 2221-2225 in part IV. Shown in Appendix I.
- Søren Holm, **Peter Toft**, and Mikael Jensen: “*Estimation of the noise contributions from Blank, Transmission and Emission scans in PET*”. Accepted for publication in the Conference Issue of IEEE Medical Imaging Conference 95. Furthermore, the corresponding abstract can be found in the abstract collection of Medical Imaging Conference 95. Shown in Appendix J.
- Søren Holm, **Peter Toft**, and Mikael Jensen: “*Estimation of the noise contributions from Blank, Transmission and Emission scans in PET*”. Submitted to IEEE Transactions on Nuclear Science. Shown in Appendix K.
- **Peter Toft** and Jesper James Jensen: “*A very fast implementation of 2D Iterative Reconstruction Algorithms*”. Submitted to IEEE Medical Imaging Conference 1996. Summary and abstract can be found in Appendix L.
- **Peter Toft** and Jesper James Jensen: “*Accelerated 2D Iterative Reconstruction*”. Submitted to IEEE Transactions on Medical Imaging. Shown in Appendix M.
- Peter Alshede Philipsen, Lars Kai Hansen and **Peter Toft**: “*Mean Field Reconstruction with Snaky Edge Hints*”. Accepted for publication in the book “*INVERSE METHODS - Interdisciplinary elements of Methodology, Computation and Application*”. Will be published by Springer-Verlag in 1996. An (almost) identical paper can be found in Proceedings of the *Fourth Danish Conference on Pattern Recognition and Image Analysis 95*, pages 155-161. This paper can be found in Appendix N.
- **Peter Toft**: “*Detection of Lines with Wiggles using the Radon Transform*”. Submitted to the *NORSIG'96 - 1996 IEEE Nordic Signal Processing Symposium* in Espoo, Finland. This paper can be found in Appendix O.

May 31. 1996, Peter Aundal Toft.

Department of Mathematical Modelling, Section for Digital Signal Processing,
Technical University of Denmark, 2800 Lyngby, Denmark, E-mail: pto@imm.dtu.dk

Style Conventions

- References to the bibliography placed in the end of the thesis will appear as [13] or [14, 15].
- Equation and Figure has been abbreviated to Eq. and Fig. Likewise Equations and Figures to Eqs. and Figs.
- Equations, tables, and figures are numbered by the chapter, i.e., Eq. 10.2 is the second equation in Chapter 10.
- The appendices are numbered using capitals, starting with Appendix A.
- Normally the letters m, n, k, h, l are used to denote integer type variables, and the letters x, y, z denote continuous variables.
- Vectors are denoted by bold-faced small letters or Greek letters, like \mathbf{b} or $\boldsymbol{\xi}$.
- Transpose of a vector \mathbf{b} is shown as \mathbf{b}^T .
- Matrices are denoted by bold-faced capital letters, like \mathbf{A} . A real valued matrix with I rows and J columns are denoted $\mathbf{A} \in \mathbb{R}^{I \times J}$. The individual elements are $a_{i,j}$ corresponding to row i and column j .
- Vectors are always column vectors, and the i 'th rows of the matrix are denoted \mathbf{a}_i , i.e.,

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_i^T \\ \vdots \\ \mathbf{a}_I^T \end{bmatrix} \quad (0.1)$$

- Program examples in pseudo C-code will use the sans serif font and symbols using that font refers to the pseudo-code. An examples of pseudo-code are called an Algorithm, which looks like

ALGORITHM 0.1 : SMALL EXAMPLE

```

For k = 0 to K-1
    p(k) = p_min + k*Delta_p
End
    
```

- Vector indices in equations starts from 1, but in order to aid implementation in C or C++ the indices start from 0 in the pseudo code. Given that the pseudo code is intended for overview, this should not lead to confusion.
- The delta function is denoted by $\delta(\cdot)$. This function is reviewed in Appendix A.
- The Hamilton step function is denoted by $\mu(\cdot)$. The function is zero if the argument is negative, and one if the argument is positive.
- Rounding to the nearest upper integer, $\lceil \cdot \rceil$.
- Rounding to the nearest lower integer, $\lfloor \cdot \rfloor$. In an algorithm the function `floor` is used.
- Rounding to the nearest integer, $\lfloor \cdot \rfloor$. In an algorithm the function `round` is used.
- Fourier transform pairs are marked as $g(t) \leftrightarrow G(f)$.
- In equation convolutions are denoted by $*$ for a one dimensional, $**$ for a two-dimensional, and $***$ for a three-dimenisonal convolution.
- In the algorithms, i.e., pseudo-code, no convolutions are found and the symbol $*$ will be used as a simple multiplication.
- The scalar product between two (column) vectors \mathbf{a} and \mathbf{b} are denoted by $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b}$.
- The symbol \forall denotes ‘for all’.
- The length of vectors are normally denoted by $|\cdot|$, but in Chapter 9 the linear algebra notation $\|\cdot\|_2$ has been used, given by $\|\mathbf{v}\|_2 = \sqrt{\mathbf{v}^T \mathbf{v}} = \sqrt{\sum_j v_j^2}$.

Contents

Abstract	i
Resume på dansk (Abstract in Danish)	iii
Preface	v
Contents	v
Collaborations	vi
Acknowledgments	vii
Papers	viii
Style Conventions	ix
I Curve Parameter Detection using the Radon Transform	1
1 Slant Stacking	3
1.1 Why Consider the Radon Transform	3
1.2 Defining the (p, τ) Radon Transform	3
1.3 Basic Properties of the (p, τ) Radon Transform	4
1.3.1 Linearity	4
1.3.2 Shifting	5
1.3.3 Scaling	5
1.3.4 The Point Source	5
1.3.5 The Line	6
1.4 Discrete Slant Stacking	7
1.4.1 Nearest Neighbour Interpolation	8
1.4.2 Linear Interpolation	9
1.4.3 Sinc Interpolation	10
1.4.4 Sampling Properties of the Discrete Radon Transform	11
1.5 Discrete Radon Transform of a Discrete Line	12
1.5.1 Comparison of Different Interpolation Methods	13
1.6 Discrete Radon Transform of Points	16
1.7 Slant Stacking and Images with Steep Lines	19
1.8 Detection of Lines Convolved with a Wavelet	21
1.9 Summary	21

2 The Normal Radon Transform	23
2.1 Defining the (ρ, θ) Radon Transform	23
2.1.1 The Point Source	25
2.1.2 The (ρ, θ) Radon Transform of a Line	26
2.2 The Discrete (ρ, θ) Radon Transform	26
2.2.1 Sampling Properties of the (ρ, θ) Radon Transform	29
2.2.2 The Discrete (ρ, θ) Radon Transform of Several Lines	30
2.2.3 The Discrete (ρ, θ) Radon Transform of Points	34
2.3 Summary	36
3 The Hough Transform	37
3.1 The (p, τ) Hough Transform	37
3.1.1 Line Detection Using The Hough Transform	39
3.1.2 Choosing Sampling Parameters with the (p, τ) Hough Transform	42
3.2 The (ρ, θ) Hough Transform	43
3.2.1 Choosing sampling parameters with the (ρ, θ) Hough Transform	44
3.2.2 Comparison Between Different Optimization Strategies	45
3.2.3 Other Hough-like Algorithms	47
3.3 Summary	48
4 The FCE-Algorithm	49
4.1 The Generalized Radon Transform	50
4.1.1 The Continuous Generalized Radon Transform	50
4.1.2 The Discrete Generalized Radon Transform	51
4.2 Image Point Mapping	52
4.3 Parameter Domain Sampling	54
4.4 Parameter Domain Blurring	55
4.5 The Fast Curve Estimation Algorithm	56
4.6 The Hyperbolic Transformation Curve	58
4.6.1 Clusters in the Hyperbolic case	59
4.6.2 An Example with Eight Hyperbolas	60
4.6.3 An Example with a Noise Corrupted Synthetic CMP-gather	65
4.7 Summary	67
5 Curve Parameter Estimation in Noisy Images	69
5.1 Lines with Wiggles	69
5.2 A “Fuzzy” Radon Transform	74
5.3 Detection of Curves in Noisy Images	76
5.3.1 The Generalized Radon Transform	76
5.3.2 Curve Detection using the Generalized Radon Transform	77
5.3.3 Discussion	79
5.3.4 An Example of Line Detection in a very Noisy Image	79
5.4 Summary	80

II The Inverse Radon Transform and PET	83
6 Introduction to Computerized Tomography	85
6.1 Fundamental Theory of the CT-Scanner	86
6.2 The PET Scanner	88
6.2.1 Correction for Attenuation in PET	92
6.3 Summary	93
7 Inversion of the Radon Transform	95
7.1 The Fourier Slice Theorem	95
7.2 Filtered Backprojection	96
7.3 Filtering after Backprojection	97
7.4 Calculation using Operators	98
7.4.1 The Zero Frequency Problem	99
7.5 Sampling Considerations	99
7.6 Inversion of the (p, τ) Radon Transform	100
7.6.1 Fourier Slice Theorem	101
7.6.2 Filtered Backprojection	102
7.6.3 An Inversion Formula using the Hilbert Transform	102
7.6.4 Filtering after Backprojection	103
7.7 Summary	104
8 Numerical Implementation of Direct Reconstruction Algorithms	105
8.1 Using the DFT to Approximate the Fourier Transformation	105
8.1.1 The FFT Applied for Filtering	106
8.2 Discrete Implementation of Backprojection	111
8.3 Implementation of Filtering after Backprojection	114
8.4 Implementation of The Fourier Slice Theorem	116
8.4.1 Non-linear sampling of the Radon domain	119
8.5 Examples Using Direct Reconstruction Algorithms	121
8.5.1 Reconstruction using Different Methods	121
8.5.2 Sinogram with very few Samples in the Angular Direction	122
8.5.3 Reconstruction with Varying Image Size	123
8.5.4 Reconstruction into a Oversampled Image	125
8.5.5 Noise in the Sinogram	126
8.6 Summary	127
9 Reconstruction Algorithms Based on Linear Algebra	129
9.1 From the Radon Transform to Linear Algebra based Reconstruction	129
9.2 The Calculation of Matrix Elements	131
9.2.1 Pixel Oriented Nearest Neighbour Approximation	132
9.2.2 Discrete Radon Transform	132
9.2.3 First Order Pixel Oriented Interpolation Strategy	132
9.2.4 The Sinc Interpolation Strategy	133
9.3 Duality between Matrix Operations and the Radon Transform	133
9.4 Regularization and Constraints	134
9.5 Singular Value Decomposition	135
9.6 Iterative Reconstruction using ART	137
9.6.1 ART with Constraints	139

9.6.2 Initialization	140
9.7 Multiplicative ART	141
9.8 The EM algorithm	141
9.9 The Conjugate Gradient Method	145
9.10 Accelerated Iterative Reconstruction	146
9.10.1 A Fast 2D Iterative Reconstruction Package	147
9.11 Examples Using Iterative Reconstruction Algorithms	148
9.11.1 A Small Reconstruction Example	149
9.11.2 A Larger Reconstruction Example	149
9.11.3 Comparison with Different Noise Levels	150
9.11.4 Reconstruction Using Constraints	153
9.11.5 Reconstruction Using Regularization	156
9.12 Summary	156
10 The 3D Radon Transform for Lines	159
10.1 Lines in a Three Dimensional Space	160
10.1.1 Limiting the 3D line parameters	162
10.2 Fourier Slice Reconstruction in 3D	163
10.3 Backprojection Based Inversion of Line Integrals in 3D	164
10.4 Filtering after Backprojection of Line Integrals in 3D	165
10.5 Filtered Backprojection of Line Integrals in 3D	166
10.6 Reconstruction Scheme for 3D Multi Ring PET Scanners	168
10.7 A 3D Reconstruction Package	169
10.8 Implementation of the 3D Reconstruction Methods	169
10.8.1 Implementation of the Backprojection Operator	170
10.8.2 Implementation of the Radon Transform Operator	171
10.9 Examples of 3D Reconstructed Volumes	173
10.9.1 Reconstruction of a Ball	173
10.9.2 Reconstruction of the Mickey Phantom	174
10.10 Summary	177
11 Noise Contributions from Blank, Transmission and Emission Scans in PET	179
11.1 Introduction	179
11.2 Theory	180
11.2.1 One Emission and One Transmission Scan	180
11.2.2 Two Emission Scans	181
11.2.3 Two Emission and Two Transmission Scans	181
11.2.4 Zeroes in the Transmission Sinogram	182
11.2.5 Limitations	182
11.3 Overview of the Measurements	182
11.3.1 Phantom studies	182
11.3.2 Human studies	184
11.3.3 Fitting data	184
11.4 Results	184
11.5 Optimization	187
11.6 Discussion of the Results	188
11.7 Summary	189

Conclusion and Topics for Further Research	191
III Appendices	193
A The Dirac Delta Function	195
B Properties of the Normal Radon Transformation	197
B.1 Basic Properties of the Radon Transform	197
B.1.1 Linearity	197
B.1.2 Shifting	197
B.1.3 Rotation	197
B.1.4 Scaling	198
B.1.5 Convolution	199
B.2 The Shepp-Logan Phantom Brain	199
B.3 Analytical Radon Transform of Primitives	201
B.3.1 The Circular Disc	201
B.3.2 The Square	202
B.3.3 The Triangle	203
B.3.4 The Gaussian Bell	205
B.3.5 The Pyramid	205
C Usage of the 2D Program Packages	209
C.1 The Analytical Sinogram Program “RadonAna”	209
C.2 The Direct Reconstruction Program “iradon”	212
C.3 The Fast Iterative Reconstruction Program “it”	214
D The Three Dimensional Radon Transformation	217
D.1 The Three Dimensional Fourier Slice Theorem	218
D.2 Filtered Backprojection in 3D	218
D.3 Connection between the 3D plane integrals and 3D line integrals	219
E Properties of the 3D line Radon Transform	221
E.1 Basic Properties of the 3D line Radon Transform	221
E.1.1 Linearity	221
E.1.2 Translation	221
E.1.3 Rotation and Scaling	222
E.2 Analytical Radon Transformation of Primitives	224
E.2.1 The ball	224
E.2.2 The Gaussian bell	224
F Usage of the 3D Reconstruction Tools	225
F.1 The 3D Reconstruction Program “Recon3D”	225
F.2 The Analytical Sinogram Program “3D_RadonAna”	227

IV Papers	231
G Fast Radon Transform for Detection of Seismic Reflections	233
H Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform	239
I Using the Generalized Radon Transform for Detection of Curves in Noisy Images	255
J Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET	261
K Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET	267
L A very fast Implementation of 2D Iterative Reconstruction Algorithms	275
M Accelerated 2D Iterative Reconstruction	279
N Mean Field Reconstruction with Snaky Edge Hints	285
O Detection of Lines with Wiggles using the Radon Transform	293
Thesis Bibliography	297
Index	306

Part I

Curve Parameter Detection using the Radon Transform

Chapter 1

Slant Stacking

In this chapter the Radon transform will be introduced. The definition of the Radon transform is, e.g., used within seismics, where it is known as slant stacking. The discrete Radon transform is also introduced and several properties are reviewed. Several examples mostly regarding detection of straight lines from digital images are given [16].

1.1 Why Consider the Radon Transform

First a small example is presented only to motivate that the Radon transform is suited for line parameter extraction even in presence of noise. Fig. 1.1 shows an image containing three lines of which two are very close and the image has been corrupted by additive uniformly distributed noise. Fig. 1.2 shows the discrete Radon transform of the image, i.e., the figure shows a space of possible line parameters. It will be shown that the Radon transform are able to transform each of the lines into peaks positioned corresponding to the parameters of the lines. In this way, the Radon transform converts a difficult global detection problem in the image domain into a more easily solved local peak detection problem in the parameter domain, and the actual parameters can be recovered by, e.g., thresholding the Radon transform.

Note that especially in this noisy case other algorithms in general fail. An alternative could be to use a local detection algorithm, e.g. edge detection filters [17, 18], succeeded by a procedure for linking the individual pixels together, and finally to use linear regression for estimation of parameters. Algorithms of this kind have problems with intersecting lines, and in case of a high noise level, it is difficult to stabilize the edge detection filters. The Radon transform is not limited in the same sense by these problems.

1.2 Defining the (p, τ) Radon Transform

The Radon transform can be defined in different ways. The definition used, e.g., within seismics [19] is perhaps the easiest to comprehend. The Radon transform $\check{g}(p, \tau)$ of a (continuous) two-dimensional function $g(x, y)$ is found by stacking or integrating values of g along slanted lines. The location of the line is determined from the line parameters; slope p and line offset τ .

$$\check{g}(p, \tau) = \int_{-\infty}^{\infty} g(x, px + \tau) dx \quad (1.1)$$

Within seismics this linear Radon transform is also known as slant stacking or the τ - p transform. Note that different notation exists, and here the notation found in [14] is used.

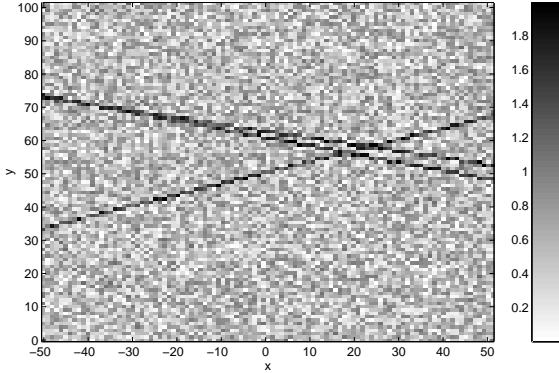


Figure 1.1 An image with two lines where uniformly distributed and uncorrelated noise has been added.

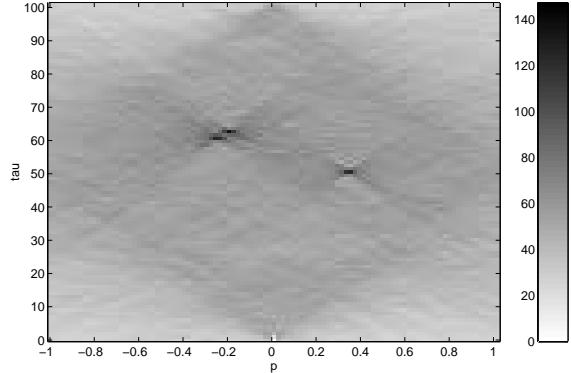


Figure 1.2 The corresponding discrete Radon transform. Each of the peaks correspond to the curves in the image. The line parameters can here be recovered by simple means, e.g., thresholding.

The Dirac delta function, $\delta(\cdot)$ (see Appendix A for basic properties), will be used extensively in the following. Using the delta function implies that slant stacking can be written as

$$\check{g}(p, \tau) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(y - px - \tau) dx dy \quad (1.2)$$

The values of $\check{g}(p, \tau)$ is function in two-dimensional (p, τ) -space, i.e., Radon space or parameter domain. In principle, the two parameters do not have lower or higher limits, though, as it will be shown, discrete implementation will use a limited number of samples in both parameter directions.

1.3 Basic Properties of the (p, τ) Radon Transform

From Eqs. 1.1 or 1.2, a set of fundamental properties can be derived. It can also be noted, that it is possible to find the Radon transform analytically of some simple mathematical functions [20].

1.3.1 Linearity

From Eq. 1.1 it is directly found, that the Radon transform of a weighted sum of functions is the same weighted sum of the individually Radon transformed functions.

$$\begin{aligned} h(x, y) &= \sum_i w_i g_i(x, y) \Rightarrow \\ \check{h}(p, \tau) &= \sum_i w_i \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g_i(x, y) \delta(y - px - \tau) dx dy \\ &= \sum_i w_i \check{g}_i(p, \tau) \end{aligned} \quad (1.3)$$

This property is naturally very important.

1.3.2 Shifting

The next property is the Radon transform of a shifted function.

$$\begin{aligned}
h(x, y) &= g(x - x^*, y - y^*) \Rightarrow \\
\check{h}(p, \tau) &= \int_{-\infty}^{\infty} g(x - x^*, px + \tau - y^*) dx \\
&= \int_{-\infty}^{\infty} g(\tilde{x}, p(\tilde{x} + x^*) + \tau - y^*) d\tilde{x}, \text{ where } \tilde{x} = x - x^* \\
&= \check{g}(p, \tau - y^* + px^*)
\end{aligned} \tag{1.4}$$

Thus only the offset parameter is changed by shifting the function. From a geometrical point of view the result is obvious. The slope of a line cannot be altered by a translation, and the offset must be changed as shown in Eq. 1.4.

1.3.3 Scaling

A scaled function is assumed

$$\begin{aligned}
h(x, y) &= g\left(\frac{x}{a}, \frac{y}{b}\right) \text{ where } a > 0 \text{ and } b > 0 \\
\check{h}(p, \tau) &= \int_{-\infty}^{\infty} g\left(\frac{x}{a}, \frac{px + \tau}{b}\right) dx \\
&= a \int_{-\infty}^{\infty} g\left(\tilde{x}, \frac{pa\tilde{x} + \tau}{b}\right) d\tilde{x} \text{ where } \tilde{x} = \frac{x}{a} \\
&= a \check{g}\left(\frac{pa}{b}, \frac{\tau}{b}\right)
\end{aligned} \tag{1.5}$$

The result can also here be understood from the parameters of a line. It is clear that a compression in the y -direction must be followed by a compression in the line offset τ . And it is not difficult to understand that any slope will be scaled with the ratio between a and b , and so will the Radon transform.

1.3.4 The Point Source

A point source is modelled as a product of two delta functions. Initially the point source is placed in the origin of the coordinate system.

$$\begin{aligned}
g(x, y) &= \delta(x) \delta(y) \Rightarrow \\
\check{g}(p, \tau) &= \int_{-\infty}^{\infty} \delta(x) \delta(px + \tau) dx = \delta(\tau)
\end{aligned} \tag{1.6}$$

The point source could easily have been placed at any given position, but now Eq. 1.4 is used to do that

$$g(x, y) = \delta(x - x^*) \delta(y - y^*) \Rightarrow \check{g}(p, \tau) = \delta(\tau - y^* + px^*) \tag{1.7}$$

This result is interesting, because any function can be written as a weighted integral of point sources.

$$\begin{aligned}
g(x, y) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(y - y^*) dx^* dy^* \Rightarrow \\
\check{g}(p, \tau) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(\tau + px - y^*) dx^* dy^* dx
\end{aligned} \tag{1.8}$$

$$\begin{aligned}
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(y^* - \tau - px^*) dx^* dy^*
\end{aligned} \tag{1.9}$$

Eq. 1.7 and especially Eq. 1.9 demonstrates that any point of the function will contribute along an infinitely long line in the parameter domain. This aspect relates to the Hough transform, that will be further analyzed in Chapter 3.

In Fig. 1.3, the result is shown symbolically. Note that the figure shows the two dimensional function $\check{g}(p, \tau)$ by use of color, white is used to indicate (approximately) zero value and black is a high (in this case infinite) value. This type of graph will be used many times in the following.

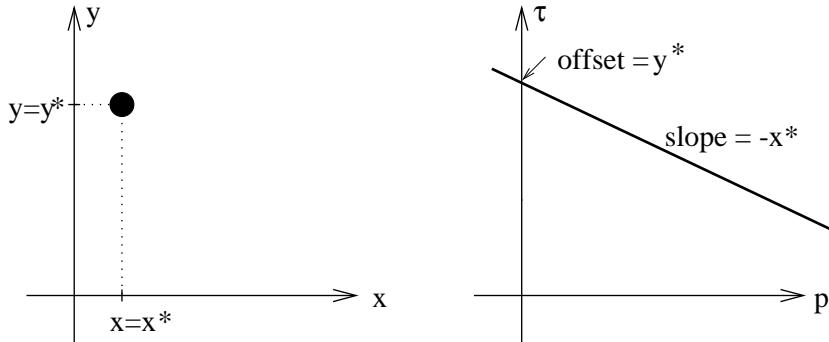


Figure 1.3 Left: A two dimensional function that only is non-zero in the point $(x, y) = (x^*, y^*)$. Right: The corresponding Radon transform (slant stacking result). Only when the Radon domain parameters matches the parameters of the line a non-zero result is found.

1.3.5 The Line

This very important property assumes a function that contains a certain line, here modelled with a delta function.

$$g(x, y) = \delta(y - p^*x - \tau^*) \quad (1.10)$$

hence the function has non-zero values only if (x, y) lies on the line with certain fixed parameters (p^*, τ^*) . In this case the Radon transform is given by

$$\begin{aligned} \check{g}(p, \tau) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \delta(y - p^*x - \tau^*) \delta(y - px - \tau) dx dy \\ &= \int_{-\infty}^{\infty} \delta((p - p^*)x + \tau - \tau^*) dx \\ &= \begin{cases} \frac{1}{|p - p^*|} & \text{for } p \neq p^* \\ 0 & \text{for } p = p^* \text{ and } \tau \neq \tau^* \\ \int_{-\infty}^{\infty} \delta(0) dx & \text{for } p = p^* \text{ and } \tau = \tau^* \end{cases} \end{aligned} \quad (1.11)$$

Note that for $p = p^*$ and $\tau = \tau^*$, the result is written as an infinite function integrated over an infinite interval, hence the result is infinite in that point. If, for now, the finite terms are neglected, the result is that the Radon transform of a line produces a peak (with infinite value) in the parameter domain, and the position of the peak matches the line parameters. This property has naturally been the basis of many curve parameter detection algorithms. In Fig. 1.4 the result is shown symbolically. Again, white color indicates (approximately) zero value and black color (infinite) values.

It can be noted that in the way slant stacking is defined, there is duality between the two domains. A point in the image domain, i.e., the (x, y) -space, is transformed into a line in the

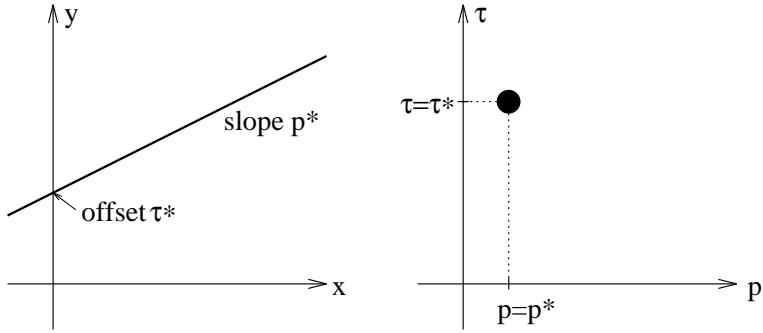


Figure 1.4 Left: A two dimensional function that is only non-zero when on the line. Right: The corresponding Radon transform (slant stacking result). When the Radon domain parameters matches the parameters of the line, a peak is found positioned at the parameters of the line in the image. The finite terms in the parameter domain are here ignored for sake of clarity.

Radon domain, and, if the finite terms are ignored, a line in the image domain will be transformed into a point in the Radon domain. This property is a direct consequence of the form of the integration kernel in Eq. 1.2. This property motivates why the Radon transform can be used for curve detection algorithms.

1.4 Discrete Slant Stacking

Given that only a subset of functions can be Radon transformed analytically, a discrete approximation to the Radon transform that transforms a digital image is very useful. Depending of the aim, e.g. speed, artifacts, or simplicity, there exist nearly as many definitions of the discrete Radon transform as people working with the Radon transform. Though an easy way is to sample the four variables linearly and only work with a limited set of samples.

$$\begin{aligned} x &= x_m = x_{\min} + m\Delta x, & m &= 0, 1, \dots, M - 1 \\ y &= y_n = y_{\min} + n\Delta y, & n &= 0, 1, \dots, N - 1 \\ p &= p_k = p_{\min} + k\Delta p, & k &= 0, 1, \dots, K - 1 \\ \tau &= \tau_h = \tau_{\min} + h\Delta\tau, & h &= 0, 1, \dots, H - 1 \end{aligned} \quad (1.12)$$

Here the x_{\min} is the position of the first sample, Δx the sampling distance of x , and m is a discrete index used to number the M samples of x . These symbols will be used many times in the following.

By using Eq. 1.1, a simple and common approach to approximate the linear Radon transform is to use a sum approximation

$$\check{g}(p_k, \tau_h) = \int_{-\infty}^{\infty} g(x, p_k x + \tau_h) dx \approx \Delta x \sum_{m=0}^{M-1} g(x_m, p_k x_m + \tau_h) \quad (1.13)$$

Sampling of the function $g(x, y)$ gives a digital image

$$g(m, n) = g(x_m, y_n) \quad (1.14)$$

A new symbol could be assigned to the digital image, though from the indices it should be clear whether the continuous or the discrete version is used. Likewise will the discrete Radon transform be written as

$$\check{g}(k, h) = \check{g}(p_k, \tau_h) \quad (1.15)$$

Hence, the discrete Radon transform can and will be presented as a digital image.

1.4.1 Nearest Neighbour Interpolation

Given the digitally sampled image $g(m, n)$ a fundamental problem arises. Eq. 1.13 requires samples not found in the digital image, because linear sampling of all variables implies that $p_k x_m + \tau_h$ in general never coincides with the samples y_n . The problem could be corrected by using, e.g., a nearest neighbour approximation in the y -direction, hence the Radon transform can be approximated by the discrete Radon transform

$$\check{g}(k, h) = \Delta x \sum_{m=0}^{M-1} g(m, n(m; k, h)) \text{ where } n(m; k, h) = \left[\frac{p_k x_m + \tau_h - y_{\min}}{\Delta y} \right] \quad (1.16)$$

where $[\cdot]$ means rounding the argument to nearest integer. Another problem is that the discrete point $(m, n(m; k, h))$ need not lie within the finite image. If the point lies outside the image the value needed could be set to zero, i.e., the point gives no contribution to the discrete Radon transform.

Note that the constant term Δx can be neglected. This term does not carry information, and is only needed if the discrete Radon transform should quantitatively approximate the (continuous) Radon transform.

It can easily be proven that the discrete Radon transform is a linear function, though the other properties shown for the continuous Radon transform can only be used to the discrete Radon transform with approximation.

Before showing a small part of an algorithm to compute the discrete Radon transform, the expression for n should be rewritten to a very simple linear form, in order to reduce the computational cost.

$$n = [n^*] \text{ and } n^* = \frac{p_k x_m + \tau_h - y_{\min}}{\Delta y} = \alpha m + \beta \quad \text{where} \quad \begin{cases} \alpha = \frac{p_k \Delta x}{\Delta y} \\ \beta = \frac{p_k x_{\min} + \tau_h - y_{\min}}{\Delta y} \end{cases} \quad (1.17)$$

The following program is written in pseudo-code, where all array indices start at 0. Comments to the code are shown in C++ style using $//$.

ALGORITHM 1.1 : THE DISCRETE SLANT STACKING

```

For k = 0 to K-1                                //For all values of p
  For h = 0 to H-1                               //For all values of tau
    alpha = p(k)*Delta_x/Delta_y                 //Calculate digital slope
    beta = (p(k)*x_min+tau(h)-y_min)/Delta_y   //Calculate digital offset
    sum = 0                                         //Initialize sum
    For m = 0 to M-1                             //For all values of x
      n = round(alpha*m+beta)                    //Use nearest neighbour approx.
      If n≥0 and n<N                           //Check if pixel lies in image
        sum=sum+g(m,n)                          //If so, then increment sum
      End
    End
    g_radon(k,h)=Delta_x*sum                   //Store Radon value
  End
End

```

In Algorithm 1.1 p_k has been replaced by $p(k)$, and the initialization of the arrays has been removed in order to reduce the size of the pseudo-code. Note that actions have to be taken to

assure that pixel going into the sum actually lies in the image. In the time-consuming part of the loop, i.e., within the m -loop, optimization should be done very carefully depending on the platform (computers or similar digital equipment), used for computing the discrete Radon transform. A trick that can boost the performance considerably on most platforms, is to directly calculate which values of m , that gives pixels lying within the image.

$$0 \leq n = [\alpha m + \beta] \leq N - 1 \Rightarrow \begin{cases} -\frac{\beta + \frac{1}{2}}{\alpha} \leq m < \frac{N - \frac{1}{2} - \beta}{\alpha} & \text{if } \alpha > 0 \\ \frac{N - \frac{1}{2} - \beta}{\alpha} \leq m < -\frac{\beta + \frac{1}{2}}{\alpha} & \text{if } \alpha < 0 \end{cases} \Rightarrow \quad (1.18)$$

$$m_{min} = \max \left\{ 0, \left\lceil -\frac{\beta + \frac{1}{2}}{\alpha} \right\rceil \right\} \text{ and } m_{max} = \min \left\{ M - 1, \left\lfloor \frac{N - \frac{1}{2} - \beta}{\alpha} \right\rfloor \right\} \text{ if } \alpha > 0$$

$$m_{min} = \max \left\{ 0, \left\lceil \frac{N - \frac{1}{2} - \beta}{\alpha} \right\rceil \right\} \text{ and } m_{max} = \min \left\{ M - 1, \left\lfloor -\frac{\beta + \frac{1}{2}}{\alpha} \right\rfloor \right\} \text{ if } \alpha < 0 \quad (1.19)$$

where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ rounds to the nearest lower and higher integer, respectively. So instead of using the full interval of m -values, only m from m_{min} to m_{max} should be used, in order to avoid the time consuming testing of valid limits.

Another issue of optimization concern the expression `n=round(alpha*m+beta)`, which could be replaced by an initial `nfloat=beta+m_min*alpha` (if using the alteration described in Eq. 1.19) and then within the loop `n=round(nfloat)` and the line `nfloat=nfloat+alpha`. Hence, multiplications can be avoided in the (time) important part of the program. The extra accumulation of noise will normally only cause problems if using a very poor (small) representation of the floating point numbers or using extremely large images (hardly the case). This alteration might give an additional speedup, but it depends on the platform, i.e., the memory access rate, and the adding/multiplication rate of the platform.

1.4.2 Linear Interpolation

So far only a nearest neighbour approximation has been considered. It can also be chosen to use linear interpolation in the y -direction of the image.

$$\check{g}(k, h) = \Delta x \sum_{m=0}^{M-1} ((1 - w)g(m, n) + wg(m, n + 1)) \quad (1.20)$$

$$\text{where } n^* = \frac{p_k x_m + \tau_h - y_{min}}{\Delta y} = \alpha m + \beta, \quad n = \lfloor n^* \rfloor, \text{ and } w = n^* - n \quad (1.21)$$

It can be noted that Eq. 1.20 requires the sum of twice as many samples compared to Eq. 1.16. In general, a better (higher order) interpolation scheme implies more computational work. This tradeoff must be determined by sampling properties and the time available for the algorithm.

As shown in the following equation, linear interpolation does not imply much change to the optimization strategy of computing the range of m -values, corresponding to pixels in the image.

$$0 \leq n = \lfloor \alpha m + \beta \rfloor \leq N - 2 \Rightarrow \begin{cases} -\frac{\beta}{\alpha} \leq m < \frac{N - 1 - \beta}{\alpha} & \text{if } \alpha > 0 \\ \frac{N - 1 - \beta}{\alpha} \leq m < -\frac{\beta}{\alpha} & \text{if } \alpha < 0 \end{cases} \Rightarrow \quad (1.22)$$

$$m_{min} = \max \left\{ 0, \left\lceil -\frac{\beta}{\alpha} \right\rceil \right\} \text{ and } m_{max} = \min \left\{ M - 1, \left\lfloor \frac{N - 1 - \beta}{\alpha} \right\rfloor \right\} \text{ if } \alpha > 0$$

$$m_{min} = \max \left\{ 0, \left\lceil \frac{N - 1 - \beta}{\alpha} \right\rceil \right\} \text{ and } m_{max} = \min \left\{ M - 1, \left\lfloor -\frac{\beta}{\alpha} \right\rfloor \right\} \text{ if } \alpha < 0 \quad (1.23)$$

In Algorithm 1.2, it is shown how to compute the Discrete Radon transform using the linear interpolation approximation. Note also, that Eq. 1.23 should be used to improve the performance.

ALGORITHM 1.2 : THE DISCRETE SLANT STACKING WITH LINEAR INTERPOLATION

```

For k = 0 to K-1                                //For all values of p
    alpha = p(k)*Delta_x/Delta_y                //Calculate digital slope
    For h = 0 to H-1                            //For all values of tau
        beta = (p(k)*x_min+tau(h)-y_min)/Delta_y //Calculate digital offset
        Set m_min and m_max using Eq. 1.23       //Not shown
        sum = 0                                     //Initialize sum
        For m = m_min to m_max                    //For all valid values of x
            nfloat = alpha*m+beta                 //Use nearest neighbour approx.
            n=floor(nfloat)                      //Calculate lower index
            w=nfloat-n                           //Calculate weight factor
            sum=sum+g(m,n)*(1-w)+g(m,n+1)*w     //Increment sum
        End
        g_radon(k,h)=sum*Delta_x                //Update matrix element
    End
End

```

In Sections 1.5 and 1.6, examples will be given in order to show differences between the simple nearest neighbour approximation, given in Eq. 1.16, the linear interpolation approximation, and a sinc interpolation Radon transform.

1.4.3 Sinc Interpolation

Assuming that the function $g(x, y)$ was sampled according to the Whittaker-Shannon sampling theorem [21], then the function can be recovered from the digital image $g(m, n)$ by convolution with a sinc function,

$$g(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) \frac{\sin(\frac{\pi}{\Delta x}(x - x_m))}{\frac{\pi}{\Delta x}(x - x_m)} \frac{\sin(\frac{\pi}{\Delta y}(y - y_n))}{\frac{\pi}{\Delta y}(y - y_n)} \quad (1.24)$$

It is not quite true that $g(x, y)$ is exactly recovered because the summations should in principle be infinitely long, but these samples are assumed to have value zero.

Instead of approximating the Radon transform to a digital image, a new way of analyzing the Radon transform of a digital image is proposed [16].

Eq. 1.1 will now be applied to Eq. 1.24 in order to gain information of the Radon transform and derive a sampling criterion for the parameter domain

$$\check{g}(p, \tau) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) I(p, \tau, x_m, y_n) \quad (1.25)$$

$$I(p, \tau, x_m, y_n) = \int_{-\infty}^{\infty} \frac{\sin(\frac{\pi}{\Delta x}(x - x_m))}{\frac{\pi}{\Delta x}(x - x_m)} \frac{\sin(\frac{\pi}{\Delta y}(px + \tau - y_n))}{\frac{\pi}{\Delta y}(px + \tau - y_n)} dx \quad (1.26)$$

$$= \frac{\Delta x}{\pi} \int_{-\infty}^{\infty} \frac{\sin(t)}{t} \frac{\sin(\alpha t + \gamma)}{\alpha t + \gamma} dt \text{ where } \begin{cases} t = \frac{\pi}{\Delta x}(x - x_m) \\ \alpha = \frac{\Delta x}{\Delta y}p \\ \gamma = \frac{\pi}{\Delta y}(px_m + \tau - y_n) \end{cases} \quad (1.27)$$

The last integral can be interpreted as a convolution. Using that the Fourier transform of a sinc-function is a square, and a square multiplied with a square gives a square, implies that the last integral is a sinc function.

$$I(p, \tau, x_m, y_n) = \frac{\Delta x}{\gamma} \sin \left(\gamma \min \left\{ 1, \frac{1}{|\alpha|} \right\} \right) \Rightarrow \quad (1.28)$$

$$\check{g}(p, \tau) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) \Delta x \frac{\sin \left(\gamma \min \left\{ 1, \frac{1}{|\alpha|} \right\} \right)}{\gamma} \quad (1.29)$$

The result is a sinc-interpolation discrete Radon transform, but right now the expression will be used to analyze the sampling properties of the Radon transform.

1.4.4 Sampling Properties of the Discrete Radon Transform

Assuming that the absolute slope $|p|$ is limited (in Section 1.7, it will be shown that this is reasonable.), i.e., $|\alpha| < 1$, implies that the Radon transform is determined by the function $\sin(\gamma)/\gamma$, which, as a function of γ has an upper limit frequency of $\frac{1}{2\pi}$, thus if γ should be sampled, it should be done with a rate faster than

$$\Delta\gamma = \pi \quad (1.30)$$

A discrete version of the Radon transform demand that the two parameters have to be sampled, and Eq. 1.30 is used for each of the Radon parameters, i.e., p and τ .

$$\Delta\gamma = \left| \frac{\partial\gamma}{\partial\tau} \right| \Delta\tau = \frac{\pi}{\Delta y} \Delta\tau \leq \pi \Rightarrow \Delta\tau \leq \Delta y \quad (1.31)$$

$$\Delta\gamma = \left| \frac{\partial\gamma}{\partial p} \right| \Delta p = \frac{\pi}{\Delta y} |x_m| \Delta p \leq \pi \Rightarrow \Delta p \leq \frac{\Delta y}{\max|x_m|} \quad (1.32)$$

Note that a conservative evaluation is used to assure that the limit in Eq. 1.32 is valid for all values of x_m . The last equation shows that the sampling distance Δp must be small if $\max|x_m|$ is large. Regarding the sampling of p , it is optimal to choose symmetrical sampling points of x , hence minimizing $\max|x_m|$.

$$x_{\min} = -x_{\max} \Rightarrow x_{\min} = -\frac{M-1}{2} \Delta x \quad (1.33)$$

Even if x_{\min} is fixed by the underlying physics, this little trick can be used due to the shifting property of the Radon transform, shown in Eq. 1.4.

Another reasonable way of defining bounds on the sampling distances in the parameter domain is to demand that changing one of the Radon parameters with its sampling interval, cannot lead to more than a pixel of difference in the image, otherwise some of the pixels might not add weight to the parameter domain, hence information is lost. In other terms

$$p : \max\{|(p + \Delta p)x + \tau - (p x + \tau)|\} = \max\{|\Delta p x|\} \leq \Delta y \quad (1.34)$$

$$\tau : \max\{|p x + (\tau + \Delta\tau) - (p x + \tau)|\} = \Delta\tau \leq \Delta y \quad (1.35)$$

hence, the interesting conclusion is that the two approaches give the same result.

Assuming that the task is to identify line parameters, then the sampling distances Δp and $\Delta\tau$ can be set with Eq. 1.32 and Eq. 1.31, respectively. The two parameters τ_{\min} and H can be set from a requirement that, e.g., half of the samples in the discrete Radon transform should lie within the image. If assuming Eq. 1.33, it is found that $H = N$ and $\tau_{\min} = y_{\min}$, i.e., the

sampling of τ follows the sampling of y . The two parameters p_{min} and K must be set to cover the interval of expected slopes, e.g., if the underlying physics can be used to predict the resulting interval of slopes, this should be incorporated. Later, in Section 1.7, it will be shown that there are limitations to this strategy.

The conclusion is that with a given digital image the parameter domain must be sampled sufficiently dense in order to avoid aliasing problems. The limits derived in Eqs. 1.32 and 1.31 will now be tested with discrete implementations of the Radon transform.

1.5 Discrete Radon Transform of a Discrete Line

An digital image was generated with 101×101 samples. The synthetic image contains two lines as seen from Fig. 1.5 and the corresponding discrete Radon transform is shown in Fig. 1.6. Sampling parameters for this example can be found in Table 1.1. Here Δp and $\Delta \tau$ is chosen according to Eqs. 1.31 and 1.32.

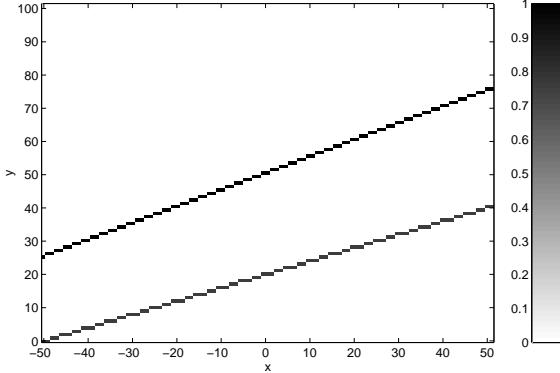


Figure 1.5 The image containing two lines.

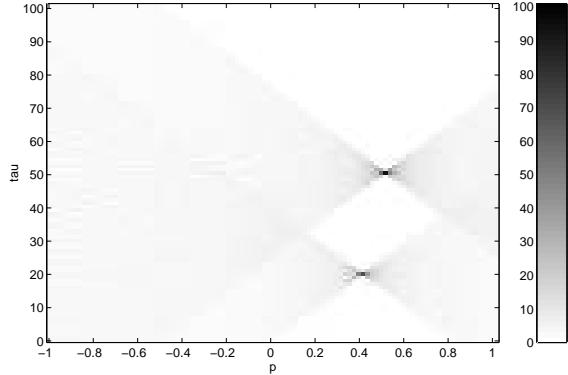


Figure 1.6 The corresponding discrete Radon transform (slant stacking).

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	101	H	101
N	101	K	101
Δx	1	Δp	0.02
Δy	1	$\Delta \tau$	1
x_{min}	-50	p_{min}	-1
y_{min}	0	τ_{min}	0

Table 1.1 Sampling parameter settings.

The linearity of the Radon transform implies that two peaks should be expected in the discrete parameter domain, which is shown in from Fig. 1.6. From the position of the peaks, it is clear that the line parameters are $(p_1 = 0.5, \tau_1 = 50)$ and $(p_2 = 0.4, \tau_2 = 20)$.

Now the sampling parameters of the parameter domain are changed, in order to zoom in around one of the peaks. Sampling parameters are shown in left side of Table 1.2 and the Radon Transform is shown in Fig. 1.7. Now the peak is very broad, which suggests that the sampling

of the parameter domain is unnecessary dense. It can also be noted that the number of samples in the discrete parameter domain has been maintained so the computational cost is (almost) the same as in Fig. 1.6.

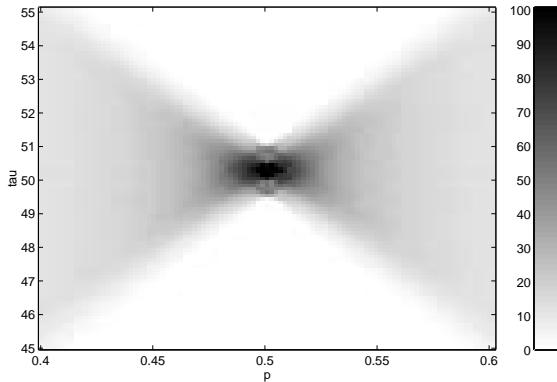


Figure 1.7 The discrete Radon transform of the image shown in Fig. 1.5. Dense sampling of the parameter domain has been used.

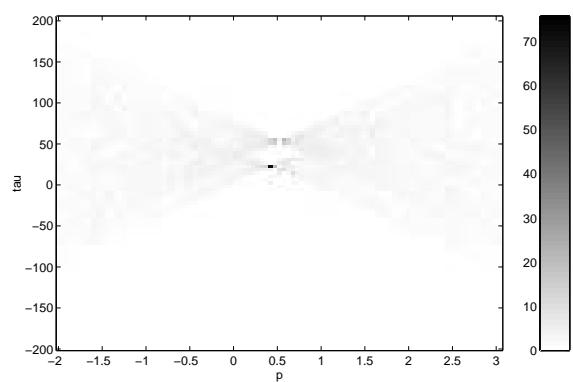


Figure 1.8 The discrete Radon transform of the image shown in Fig. 1.5. Very sparse sampling of the parameter domain has been used.

In Fig. 1.8 the parameter domain has been sampled very sparsely, and note how the upper peak has vanished. The sampling parameters Δp and $\Delta \tau$ are here set to strongly under-sample the parameter domain, i.e., the limits found in Eqs. 1.32 and 1.31 are strongly violated. This example shows how the sampling parameters of the parameter domain should be set. If the parameter domain is sampled too densely, redundant information is acquired. On the other hand sampling the parameter domain very sparsely implies important information *might* be lost. Here information about one of the lines is lost, but the parameters of the other line is maintained.

Parameter Domain (zoom in)		Parameter domain (zoom out)	
Parameter	Value	Parameter	Value
H	101	H	101
K	101	K	101
Δp	0.002	Δp	0.05
$\Delta \tau$	0.1	$\Delta \tau$	4
p_{min}	0.4	p_{min}	-2
τ_{min}	45	τ_{min}	-200

Table 1.2 Sampling parameter settings.

1.5.1 Comparison of Different Interpolation Methods

The sampling of the parameter domain also depends on the type of interpolation being used, and the type of image. If the image $g(m, n)$ does not include very sharp edges with many high frequency components, a nearest neighbour approximation might be sufficient, but an image containing a line, like Fig. 1.9 implies that interpolation method affect the result near the peak in the discrete parameter domain.

In the following the discrete parameter domain is found by use of the three approximations, i.e., the nearest neighbour interpolation in the image (Eq. 1.16), the linear interpolation (Eq. 1.20), and finally the sinc-interpolation (Eq. 1.29). In all three cases the line parameters, corresponding to Fig. 1.9, are found.

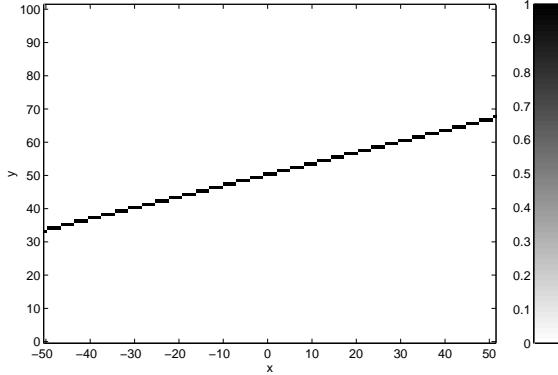


Figure 1.9 An image with one line.

Figs. 1.10, 1.12, and 1.14 use the same color scaling to show the discrete parameter domains. It can be seen that the difference is limited, but the value in the peak is lower in the last two images, compared to the first (nearest neighbour). In Figs. 1.11, 1.13, and 1.15 the discrete parameter domains are shown using a dense sampling in the area of the peak. The images have a common color scaling. It can be seen that the peak in Fig. 1.11 is very narrow, though the largest value (≈ 101) is much higher compared to Fig. 1.13, and 1.15. This indicates that using nearest neighbour approximation requires a dense sampling of the parameter domain in order to assure that the peak is found. Using linear interpolation with this sparse image implies that the peak value will be rather low (≈ 78) but broad. This can be understood from Fig. 1.9, where at least one of the two image points going into Eq. 1.20 is identically zero at any m , so the weight factors will lower the result in the peak.

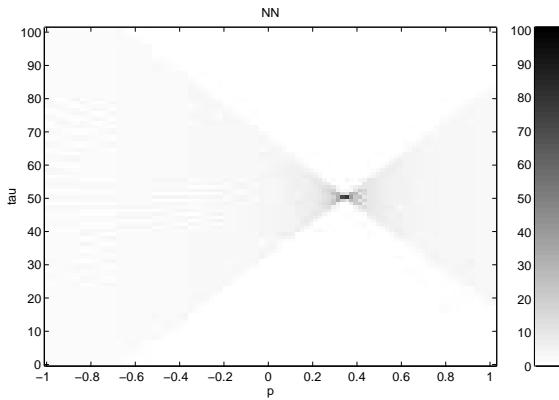
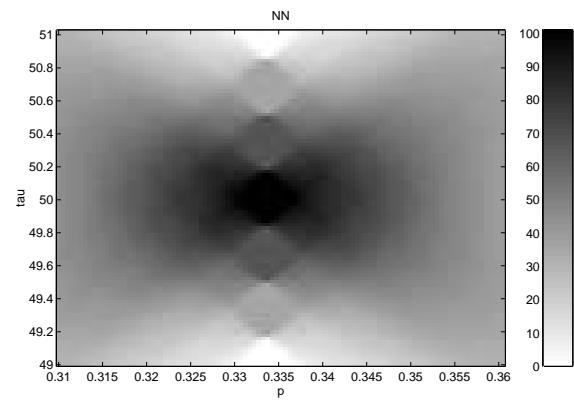
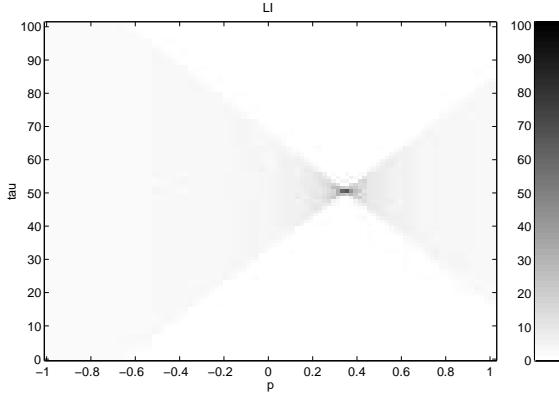
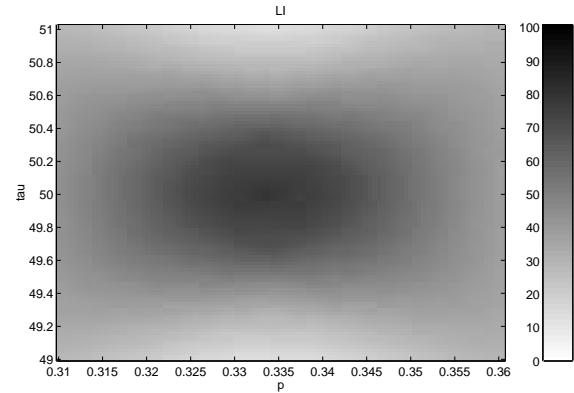
It is important is that a peak is generated in the discrete parameter domain of a sufficient size, and the adequate sampling of the parameter domain should be compared to the theoretically derived ones. From the Fig. 1.15 a somewhat higher peak value (≈ 89) is found, and the effective size of the peak resembles that found with linear interpolation. For Figs. 1.10, 1.12, and 1.14 the sampling parameters of the image and parameter domain can be found in Table 1.1, and for Figs. 1.11, 1.13, and 1.14 in Table 1.3. Note the special form of the parameter domain in the nearest neighbour case. It is a digital effect, and will be found when the line slope can be written as a fraction with small numbers, e.g., $\frac{1}{3}$, $\frac{1}{2}$, or $\frac{3}{4}$.

The computational complexity of the three approximations is also very interesting. Using the symbols defined in Eq. 1.12 the computational complexity of the three ways of implementing the discrete Radon transform is approximately given by

$$\begin{aligned} \mathcal{O}_{\text{Nearest Neighbour}} &= \mathcal{O}(K H M) \approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Linear Interpolation}} &= \mathcal{O}(K H M) \approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Sinc Neighbour}} &= \mathcal{O}(K H M N) \approx \mathcal{O}(M^4) \end{aligned} \quad (1.36)$$

where $\mathcal{O}(\cdot)$ is the order function, and some lower order terms have been skipped. Assuming M in the order of 100, then the computational complexity of the sinc interpolation is about 100

Parameter	Value
H	101
K	101
Δp	0.0005
$\Delta \tau$	0.02
p_{min}	0.31
τ_{min}	49

Table 1.3 Sampling parameter settings for Figs. 1.11, 1.13, and 1.15.**Figure 1.10** The discrete Radon transform using nearest neighbour approximation.**Figure 1.11** The discrete Radon transform in the area of the peak using nearest neighbour approximation.**Figure 1.12** The discrete Radon transform using linear interpolation.**Figure 1.13** The discrete Radon transform in the area of the peak using linear interpolation.

times higher than the other two methods. Note that linear approximation is slower than nearest neighbour approximation, due to the summation of two samples for every m .

In Table 1.4 the time needed for computing the discrete parameter domain is listed, corresponding to the three different approximations. The algorithms have been implemented in the C-language. As it can be seen a speedup-factor of approximately 3.5 has been gained by optimizing the code as described. The time used with linear approximation can be compared to the (not optimized) nearest neighbour time of 1.4, so a minor difference is found there. As could be expected from the Eq. 1.36 the sinc approximation is very slow, due to the fact that this al-

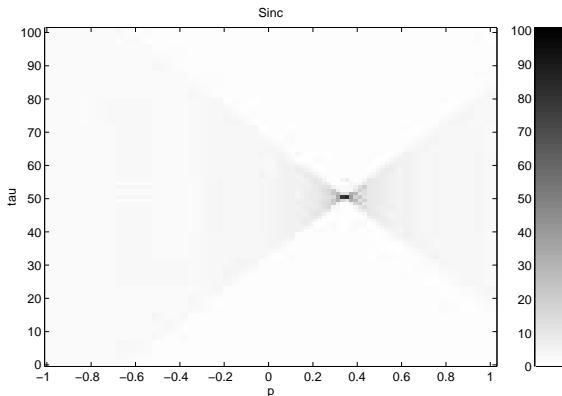


Figure 1.14 The discrete Radon transform using the sinc expansion. Note the ripples away from the peak.

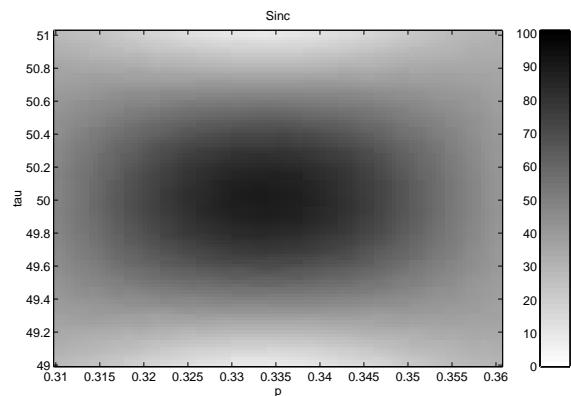


Figure 1.15 The discrete Radon transform in the area of the peak using the sinc expansion.

gorithm has to calculate a sinc function for every (m, n, k, h) . No flop-ratings are given because the performance is not totally dominated by the arithmetic statements, but also on the memory access-rate and the number of comparisons needed in the algorithm.

Method	Time
Nearest Neighbour	1.4 sec
Nearest Neighbour (optimized)	0.45 sec
Linear Interpolation	1.7 sec
Sinc Interpolation	287 sec

Table 1.4 Time for calculating the discrete Radon transform for the three types of approximation, corresponding to Figs. 1.11, 1.13, and 1.15. A 120 MHz Pentium PC has been used for the measurements.

One way of analyzing the resolution of the discrete Radon transform is to visualize the shape of the discrete parameter domain with a threshold value corresponding to, e.g., half of the potential max of the parameter domain, i.e., in this case 50. This is a kind of fwhm-measure (full width half max). Figs. 1.16, 1.17, and 1.18 show the parameter domain using a threshold level of 50. From the figures it can be seen that the extension of the peak, which could be interpreted as the resolution of the discrete Radon transform in the τ -direction, is very close to the sampling distances derived in Eq. 1.31, and the resolution in the p -direction is approximately 3 time the one derived in Eq. 1.32. But it should be noted that this kind of analysis depends on the actual threshold level. Regarding this example the conclusion is that the simple approximations actually produce good results and much faster than using the sinc interpolation.

1.6 Discrete Radon Transform of Points

In this subsection, the discrete Radon transform of individual points will be analyzed. Images with an increasing number of points lying on a line will be analyzed in order to see the results of Section 1.5 from another point of view.

Fig. 1.19 is generated from Fig. 1.5 (an image with two lines), and the image only has eight non-zero pixels lying on two lines. The pixel values are now attenuated horizontally across the image (lowest pixel value in the right side of the image). In Fig. 1.20 the corresponding discrete

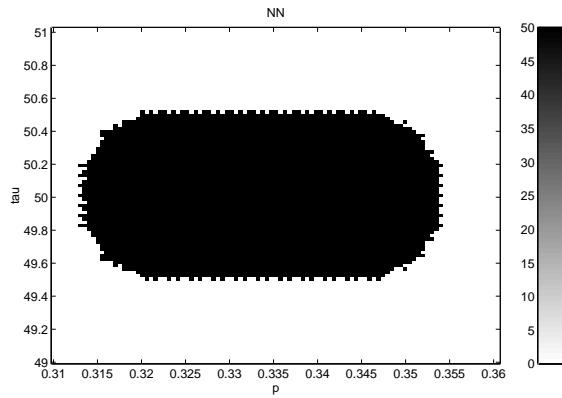


Figure 1.16 The thresholded discrete Radon transform (nearest neighbour).

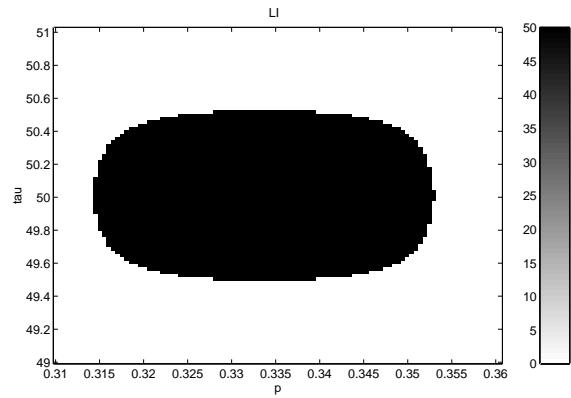


Figure 1.17 The thresholded discrete Radon transform (linear interpolation).

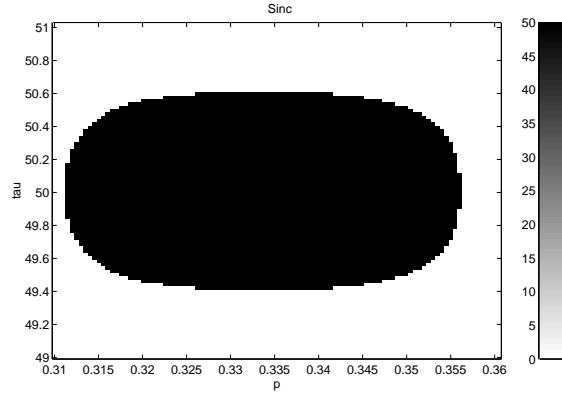


Figure 1.18 The thresholded discrete Radon transform (sinc interpolation).

Radon transform is shown using nearest neighbour interpolation. As could be expected from Eq. 1.7 each of the eight pixels is transformed into a line in the parameter domain. Due to the different values of the non-zero pixels in the image, it is possible to identify which non-zero pixel in the image corresponds to a certain line in the parameter domain. Another important issue is that the lines found in the parameter domain intersect, and amplify in two points, corresponding to the underlying line-parameters. Sampling parameters for this example can be found in Table 1.1.

Fig. 1.21 shows a similar image generated having ten points on each line. From the corresponding discrete Radon transform, shown in Fig. 1.22, the underlying line parameters can be extracted, due to clearly marked position of the peaks in the parameter domain. Finally Fig. 1.23 shows an image with two complete lines, and in Fig. 1.24 are shown the corresponding parameter domain. The images have been scaled individually corresponding to the minimal and maximal pixel value, hence it can be seen that the peak value in the last case is very high compared to the two others.

It has been demonstrated, that each point in the image is transformed into a line in the discrete parameter domain, where the offset and slope (in the parameter domain) vary according to the position of the pixel in the image. This leads to an important issue, namely that the discrete Radon transform of any given pixel cannot be confined in the finite, thus truncated, parameter domain. In this way (perhaps important) information will get lost using discrete slant stacking.

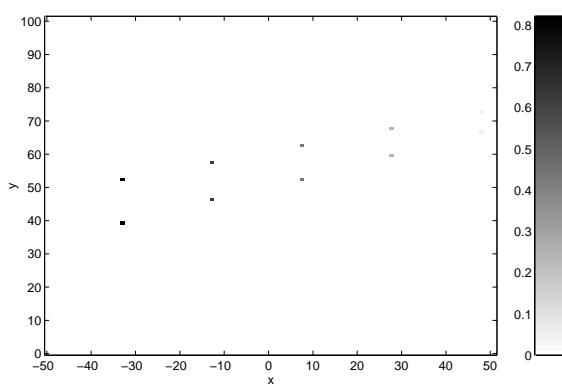


Figure 1.19 Image with ten points lying on two lines.

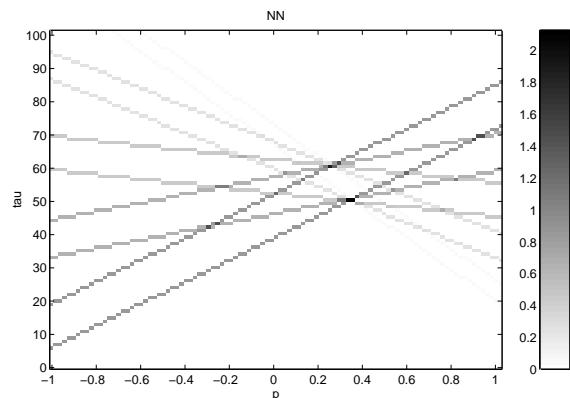


Figure 1.20 The corresponding discrete Radon domain.

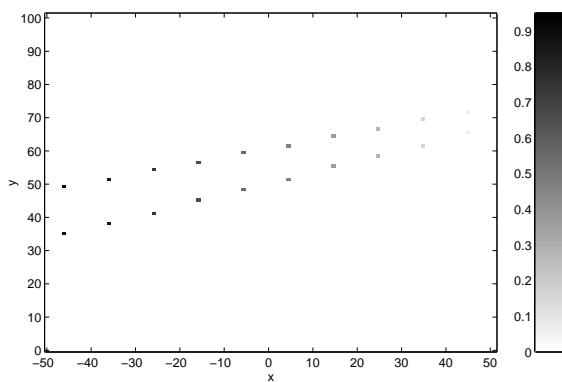


Figure 1.21 Image with 20 points lying on two lines.

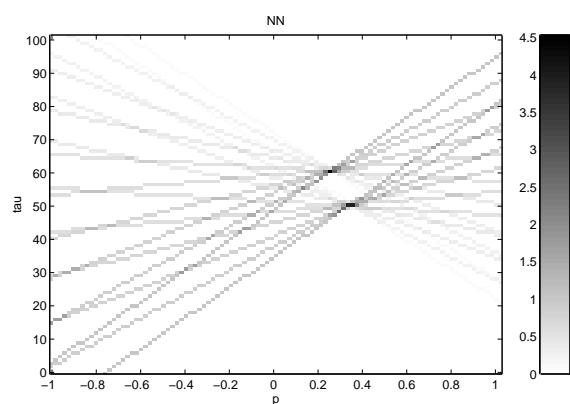


Figure 1.22 The corresponding discrete Radon domain.

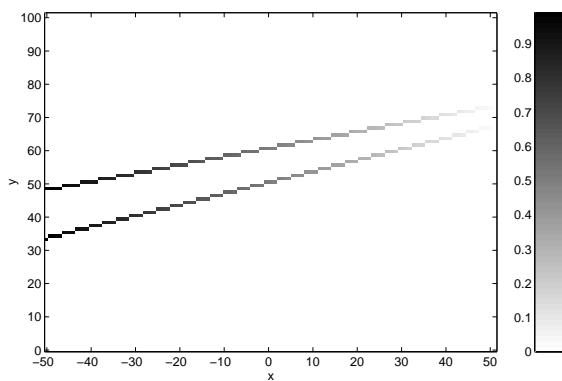


Figure 1.23 Image with 202 points lying on two lines.

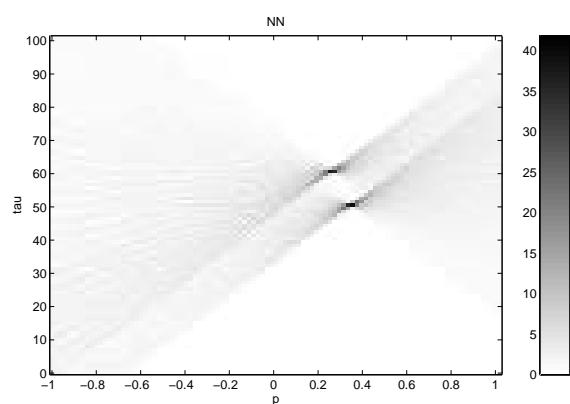


Figure 1.24 The corresponding discrete Radon domain.

1.7 Slant Stacking and Images with Steep Lines

It has been demonstrated that the parameter domain must be sampled sufficiently dense, in order to assure that the discrete approximations to the Radon transform does not cause aliasing problems.

Another problem arises when images include one or more lines with high slope (nearly vertical lines). First assume an image with two lines as shown in Fig. 1.25. One line has a slope of $\frac{1}{9}$ and the other a slope of 9. The corresponding discrete Radon transform is shown in Fig. 1.26, where the parameter p has been limited between -1 and 1 . The parameters of the first line is clearly identified, but due to the high slope and the truncated parameter domain, the other line is not detected. This illustrates that only lines with parameters that lie within the limits of the parameter domain can be identified. It is a very general property of the discrete Radon transform that a parameter domain with many samples is required when no prior information is provided. If, e.g., the slopes of the lines are limited between -0.1 and 0.1 , then this should be used to limit the discrete parameter domain, in order to reduce the computational work.

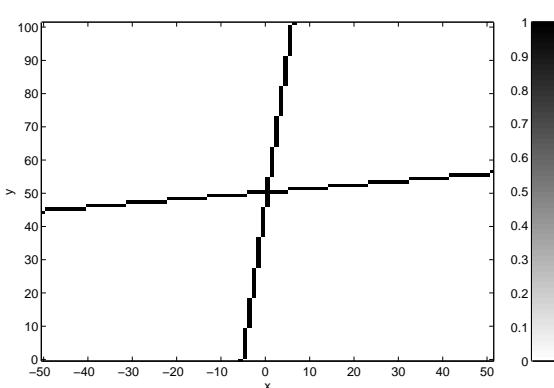


Figure 1.25 An image with two lines having a slope of $\frac{1}{9}$ and 9 respectively.

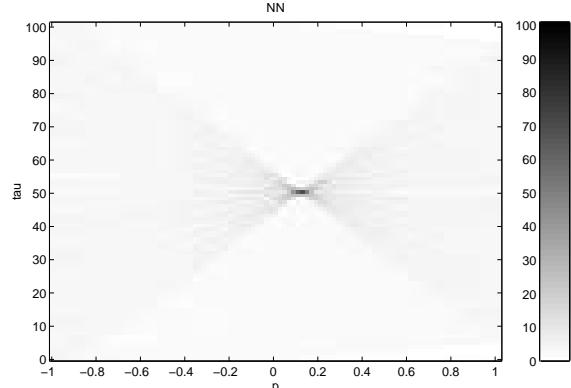


Figure 1.26 The corresponding discrete parameter domain, with the slope parameter p limited between -1 and 1 .

One way to enable detection of both lines, is to increase K , i.e., the number of samples in the p -direction, in order to have $p_{max} = p_{min} + (K-1)\Delta p > 9$. This is possible, but the computational cost would be unnecessary high, due to the large number of samples in the discrete parameter domain.

Another problem is that the peak value in the discrete parameter domain should be around 100, (or perhaps somewhat lower as shown in Subsection 1.5.1), due to the fact that the each line is made of 101 non-zero samples with value of 1. Assume that the parameter domain was expanded to include the very steep line, then the value of the discrete Radon transform at the peak would only be in the region of 10. This problem is indicated in Fig. 1.27. The dashed line show the line under which the values are summed. But stepping forward in m , in this case only four samples contribute to the discrete Radon transform, and several samples between the four are skipped. In Fig. 1.25 only around 10 pixels would contribute to the discrete Radon transform in the region of the parameter of the second line parameters, hence the peak value would be much lower than possible from the image.

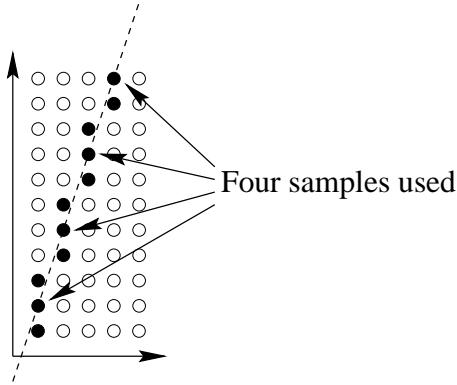


Figure 1.27 Image with steep line. The filled circles are pixels with a value of one, and the others have a value of zero. The dashed line indicate the line of integration. Note how only four samples go into the discrete Radon transform and several pixels are skipped.

Another serious reason for concern, comes from the analysis of functions with vertical lines. The corresponding Radon transform can be found from the analysis of a (continuous) vertical line.

$$g(x, y) = A \delta(x - x^*) \Rightarrow \check{g}(p, \tau) = A \int_{-\infty}^{\infty} \delta(x - x^*) dx = A \quad (1.37)$$

where it should be noted that the Radon transform does not depend on the position of the line x^* . This implies that this information is not maintained in the parameter domain, hence the position of vertical lines cannot be detected using slant stacking.

A way to overcome the described problems is to compute two parameter domains. The first $\check{g}(p, \tau)$ is restricted to $-p_{lim} < p \leq p_{lim}$, and the other $\check{g}(r, \eta)$ is used to manage the remaining line orientations.

$$\check{g}(p, \tau) = \int_{-\infty}^{\infty} g(x, p x + \tau) dx \quad (1.38)$$

$$\check{g}(r, \eta) = \int_{-\infty}^{\infty} g(r y + \eta, y) dy \quad (1.39)$$

The two ways of describing lines are related

$$y = p x + \tau \Rightarrow x = r y + \eta \text{ where } \begin{cases} r = \frac{1}{p} \\ \eta = -\frac{\tau}{p} \end{cases} \quad (1.40)$$

Thus if $-p_{lim} \leq p \leq p_{lim}$ then the other slope parameter should be bounded by $-\frac{1}{p_{lim}} \leq r \leq \frac{1}{p_{lim}}$. A reasonable value of the limiting slope is

$$p_{lim} = \frac{\Delta y}{\Delta x} \quad (1.41)$$

This limit assures that the problem illustrated in Fig. 1.27 is eliminated. The change in vertical direction when moving to the next m , during discrete Radon transform, will be less than 1. From Eq. 1.17, this could be interpreted as the digital slope of the line, and $p \frac{\Delta x}{\Delta y}$ lies between -1 and 1. This way of choosing p_{lim} also fulfills the assumption in Eq. 1.30 that $|\alpha| > 1$.

Note that a discrete implementation of the Radon transform defined in Eq. 1.39 does not imply additional programming. By applying the discrete Radon transform to the transpose of the digital image, i.e., $h(m, n) = g(n, m)$, and using Eq. 1.40 to convert the sampling parameters, then the obtained discrete parameter domain estimates the wanted Radon transform.

1.8 Detection of Lines Convolved with a Wavelet

So far only positive valued images have been considered. In this subsection the restriction will be abandoned, by assuming a line convolved with a wavelet. Here a wavelet $\psi(x)$ is defined as a (short) signal centered around $x = 0$. Note that this definition does not agree totally with the one used in wavelet analysis. In wavelet analysis, the wavelet is also restricted to $\int \psi(x) dx = 0$. This setup is relevant in connection with filtering of edges, where the filtering output is both positive and negative.

The model considered is

$$g(x, y) = \psi(y - p^*x - \tau^*) \quad (1.42)$$

hence the line has certain parameters (p^*, τ^*) . Comparing Eq. 1.42 with Eq. 1.10, it can be noted that the wavelet $\psi(\cdot)$, has been convolved in the y -direction. The wavelet could also have been convolved perpendicularly to the line, and this could be included in the wavelet as a angular dependent scaling parameter. The corresponding Radon transform is given by

$$\check{g}(p, \tau) = \begin{cases} \int_{-\infty}^{\infty} \psi(\tau - \tau^*) dx & \text{for } p = p^* \\ \frac{1}{|p - p^*|} \int_{-\infty}^{\infty} \psi(\tilde{x}) d\tilde{x} & \text{for } p \neq p^* \end{cases} \quad (1.43)$$

This result is interesting in various ways. In order to have a peak in the parameter domain at the correct parameters, the wavelet should have a global maximum when the argument is zero, i.e., corresponding to the position of the line.

$$\psi(0) > \psi(x) \quad \forall x \neq 0 \quad (1.44)$$

Another interesting result is that when $p \neq p^*$ the result only depends on the integral of the wavelet. Consider that the wavelet is chosen to $\psi(x) = \delta''(x)$. The integral (from minus infinity to infinity) of this function is 0, and the value of $\delta''(\cdot)$ is infinite, hence the parameter domain would be infinite at the position of the line parameters $(p, \tau) = (p^*, \tau^*)$, and zero otherwise.

In Fig. 1.28 an image containing two lines is created using the digital wavelet $\{-0.5, 1.0, 0.5\}$. It is found that the difference between the discrete parameter domains obtained by using nearest neighbour, linear interpolation, and the sinc expansion approximation is limited, hence Fig. 1.29 only shows the corresponding discrete Radon transform using the nearest neighbour approximation. Two peaks are found corresponding to the line parameters, and the values in the parameter domain are in fact small away from the peaks. Note also the negative value, just over and under the (positive) peak.

1.9 Summary

Slant stacking or τ - p transform was defined as it is common within seismics. The transform was analyzed as a special case of the Radon transform. Discrete approximations were derived, and sampling properties were derived and analyzed by introducing a new sinc expansion strategy. It was demonstrated for curve detection purposes, that a nearest neighbour approximation is adequate if sampling of the parameter domain is sufficiently dense. It was also demonstrated that linear interpolation gave approximately the same parameter domain as the one found with the sinc interpolation strategy.

It has been shown that slant stacking is capable of detecting curve parameters if the slope of the line is below a certain limit related to the sampling parameters. A simple cure for this limitation was also reviewed.

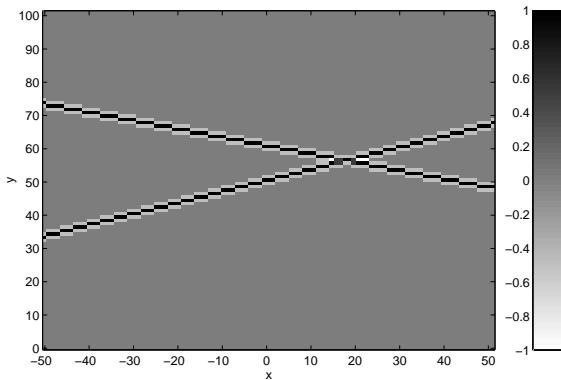


Figure 1.28 Image with two intersecting lines convolved with a wavelet in the vertical direction. Outside the lines the pixels values are zero.

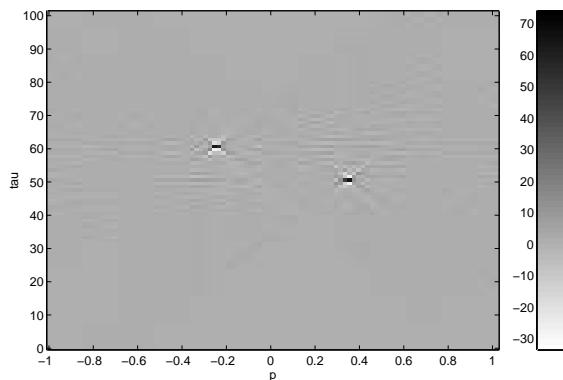


Figure 1.29 The corresponding discrete Radon transform, where the color scaling is determined by the minimum and maximum value, respectively. Besides the areas containing peaks, the values are close to zero.

Chapter 2

The Normal Radon Transform

In this chapter another very popular form of the linear Radon transform are reviewed. Several ways to define a discrete Radon transform is presented and an analysis of the sampling relationships is presented using the same techniques as in Chapter 1. A set of examples are included to illustrate the theory.

2.1 Defining the (ρ, θ) Radon Transform

In Chapter 1 one form of the linear Radon transform has been considered, namely slant stacking. This is merely one linear form of many, and the linear Radon transform can be defined on a more general form as

$$\check{g}(\xi_0, \xi_1, \xi_2) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\xi_0 - \xi_1 x - \xi_2 y) dx dy \quad (2.1)$$

On this form the line is described with (apparently) three degrees of freedom, which is one to many to describe a line, so the three parameters should always have a link, which removes one degree of freedom. In the literature two forms are by far the most common. The first, slant stacking, where the parameters are

$$(\xi_0, \xi_1, \xi_2) = (-\tau, p, -1) \quad (2.2)$$

Another definition of the Radon transform is used in many fields of science, e.g., tomography, astronomy and microscopy [14], where the fundamental function $g(x, y)$ has no preferred orientation. This has lead to describing the line on its normal form

$$\rho = x \cos \theta + y \sin \theta \quad (2.3)$$

i.e., $(\xi_0, \xi_1, \xi_2) = (\rho, \cos \theta, \sin \theta)$, and the normal Radon transform. The term normal Radon transform is not generally accepted, but here it is a very practical term and is only used to be able to distinguish slant stacking from the form shown in Eq. 2.4 can be written as

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (2.4)$$

This definition can be compared to Eq. 1.2, found for slant stacking. Note again that no new symbol for the normal Radon transform has been assigned, but the arguments are used to show which definition is used.

Often another equivalent way of writing Eq. 2.4 is used

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} g(\rho \cos \theta - s \sin \theta, \rho \sin \theta + s \cos \theta) ds \quad (2.5)$$

$$= \frac{1}{|\sin \theta|} \int_{-\infty}^{\infty} g\left(x, \frac{\rho}{\sin \theta} - x \cot \theta\right) dx \quad (2.6)$$

$$= \frac{1}{|\sin \theta|} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta\left(y - \frac{\rho}{\sin \theta} + x \cot \theta\right) dx dy \quad (2.7)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (2.8)$$

where the s -axis lies along the line. Eq. 2.8 can be compared to Eq. 1.1.

The meaning of the normal parameters used to specify the position of the line are shown in Fig. 2.1. The parameter ρ is the shortest distance from the origin of the coordinate system to the line, and θ is an angle corresponding to the angular orientation of the line.

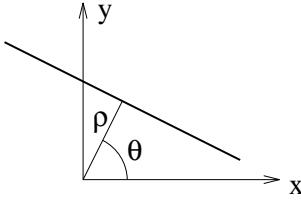


Figure 2.1 The two parameters ρ and θ used to specify the position of the line.

The first thing to notice about the normal Radon transform is that all lines can be described by choosing that $0 \leq \theta < 2\pi$ and $\rho \geq 0$, but frequently other limits are used. If negative values of ρ are introduced the parameter domain are bounded by

$$0 \leq \theta < \pi \text{ and } -\rho_{max} \leq \rho \leq \rho_{max} \quad (2.9)$$

where ρ_{max} is positive, and finite when any discrete implementation is considered. Both limitations of the parameter domain are valid, and they are very much related by

$$\check{g}(\rho, \theta) = \check{g}(-\rho, \theta + \pi) \quad (2.10)$$

which is easily found from Eq. 2.4 using the fact that the delta function is even.

The normal Radon transform is (not surprisingly) linked to slant stacking, and the one can be expressed from the other. From Eq. 2.6 it is found that

$$\check{g}(\rho, \theta) = \frac{1}{|\sin \theta|} \int_{-\infty}^{\infty} g\left(x, \frac{\rho}{\sin \theta} - x \cot \theta\right) dx = \frac{1}{|\sin \theta|} \check{g}\left(p = -\cot \theta, \tau = \frac{\rho}{\sin \theta}\right) \quad (2.11)$$

The relation is very simple, and the link between the two set of parameters can be found directly from Figs. 2.1 and 1.3. It is clear that Eq. 2.11 impose a problem when θ is close to 0, but in that case a link can be made to the Radon transform defined in Eq. 1.39,

$$\check{g}(\rho, \theta) = \frac{1}{|\cos \theta|} \int_{-\infty}^{\infty} g(ry + \eta, y) dy \text{ where } \begin{cases} r = -\tan \theta \\ \eta = \frac{\rho}{\cos \theta} \end{cases} \quad (2.12)$$

Using Eq. 2.4 several properties can be derived. Some of these can be found in appendix B. The most important property is that the normal Radon transform is a linear function. It should also be mentioned that several functions can be transformed analytically [14, 22]. As described in Sections B.3 and C.1, a set of those has been implemented in order to, e.g., test numerical approximations to the normal Radon transform or its inverse.

2.1.1 The Point Source

Again the point source is used to gain information about the fundamental behavior of the normal Radon transform. Here an arbitrary position of the point source (x^*, y^*) is assumed.

$$g(x, y) = \delta(x - x^*) \delta(y - y^*) \Rightarrow \quad (2.13)$$

$$\begin{aligned} \check{g}(\rho, \theta) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \delta(x - x^*) \delta(y - y^*) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \\ &= \delta(\rho - x^* \cos \theta - y^* \sin \theta) \end{aligned} \quad (2.14)$$

Fig. 2.2 illustrates a point source and the corresponding Radon transform.

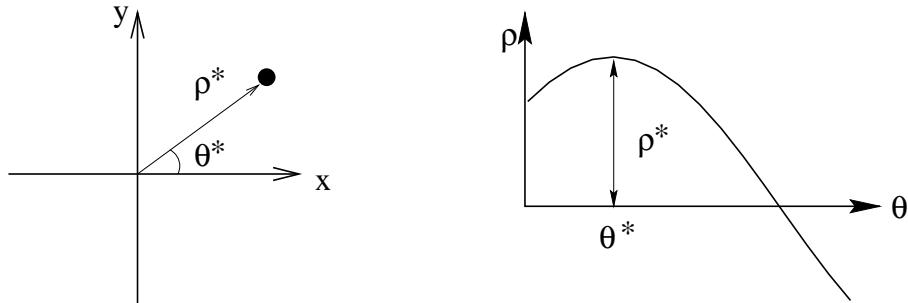


Figure 2.2 To the left is shown a point source, and to the right is shown the corresponding normal Radon transform.

From Eq. 2.14, it is derived that for any function $g(x, y)$ the Radon transform is given by

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(y - y^*) dx^* dy^* \Rightarrow \quad (2.15)$$

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\rho - x^* \cos \theta - y^* \sin \theta) dx^* dy^* \quad (2.16)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\rho - \rho^* \cos(\theta - \theta^*)) dx^* dy^* \quad (2.17)$$

$$\text{where } \rho^* \cos \theta^* = x^* \text{ and } \rho^* \sin \theta^* = y^* \quad (2.18)$$

This result is very interesting with respect to numerical algorithms. It shows that a point is transformed into a sinusoid in the parameter domain, and the Radon transform of a point source can be confined in a truncated parameter domain.

$$g(x, y) = 0 \text{ for } \sqrt{x^2 + y^2} > \rho_{max} \Rightarrow \check{g}(\rho, \theta) = 0 \text{ for } |\rho| > \rho_{max} \quad (2.19)$$

which comes directly from Eq. 2.17. This result should be compared to Eq. 1.7, found for slant stacking.

2.1.2 The (ρ, θ) Radon Transform of a Line

One of the major advantages of the normal Radon transform regards line detection. Modelling a line with certain parameters (ρ^*, θ^*) with the delta function gives

$$g(x, y) = \delta(\rho^* - x \cos \theta^* - y \sin \theta^*) \Rightarrow \quad (2.20)$$

$$\begin{aligned} \check{g}(\rho, \theta) &= \int_{-\infty}^{\infty} \delta(\rho^* - (\rho \cos \theta - s \sin \theta) \cos \theta^* - (\rho \sin \theta + s \cos \theta) \sin \theta^*) \, ds \\ &= \int_{-\infty}^{\infty} \delta(\rho^* - \rho \cos(\theta - \theta^*) + s \sin(\theta - \theta^*)) \, ds \\ &= \int_{-\infty}^{\infty} \frac{1}{|\sin(\theta - \theta^*)|} \delta\left(\frac{\rho^* - \rho \cos(\theta - \theta^*)}{\sin(\theta - \theta^*)} + s\right) \, ds, \quad \text{if } \sin(\theta - \theta^*) \neq 0 \\ &= \frac{1}{|\sin(\theta - \theta^*)|} \end{aligned} \quad (2.21)$$

and if $\theta = \theta^*$, i.e., $\sin(\theta - \theta^*) = 0$, it is found that

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \delta(\rho^* - \rho) \, ds = \begin{cases} 0, & \text{if } \rho^* \neq \rho \\ \int_{-\infty}^{\infty} \delta(0) \, ds, & \text{if } \rho^* = \rho \end{cases} \quad (2.22)$$

Not surprisingly, the result is that a peak is formed, when $\rho = \rho^*$ and $\theta = \theta^*$ (and finite terms will also be found). Furthermore note that there is no limitations on the orientation of the line, which was a problem with the way slant stacking was defined.

2.2 The Discrete (ρ, θ) Radon Transform

A discrete approximation to the continuous slant stacking was straight forward, cf. Eqs. 1.12 and 1.13. Again all parameters are sampled linearly.

$$\begin{aligned} x &= x_m = x_{min} + m\Delta x, \quad m = 0, 1, \dots, M-1 \\ y &= y_n = y_{min} + n\Delta y, \quad n = 0, 1, \dots, N-1 \\ \theta &= \theta_t = \theta_{min} + t\Delta\theta, \quad t = 0, 1, \dots, T-1 \\ \rho &= \rho_r = \rho_{min} + r\Delta\rho, \quad r = 0, 1, \dots, R-1 \end{aligned} \quad (2.23)$$

Apparently the transform has a lot of free parameters, but normally several of them are fixed. Eq. 2.19 implies that the interesting part of the function $g(x, y)$ should be shifted towards the center of the coordinate system, in order to get the smallest number of samples in the ρ -direction.

Now it will be assumed that the image is square

$$\Delta x = \Delta y \quad (2.24)$$

$$M = N \quad (2.25)$$

which implies that the samples should lie in a symmetrical interval around zero, i.e.,

$$x_{min} = -x_{max} = -\frac{(M-1)}{2} \Delta x \quad (2.26)$$

$$y_{min} = x_{min} = -y_{max} = -\frac{(M-1)}{2} \Delta x \quad (2.27)$$

$$\rho_{min} = -\rho_{max} = -\frac{(R-1)}{2} \Delta \rho \quad (2.28)$$

Considering the angular sampling, the angular starting point can be chosen to

$$\theta_{min} = 0 \quad (2.29)$$

Secondly, the sampling interval of θ should be set to span π , i.e.,

$$\Delta\theta = \frac{\pi}{T} \quad (2.30)$$

This leaves only a few free parameters, namely Δx , M , T , R , and $\Delta\rho$. These limitations are assumed in the following.

One way of defining a discrete normal Radon transform could be to approximate Eq. 2.8,

$$\check{g}(\rho_r, \theta_t) = \int_{-\infty}^{\infty} g(\rho_r \cos \theta_t - s \sin \theta_t, \rho_r \sin \theta_t + s \cos \theta_t) ds \quad (2.31)$$

$$\approx \Delta s \sum_{j=0}^{J-1} g(\rho_r \cos \theta_t - s_j \sin \theta_t, \rho_r \sin \theta_t + s_j \cos \theta_t) \quad (2.32)$$

where s_j is a linear sampling of the variable s , done as in Eq. 2.23. This approach have some implications. The most serious is that for a given value of j , the image points found in Eq. 2.32, i.e., $(x, y) = (\rho_r \cos \theta_t - s \sin \theta_t, \rho_r \sin \theta_t + s \cos \theta_t)$ (in principle) never coincide with samples in the (assumed) image $g(m, n) = g(x_m, y_n)$, hence an interpolation in both variables (x, y) is needed. This two-dimensional interpolation can and should be avoided, due to the extra noise added, compared to a one-dimensional interpolation. Another question is how densely should the sampling interval of the parameter s , i.e., how Δs should be adjusted.

Another way of approximating the normal Radon transform and only get a one-dimensional interpolation in the image is to reformulate Eq. 2.8 using the slant stacking definition, like it was done in Eq. 2.11 and Eq. 2.12. That solution only requires an one-dimensional interpolation. Either implemented as a (fast) nearest neighbour approximation like below, or e.g., a linear interpolation.

$$\sin \theta > \frac{1}{\sqrt{2}} : \check{g}(\rho, \theta) = \frac{1}{|\sin \theta|} \check{g} \left(p = -\cot \theta, \tau = \frac{\rho}{\sin \theta} \right) \quad (2.33)$$

$$\approx \frac{\Delta x}{|\sin \theta|} \sum_{m=0}^{M-1} g(m, [\alpha m + \beta]) \quad (2.34)$$

$$\text{where } \alpha = -\cot \theta \text{ and } \beta = \frac{\rho - x_{min}(\cos \theta + \sin \theta)}{\Delta x \sin \theta} \quad (2.35)$$

$$\sin \theta \leq \frac{1}{\sqrt{2}} : \check{g}(\rho, \theta) = \frac{1}{|\cos \theta|} \check{g}(r = -\tan \theta, \eta = \frac{\rho}{\cos \theta}) \quad (2.36)$$

$$\approx \frac{\Delta x}{|\cos \theta|} \sum_{n=0}^{M-1} g([\alpha n + \beta], n) \quad (2.37)$$

$$\text{where } \alpha = -\tan \theta \text{ and } \beta = \frac{\rho - x_{min}(\cos \theta + \sin \theta)}{\Delta x \cos \theta} \quad (2.38)$$

where it has been used that $\Delta x = \Delta y$, $M = N$, and $x_{min} = y_{min}$. Note that besides the (projection) factors $\frac{1}{|\sin \theta|}$ and $\frac{1}{|\cos \theta|}$ in front of the integrals, the way to approximate the two integrals follow section 1.4 and especially Section 1.7. The idea is shown in Algorithm 2.1.

ALGORITHM 2.1 : THE DISCRETE NORMAL RADON TRANSFORM

```

For t = 0 to T-1                                //For all values of theta
    costheta = cos(theta(t))
    sintheta = sin(theta(t))
    rhooffset = x_min*(costheta+sintheta)          //Common factor
    If sintheta>1/sqrt(2)
        alpha = -costheta/sintheta                 //Project onto x-axis
    For r = 0 to R-1                               //Digital slope is set
        alpha = -costheta/sintheta                 //For all values of rho
        beta = (rho(r)-rhooffset)/(Delta_x*sintheta) //Offset is set
        mmin and mmax are set using Eq. 1.19       //Expression omitted
        sum = 0                                      //Initialize sum
        For m = mmin to mmax                        //For all legal values of x
            sum=sum+g(round(alpha*m+beta))           //Increment sum
        End
        g_radon(r,t)=Delta_x*sum/sintheta          //Update matrix element
    End
Else
    alpha = -sintheta/costheta                  //Project onto y-axis
    For r = 0 to R-1                           //Digital slope is set
        alpha = -sintheta/costheta                //For all values of rho
        beta = (rho(r)-rhooffset)/(Delta_x*cosheta) //Offset is set
        nmin and nmax are set using Eq. 1.19      //Expression omitted
        sum = 0                                      //Initialize sum
        For n = nmin to nmax                      //For all legal values of x
            sum=sum+g(round(alpha*n+beta),n)         //Increment sum
        End
        g_radon(r,t)=Delta_x*sum/abs(cosheta)      //Update matrix element
    End
End

```

As it can be seen in Algorithm 2.1 the program is actually split up in two parts, corresponding to the axis on which the line integral has been projected. Each of the two parts have been optimized like it has been described below Algorithm 1.1. The expressions for `mmin` and `mmax` are given in Eq. 1.19, and the expressions for `nmin` and `nmax` can be derived from the same equation with some minor changes in variables.

2.2.1 Sampling Properties of the (ρ, θ) Radon Transform

In order to gain knowledge of the sampling relationships, the sinc-expansion procedure, shown in Subsection 1.4.3, is used. But instead of integrating to get the result, Eq. 1.24, i.e., the sinc interpolation result from slant stacking and the relation between the normal Radon transform and slant stacking, Eq. 2.11, are coupled in order to get the result. This shows an important way to gain more knowledge about the Radon transform. Some issues are better understood or easier derived with, e.g., slant stacking, but can easily be translated to the normal Radon transform (and reversely).

$$g(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) \frac{\sin(\frac{\pi}{\Delta x}(x - x_m))}{\frac{\pi}{\Delta x}(x - x_m)} \frac{\sin(\frac{\pi}{\Delta y}(y - y_n))}{\frac{\pi}{\Delta y}(y - y_n)} \Rightarrow \quad (2.39)$$

$$\check{g}(\rho, \theta) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) I\left(p = -\cot \theta, \tau = \frac{\rho}{\sin \theta}, x_m, y_n\right) \quad (2.40)$$

$$= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) I(\rho, \theta, x_m, y_n) \quad (2.41)$$

$$I(\rho, \theta, x_m, y_n) = \frac{\Delta x}{\psi} \sin\left(\psi \min\left\{\frac{1}{|\sin \theta|}, \frac{1}{|\cos \theta|}\right\}\right) \quad (2.42)$$

$$\text{where } \psi = \frac{\pi}{\Delta x}(\rho - x_m \cos \theta - y_n \sin \theta) \quad (2.43)$$

Using the technique shown in Subsection 1.4.3, the sampling properties of the normal Radon transform is determined by ψ . As a function of θ the function $\min\{1/|\sin \theta|, 1/|\cos \theta|\}$ has values between 1 and $\sqrt{2}$. Using the maximum of $\sqrt{2}$, the sampling interval of the ψ should not be greater than $\frac{\pi}{\sqrt{2}}$. Using this limit for both parameters gives

$$\Delta\psi = \left| \frac{\partial\psi}{\partial\rho} \right| \Delta\rho = \frac{\pi}{\Delta x} \Delta\rho \leq \frac{\pi}{\sqrt{2}} \Rightarrow \Delta\rho \leq \frac{\Delta x}{\sqrt{2}} \quad (2.44)$$

$$\Delta\psi = \left| \frac{\partial\psi}{\partial\theta} \right| \Delta\theta \leq \frac{\pi}{\sqrt{2}} \Rightarrow \Delta\theta \leq \min_{x_m, y_m} \frac{\Delta x}{\sqrt{2}\sqrt{x_m^2 + y_m^2}} = \frac{\Delta x}{2|x_{min}|} \quad (2.45)$$

Note that some rather conservative assumptions are made, in order to make the expressions valid for all values of (x_m, y_n) . The results show that for a given image a lower limit of sampling intervals in the parameter domain exist. Violating these limits can lead to aliasing problems in the parameter domain. It can be shown that Eq. 2.44 and Eq. 2.45 corresponds to a demand, that the change of line positions should be below one sample, when changing either ρ or θ with their respective sampling intervals.

With a given image $g(m, n)$ and sampling distance Δx , the angular sampling distance $\Delta\theta$ can be set using Eq. 2.45, hence the angular sampling parameters can be chosen to

$$T = \lceil \pi(M-1) \rceil \quad \text{and} \quad \Delta\theta = \frac{\pi}{T} \quad (2.46)$$

where it has been used that $x_{min} = -\Delta x(M-1)/2$. When using Eq. 2.44 and Eq. 2.28, the only remaining parameter to determine is R (or $\rho_{min} = -\rho_{max}$), which can be determined by assuming $\Delta\rho = \frac{\Delta x}{\sqrt{2}}$, cf. Eq. 2.44, and inserting Eqs. 2.26-2.28 into Eq. 2.19. In total,

$$\rho_{max} = \max_{x_m, y_n} \sqrt{x^2 + y^2} = \sqrt{2}|x_{min}| \Rightarrow \quad (2.47)$$

$$R = 2 \frac{\sqrt{2}|x_{min}|}{\Delta\rho} + 1 = 2M - 1 \quad \text{and} \quad \rho_{min} = -\frac{R-1}{2}\Delta\rho \quad (2.48)$$

2.2.2 The Discrete (ρ, θ) Radon Transform of Several Lines

Fig. 2.3 shows an image with $101 * 101$ pixels, where seven lines can be found with different orientations and amplitudes μ (pixel value on the line) as shown in Table 2.1.

Fig. 2.4 shows the discrete parameter domain using Algorithm 2.1, i.e., a nearest neighbour approximation. Seven peaks are found and the values in the peaks correspond to the curve amplitudes μ . As seen from Table 2.2, the sampling parameters were chosen according to Eq. 2.44 and Eq. 2.45. Note that a high number of samples is required in the parameter domain compared to the original image.

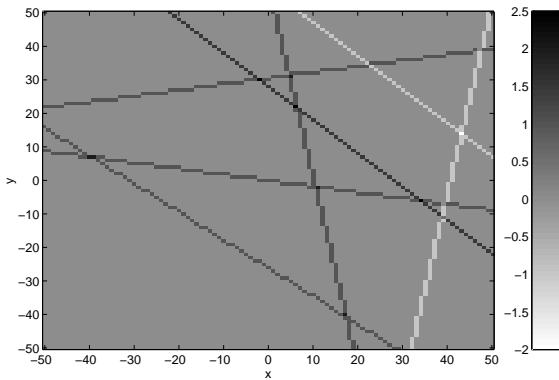


Figure 2.3 An image with seven lines. The parameters are listed in Table 2.1. The background color correspond to a value of zero.

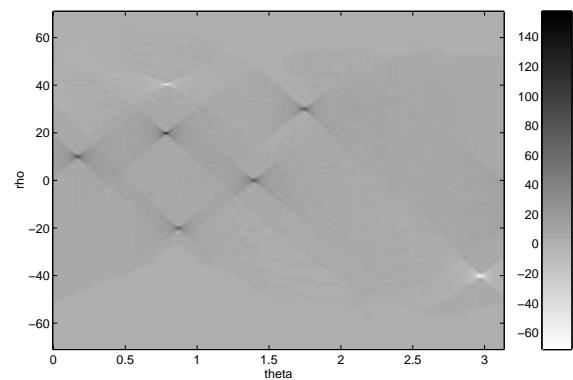


Figure 2.4 The corresponding discrete Radon transform using the nearest neighbour approximation. The background color correspond to a value of zero.

Line number	1	2	3	4	5	6	7
ρ	10	20	40	-20	0	30	-40
θ	0.175	0.785	0.785	0.873	1.396	1.745	2.967
μ	1	1.5	-1	1	1	1	-1

Table 2.1 Line parameters corresponding to Fig. 2.3.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	101	R	201
N	101	T	315
Δx	1	$\Delta \rho$	0.707
Δy	1	$\Delta \theta$	0.01
x_{min}	-50	ρ_{min}	-70.7
y_{min}	-50	θ_{min}	0

Table 2.2 Sampling parameter settings corresponding to Figs. 2.3 and 2.4.

Now the sampling parameters are changed. Firstly, the sampling parameters corresponding to the discrete parameter domain shown in Fig. 2.5, violates Eq. 2.44 and Eq. 2.45. Here the peak corresponding to line number three have disappeared. Secondly, in Fig. 2.6 a discrete parameter domain is shown, where the sampling parameters are set to zoom in the area of parameters of line number five.

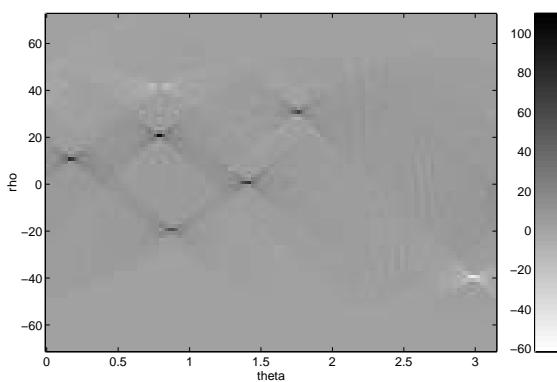


Figure 2.5 The sparsely sampled parameter domain. Note how the peak corresponding to line number three is missing, i.e., at $(\rho = 40, \theta = 0.79)$. The sampling parameters are listed in the left part of Table 2.3.

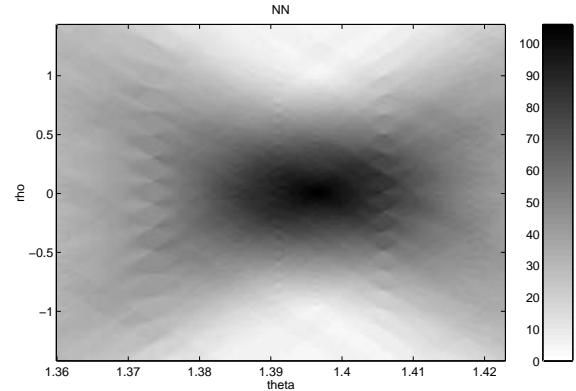


Figure 2.6 The parameter domain with focus in the area of line number five, i.e., at $(\rho = 0, \theta = 1.396)$. The sampling parameters are listed in the right part of Table 2.3.

Parameter domain (zoom out)		Parameter domain (zoom in)	
Parameter	Value	Parameter	Value
R	101	R	201
T	128	T	315
$\Delta\rho$	1.41	$\Delta\rho$	0.0141
$\Delta\theta$	0.02	$\Delta\theta$	$2 \cdot 10^{-4}$
ρ_{min}	-70.7	ρ_{min}	-1.41
θ_{min}	0	θ_{min}	1.36

Table 2.3 Sampling parameter settings corresponding to Figs. 2.5 and 2.6.

Next the discrete parameter domain using linear interpolation and sinc interpolation are displayed in the area of line number five. From Fig. 2.6 it can be seen that the nearest neighbour approximation actually produces a peak, though it is a somewhat different shape compared the peaks found in Fig. 2.7 and Fig. 2.8, which look very much alike. Again for this example, it is found that the peak value is lowest when using linear interpolation.

The evaluation of the results should also incorporate the time used to calculate the three zoomed parameter domains, which is shown in Table 2.4. The complexity of the three approximations are given by

$$\begin{aligned} \mathcal{O}_{\text{Nearest Neighbour}} &= \mathcal{O}(R T M) \approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Linear Interpolation}} &= \mathcal{O}(R T M) \approx \mathcal{O}(M^3) \\ \mathcal{O}_{\text{Sinc Interpolation}} &= \mathcal{O}(R T M^2) \approx \mathcal{O}(M^4) \end{aligned} \quad (2.49)$$

which is analogous to Eq. 1.36. And again it should be noted that linear interpolation is somewhat

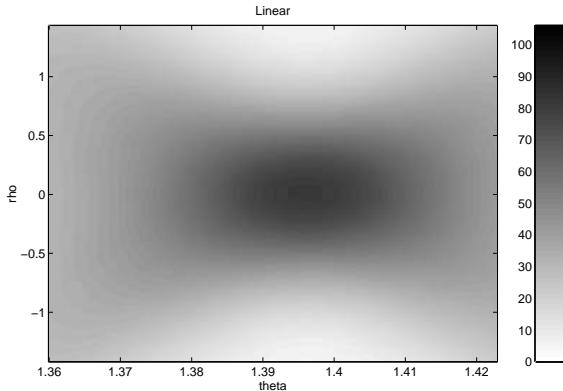


Figure 2.7 The parameter domain with focus in the area of line number five using linear interpolation.

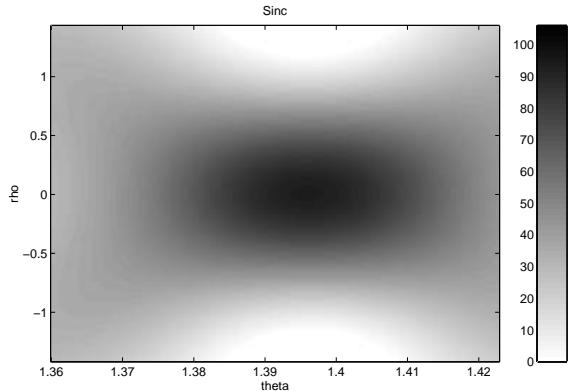


Figure 2.8 The parameter domain with focus in the area of line number five using sinc interpolation.

slower than using nearest neighbour interpolation.

All programs were implemented in C, and for the first two algorithms, the code was optimized and implemented as described below Algorithm 1.1. The huge difference found in Table 2.4 from the first two algorithms to the 1500 seconds used with the sinc interpolation, is due to two things. Firstly, the sinc approximation requires a sinc function, which have to be computed once per iteration. Secondly, the first two algorithms have, cf. Eq. 1.36, a much smaller computational cost, compared to the sinc interpolation strategy, as shown in Eq. 2.49.

Method	Time
Nearest Neighbour	2.9 sec
Linear Interpolation	4.2 sec
Sinc Interpolation	1450 sec

Table 2.4 Time for calculating the discrete Radon transform for the three types of approximation, corresponding to Figs. 2.6, 2.7, and 2.8. All three times are measured on a 120 MHz Pentium.

To visualize the resolution of the discrete normal Radon transform, Figs. 2.9 (nearest neighbour), 2.10 (linear interpolation), and 2.11 (sinc interpolation) show the parameter domain in the area of the peak using a threshold level of 50.

It can be seen that all three implementations give approximately nearly the same resolution, and the resolution in the ρ -direction is a bit over the one derived in Eq. 2.44. The resolution in the θ -direction is approximately five times the one derived in Eq. 2.45, but it must be noted, that this limit was derived by using rather conservative assumptions.

In Fig. 2.12 the discrete parameter domain is shown in the area of the line number three in Table 2.1 by using nearest neighbour approximation, and in Fig. 2.13 the parameter domain has been truncated at the value -50 (note that line number three has amplitude -1, and note that the peak value is approximately -63). It can be seen that resolution here is below the one derived value in Eq. 2.44, in the ρ -direction, which again indicates that the threshold changes the result of this kind of analysis. Likewise Fig. 2.14 shows the same part of the parameter domain found by use of linear interpolation. Finally in Fig. 2.15 the parameter domain has been computed using linear interpolation and then truncated at -50. A significantly smaller region is found, which is to be expected with the high threshold level.

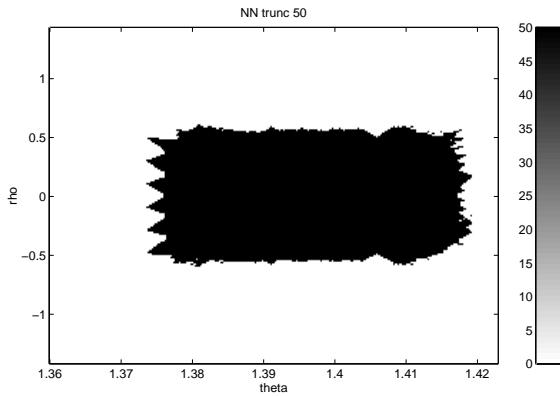


Figure 2.9 The discrete parameter domain using a threshold level of 50 (nearest neighbour).

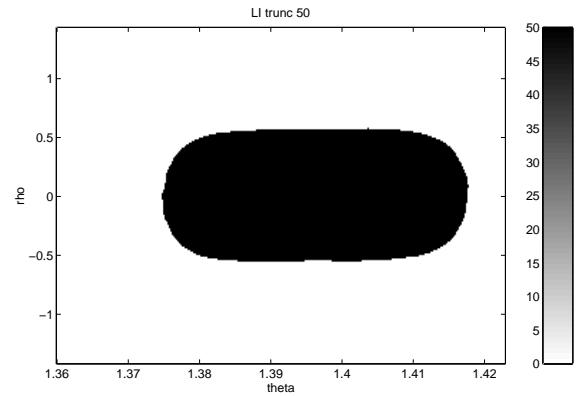


Figure 2.10 The discrete parameter domain using a threshold level of 50 (linear interpolation).

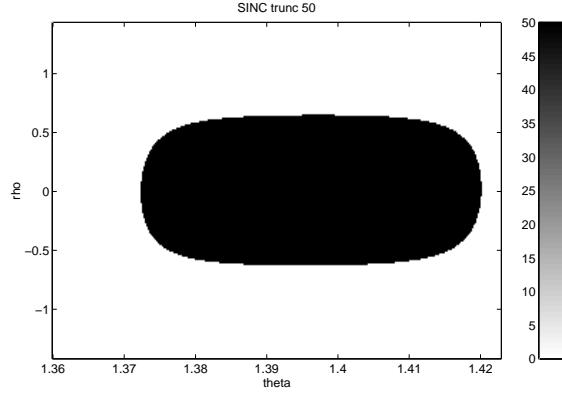


Figure 2.11 The discrete parameter domain using a threshold level of 50 (sinc interpolation).

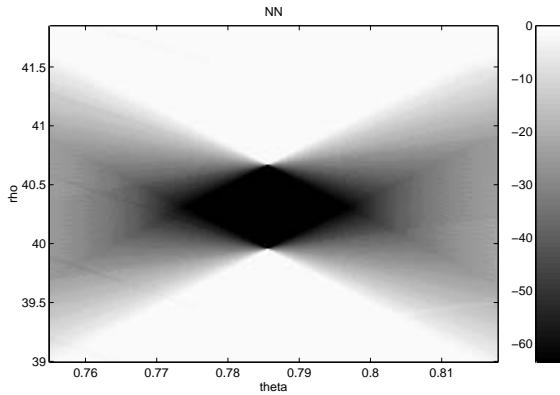


Figure 2.12 The discrete parameter domain (nearest neighbour) in the area of the third line, see Table 2.1.

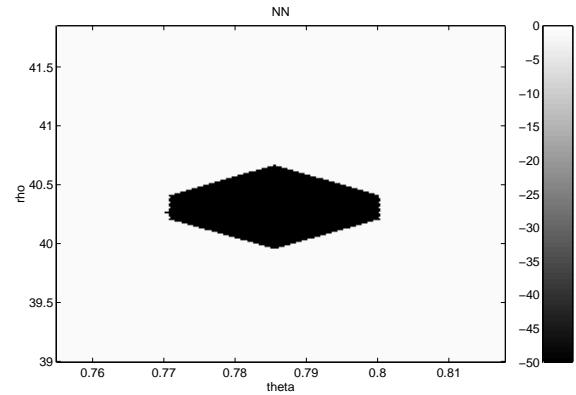


Figure 2.13 The same discrete parameter domain using a threshold level of -50.

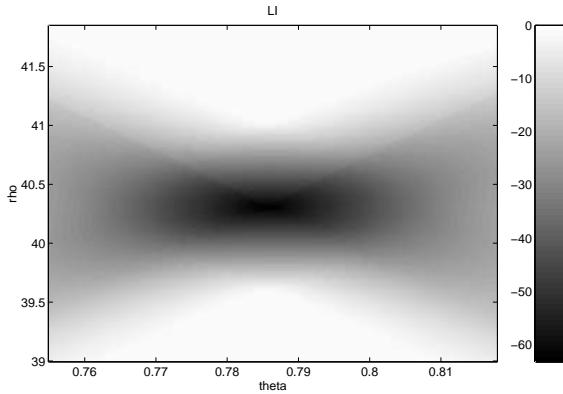


Figure 2.14 The discrete parameter domain (linear interpolation) in the area of the third line, see Table 2.1.

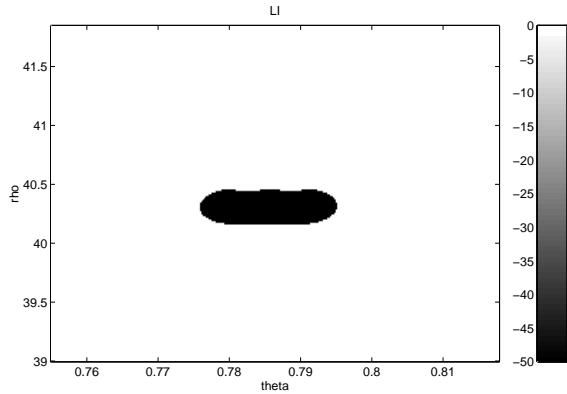


Figure 2.15 The same discrete parameter domain using a threshold level of -50 (linear interpolation).

2.2.3 The Discrete (ρ, θ) Radon Transform of Points

Fig. 2.16 shows an image with only 14 pixels having a value different to zero. The value drops off linearly to the right of the image. The 14 pixels lie on two lines with the same angular orientation, and in Fig. 2.17 the corresponding discrete parameter domain is shown, by use of a nearest neighbour approximation. Sampling parameters can be found in Table 2.2. Each of the 14 points are clearly represented by a sinusoid, and note how the linearity of the Radon transform implies that the sinusoids intersect and peaks are positioned corresponding to the parameters of the two supporting lines.

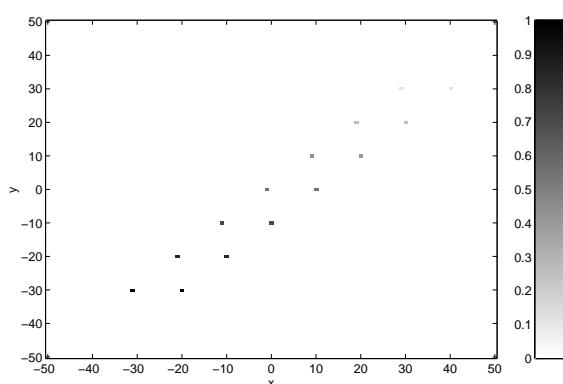


Figure 2.16 An image with 14 pixels, lying on two lines. The values are attenuated towards the right side of the image.

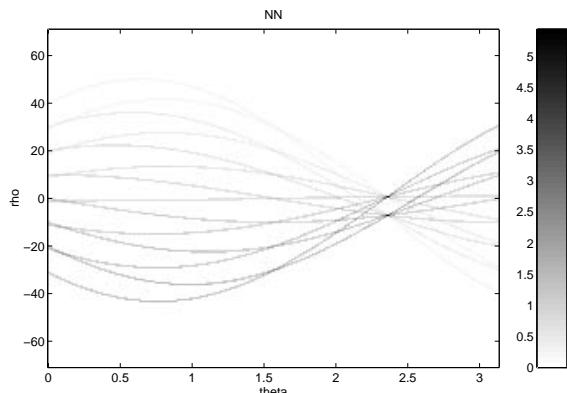


Figure 2.17 The corresponding discrete parameter domain using a nearest neighbour approximation. Note how individual sinusoids form connected curves.

Figs. 2.18 and 2.19 show the discrete parameter domains found by using linear interpolation and the sinc interpolation respectively. Fig. 2.18 shows approximately the same result as Fig. 2.17, but the sinusoids are more smooth. A closer examination of Fig. 2.19 reveals that each of the sinusoids have ringing effects, resulting in undesired non-zero values in the parameter domain also in areas where no sinusoids are found.

Transformation times corresponding to the three approximations are given in Table 2.5.

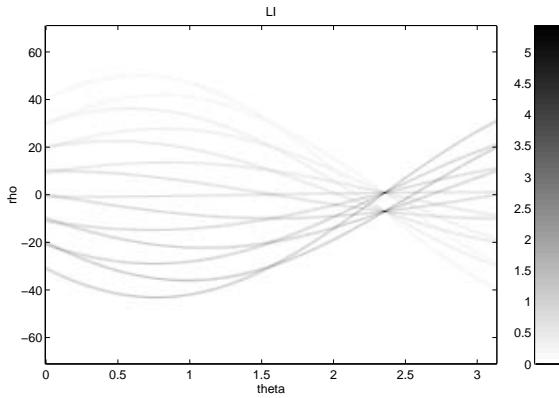


Figure 2.18 The corresponding discrete parameter domain using linear interpolation.

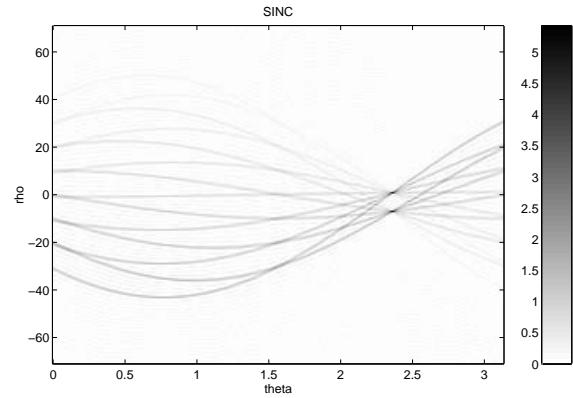


Figure 2.19 The corresponding discrete parameter domain using sinc interpolation.

In Fig. 2.20 is shown the discrete sparsely sampled parameter domain using a nearest neighbour approximation. Sampling parameters are shown in the left side of Table 2.3. It should be noted that the upper peak has a very low value. This is due to the violation of the sampling criterion. It can also be seen that the sinusoids now are broken at certain angles. Using linear interpolation as in Fig. 2.21 only implies that the curves does not break, but note again the low value of the upper peak.

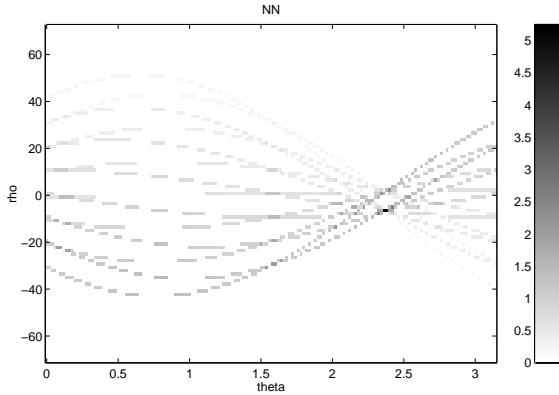


Figure 2.20 The discrete parameter domain using a sparse sampling (nearest neighbour). Note how the curves break up at certain angles.

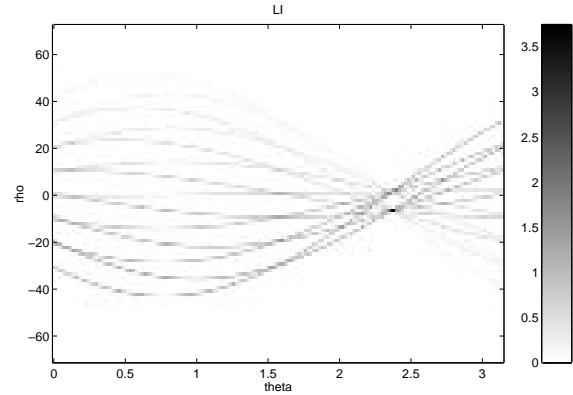


Figure 2.21 The discrete parameter domain using a sparse sampling (linear interpolation). Here the curves nearly break up at certain angles.

Method	Time
Nearest Neighbour	1.9 sec
Linear Interpolation	2.7 sec
Sinc Interpolation	1450 sec

Table 2.5 Time for calculating the discrete Radon transform for the three types of approximation, corresponding to Figs. 2.17, 2.18, and 2.19. All times are measured on a 120 MHz Pentium.

2.3 Summary

The Radon transform was defined on (ρ, θ) form and discrete forms were derived. Both forms were analyzed, and sampling properties were derived and analyzed using a sinc expansion strategy, like it was done in the Chapter 1.

It was demonstrated for curve detection purposes, that a nearest neighbour approximation is adequate if sampling of the parameter domain is sufficiently dense. It was also demonstrated that linear interpolation gave approximately the same parameter domain as the one found with the sinc interpolation strategy.

As expected, it has been demonstrated that the discrete normal Radon transform can detect lines with arbitrary orientation, which could not be fulfilled with slant stacking. The chapter was concerned with quadratic images and the arbitrary line orientation, and that is one of the major differences from the results found in Chapter 1.

Chapter 3

The Hough Transform

So far the Radon transform has been investigated as a tool for line detection. Two different ways of selecting parameters gave mainly the same type of discrete transform, namely, for each position in the parameter domain, sum up pixel values along a line in the image and store the sum with proper scaling at that position.

If the image is very sparse, e.g., a binary image with few non-zero pixels, most of the computer time is spent summing up zeros, that does not contribute to the parameter domain. In one of the most cited patents in the image processing literature, [23], P.V.C. Hough proposed a way of incorporating that prior knowledge into the transform, in order to reduce the computational cost. The idea was initially formulated for lines described with slope and offset parameter (like slant stacking), but many authors have translated the idea into the domain of normal line parameters (ρ, θ) , e.g., [24, 25, 26]. It should be mentioned, that there is some confusion in the literature, regarding how the Hough and the Radon transform are related. It is a problem of definition. A common opinion, which here will be used for defining the Hough transform, is that the Hough transform maps the individual pixels from the image domain into a shape in the parameter domain, and the Radon transform transforms a shape in the image domain into a single pixel in the parameter domain. As written in [23], “For a given point on a line segment in framelet 108¹, a line is drawn in the transformed plane...”, thus the main idea is to transform each pixel in the image individually into the parameter domain.

From a historical point of view it is not clear what motivated the Hough transform. It is likely that it was a new idea, and not motivated by results like the ones shown in Section 1.6. Apparently it was first later on, that it was realized, that the Radon transform and the Hough transform are highly related.

3.1 The (p, τ) Hough Transform

The Hough transform can be derived from the Radon transform [15]. In Eq. 1.9 it was found that

$$\begin{aligned} g(x, y) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(y - y^*) dx^* dy^* \Rightarrow \\ \check{g}(p, \tau) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\tau - y^* + px^*) dx^* dy^* \end{aligned} \quad (3.1)$$

Eq. 3.1 actually expresses what Hough meant. For each point in the image $g(x^*, y^*)$, draw a line in the transformed plane, i.e., the parameter domain. The line is found by setting the argument of the last delta function to zero

$$\tau = y^* - px^* \quad (3.2)$$

¹Reference to an image in the patent.

Eq. 3.1 states that the extra weight, or vote, that the parameter domain should be updated with is proportional to the function value $g(x^*, y^*)$. This is a natural choice, so in this light the Hough transform on a continuous form, c.f. Eq. 3.1, can be viewed as a special case of the Radon transform. But this is only true in the continuous case, and as it will be demonstrated the two transforms are in general not identical in their discrete forms. Note also that the Hough transform commonly is only used in a discrete form.

Eq. 3.1 motivates a new scheme for estimation of the parameter domain. Again in order to reduce the size of the algorithm $y(n)$, is assumed to be set to y_n and so on. Note that the sampling parameters defined in Eq. 1.12 are used again.

ALGORITHM 3.1 : THE HOUGH TRANSFORM

```

Set g_hough(k,h)=0 for all (k,h)           //Initialize Hough space
For m=0 to M-1                            //For all values of x
    For n=0 to N-1                          //For all values of y
        gvalue = g(m,n)                    //Store in simple variable
        If gvalue≠0                         //Only consider non-zero pixels
            For k=0 to K-1                  //For all values of p
                tau=y(n)-p(k)*x(m)          //Calculate the tau
                h=round((tau-tau_min)/Delta_tau) //Find the right sample
                If h≥0 and h<H              //Check if sample is valid
                    g_hough(k,h)=g_hough(k,h)+gvalue //Update parameter domain
            End
        End
    End
End
End

```

Algorithm 3.1 describes the core of the Hough transform, though it should be noted that nearly all people working with the Hough transform use the (ρ, θ) form. The key point is that for the non-zero pixels in the image and each value of p_k , a value of τ is calculated, which (here) is rounded to the nearest neighbour. In the following the implications of this strategy will be analyzed and compared to the Radon transform.

The first issue is to compare the computational complexities. Assuming that only some of the image pixels have a non-zero value, then the computational complexity of the Hough transform is given in Eq. 3.4. For easy comparison the first line of Eq. 1.36 is repeated in Eq. 3.3, corresponding to the nearest neighbour discrete Radon transform.

$$\mathcal{O}_{\text{Radon}} = \mathcal{O}(K H M) \approx \mathcal{O}(M^3) \quad (3.3)$$

$$\mathcal{O}_{\text{Hough}} = \mathcal{O}((M N)_r K) \approx \mathcal{O}((M N)_r M) \quad (3.4)$$

where the index r is used to indicate that a reduced number of the pixel has to be transformed. If the image only contains one non-zero pixel then $(M N)_r = 1$, and the Hough transform will be the fastest of the two transforms, but a more realistic value of $(M N)_r$ lies between M and $M N$, depending on the structure of the image. This also indicates that most applications of the Hough transform is in connection with binary images, often rather sparse (many zeros and few ones).

Again the procedure described right after Algorithm 1.1, is applied to reduce the computational cost, i.e., the continuous mapping equation is rewritten using the discrete parameters

$$\tau = y - px \Rightarrow \quad (3.5)$$

$$h \approx \frac{y_n - (k\Delta p + p_{min})x_m - \tau_{min}}{\Delta\tau} = \kappa k + \zeta \quad \text{where } \begin{cases} \kappa = -\frac{x_m \Delta p}{\Delta\tau} \\ \zeta = \frac{y_n - x_m p_{min} - \tau_{min}}{\Delta\tau} \end{cases} \quad (3.6)$$

If a nearest neighbour approximation is applied then Eq. 1.19 can be used to determine the values of h , that corresponds to parameters lying in the truncated discrete parameter domain, i.e.,

$$0 \leq h = [\kappa k + \zeta] \leq H - 1 \Rightarrow \begin{cases} -\frac{\zeta + \frac{1}{2}}{\kappa} < k < \frac{H - \frac{1}{2} - \zeta}{\kappa} & \text{if } \kappa > 0 \\ \frac{H - \frac{1}{2} - \zeta}{\kappa} < k < -\frac{\zeta + \frac{1}{2}}{\kappa} & \text{if } \kappa < 0 \end{cases} \Rightarrow \quad (3.7)$$

$$k_{min} = \max \left\{ 0, \left\lceil -\frac{\zeta + \frac{1}{2}}{\kappa} \right\rceil \right\} \text{ and } k_{max} = \min \left\{ K - 1, \left\lfloor \frac{H - \frac{1}{2} - \zeta}{\kappa} \right\rfloor \right\} \text{ if } \kappa > 0$$

$$k_{min} = \max \left\{ 0, \left\lceil \frac{H - \frac{1}{2} - \zeta}{\kappa} \right\rceil \right\} \text{ and } k_{max} = \min \left\{ K - 1, \left\lfloor -\frac{\zeta + \frac{1}{2}}{\kappa} \right\rfloor \right\} \text{ if } \kappa < 0 \quad (3.8)$$

In Algorithm 3.2 an implementation of the Hough transform is shown using (p, τ) -parameters and a nearest neighbour approximation in the parameter domain, and for optimization purposes Eq. 3.8 is used to assure that the pixels actually lie in the discrete parameter domain.

ALGORITHM 3.2 : THE OPTIMIZED HOUGH TRANSFORM

```

Set g_hough(k,h)=0 for all (k,h)                                //Initialize Hough space
For m=0 to M-1                                                 //For all values of x
  For n=0 to N-1                                               //For all values of y
    gvalue = g(m,n)                                           //Store in simple variable
    If gvalue≠0
      kappa = -x(m)*Delta_p/Delta_tau                         //Calculate digital slope
      zeta = (y(n)-x(m)*p_min-tau_min)/Delta_tau             //Calculate digital offset
      Calculate k_min and k_max from Eq. 3.8
      For k=k_min to k_max                                     //For all valid values of p
        h=round(kappa*k+zeta)                                 //Find the right tau-value
        g_hough(k,h)=g_hough(k,h)+gvalue                      //Update parameter domain
      End
    End
  End
End

```

In the algorithm, it is used that only the image pixels with value different from zero are transformed. For non-binary (floating point valued) images a threshold can be applied, but the threshold level strongly depends on the nature of the image.

3.1.1 Line Detection Using The Hough Transform

In Fig. 3.1 yet another synthetic image containing six line pieces is shown, and the line parameters can be found in Table 3.1. Fig. 3.2 shows the corresponding discrete parameter domain by use of discrete slant stacking. Sampling parameters can be found in Table 3.2. Six clearly marked

peaks positioned at the correct line parameters can be found in the discrete parameter domain, even now, where only segments of lines are available. The starting and ending points of the line segments cannot directly be found from the parameter domain, but a more interesting discovery is that the parameter domain found by use of Hough transform in this case is *identical* to the one found with the discrete Radon transform. This is due to the choice of sampling parameters and the way the two algorithms compute the discrete indices.

In Eq. 1.17 it is found that

$$n = \left[\frac{p_k x_m + h\Delta\tau + \tau_{min} - y_{min}}{\Delta y} \right] \quad (3.9)$$

If the sampling parameters have been chosen as $\Delta\tau = \Delta y$ and $\tau_{min} = y_{min}$ as suggested in Subsection 1.4.4, then Eq. 3.9, corresponding to the discrete Radon transform, becomes very simple

$$n = \left[\frac{p_k x_m + h\Delta y}{\Delta y} \right] = h + \left[\frac{p_k x_m}{\Delta y} \right] \quad (3.10)$$

and for the Hough transform with the same choice of $\Delta\tau = \Delta y$ and $\tau_{min} = y_{min}$, the mapping function is given by Eq. 3.6:

$$h = \left[\frac{n\Delta y + y_{min} - p_k x_m - \tau_{min}}{\Delta\tau} \right] \quad (3.11)$$

$$= \left[\frac{n\Delta y - p_k x_m}{\Delta y} \right] \quad (3.12)$$

$$= n - \left[\frac{p_k x_m}{\Delta y} \right] \quad (3.13)$$

which is exactly the same as in Eq. 3.10. So the interesting conclusion is that using slope and offset parameters and nearest neighbour approximation, the two algorithms give *exactly* the same discrete parameter domain, when choosing $\Delta\tau = \Delta y$ and $\tau_{min} = y_{min}$. It should be noted that the Hough transform needs a general multiplicative factor of Δx in order to get the same scaling as the discrete Radon transform.

A major difference of the two algorithms comes with the time needed to compute the discrete parameter domains. Using optimized discrete Radon transform requires 0.42 seconds, and the optimized Hough transform requires only 0.026 seconds, i.e., a factor of 16 in favor of the Hough transform. In this case only 3.34% of the pixels is non-zero. Both times are measured on a 120 MHz Pentium PC.

Line number	1	2	3	4	5	6
p	-0.33	-0.10	0.25	-0.25	0.50	0.00
τ	50	70	60	40	50	20
x_{start}	-41	-11	-50	-11	-50	-41
x_{end}	29	50	9	29	9	9

Table 3.1 Line parameters corresponding to Fig. 3.1 are given along with the starting point x_{start} , and the ending point x_{end} .

Now the sampling parameters of the discrete parameter domain are changed in order to focus on two of the peaks. Sampling parameters are shown in the left side of Table 3.3. In Fig. 3.3 the discrete parameter domain is shown using discrete Radon transform, and likewise in Fig. 3.4

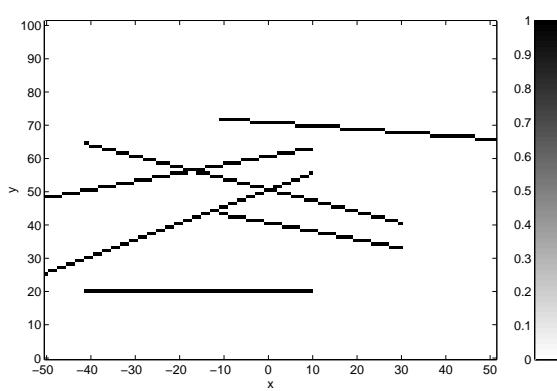
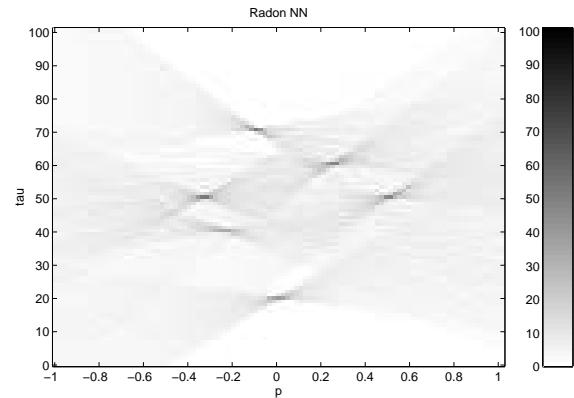
**Figure 3.1** An image with 6 line pieces**Figure 3.2** The discrete parameter domain found by either the Radon transform or the Hough transform.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	101	H	101
N	101	K	101
Δx	1	Δp	0.02
Δy	1	$\Delta \tau$	1
x_{min}	-50	p_{min}	-1
y_{min}	0	τ_{min}	0

Table 3.2 Sampling parameter settings.

by use of Hough transformation. It can be seen that two clearly marked peaks are found by use of the discrete Radon transform, just as expected from the previous chapters. From Fig. 3.4 it is seen that the Hough transform malfunctions. Firstly, it is found, that the levels are very low. Secondly, the regions of expected line parameters are only weakly marked, hence the parameter domain is useless.

Next, a sparse sampling of the parameter domain in Figs. 3.5 and 3.6 is used corresponding to the right side of Table 3.3. It can be seen that the discrete Radon transform malfunctions, and only two peaks are clearly marked. This is due to the under-sampling of the parameter domain. In contrast to Fig. 3.5, Fig. 3.6 shows that the Hough transform performs well. Six peaks can be found, but it should be noted that the resolution is poor.

Parameter domain (zoom in)		Parameter domain (zoom out)	
Parameter	Value	Parameter	Value
H	101	H	33
K	101	K	51
Δp	0.004	Δp	0.04
$\Delta \tau$	0.15	$\Delta \tau$	3
p_{min}	0.2	p_{min}	-1
τ_{min}	48	τ_{min}	0

Table 3.3 Sampling parameter settings for the zoom figures.

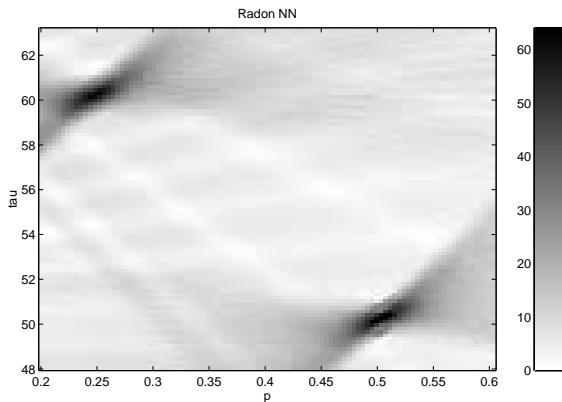


Figure 3.3 The discrete parameter domain found by use of the discrete Radon transform in the area with two line parameters.

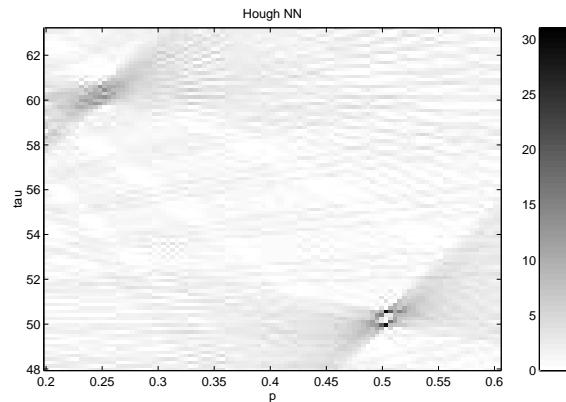


Figure 3.4 The discrete parameter domain found by use of the discrete Hough transform in the area with two of the line parameters.

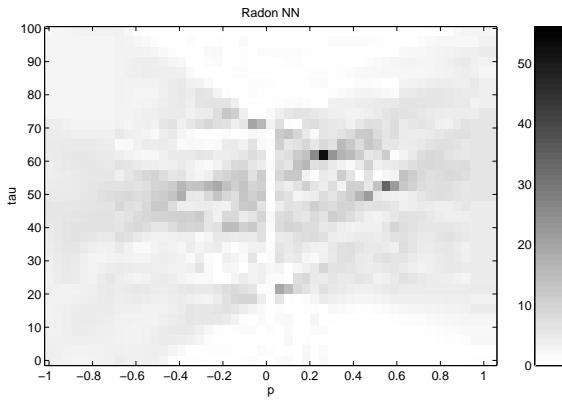


Figure 3.5 The sparsely sampled discrete parameter domain found by use of the discrete Radon transform.

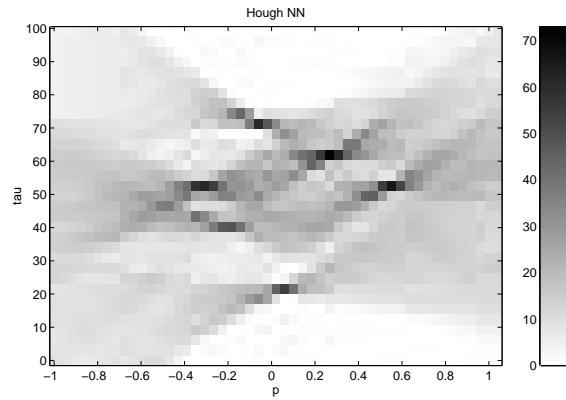


Figure 3.6 The sparsely sampled discrete parameter domain found by use of the discrete Hough transform.

3.1.2 Choosing Sampling Parameters with the (p, τ) Hough Transform

The reason why the Hough transform malfunctions when the parameter domain is sampled densely, and apparently works well with a coarsely sampled parameter domain can be investigated from the mapping procedure using a nearest neighbour approximation. From Eq. 3.6 it will be shown that a band of image points are mapped into the same parameter vector.

$$h = \left[\frac{y_n - p_k x_m - \tau_{min}}{\Delta\tau} \right] \Leftrightarrow \quad (3.14)$$

$$0 = \left[\frac{(n\Delta y + y_{min}) - p_k(m\Delta x + x_{min}) - \tau_h}{\Delta\tau} \right] \Leftrightarrow \quad (3.15)$$

$$|n - \alpha m - \beta| < \frac{1}{2} \frac{\Delta\tau}{\Delta y} \quad \text{where} \quad \begin{cases} \alpha = p_k \frac{\Delta x}{\Delta y} \\ \beta = \frac{p_k x_{min} + \tau_h - y_{min}}{\Delta y} \end{cases} \quad (3.16)$$

The important result is that the way the Hough transform is defined implies that all pixels lying around the (digital) line $n = \alpha m + \beta$ within a band of width $\frac{\Delta\tau}{\Delta y}$, measured in pixels in the n -direction, will be mapped into the same point, (k, h) , in the discrete parameter domain. This explains the results shown in Figs. 3.4 and 3.6. Using a densely sampled parameter domain, with,

e.g., $\Delta\tau = 0.1\Delta y$ implies that the width of the band in the image domain only amounts to $\frac{1}{10}$ of a sample, hence from a statistical point of view, only every tenth pixel along the digital line contributes to the discrete parameter domain, and a coarsely sampled parameter domain, with, e.g., $\Delta\tau = 5\Delta y$, implies that for a given point in the discrete parameter domain, and a certain value of x_m , five pixels will contribute to the same position in the parameter domain, if their value is different from zero. If no prior knowledge of the image is known, the conclusion is that a reasonable value of sampling distance of τ is, $\Delta\tau = \Delta y$.

Using the Hough transform with a coarsely sampled parameter domain can imply that the pixels need not lie on a perfect line, and still result in a peak in the parameter domain. For the Radon transform, further implications and limitations with respect to noise are discussed in Chapter 5.

It has been shown above, that the Hough transform requires that $\Delta\tau$ is not chosen too small. The parameter Δp can be chosen from different criteria. One criteria is that the absolute (digital) slope of the line in Eq. 3.6 does not exceed 1, in order to avoid that a pixel in the image is mapped into a perforated line in the discrete parameter domain, as shown in Figs. 3.7 and 3.8. This perforation problem can imply that the peak is not found in the discrete parameter domain.

$$|\kappa| = \left| -\frac{x_m \Delta p}{\Delta \tau} \right| \leq 1 \Rightarrow \Delta p < \frac{\Delta \tau}{\max |x|} = \frac{2\Delta \tau}{(M-1)\Delta x} \quad (3.17)$$

It can be noted that this limit matches Eq. 1.32 derived for the discrete (p, τ) Radon transform, when choosing $\Delta\tau = \Delta y$.

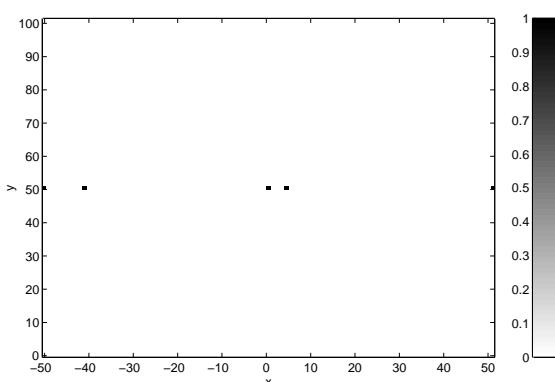


Figure 3.7 An image with 5 points lying on a line.

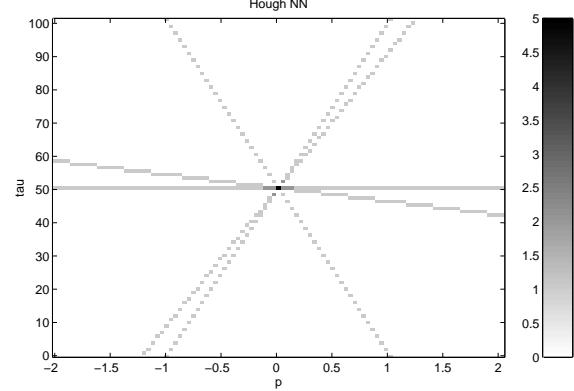


Figure 3.8 The corresponding Hough transform.

3.2 The (ρ, θ) Hough Transform

The Hough transform can be defined using normal parameters (ρ, θ) and Eq. 2.16 motivates the definition of the Hough transform

$$\begin{aligned} g(x, y) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(y - y^*) dx^* dy^* \Rightarrow \\ \check{g}(\rho, \theta) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\rho - x^* \cos \theta - y^* \sin \theta) dx^* dy^* \end{aligned} \quad (3.18)$$

so the core of (ρ, θ) Hough transform is that each point in the image is transformed into a sinusoid in the discrete parameter domain, and in Algorithm 3.3 the Hough transform is shown using (ρ, θ)

parameters. The easiest implementation distributes the pixel value for every value of θ_t . In the following the image is assumed quadratic, cf. Eqs. 2.24 and 2.25, and the sampling parameters defined in Eq. 2.23 are used.

ALGORITHM 3.3 : THE (ρ, θ) HOUGH TRANSFORM

```

Set g_hough(r,t)=0 for all (r,t)           //Initialize Hough space
For m=0 to M-1                            //For all values of x
  For n=0 to N-1                          //For all values of y
    gvalue = g(m,n)                      //Store value in a simple variable
    If gvalue≠0                           //Only consider non-zero pixels
      For t=0 to T-1                     //For all values of theta
        rho=x(m)*cos(theta(t))+y(n)*sin(theta(t)) //Calculate the rho
        r=round((rho-rho_min)/Delta_rho)          //Find the right sample
        If r≥0 and r<R                   //Check if sample is valid
          g_hough(r,t)=g_hough(r,t)+gvalue       //Update parameter domain
      End
    End
  End
End
End

```

This algorithm is valid, but the algorithm still needs a lot of checking in the inner parts of the loops, where it is computational expensive, so for increasing the efficiency several parts should be optimized. Ideas for this is shown in Subsection 3.2.2.

3.2.1 Choosing sampling parameters with the (ρ, θ) Hough Transform

Following the procedure shown in Subsection 3.1.2, a similar expression for the nearest neighbour mapping is found

$$r^* = \frac{x_m \cos \theta_t + y \sin \theta_t - \rho_{min}}{\Delta\rho} \text{ and } r = [r^*] \Rightarrow \quad (3.19)$$

$$\left| m \cos \theta_t + n \sin \theta_t - \left(\frac{\rho_{min} - x_{min}(\cos \theta_t + \sin \theta_t)}{\Delta x} \right) \right| < \frac{1}{2} \frac{\Delta\rho}{\Delta x} \quad (3.20)$$

which shows that the Hough transform maps all pixels in the image that lies within a band of width $\frac{\Delta\rho}{\Delta x}$ into the same point in the discrete parameter domain. The width is measured in pixels perpendicular to the line. This implies that $\Delta\rho$ has to be chosen sufficiently large, and not below Δx , i.e.,

$$\Delta\rho \geq \Delta x \quad (3.21)$$

Concerning the choice of $\Delta\theta$, it should be avoided that a pixel in the image is mapped into a perforated sinusoid. Using a first order approximation gives

$$\max_{x_m, y_n} \left| \frac{\partial r^*}{\partial \theta} \right| \Delta\theta < 1 \Rightarrow \max_{x_m, y_n} \frac{\sqrt{x_m^2 + y_n^2}}{\Delta\rho} \Delta\theta < 1 \Rightarrow \Delta\theta < \frac{\sqrt{2}}{(M-1)} \frac{\Delta\rho}{\Delta x} \quad (3.22)$$

In Fig. 3.9 an image with five non zero pixels is shown, and in Fig. 3.10 the corresponding discrete Hough transform. The sampling parameters have been set from Eq. 3.21 and only using $T = 50$, where Eq. 3.22 and Eq. 2.30 implies that $T = 142$ is necessary.

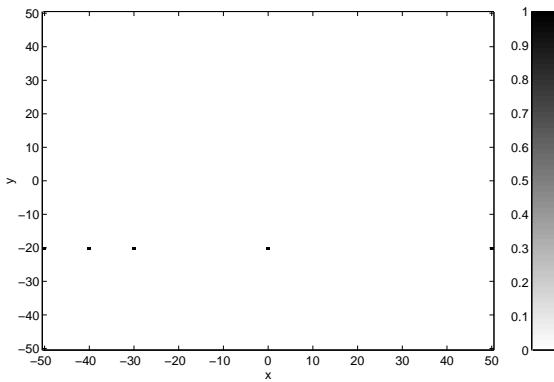


Figure 3.9 An image with 5 points lying on a line.

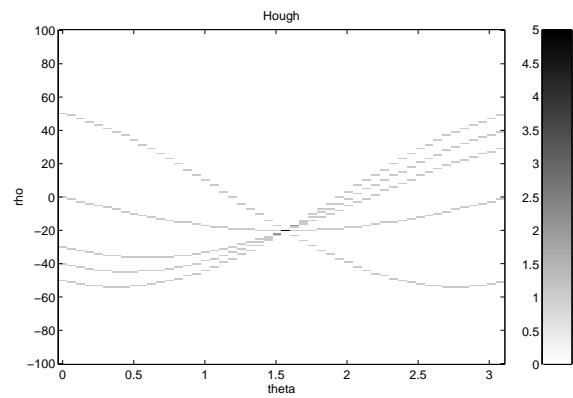


Figure 3.10 The corresponding Hough transform. Note the perforation of the sinusoids.

Note that the discrete Radon and discrete Hough transform can give identical result in the (p, τ) case, but in the (ρ, θ) case the results will always (somewhat) differ. This is also clear from the way sampling of the discrete parameter domain should be chosen.

For further reading on choosing sampling parameters [27] is recommended.

3.2.2 Comparison Between Different Optimization Strategies

Here four different optimizations strategies are compared.

Case 1 No particular optimization as it is shown in Algorithm 3.3.

Case 2 In this case, it is exploited that the fundamental mapping can be written by computing the four vectors containing relative x- and y-values, and cosine and sine to the possible values of θ_t .

$$x_{\text{rel}}(m) = \frac{x_m}{\Delta\rho}, \quad y_{\text{rel}}(n) = \frac{y_n}{\Delta\rho}, \quad \text{and } \text{rhooff} = \frac{\rho_{\min}}{\Delta\rho} \quad (3.23)$$

$$\text{costheta}(t) = \cos \theta_t \text{ and } \text{sintheta}(t) = \sin \theta_t \quad (3.24)$$

These arrays are computed one time only and used in the t -loop, hence extra calculations of sines and cosines are avoided, as shown in following equation

$$r = [x_{\text{rel}}(m)\text{costheta}(t) + y_{\text{rel}}(n)\text{sintheta}(t) + \text{rhooff}] \quad (3.25)$$

Case 3 In Algorithm 3.3 it is necessary to check that the value of r lies within the image. The cure for this problem can be incorporated in the design of discrete parameter domain, simply by increasing R , i.e., the number of samples in the ρ -direction to fulfill

$$\left. \begin{aligned} \rho_{\min} &= -\frac{R-1}{2}\Delta\rho \\ 0 &\leq \left[\frac{x_m \cos \theta_t + y_n \sin \theta_t - \rho_{\min}}{\Delta\rho} \right] < R \text{ for all } (m, n) \end{aligned} \right\} \Rightarrow \quad (3.26)$$

$$\max_{x_m, y_n} |x_m \cos \theta_t + y_n \sin \theta_t| = \max_{x_m, y_n} \sqrt{x_m^2 + y_n^2} = \sqrt{2}|x_{\min}| < \frac{R}{2}\Delta\rho \Rightarrow \quad (3.27)$$

$$R > \frac{\Delta x}{\Delta\rho} \sqrt{2}(M-1) \quad (3.28)$$

where it has been used that $x_{\min} = -\Delta x(M-1)/2$.

Case 3 assumes that Eq. 3.28 is fulfilled and otherwise the implementation is like in Case 2. Normally the expansion of the parameter domain will require extra memory, and in practice no extra computations are needed for that purpose.

Case 4 This case is like Case 3, but for the mapping two matrices \mathbf{a} calculated from the vectors shown in Case 2.

$$\mathbf{xc}(m, t) = \mathbf{xrel}(m) * \text{costheta}(t) \text{ and } \mathbf{ys}(n, t) = \mathbf{yrel}(n) * \text{sintheta}(t) - \text{rhooff} \quad (3.29)$$

Hence for the mapping it can simply be used that

$$r = [\mathbf{xc}(m, t) + \mathbf{ys}(n, t)] \quad (3.30)$$

A further reduction of the computational cost can be achieved from the fact that some platforms provides fast truncation of floating-point numbers compared with the round-function. In this case the constant rhooff can be added with 0.5, and truncation is used instead of the rounding function.

This optimization technique will require two matrices and the computational load is lowered significantly if the tables are used many times, i.e., if the image contains few non-zero pixels and the algorithm only has to be used once, then the overhead associated with the computation of the matrices \mathbf{xc} and \mathbf{ys} will degrade the performance.

Yet another strategy to optimize the calculation of the sinusoid, can be found in [26]. The idea is to use the fact that ρ can be written as $\tilde{\rho}_{m,n} \cos(\theta_t - \tilde{\theta}_{m,n})$, where $x_m + jy_n = \tilde{\rho}_{m,n} e^{j\tilde{\theta}_{m,n}}$. This strategy implies that one table of cosine values is needed, but additional multiplications are needed and interpolation in the table is necessary. The techniques demonstrated in Cases 2, 3, and 4 involve no additional approximation.

First, the times for generating Fig. 3.10 are measured in the four cases shown above. This corresponds to Fig. 3.9 with only five non-zero pixels. Second, the image has been altered by adding a very small bias in order to have all pixel values considered non-zero, hence all 10201 pixels take up time in the transform. From Table 3.4 it can be seen that initialization of the arrays uses too much time in the case of only 5 non-zero pixels, so that the four cases are more or less identical in performance, but in the case of 10201 pixels, it can be seen that the case 3 and 4 outperforms the two other cases. The measurements have been made using a 120 MHz Pentium, hence the ratios look somewhat different with other processor types. Table 3.4 also demonstrates the huge time-difference found when going from 5 to 10201 non-zero pixels in the Hough transform, and in this case, the small additive bias does not imply any visible change of the parameter domain. This also demonstrates that preprocessing, e.g., in the simplest case as a thresholding of the images can imply a huge speedup of the program.

Case	Time (5 Pixels)	Time (10201 Pixels)
1: No optimization	8 msec	1884 msec
2: Vector mapping	10 msec	742 msec
3: Avoid limit check and use vector mapping	8 msec	478 msec
4: Avoid limit check and use matrix mapping	7 msec	454 msec

Table 3.4 Time consumption for the different types of optimization.

3.2.3 Other Hough-like Algorithms

A huge number of authors have proposed other kinds of Radon or Hough-like algorithms. Here only a few of them will be mentioned. Furthermore [25, 28, 29] are recommended for further reading.

3.2.3.1 Random Radon transform

One of the more radically different ideas is the Random Radon transform [30, 31]. The output of the algorithm is still peaks in the parameter domain at the positions of the lines. The main idea can be generalized to handle more general types of curves [32], but here it will only be presented for detection of lines in binary images using normal line parameters.

Basically the idea is to take two non-zero pixels randomly from the image from positions (x_1, y_1) and (x_2, y_2) , and then increase the parameter domain with one vote at the line parameter (ρ, θ) matching both pixels, which is given by

$$\tan \theta = \frac{x_2 - x_1}{y_1 - y_2} \text{ and } \rho = x_1 \cos \theta + y_1 \sin \theta \quad (3.31)$$

This procedure is repeated over and over again. Each iteration is very fast, and a lot of refinements are possible with this strategy, e.g., with respect to storage of the parameter domain [33], but the really tricky part is to choose a good stopping criterion. Stopping the iteration too fast a bad estimate of the parameter domain will be obtained, and stopping too late means waste of computing time. The method works and is potentially very fast, but is iterative.

3.2.3.2 The Gradient Method

In [34] a method for increasing the speed of the (generalized) Radon transform. The idea is here demonstrated for the slant stacking using continuous parameters. If the tangent of the line (curve) can be estimated locally in the image domain, then the computational cost can be lowered significantly. Given an image point with non-zero value at position (x_0, y_0) and that the local estimate of the slope is \hat{p} , then the image point only needs to get mapped into a single point in the parameter domain

$$(x_0, y_0)|\hat{p} \rightarrow (p = \hat{p}, \tau = y_0 - \hat{p}x_0) \quad (3.32)$$

which implies that the computational cost will decrease with an order or magnitude, due to the point to point mapping. The method heavily depends on the quality of the local gradient detector and the noise level.

3.2.3.3 The Hierarchical Hough Transform

In [35] and [25] a hierarchical Hough algorithm for line detection in binary images is presented. The method first uses a very coarse sampling of the parameter domain, to which all non-zero pixels are transformed. Due to the very coarse sampling this operation is very fast. Then the areas of the parameter with zero or almost zero votes are rejected for further investigation. The areas of high votes are very likely to hold line parameters, hence they are divided into smaller pieces (the sampling intervals in the parameter domain are lowered), and the method is repeated until line parameters have been detected with a high accuracy, compared to the initial resolution. The method is potentially fast if large areas of the parameter domain can be rejected in the early stages of the method.

3.2.3.4 A Length Invariant Transform

In some curve detection algorithms the actual length of the curve is without interest, hence the Radon transform using (ρ, θ) as parameters could be redefined as

$$\check{g}(\rho, \theta) = \frac{1}{L(\rho, \theta)} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (3.33)$$

where $L(\rho, \theta)$ is the length of the line with parameters (ρ, θ) that lies within the image. That length is actually the Radon transform of a square box, so it is given in subsection B.3.2 along with the scaling property given in Eq. B.22.

In that way the parameter domain will show a value close to 1 if the image contains lines with amplitude 1. The price of this is that the Radon transform will be highly dominated by noise when $|\rho|$ gets large, due to a decrease in $L(\rho, \theta)$.

This alteration can very easily be incorporated in the Hough transform as well, but it should solely be used as a post-Hough weighting function of the parameter domain, and not used directly in the mapping, due to a heavy and very unneeded rise in the computational load.

3.3 Summary

It has been shown that the (p, τ) Hough transform can be defined in a way that gives exactly the same discrete parameter domain as found with the nearest neighbour approximation of the discrete Radon transform. The (ρ, θ) Hough transform does not have the same property.

It has also been shown that the Hough transform behaves very differently when changing the sampling intervals in the discrete parameter domain, compared to the discrete Radon transform.

Different optimization schemes has been presented for the (ρ, θ) Hough transform and a set of other Hough-like transforms have been described briefly.

Chapter 4

The FCE-Algorithm

A recurring problem in computer image processing is the identification of curves with specific shapes. The transform developed by Hough [23] for detection of complex patterns in binary digital images has been discussed by several authors, e.g., [24], [25] and generalized for multi-dimensional pattern detection in [35].

As the previous chapters have shown, the Radon transform is a mapping from an image domain to a parameter domain, where the parameters characterize the curves to be identified. The Radon transform can be generalized to handle arbitrary curves [14]. The Hough version of the generalized Radon transform detects specific values of parameters, as spatially extended curves are transformed to produce spatially compact features in the parameter domain. In this way, the generalized Radon transform converts a difficult global detection problem in the image domain into a more easily solved local peak detection problem in the parameter domain.

In recent years, progress has been made to understand and increase the speed of the generalized Radon transform [36, 37, 35]. Investigations concerning both the traditional and more recent generalized Radon transform for curve identification schemes point out two problems with respect to computational cost. First, the estimation schemes are not capable of exploiting image points with zero value (zero image points), and second, estimation is computationally expensive, as it estimates unneeded information. This is a consequence of parameter estimation, in areas of the parameter domain, where curve parameters are very unlikely.

The basis for curve parameter estimation as described in this chapter is a binary image. The binary image can be produced in several ways, e.g., by edge filtering, deconvolution, or mean field annealing.

In this chapter, a new algorithm for fast curve estimation, called the FCE-algorithm, is presented. The key idea of the FCE-algorithm is to pre-condition the parameter domain, creating irregular regions corresponding to the parameter regions of interest. Furthermore, the FCE-algorithm takes advantage of pixels with zero value in generating the pre-condition map. Initially, the FCE-algorithm identifies regions of the parameter domain which contain peaks representing curves in the image. Subsequently, it estimates a traditional generalized Radon transform within these regions.

The identification of hyperbolas is of particular interest within seismic signal processing. Over the years the seismic industry has developed a method of recording seismic signals resulting in images in which the investigated reflections produce curves. The recording geometry of seismic data acquisition results in signals having the same geometrical mid point between source and receiver, see, e.g., [38, 39]. The set of signals corresponding to the same mid point is called a common mid point gather (CMP-gather). A common technique in seismic data analysis for estimation of hyperbola parameters is velocity analysis, e.g., [19]. Within the presented work

some features from velocity analysis are incorporated, however, the presented method has less computational cost.

Recently, the multipulse excited speech coding technique, which was originally proposed by Atal *et al* [40], has been suggested for seismic deconvolution by Cookey *et al.* [41]. Basically, the multipulse model for speech assumes the speech signal to be a result of an impulse train transformed by the shape of the vocal tract filter. In seismic deconvolution, the input signal is often modeled as an all-pole impulse and the layered earth reflectivity model is considered as an impulse train, see, e.g., [39]. Thus, the multipulse technique can be used to map seismic signals in a CMP-gather into a binary image, where the image value is set to one in case of a reflection and zero otherwise. The reflections represent information concerning the geological structure of the subsurface.

The generalized Radon transform is described in Section 4.1 with some of its properties, regarding curve detection. Creation of the pre-conditioning map is described in Section 4.2, where the *image point mapping* procedure is developed. In section 4.3, parameter domain sampling is discussed and section 4.4 describes parameter domain blurring, which is a result of discretization. The FCE-algorithm is presented in section 4.5, and is applied to two numerical examples for detection of hyperbolas in Section 4.6. The FCE-algorithm is also presented in [1, 2, 42].

It should be mentioned that many authors have used slant stacking for interpretation of the seismic gathers [43, 44, 45, 46, 47]. Note also that the use of the generalized Radon transform for detection of curve parameters without forming a binary image can be found in [48, 42].

4.1 The Generalized Radon Transform

Let $g(x, y)$ be a continuous signal of the continuous variables x and y and let ξ denote an η -dimensional parameter vector defined as

$$\xi = (\xi_1, \dots, \xi_i, \dots, \xi_\eta) \quad (4.1)$$

where ξ spans the parameter domain.

4.1.1 The Continuous Generalized Radon Transform

The generalized Radon transform can be defined in various ways [15, 13]. A very general form is

$$\check{g}(\xi) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\phi(x, y; \xi)) dx dy \quad (4.2)$$

Using the definition given in Eq. 4.2, shapes expressed by the parameter form $\phi(x, y; \xi) = 0$ can be detected. However, a subset of Eq. 4.2 will be used in the following, noting that most of the techniques to be presented later in this chapter can be modified to conform to the more general forms of the generalized Radon transform.

Let $\check{g}(\xi)$ denote the continuous generalized Radon transform of the function $g(x, y)$, which here is defined as

$$\check{g}(\xi) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(y - \phi(x; \xi)) dx dy = \int_{-\infty}^{\infty} g(x, \phi(x; \xi)) dx \quad (4.3)$$

where $y = \phi(x; \xi)$ denotes the transformation curve. The generalized Radon transform is the integration of $g(x, y)$ along the transformation curve. A Jacobian-like weighting function in front of the integral can be included if other weighting schemes is required.

Eq. 4.3 can be interpreted as the generalization of the slant stacking technique described in Chapter 1. Note that Eq. 4.3 only allows curves to have one value of y for each value of x and the parameter ξ . This excludes, e.g., closed curves such as circles. In this case Eq. 4.2 can be used.

The curve to be detected is now modelled using a delta function following the curve shape.

$$g(x, y) = \delta(y - \phi(x; \xi^*)) \quad (4.4)$$

hence the generalized Radon transform is given by

$$\check{g}(\xi) = \int_{-\infty}^{\infty} \delta(\phi(x; \xi) - \phi(x; \xi^*)) dx \quad (4.5)$$

$$= \int_{-\infty}^{\infty} \sum_{i=1}^I \frac{\delta(x - x_i)}{\left| \frac{\partial \phi(x; \xi)}{\partial x} - \frac{\partial \phi(x; \xi^*)}{\partial x} \right|} dx \quad (4.6)$$

$$= \sum_{i=1}^I \frac{1}{\left| \frac{\partial \phi(x_i; \xi)}{\partial x} - \frac{\partial \phi(x_i; \xi^*)}{\partial x} \right|} \quad (4.7)$$

where each of the I values x_i , cf. Eq. A.4, fulfills $\phi(x_i; \xi) = \phi(x_i; \xi^*)$.

Eq. 4.7 shows that the generalized Radon transform will give an infinite peak, when $\xi = \xi^*$, and the parameter domain will also contain some other non-zero values, depending on the value of I and the actual function ϕ . The equation also shows that the transform will malfunction when slope of the curve $\partial \phi / \partial x$ is infinite, i.e., when the tangent to the curve in a (x, y) plane is vertical. In conclusion, Eq. 4.3 is well suited for curves like parabolas, hyperbolas, and other parameterized curves with limited slope. Note that a generalization of the technique described in Section 1.7 can be used to expand the applicability of the transform.

4.1.2 The Discrete Generalized Radon Transform

Let \mathbf{j} denote the η -dimensional discrete index parameter vector defined as

$$\mathbf{j} = (j_1, \dots, j_i, \dots, j_\eta) \quad (4.8)$$

The correspondence between the index vector \mathbf{j} and the sampled version of the parameter vector, denoted $\xi_{\mathbf{j}}$ can be written as

$$\xi = \xi_{\mathbf{j}} = \boldsymbol{\theta}(\mathbf{j}), \quad \text{where } \xi_i = \theta_i(j_i) \quad (4.9)$$

where $\theta_i(j_i)$ is a parameter sampling function.

If a uniform sampling of the parameter domain is chosen, the function θ_i can be written

$$\xi_i = \theta_i(j_i) = \xi_{i,min} + j_i \Delta \xi_i, \quad j_i = 0, \dots, J_i - 1 \quad (4.10)$$

where $\xi_{i,min}$ denotes the lower limit and $\Delta \xi_i$ the sampling interval of ξ_i .

The parameter domain and image domain sampling are assumed to be uniform, as defined in Eq. 1.12, i.e.,

$$x = x_m = x_{min} + m \Delta x, \quad m = 0, 1, \dots, M - 1 \quad (4.11)$$

$$y = y_n = y_{min} + n \Delta y, \quad n = 0, 1, \dots, N - 1 \quad (4.12)$$

Substitution of Eqs. 4.9, 4.11 and 4.12 into the transformation curve gives

$$y = y_{min} + n \Delta y = \phi(x_{min} + m \Delta x; \boldsymbol{\theta}(\mathbf{j})) \quad (4.13)$$

Using a nearest neighbour approximation of Eq. 4.13, the discrete index transformation curve $\phi(m; \mathbf{j})$ can be expressed as (note the change of arguments)

$$\phi(m; \mathbf{j}) = n = \left\lceil \frac{\phi(x_{min} + m \Delta x; \boldsymbol{\theta}(\mathbf{j})) - y_{min}}{\Delta y} \right\rceil \quad (4.14)$$

Let $g(m, n)$ denote the discretized version of the signal $g(x, y)$, i.e., $g(m, n) = g(x_m, y_n)$ and let $\check{g}(\mathbf{j})$ denote the discrete generalized Radon transform (GRT) of $g(m, n)$ defined as

$$\check{g}(\mathbf{j}) = \sum_{m=0}^{M-1} g(m, \phi(m; \mathbf{j})) \quad (4.15)$$

Notice, that the term dx has been omitted in the discrete generalized Radon transform as it is a constant due to the uniform sampling of x . If the discrete generalized Radon transform should quantitatively approximate Eq. 4.3, then Eq. 4.15 must be multiplied by Δx .

The use of rounding instead of, e.g., linear interpolation will normally not lead to problems, when using a sufficient sampling of the parameter domain, but interpolations schemes can easily be incorporated in Eqs. 4.14 and 4.15.

4.2 Image Point Mapping

The first part of Eq. 4.3 can be used to define another way of estimating a discrete parameter domain

$$\check{g}(\mathbf{j}) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) \delta(n - \phi(m; \mathbf{j})) \quad (4.16)$$

where $\delta(\cdot)$ denotes the Kronecker delta function.

Eq. 4.16 shows that each image point (m, n) is transformed into a parameter curve where $n = \phi(m; \mathbf{j})$. Thus mapping of image points with zero value is needless, since they will not contribute to $\check{g}(\mathbf{j})$. The proposed estimation scheme for the Radon transform accounts for the fact that zero image values do not contribute, and does not include them in the summation in Eq. 4.16. This has the potential to reduce the computational cost considerably.

Now, assume that ϕ is invertible in one of the parameters ξ_i , e.g., ξ_η . The inverse function $\phi_{\xi_\eta}^{-1}$ with respect to ξ_η can be written as

$$\xi_\eta = \phi_{\xi_\eta}^{-1}(x, y; \boldsymbol{\xi}_r), \quad \boldsymbol{\xi}_r = (\xi_1, \dots, \xi_{\eta-1}) \quad (4.17)$$

Furthermore, assume that the sampling function θ_η is invertible. Then

$$j_\eta = [\theta_\eta^{-1}(\xi_\eta)] = [(\theta_\eta^{-1}(\phi_{\xi_\eta}^{-1}(x, y; \boldsymbol{\xi}_r))] \equiv \Omega(m, n; \mathbf{j}_r), \quad \mathbf{j}_r = (j_1, \dots, j_{\eta-1}) \quad (4.18)$$

Eqs. 4.16 and 4.18, are the basis for an estimation scheme for the generalized Radon transform which here is referred to here as Image Point Mapping (IPM). The key steps of IPM can be summarized as

1. Initialize $\check{g}(\mathbf{j}) = 0$ for all \mathbf{j}
2. For all image points $g(m, n)$ with a value different from zero do
 - For all possible \mathbf{j}_r do

$$\mathbf{j} = (\mathbf{j}_r, \Omega(m, n; \mathbf{j}_r))$$

$$\check{g}(\mathbf{j}) := \check{g}(\mathbf{j}) + g(m, n)$$

Note that IPM is a generalized Hough transform [23], which has been extended to handle a large set of non-linear transformation curves. Like the Hough transform, the objective of IPM is parameter identification, by transforming each pixel in the image into the parameter domain.

One of the fundamental differences between GRT and IPM is, that GRT requires rounding in the image domain, whereas IPM rounds in the parameter domain, as schematically shown in Fig. 4.1.

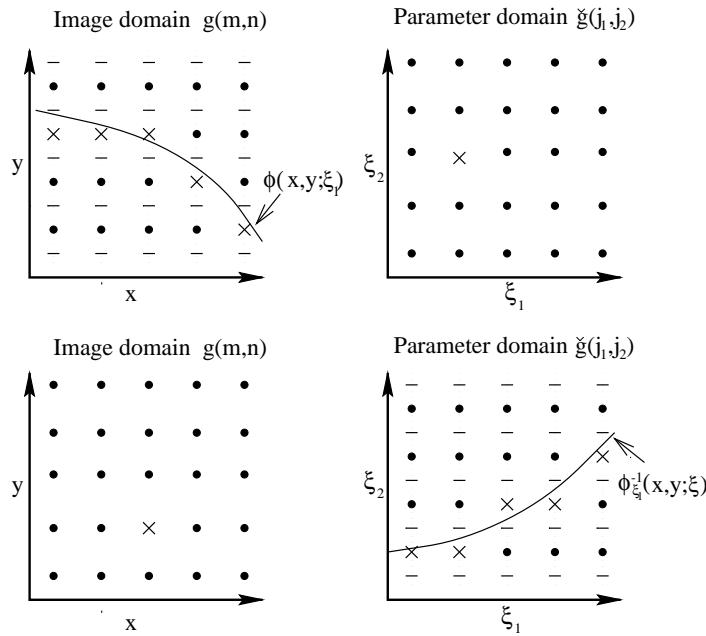


Figure 4.1 Top: Rounding by GRT in the image domain corresponding to a point in the parameter domain. Bottom: Rounding by IPM in the parameter domain corresponding to a point in the image domain.

The computational complexities of GRT and IPM, respectively, are

$$\mathcal{O}_{GRT} = \mathcal{O}\left(M \prod_{i=1}^{\eta} J_i\right) \quad \mathcal{O}_{IPM} = \mathcal{O}\left((MN)_r \prod_{i=1}^{\eta-1} J_i\right) \quad (4.19)$$

where $(MN)_r$ indicates that only a reduced number of image points (non-zero values) are to be transformed. In Eq. 4.19, the cost to test for non-zero values is assumed to be negligible.

The major advantage of IPM over GRT is the ability of IPM to ignore image points with a value of zero, making IPM especially well suited for sparse binary images.

If the absolute value between two successive j_η values is greater than one, i.e.,

$$|\Omega(m, n; \mathbf{j}_r + \boldsymbol{\varepsilon}) - \Omega(m, n; \mathbf{j}_r)| > 1 \quad (4.20)$$

where vector $\boldsymbol{\varepsilon}$ contains zeros at all $\eta - 1$ entries and holds unit value at entry i , i.e., $\varepsilon_i = 1$, it is seen that IPM will map the image point (m, n) into a perforated hyper-curve, i.e., a hyper-curve containing missing sections. Identification of curve parameters can be difficult due to the presence of perforation holes depending on the shape and depth of the holes.

For $\eta = 2$ the perforation problem can be eliminated simply by adding $g(m, n)$ to $\check{g}(\mathbf{j})$ for the parameter values skipped between two successive parameter vectors. Consider, for example,

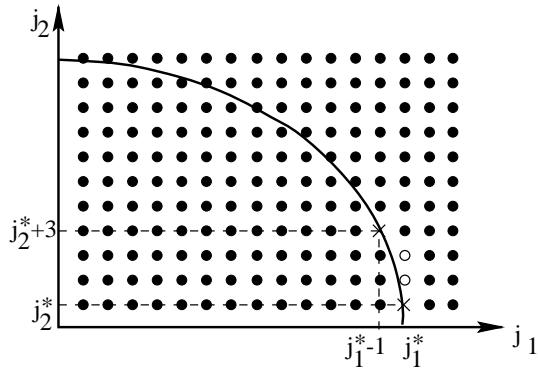


Figure 4.2 Illustration of the perforation problem.

the hyperbolic transformation curve as illustrated in Fig. 4.2, where the parameter domain is described by $(j_1, j_2) = (j_1^*, j_2^*)$.

Fig. 4.2 shows the mapping of an image point (m^*, n^*) to the parameter domain, that is, discretized parameter point accumulators to which the value $d(m^*, n^*)$ of the image point must be added. To investigate the perforation problem, two parameter points $(s^* - 1, z^* + 3)$ and (s^*, z^*) on the inverse transformation curve are considered. Following the inverse transformation curve from $(s^* - 1, z^* + 3)$ to (s^*, z^*) along the s axis gives only the two parameter points $(s^* - 1, z^* + 3)$ and (s^*, z^*) . However, following the inverse transformation curve along the z axis also gives the two intervening points $(s^*, z^* + 2)$ and $(s^*, z^* + 1)$. One way of handling such intervening points is to add the value of the image point to the parameter points skipped between two successive parameter points on the inverse transformation curve. Therefore, in this example, $d(m^*, n^*)$ must be added to the skipped parameter points $(s^*, z^* + 2)$ and $(s^*, z^* + 1)$ when following the s axis.

4.3 Parameter Domain Sampling

For a given image $g(m, n)$ an exact determination of $\Delta\xi_i$, $\xi_{i,min}$ and J_i , which match the image, is, in general, not possible. However, some guidelines can be stated. First, some of the parameters will be bounded by the underlying physics, e.g., $\xi_{i,min}$ and $\xi_{i,max}$ will normally be limited. Second, the parameter domain can be limited by requiring that at least a fraction β of the image points addressed by the transformation curve $\phi(m; \boldsymbol{\xi})$ lie inside the image, i.e.,

$$\sum_{m=0}^{M-1} I\{y_{min} < \phi(x_m; \boldsymbol{\xi}) < y_{max}\} > \beta M, \quad 0 < \beta < 1, \quad \forall \boldsymbol{\xi} \quad (4.21)$$

where $I\{\cdot\}$ equals 1 when the logical expression is true and 0 otherwise.

Concerning use of the GRT, the sampling intervals $\Delta\xi_i$ can be chosen by requiring that

$$\max\{|\phi(x; \boldsymbol{\xi} + \boldsymbol{\gamma}_i) - \phi(x; \boldsymbol{\xi})|\} = \Delta y, \quad \forall x, \boldsymbol{\xi}, \boldsymbol{\gamma}_i \quad (4.22)$$

where $\boldsymbol{\gamma}_i$ is a η -dimensional vector containing zeros at all entries except entry i which is $\Delta\xi_i$.

Eq. 4.22 states that two adjacent $\boldsymbol{\xi}$ -vectors give $y = \phi(x; \boldsymbol{\xi})$ values which cannot be separated by more than one sample in the y -direction. This design criterion is not optimal as it leads to an unnecessarily dense sampling of certain parts of the parameter domain, however, it can be used as an upper sampling rate limit. Increasing the sampling rate of the parameter domain above a certain limit will not improve the resolution obtained by GRT as adjacent \boldsymbol{j} vectors will result

in the same curve $n = \phi(m; \mathbf{j})$. For GRT, a coarse sampling cannot guarantee a given curve will be detected, due to the fact that no transformation curve $\phi(m; \mathbf{j})$ can be guaranteed to follow the image curve perfectly.

Use of IPM with a dense parameter domain sampling according to Eq. 4.22 produces curves $j_\eta = \Omega_d(m, n; \mathbf{j}_r)$ for (m, n) values corresponding to an image curve which do not intersect in a parameter point \mathbf{j} , but will spread out over several parameter points. The reason for this spread is the individual treatment of image points as regions of zero size and not regions of, e.g., rectangular shape. This spread can be compensated by the use of a coarse sampling of the parameter domain. In addition, a coarse sampling will lead to a lower computational cost.

Summarizing, GRT requires a dense parameter domain sampling, while IPM gives rise to blurring in the parameter domain in the case of dense parameter domain sampling, but works well with a coarse sampling of the parameter domain. This observation is the basis for the curve parameter estimation algorithm, presented in Section 4.5.

4.4 Parameter Domain Blurring

Parameter domain blurring achieved by use of GRT can be interpreted in a way that makes it usable for parameter clustering. According to Eq. 4.3 the continuous transformation curve can be written as

$$y = \phi(x; \xi), \quad x \in [x_{\min}, x_{\max}] \quad (4.23)$$

Let Δy denote the sampling interval of y and let $\alpha \Delta y$ be a given uncertainty of y , e.g., $\alpha = 0.5$. For a parameter vector $\xi = \theta(\mathbf{j})$ to lie inside a band of width $2\alpha \Delta y$, symmetrically positioned around the image curve $y = \phi(x; \xi^*)$, it must satisfy the following inequality

$$y - \alpha \Delta y < \phi(x; \xi) < y + \alpha \Delta y, \quad y = \phi(x; \xi^*), \quad \forall x \in [x_{\min}, x_{\max}] \quad (4.24)$$

Several parameter vectors ξ will correspond to image points within the band specified in Eq. 4.24. Therefore, in general, a band in the image around a given curve described by the parameter vector ξ^* will give rise to a region in the parameter space which consists of parameter vectors that correspond to curves inside the image band. These regions are named clusters and the corresponding image curve with parameter ξ^* is called a center curve. An example of a curve band around a center curve and the corresponding parameter cluster is illustrated in Fig. 4.3.

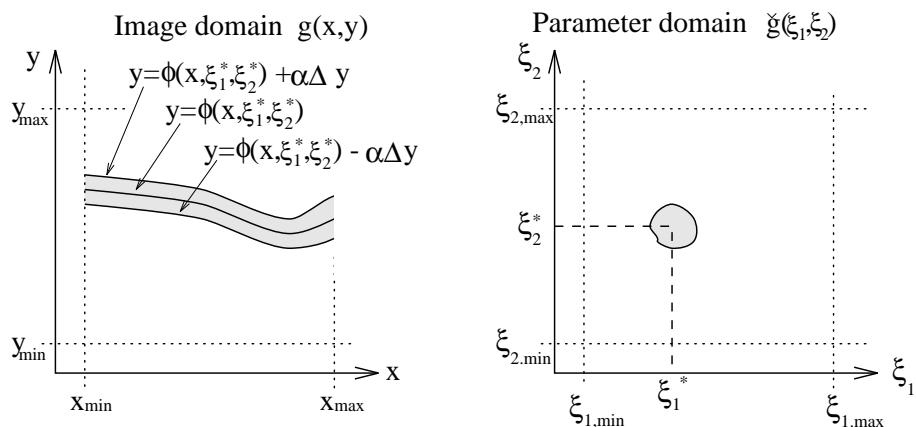


Figure 4.3 Left: Center curve with an uncertainty of $2\alpha \Delta y$ indicated. Right: Cluster corresponding to center curve.

Unfortunately, the center curves are normally unknown, and it is impossible to determine the clusters. However, it is possible to determine whether two parameter sets ξ_1 and ξ_2 belong to the same cluster. If two parameter vectors belong to the same cluster they must satisfy

$$|\phi(x; \xi_1) - \phi(x; \xi_2)| < 2 \alpha \Delta y, \quad \forall x \in [x_{min}, x_{max}] \quad (4.25)$$

Parameter domain points may be gathered into regions or clusters, with the guarantee that all possible image curves will be represented by only one cluster in the parameter domain. This partitions the parameter domain into irregular regions, which reflect the information level of the image. Clustering can also be used to estimate parameter uncertainties, e.g., the maximum and minimum parameter value within the cluster can be used to give an estimate of the parameter uncertainty as, e.g., $\frac{1}{2}(\xi_{i,max}^{\text{cluster}} - \xi_{i,min}^{\text{cluster}} + \Delta\xi_i)$. Parameter domain clustering can be performed either before or after the transform, depending on the purpose.

4.5 The Fast Curve Estimation Algorithm

Consider an image where the image values are represented by continuous values. This image can be mapped into a binary image $g(m, n)$, e.g., by edge filtering [17, 49], by deconvolution [50, 41], or by mean field annealing [42, 51]. The proposed algorithm estimates the parameters of curves in the binary image $g(m, n)$, e.g., lines or hyperbolas.

It is well-known that direct use of the GRT is computationally expensive. In the light of the characteristics of GRT and IPM, a fast curve estimation algorithm, the FCE-algorithm, is proposed which simultaneously estimates all parameters of curves having a specific shape, e.g., lines or hyperbolas. The FCE-algorithm uses IPM as a pre-conditioning procedure for GRT by selecting the regions of interest in the parameter domain. IPM is suitable for a rapid determination of regions of interest, as it works well on a coarsely sampled parameter domain and is capable of ignoring image points with a value of zero. Another important circumstance is that GRT relates the uncertainty on the estimated curve parameters to the image domain sampling, while IPM relates it to the parameter domain sampling. The FCE-algorithm is shown in Fig. 4.4, and is summarized as follows:

1. Design the discrete parameter domain, i.e., choose $\xi_{i,min}$, J_i and $\Delta\xi_i$.
2. Design a reduced parameter domain for IPM by choosing

$$J'_i = \left\lceil \frac{J_i}{2v_{\xi_i} + 1} \right\rceil, \quad \xi'_{i,min} = \xi_{min} + v_{\xi_i} \Delta\xi_i, \quad \Delta\xi'_i = (2v_{\xi_i} + 1) \Delta\xi_i \quad (4.26)$$

where $\lceil \cdot \rceil$ rounds to the nearest upper integer, and v_{ξ_i} is an integer related to the resampling (see below). Then use IPM to estimate $\check{g}_{ipm}(\mathbf{j}')$ as

$$\check{g}_{ipm}(\mathbf{j}') = \text{IPM}\{g(m, n)\} \quad (4.27)$$

3. Use the threshold function T to give

$$\check{g}_t(\mathbf{j}') = T(\check{g}_{ipm}(\mathbf{j}')) \quad (4.28)$$

The threshold function T is designed to remove all insignificant parameter combinations and a suitable choice might be

$$T(\check{g}_{ipm}(\mathbf{j}')) = \mu(\check{g}_{ipm}(\mathbf{j}')) - \lambda_1 M \quad (4.29)$$

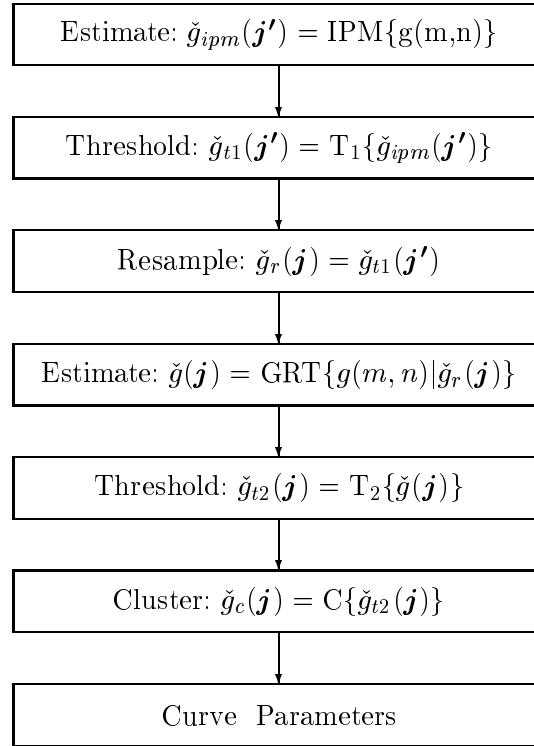


Figure 4.4 Flow diagram of the FCE-algorithm.

where M denotes the number of image lines, λ_1 a fraction defining the significance level, and $\mu(\cdot)$ the Hamilton step function. Having an image with one curve, IPM gives a corresponding parameter domain value of M . Choosing a threshold level of $\lambda_1 M$ allows some deviation for the image curve from the integration curve, e.g., to handle holes.

4. Resample the parameter domain to obtain $\check{g}_r(\mathbf{j})$. The resampling process can be written as

$$\check{g}_r(\mathbf{j}) = \check{g}_t(\mathbf{j}'), \quad j'_i = \left[\frac{j_i - v_{\xi_i}}{2v_{\xi_i} + 1} \right] \quad (4.30)$$

This quickly fills the entire parameter domain where the interesting regions have value one while other regions have value zero.

5. Use GRT to estimate $\check{g}(\mathbf{j})$ in the regions of the parameter domain where $\check{g}_r(\mathbf{j})$ is not equal to zero.

$$\check{g}(\mathbf{j}) = \text{GRT}\{g(m,n) | \check{g}_r(\mathbf{j}) \neq 0\} \quad (4.31)$$

6. As in step three, use the threshold function with a new significance level λ_2 to account for use of the GRT. After the thresholding, all points in the parameter domain having a value different from zero represent curve parameters.

7. Finally, the identified parameters are clustered into common image curves.

The resampling process takes each parameter point corresponding to the sampling interval $\Delta\xi'$ and extends it to several parameter points corresponding to the sampling interval $\Delta\xi$, $\Delta\xi_i < \Delta\xi'_i$. Thus, the FCE-algorithm operates initially in a coarsely sampled parameter domain, using IPM for determination of regions of interest. Subsequently, the sampling is refined to the required level and GRT is applied within the regions of interest.

The significance levels λ_1 and λ_2 must be chosen to reflect the parameter domain values, that can be accepted as peaks corresponding to curves in the image. In general, the precise values of λ_1 and λ_2 are not critical. For the examples, suitable ranges of values for λ_1 and λ_2 are 0.1 – 0.7 and 0.4 – 0.8, respectively. There are two reasons for the use of a lower acceptance level in step three. First, IPM has a tendency to blur in the parameter domain, and second, only regions of the parameter domain ensured not to contain image curves must be removed. A higher significance level in step six (λ_2) is justified by the required certainty for correct curve identification.

4.6 The Hyperbolic Transformation Curve

In marine seismic acquisition [39, 52, 19] it is common to use a sound emitter, e.g., an air gun attached to a vessel, which is towing an array of hydrophones (receivers). The setup is shown schematically in Fig. 4.5.

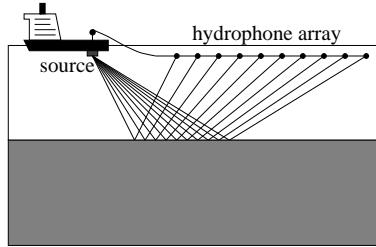


Figure 4.5 A vessel towing an array of hydrophones. A sound source emits a pulse, which is reflected at a layer-boundary.

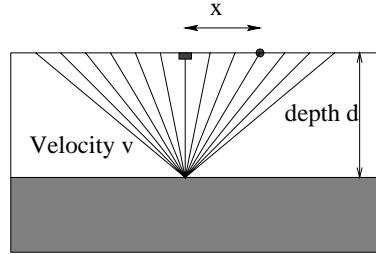


Figure 4.6 Rearranging data corresponding to a common mid point gives a CMP-gather.

As the vessel sails along a line an acquisition is made. With a certain time interval the air gun emits a pulse-like signal, which penetrates the water and top layers of the ground. A simple model is to use assume that the ground is made of homogeneous layers in which the pulse propagates linearly until the pulse is reflected at a layer-boundary and finally is recorded by the hydrophones lying near the water surface. Each of the M hydrophones record a couple of seconds of a signal after the sound emission. The signal is sampled and stored digitally. The digital signal recorded at a single hydrophone is called a trace.

Given that acquisition data is available for a large set of vessel positions relative to the earth, all the measured traces can be rearranged as if the hydrophones had a common mid point [39]. This collection of data is shown in Fig. 4.6 and is known as a CMP-gather [38, 42].

Assume that the layer between the water-level and the reflection boundary has velocity v and depth d . Then it is easy to find the time between the emission of the pulse until it is recorded at the hydrophone lying at the offset x . Using the Pythagorean theorem gives that the two way travel time t follows a hyperbolic curve as a function of x . This is called the normal moveout equation.

$$t^2 = t_0^2 + (\sigma x)^2 \text{ where } \begin{cases} \sigma = \frac{1}{v} \\ t_0 = 2d\sigma \end{cases} \quad (4.32)$$

where σ is known as the slowness, and it should be noted that the zero offset two way travel time t_0 is the value of t , when the offset x is zero.

By assuming a multi-layered earth model, it can be shown that Eq. 4.32 can model the reflection data [53], but the two parameters now depend on several layer parameters corresponding to the actual path of the wave, e.g., [54, 55, 42]. A multi-layered subsurface will generally produce several reflection hyperbolas and in the following the FCE-algorithm is used to detect the parameters of these hyperbolas. It could be mentioned that the parameters afterwards could be used in algorithms trying to reconstruct the individual layer parameters (velocity and thickness).

A digital image $f(m, n)$, containing the measurements, is formed by the finite number of different offset-values normally with a fixed distance between the hydrophones, i.e., the offset x is sampled linearly

$$x = x_m = x_{min} + m\Delta x \quad (4.33)$$

and the index n correspond to the time t , which also is sampled linearly

$$t = t_n = t_{min} + n\Delta t \quad (4.34)$$

Consider a seismic CMP-gather. The multipulse technique or mean field annealing can be used to map the seismic signals in the CMP-gather $f(m, n)$ into a binary image $g(m, n)$, where $g(m, n)$ is set to one if position n of trace number m contains a reflection, and to zero otherwise.

Hence, in this seismic case, the image parameter y is replaced by the time t , the parameter domain are spanned by σ and t_0 , and the curve to detect is given by

$$t = \phi(x; \sigma, t_0) = \sqrt{t_0^2 + (\sigma x)^2} \quad (4.35)$$

which easily can be inverted with respect to t_0 to give

$$t_0 = \phi_{t_0}^{-1}(x, t; \sigma) = \sqrt{t^2 - (\sigma x)^2} \quad (4.36)$$

Both t_0 and σ are uniformly sampled and the parameter domain sampling is chosen in agreement with Eq. 4.22. The discretized parameter domain is denoted z and s corresponding to t_0 and σ , respectively,

$$t_0 = t_{0,low} + z \Delta t_0, \quad z = 0, \dots, Z - 1 \quad (4.37)$$

$$\sigma = \sigma_{low} + s \Delta \sigma, \quad s = 0, \dots, S - 1 \quad (4.38)$$

The curve perforation described in section 4.2 occurs in the present case of a hyperbolic transformation curve. It is, however, eliminated by adding $g(m, n)$ to $\check{g}(s, z)$ for the parameter values skipped between two successive parameter vectors.

4.6.1 Clusters in the Hyperbolic case

Assume the parameter set (σ^*, t_0^*) corresponds to a center curve. The parameter sets (σ, t_0) corresponding to image curves within the band of width $2\alpha\Delta t$, symmetrically positioned around the image curve $t = \phi(x; \sigma^*, t_0^*)$, can according to Eq. 4.24 be written as

$$t - \alpha \Delta t < \sqrt{t_0^2 + (\sigma x)^2} < t + \alpha \Delta t, \quad t = \sqrt{t_0^{*2} + (\sigma^* x)^2}, \quad \forall x \in [x_{min}, x_{max}] \quad (4.39)$$

Let q denote a binary variable, $q \in \{-1, +1\}$. Thus, the lowest and highest curve within the image band specified by Eq. 4.39 can be written as

$$\sqrt{t_0^{*2} + (\sigma^* x)^2} + q \alpha \Delta t = \sqrt{t_0^2 + (\sigma x)^2} \Leftrightarrow t_0^2 = \left(\sqrt{t_0^{*2} + (\sigma^* x)^2} + q \alpha \Delta t \right)^2 - (\sigma x)^2 \quad (4.40)$$

In [42] it is shown, that x_{min} and x_{max} give the cluster limiting elliptical curves, the parameter cluster corresponding to the center curve (σ^*, t_0^*) can be written as

$$\sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{min})^2} - \alpha \Delta t\right)^2 - (\sigma x_{min})^2} < t_0 < \sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{min})^2} + \alpha \Delta t\right)^2 - (\sigma x_{min})^2} \quad (4.41)$$

and

$$\sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{max})^2} - \alpha \Delta t\right)^2 - (\sigma x_{max})^2} < t_0 < \sqrt{\left(\sqrt{t_0^{*2} + (\sigma^* x_{max})^2} + \alpha \Delta t\right)^2 - (\sigma x_{max})^2} \quad (4.42)$$

The left part of Fig. 4.7 shows a hyperbolic center curve (σ^*, t_0^*) with an uncertainty of $2\alpha\Delta t$. The right part of the figure shows the shape of the corresponding cluster, which is bounded by four elliptical curves.

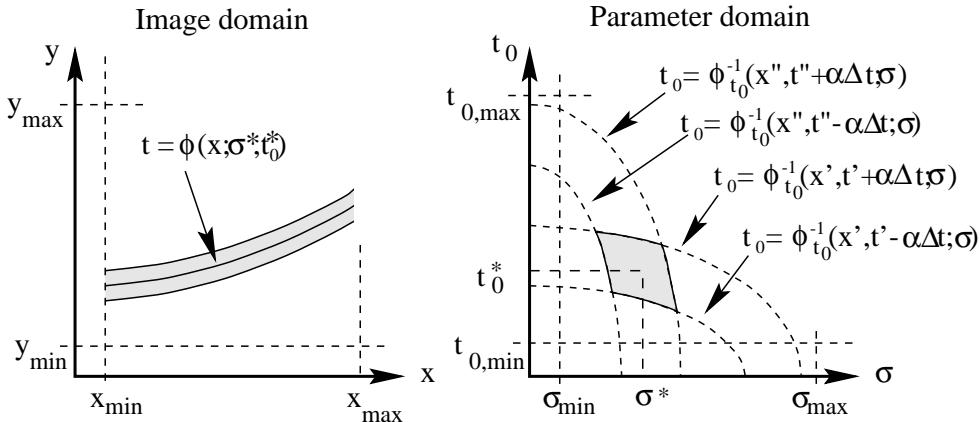


Figure 4.7 Left: A hyperbolic center curve (σ^*, t_0^*) with an uncertainty of $2\alpha\Delta t$. Right: The shape of the corresponding cluster, which is bounded by four elliptical curves, where $x' = x_{min}$, $x'' = x_{max}$, $(t')^2 = t_0^{*2} + (\sigma^* x_{min})^2$, and $(t'')^2 = t_0^{*2} + (\sigma^* x_{max})^2$.

Fig. 4.8 illustrates four clusters in the parameter domain. The four clusters correspond to four center curves with an uncertainty of $10\Delta t$, i.e., $\alpha = 5$. It should be noted that the cluster shape is highly dependent on the center curve (σ^*, t_0^*) .

4.6.2 An Example with Eight Hyperbolas

In this example the potential of the FCE-algorithm is illustrated. A synthetic image is composed of eight hyperbolas, assembled into four groups of two hyperbolas each, as shown in Fig. 4.9. In order to get a binary gather, where the pixels are set to one, when a reflection is found the gather has been deconvolved, by finding the maximum cross-correlation between the gather and the shape of the reflection wavelet a number of times. The result is shown in Fig. 4.10. Here only 2.5% of the pixels has been set to 1. It can be seen that several pixels have erroneously been identified as reflections.

Using GRT according to the settings in Table 4.1 leads to eight peaks in the parameter domain as shown in Fig. 4.11. Thus, although the eight curves intersect and lie very close, GRT is able to separate the curves in the parameter domain into eight separate peaks. The figure can be compared to Fig. 4.12 showing the actual hyperbola parameters.

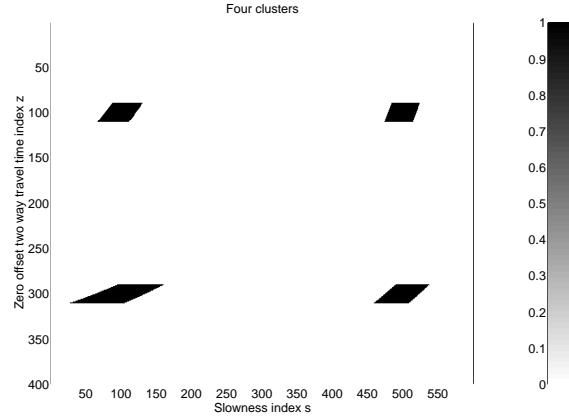


Figure 4.8 Four clusters in the parameter domain corresponding to an uncertainty of $10\Delta t$, i.e., $\alpha = 5$.

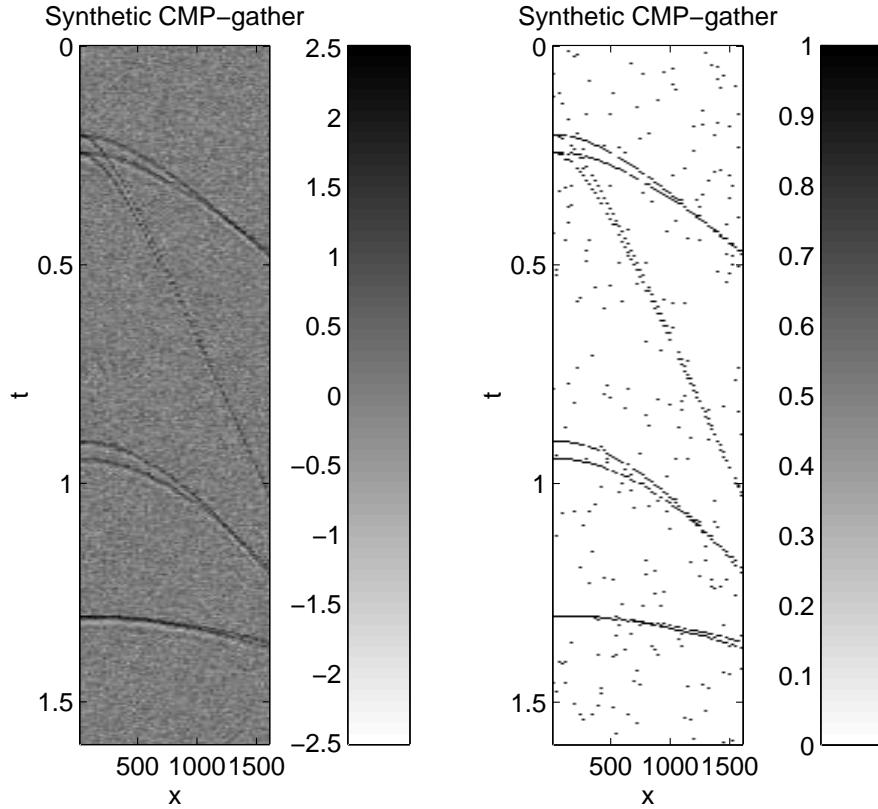


Figure 4.9 Synthetic CMP-gather composed of eight hyperbolae.

Figure 4.10 The binary deconvolved CMP-gather.

The FCE-algorithm is used for fast detection of the interesting regions in the parameter domain resulting from GRT. The following figures show the steps of the FCE-algorithm. Fig. 4.13 shows the coarsely sampled parameter domain obtained using IPM with $v_\sigma = 4$ and $v_{t_0} = 2$.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	50	S	300
N	400	Z	400
Δx	50 m	$\Delta\sigma$	$1.569 \cdot 10^{-6} \text{ s/m}$
Δt	$4 \cdot 10^{-3} \text{ s}$	Δt_0	$4 \cdot 10^{-3} \text{ s}$
x_{min}	100 m	σ_{min}	$2.3 \cdot 10^{-4} \text{ s/m}$
t_{min}	0 s	$t_{0,min}$	0 s
α	1.5	λ_1	0.5
		λ_2	0.75
		v_σ	4
		v_{t_0}	2

Table 4.1 Image and parameter domain settings.

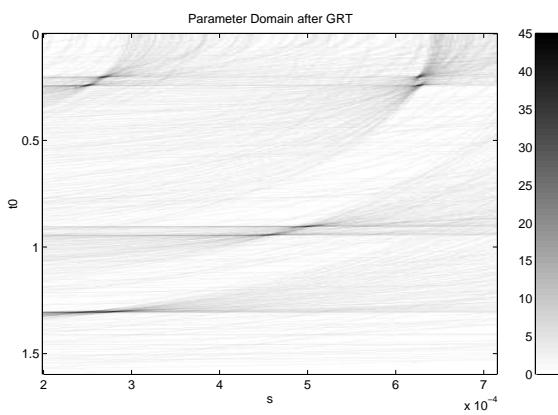


Figure 4.11 Full GRT estimated parameter domain.

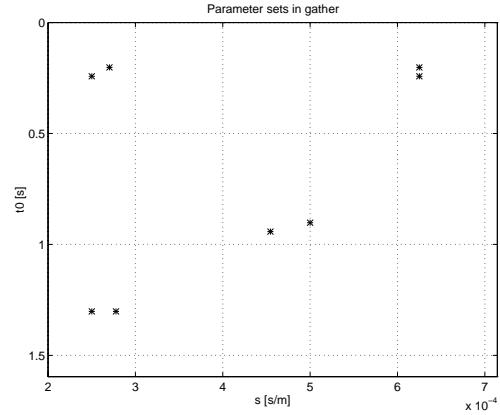


Figure 4.12 Parameters of the hyperbolae shown in the parameter domain.

The parameter domain is then thresholded using a significance level $\lambda_1 = 0.5$, and the resulting binary parameter domain is shown in Fig. 4.14.

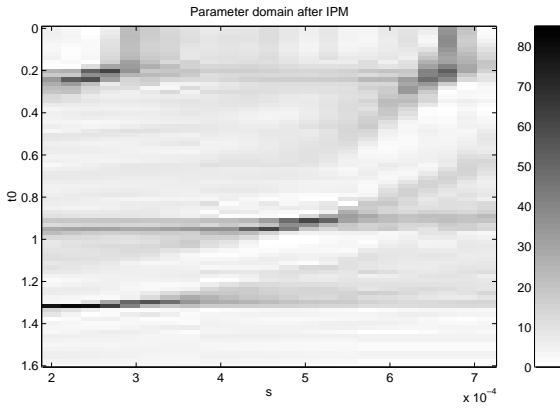


Figure 4.13 Parameter domain estimated by use of IPM.

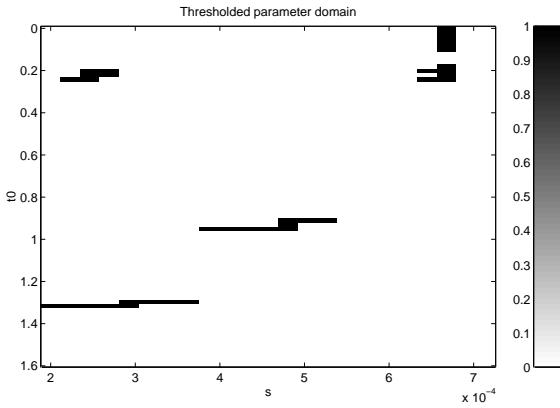


Figure 4.14 The IPM estimated parameter domain after thresholding using $\lambda_1 = 0.5$.

The binary parameter domain is then resampled to a full size parameter domain, which looks exactly like Fig. 4.14, but now the parameter domain has 27 times the number of samples used for IPM.

Comparing this pre-condition map with Fig. 4.11, shows that all regions containing curves are contained within the pre-condition map. Next, Fig. 4.15 shows the parameter domain obtained by using GRT within the regions specified by the pre-condition map. Finally, the pre-conditioned GRT parameter domain is thresholded using a threshold level $\lambda_2 = 0.75$, and the result is shown in Fig. 4.16.

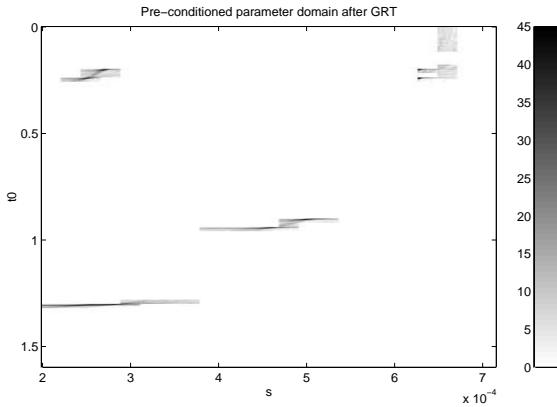


Figure 4.15 Parameter domain obtained using GRT within the regions specified by the resampled parameter domain.

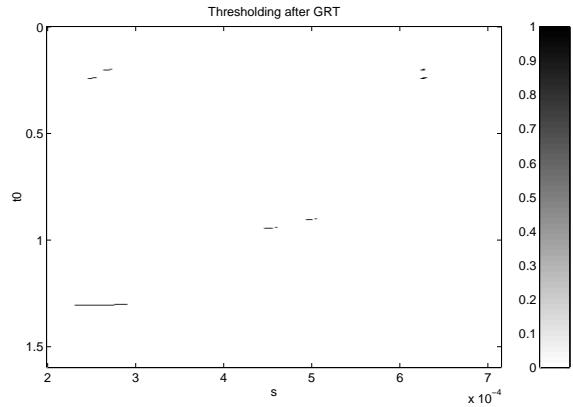


Figure 4.16 The preconditioned GRT estimated parameter domain after thresholding using $\lambda_2 = 0.75$.

The estimated parameter vectors are given in Table 4.2, where the clustering process has used a value of $\alpha = 1.5$. As seen from Table 4.2, eight groups of curve parameters are found, and the estimated curve parameters are rather close to the true parameters.

Curve Number	True parameter	Estimated parameter
1	$(0.202 s, 2.7027 10^{-4} s/m)$	$(0.204 s, 2.6832 10^{-4} s/m)$
2	$(0.202 s, 6.2500 10^{-4} s/m)$	$(0.201 s, 6.2567 10^{-4} s/m)$
3	$(0.242 s, 2.5000 10^{-4} s/m)$	$(0.242 s, 2.5517 10^{-4} s/m)$
4	$(0.242 s, 6.2500 10^{-4} s/m)$	$(0.242 s, 6.248 10^{-4} s/m)$
5	$(0.902 s, 5.0000 10^{-4} s/m)$	$(0.902 s, 4.994 10^{-4} s/m)$
6	$(0.942 s, 4.5455 10^{-4} s/m)$	$(0.943 s, 4.530 10^{-4} s/m)$
7	$(1.302 s, 2.5000 10^{-4} s/m)$	$(1.303 s, 2.571 10^{-4} s/m)$
8	$(1.302 s, 2.7778 10^{-4} s/m)$	$(1.304 s, 2.803 10^{-4} s/m)$

Table 4.2 The results of the FCE-algorithm along with the true curve parameters. The width of the cluster band is set to 3, i.e. $\alpha = 1.5$. The parameters are estimated by the center of the cluster.

The FCE-algorithm has been implemented in MATLAB and C (the loop oriented functions are made in C), and using a 120 MHz Pentium, as previously, result in the time measurements shown in Table 4.3. The table shows the individual parts of the FCE-algorithm and the total cost. It can be seen that the FCE-algorithm here outperforms the use of GRT with a factor of 10, and all eight hyperbola parameters have been estimated without any additional errors.

IPM	0.04 sec
Threshold 1	0.01 sec
Resampling	0.24 sec
Pre-Conditioned GRT	0.14 sec
Threshold 2	0.05 sec
Clustering	0.16 sec
The FCE-algorithm	0.64 sec
Full GRT estimation	6.35 sec

Table 4.3 Time measurements on a 120 MHz Pentium.

Next a small example of the sensitivity of the FCE algorithm with respect to the choice of v_σ and v_{t_0} is given. A binary image containing the same eight hyperbolas as above was created. No noise is present in the image. Fig. 4.17 shows the time used by the FCE-algorithm to estimate the hyperbola parameter as a function of v_σ and v_{t_0} , and in Fig. 4.18 is shown the number of hyperbolas identified by the FCE-algorithm. From the figures it can be seen that an optimum with respect to time is found at $v_{t_0} = 0$ and $v_\sigma = 2$, but for $v_{t_0} = 0$ Fig. 4.18 shows that one or two of the hyperbolas can be missed. This is due to the fact that IPM requires that the parameter domain is coarsely sampled especially in the t_0 -parameter, cf. Eq. 4.36.

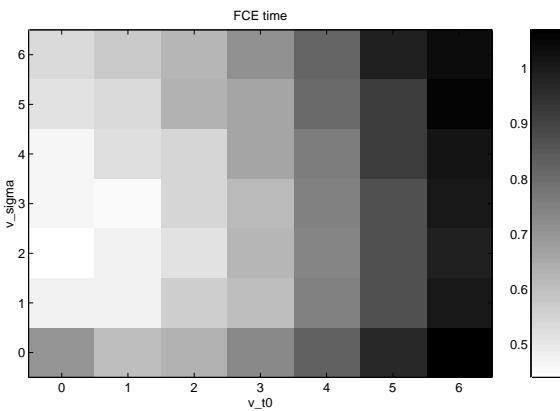


Figure 4.17 The time used by the FCE-algorithm to estimate the parameters of the hyperbolas shown as a function of v_σ and v_{t_0} . For the time measurements a 120 MHz Pentium has been used.

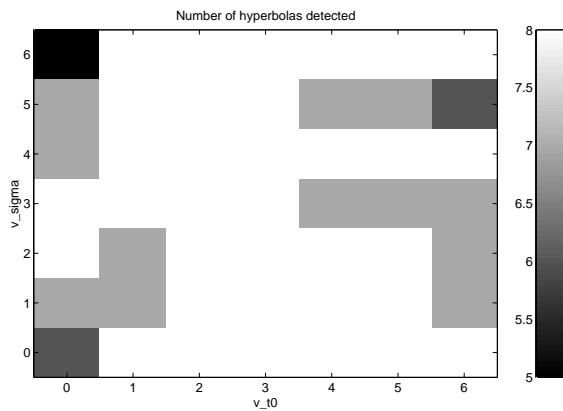


Figure 4.18 The number of hyperbolas detected by the FCE-algorithm shown as a function of v_σ and v_{t_0} .

4.6.3 An Example with a Noise Corrupted Synthetic CMP-gather

To demonstrate the performance of the FCE-algorithm with many curves, an example based on a noise corrupted synthetic CMP-gather is given. The subsurface model is a pure acoustic (compressional waves only) horizontally layered subsurface consisting of four finite layers and infinite top and bottom layers. The synthetic CMP-gather is produced by use of ray tracing [38]. The resulting CMP-gather is composed of 24 distinct reflection curves, shown in Fig. 4.19, where each vertical line represents a trace. The binary signal has been generated by adding the hyperbolas and noise is here simulated by switching the values of the individual binary pixels with a probability of 5%. The parameters of the hyperbolas are shown in Fig. 4.22 and the parameters for both image and parameter domains are given in Table 4.4.

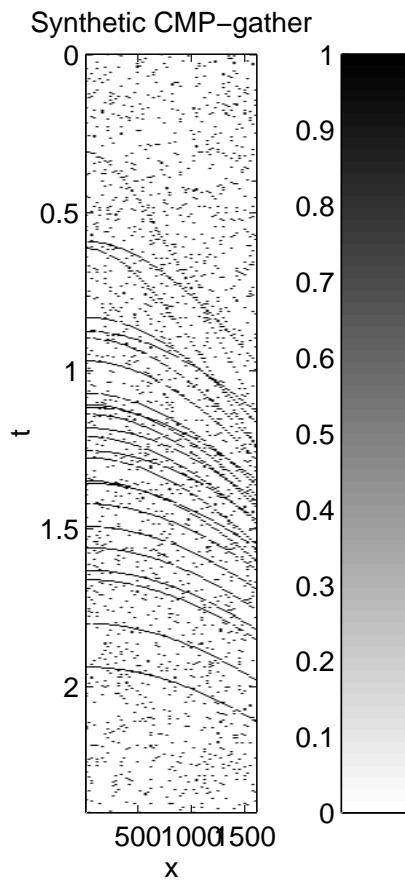


Figure 4.19 Synthetic CMP-gather composed of 24 distinct reflection curves.

Image domain		Parameter domain	
Parameter	Value	Parameter	Value
M	64	S	153
N	600	Z	600
Δx	25 m	$\Delta \sigma$	$2.5 \cdot 10^{-6} \text{ s/m}$
Δt	$4 \cdot 10^{-3} \text{ s}$	Δt_0	$4 \cdot 10^{-3} \text{ s}$
x_{min}	25 m	σ_{min}	$3.33 \cdot 10^{-4} \text{ s/m}$
t_{min}	0 s	$t_{0,min}$	0 s
α	1.5	λ_1	0.5
		λ_2	0.5
		v_σ	4
		v_{t_0}	2

Table 4.4 Image and parameter domain settings.

Applying the FCE-algorithm to the binary image gives the IPM estimated parameter domain shown in Fig. 4.20, using $v_{t_0} = 2$ and $v_\sigma = 4$. Applying GRT on the parameter domain region specified by the resampled and thresholded IPM estimated parameter domain, with $\lambda_1 = 0.5$, results in the GRT parameter domain shown in Fig. 4.21.

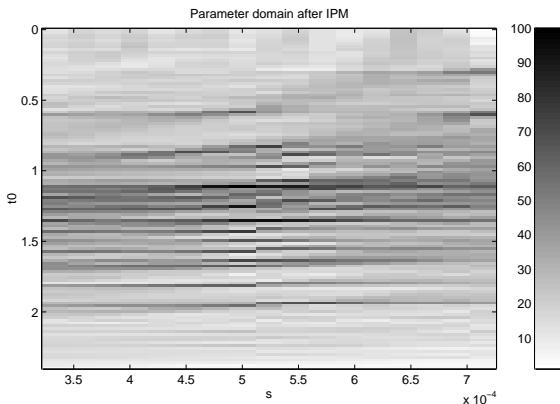


Figure 4.20 IPM estimated parameter domain.

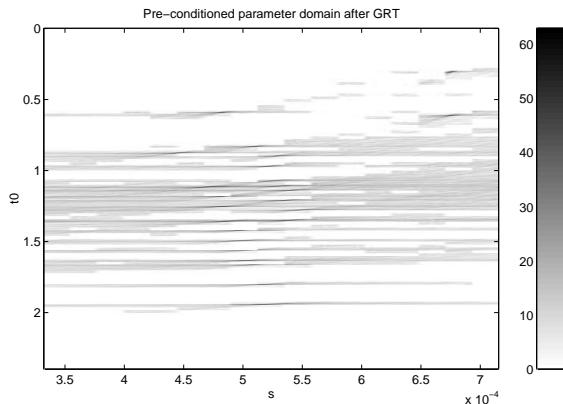


Figure 4.21 The preconditioned GRT.

Finally, Fig. 4.23 shows the estimated curve parameters, where a significance level of $\lambda_2 = 0.5$ in the final threshold process and $\alpha = 1.5$ in the clustering process have been used. In this case hyperbolas have been identified matching the original parameters very well, but two additional hyperbolas have also been identified. This is partly due to the noise level in the initial image, and that the given sampling of the parameter domain is a tradeoff between the certainty of detection and limiting the number of samples in the parameter domain. In this case the FCE-algorithm is approximately 2.6 faster than the GRT. The decrease in speed, compared to the previous example, is due to the higher number of hyperbolas in this example, covering a larger part of the discrete parameter domain.

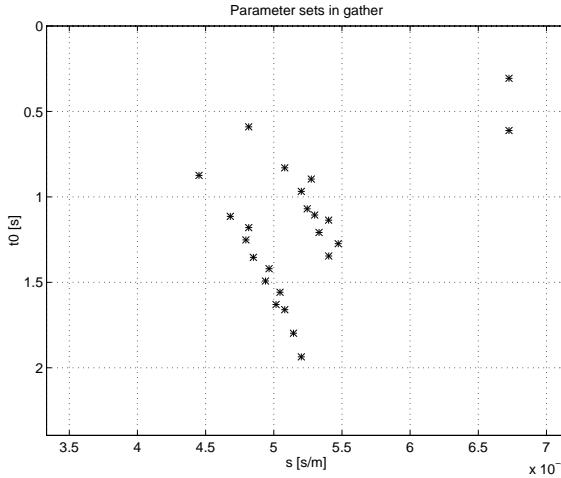


Figure 4.22 The parameters of hyperbolas shown in the parameter domain.

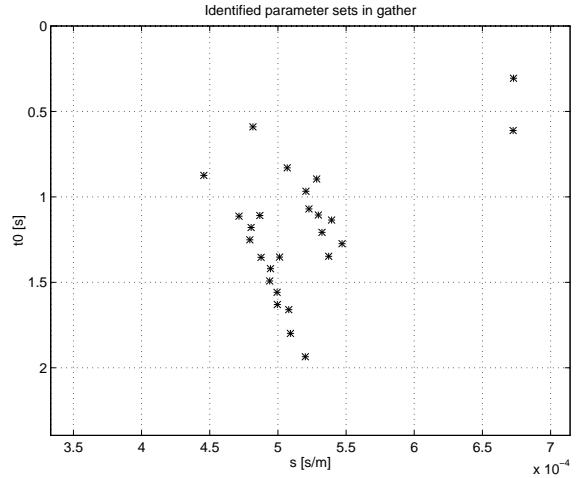


Figure 4.23 The estimated parameters of hyperbolas shown in the parameter domain.

IPM	0.14 sec
Threshold 1	0.01 sec
Resampling	0.24 sec
Pre-Conditioned GRT	2.08 sec
Threshold 2	0.07 sec
Clustering	0.17 sec
The FCE-algorithm	2.71 sec
Full GRT estimation	7.10 sec

Table 4.5 Time measurements on a 120 MHz Pentium.

4.7 Summary

A new algorithm for fast curve parameter estimation, named the FCE-algorithm, has been presented. The algorithm identifies curve parameters by operating on a binary image, obtained, e.g., by edge filtering, deconvolution, or mean field annealing.

The fundamental idea of the algorithm is the use of pre-conditioning to reduce the computational cost of the traditional generalized Radon transform. The pre-conditioning map determines regions of the parameter domain which contain peaks, and the generalized Radon transform is applied only in these regions. As the size of the regions is less than the full parameter domain, the pre-conditioning map reduces the computational costs when applying the generalized Radon transform. For fast generation of the pre-conditioning map, a generalization of the Hough transform named *image point mapping* has been developed. Image point mapping is computationally efficient by taking account of image points with value zero. The required parameter domain sampling and the resulting parameter domain blurring have been investigated.

The FCE algorithm was successfully applied to the identification of hyperbolas in seismic images and two numerical examples have been presented. One example demonstrates the potential of the algorithm for fast and accurate parameter estimation, and the other example illustrates the robustness of the algorithm with respect to noise.

Chapter 5

Curve Parameter Estimation in Noisy Images

So far it has been assumed, that the lines in the images are straight within the (digital) resolution. A very reasonable question regards the situation where lines have wiggles. In Section 5.1 it is analyzed, whether the discrete linear Radon transform still be used to detect line parameters [4]. Analytical expressions are given to quantify the theory.

In Section 5.2 it is shown that lines with wiggles can be incorporated in the Radon transform, but it is also shown that the alteration can be transformed back to the original image as a simple non-linear filter.

Finally in Section 5.3 the implications of additive noise in the image, is analyzed with respect to curve parameter detection [3]. Here a very general form of the generalized Radon transform is assumed, and the theory is illustrated in the linear Radon transform case. Simple formulas are derived, quantifying whether a curve in a noisy image can be detected or not.

5.1 Lines with Wiggles

It is assumed that the digital line in the image can be modelled by

$$g(m, n) = \delta(n - [\alpha^* m + \beta^* + \lambda]) \text{ where } \lambda \in \mathcal{N}(0, \sigma^2) \quad (5.1)$$

where $\delta(\cdot)$ is the Kronecker delta function, and $[\cdot]$ rounds to the nearest integer. The Gaussian distributed noise term λ determines the change in position of the line in the y -direction, and it will be assumed that the noise terms λ are uncorrelated as a function of m . The constants α^* and β^* correspond to actual slope and offset values as in Eq. 1.17, i.e., $\alpha = \frac{p_k \Delta x}{\Delta y}$ and $\beta = \frac{p_k x_{min} + \tau_h - y_{min}}{\Delta y}$.

What is of interest now, is to find the value and shape of the peak (if any is found) in the discrete parameter domain as a function of the noise deviation σ . Eq. 5.1 implies that the probability of the sample $g(m, n)$ being 1 is given by

$$P\{g(m, n) = 1\} = P\{n = [\alpha^* m + \beta^* + \lambda]\} \quad (5.2)$$

$$= P\{0 = [n - \alpha^* m - \beta^* - \lambda]\} \quad (5.3)$$

$$= P\left\{-\frac{1}{2} < n - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \quad (5.4)$$

and assuming that the sample point (m, n) is given from the nearest neighbour mapping in the discrete Radon transform, $n = [\alpha m + \beta]$, then the rounding function is modelled as an additive

(noise) term ω

$$P\{g(m, n) = 1\} = P\left\{-\frac{1}{2} < [\alpha m + \beta] - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \quad (5.5)$$

$$= P\left\{-\frac{1}{2} < \alpha m + \beta + \omega - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \quad (5.6)$$

$$= P\left\{-\frac{1}{2} < \zeta + \omega - \lambda < \frac{1}{2}\right\} \text{ where } \zeta = (\alpha - \alpha^*)m + \beta - \beta^* \quad (5.7)$$

$$= \Phi\left(\frac{\frac{1}{2} + \zeta + \omega}{\sigma}\right) - \Phi\left(\frac{-\frac{1}{2} + \zeta + \omega}{\sigma}\right) \quad (5.8)$$

where $\Phi(\cdot)$ is the Gaussian probability function, and ζ is a displacement between the true line parameters (α^*, β^*) and the parameters (α, β) at a given position m .

Due to many values of m used in the discrete Radon transform, it is a reasonable approximation to model ω as a uniformly distributed variable between the limits $-1/2$ and $1/2$, i.e., $\omega_m \in \mathcal{U}(-1/2, 1/2)$. Eq. 5.8 implies that the average contribution from the pixel $g(m, n)$ to the discrete Radon transform is given by

$$E\{\check{g}(k, h)|g(m, n)\} = \int_{\omega=-\frac{1}{2}}^{\omega=\frac{1}{2}} P_m(\omega) d\omega \quad (5.9)$$

$$= \int_{\omega=-\frac{1}{2}}^{\omega=\frac{1}{2}} \Phi\left(\frac{\frac{1}{2} + \zeta + \omega}{\sigma}\right) - \Phi\left(\frac{-\frac{1}{2} + \zeta + \omega}{\sigma}\right) d\omega \quad (5.10)$$

In order to get an analytical expression for $E\{\check{g}(k, h)|g(m, n)\}$ the Gaussian probability function is approximated

$$\Phi(x) \approx \frac{1}{2} \left(1 + \tanh\left(\frac{x}{\gamma}\right)\right) \text{ where } \gamma = \sqrt{\frac{\pi}{2}} \quad (5.11)$$

i.e., the integral of $\Phi(\cdot)$ can be approximated by

$$\int \Phi(x) dx \approx \frac{\gamma}{2} \left(\frac{1+x}{\gamma} + \log \cosh\left(\frac{x}{\gamma}\right)\right) \quad (5.12)$$

where $\log(\cdot)$ is the natural logarithm.

If Eq. 5.12 is inserted into Eq. 5.10, and doing some rearrangements of the expressions, it is found that

$$E\{\check{g}(k, h)|g(m, n)\} = \frac{\gamma\sigma}{2} \log\left(\frac{\cosh\left(\frac{2\zeta}{\gamma\sigma}\right) + \cosh\left(\frac{2}{\gamma\sigma}\right)}{\cosh\left(\frac{2\zeta}{\gamma\sigma}\right) + 1}\right) \quad (5.13)$$

In Fig. 5.1 the average weight to the discrete Radon transform is shown as a function of σ and ζ .

A special case of interest regards $\zeta = 0$, i.e., where the coordinates of the line in the image matches exactly a sample in the discrete parameter domain.

$$\zeta = 0 \Rightarrow E\{\check{g}(k, h)|g(m, n)\} = \sigma\sqrt{\frac{\pi}{8}} \log\left(\frac{1}{2} \left(1 + \cosh\left(\frac{\sqrt{8}}{\sigma\sqrt{\pi}}\right)\right)\right) \quad (5.14)$$

In Fig. 5.2 the average weight to the discrete Radon transform is shown as a function of σ , when the displacement is assumed negligible. It can be seen that the function agrees well with a simulated result shown in Fig. 5.3, that is found by generating 10000 images with the correct noise

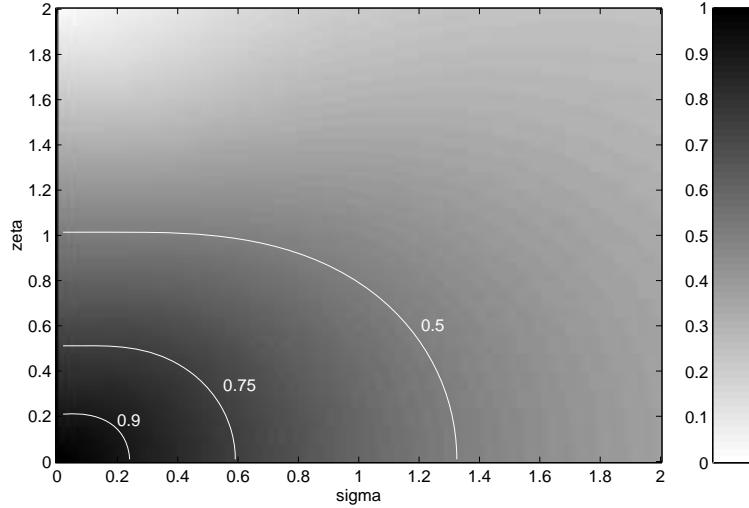


Figure 5.1 The average weight to the discrete Radon transform $E\{\bar{g}(k, h)|g(m, n)\}$ as a function of the noise deviation σ on the first axis, and the displacement ζ on the second axis.

amplitude σ and computing the discrete Radon transform only for the parameter set matching the true parameters. An even better agreement can be found using a numerical integration of Eq. 5.10, but Eq. 5.12 provides an simple analytical result that approximates the simulated results well, though it can also be seen that the theoretical model has a small bias, when $\sigma \approx 0.1$.

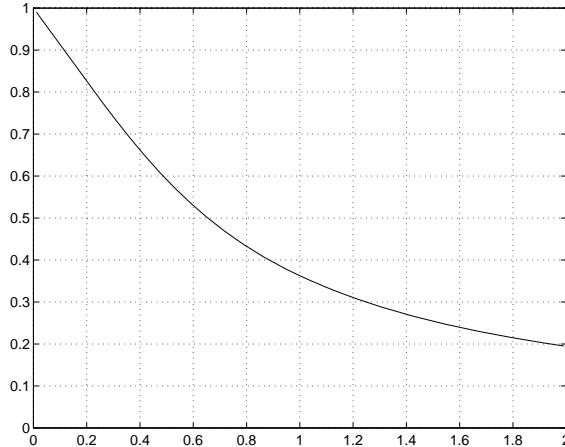


Figure 5.2 The theoretical result of the average weight to the discrete Radon transform, found in Eq. 5.14, as a function of σ when the displacement ζ is negligible.

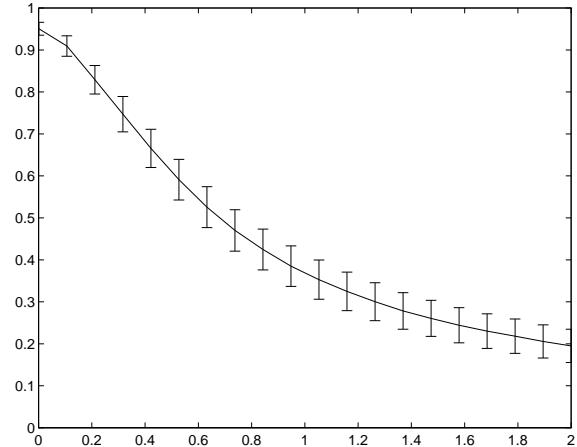


Figure 5.3 The simulated average value of the peak using 10000 images for each noise level. The vertical error bars show the measured deviation.

Fig. 5.2 or Eq. 5.14 can be used to predict whether line parameters might be estimated given that the lines have wiggles. If, e.g., demanding that the peak value cannot decrease more than 50% due to wiggles, then σ cannot exceed approximately 0.7. And a 25% decrease of the peak value is found if $\sigma \approx 0.25$. In Fig. 5.4 is shown an image with one line without noise, and Fig. 5.5 shows the discrete parameter domain in the area of the peak. Using the same sampling parameters, Fig. 5.6 shows an image where $\sigma = 0.25$, and Fig. 5.7 shows the corresponding discrete parameter domain. The images are scaled individually according to the minimal and maximal values, and it

can be seen that the maximal value here is around 80, and the shape is approximately the same as seen from Fig. 5.5. Next $\sigma = 0.5$ gives an image like it is shown in Fig. 5.8, and it can be seen that the noise level is high. In Fig. 5.9 is shown the discrete parameter domain, and it can be seen that the peak here covers a larger part of the field of view and the peak is somewhat scattered.

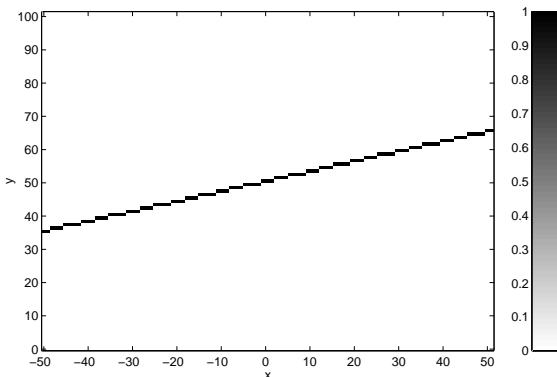


Figure 5.4 An image without noise.

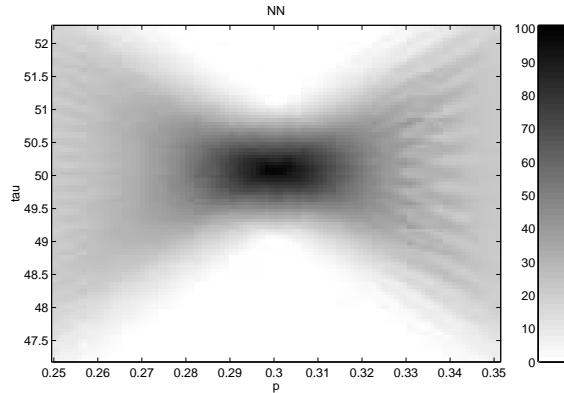


Figure 5.5 The corresponding discrete Radon transform.

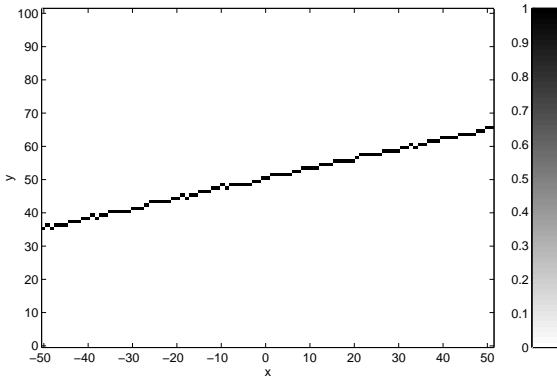


Figure 5.6 An image using $\sigma = 0.25$.

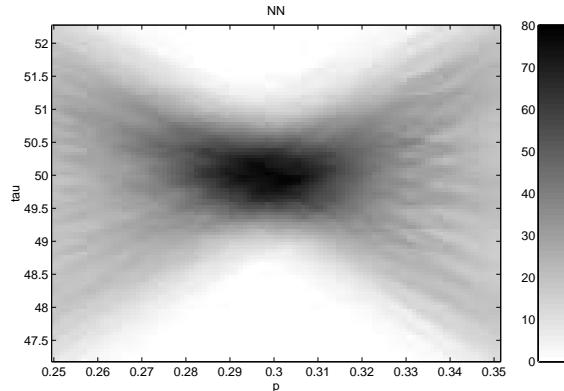


Figure 5.7 The corresponding discrete Radon transform.

Eq. 5.13 can also be analyzed in the case where $\sigma = 0$, in order to focus on the importance of the displacement. First, the expression for ζ is rewritten

$$\zeta = (\alpha - \alpha^*)m + \beta - \beta^* = (p^* - p_k) \frac{x_m}{\Delta y} + \frac{\tau^* - \tau_h}{\Delta y} \quad (5.15)$$

where (p^*, τ^*) is the line parameters corresponding to (α^*, β^*) , and (p_k, τ_h) correspond to (α, β) .

Assuming that $p^* = p_k$, the displacement can be analyzed in the τ -direction. From Fig. 5.1 it can be seen that $\zeta \approx 1$, correspond to $E\{\check{g}(k, h)|g(m, n)\} \approx 0.5$, i.e., that the peak value has decreased to approximately 50% of the possible maximum. This can be used to restrict the sampling interval $\Delta\tau$, due to the quantization of τ_h

$$E\{\check{g}(k, h)|g(m, n)\} \geq \frac{1}{2} \Rightarrow \frac{|\tau - \tau_h|}{\Delta y} < 1 \Rightarrow |\tau - \tau_h| < \Delta y \Rightarrow \Delta\tau < \Delta y \quad (5.16)$$

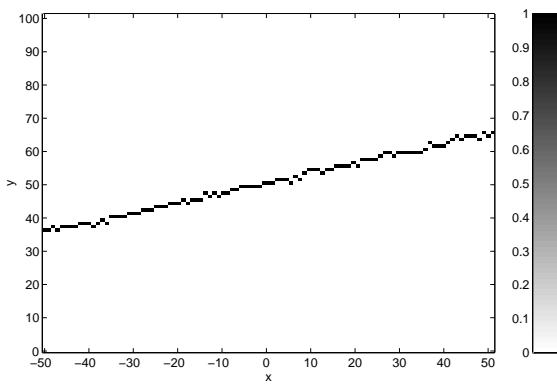


Figure 5.8 An image using $\sigma = 0.5$.

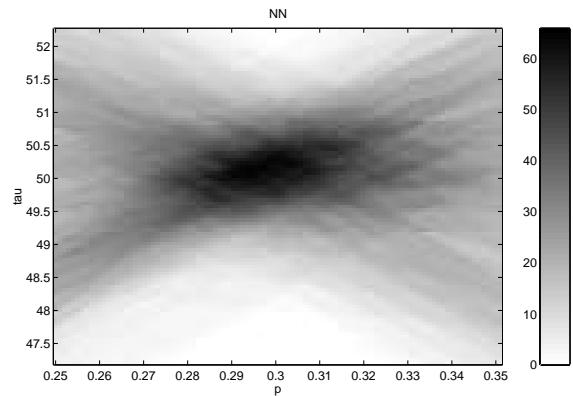


Figure 5.9 The corresponding discrete Radon transform.

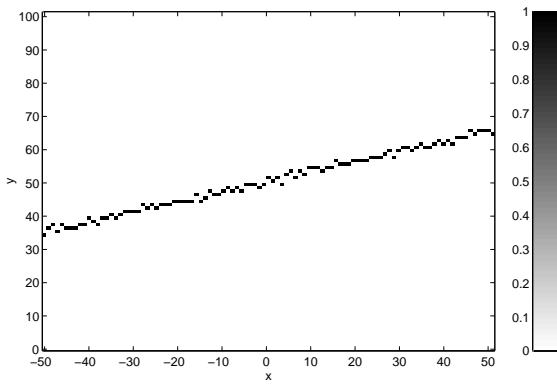


Figure 5.10 An image using $\sigma = 0.7$.

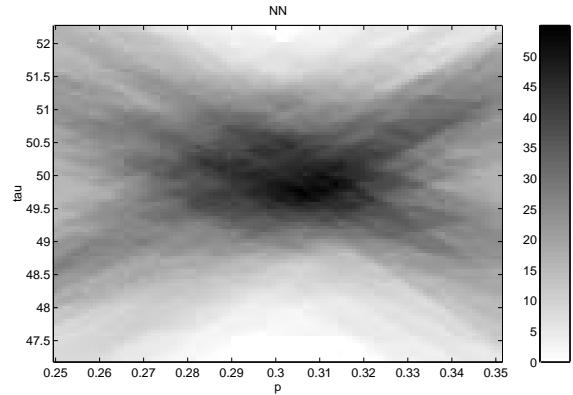


Figure 5.11 The corresponding discrete Radon transform.

This result agrees with Eq. 1.31, but here the result comes with a very important information: Using Eq. 1.31 can imply that the peak value is reduced by 50% of the possible maximum. This implies a compromise between the number of samples in the discrete parameter domain, i.e., the sampling intervals, and the time used to compute the discrete parameter domain.

It should be mentioned, that a simple and intuitively model can be proposed to analyze the case $\zeta = 0$, but the results is not as good as obtained by the model described above. The discrete Radon transform will due to the rounding have a window in the n -direction, like it is shown in Fig. 5.12, of length 1. Hence assuming that the offset ω is negligible, then the probability of detecting a 1 is given by

$$P\{g(m, n) = 1\} = P\left\{-\frac{1}{2} < \lambda < \frac{1}{2}\right\} = \Phi\left(\frac{1}{2\sigma}\right) - \Phi\left(-\frac{1}{2\sigma}\right) = 2\Phi\left(\frac{1}{2\sigma}\right) - 1 \quad (5.17)$$

which can be interpreted as the reduction factor of the peak. The probability is shown as a function of σ in Fig. 5.13.

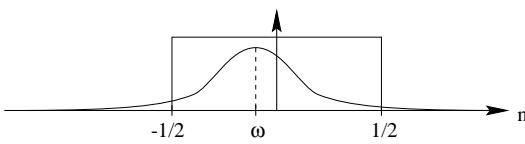


Figure 5.12 A Gaussian function and the windowing function found by using a nearest neighbour approximation of the discrete Radon transform.

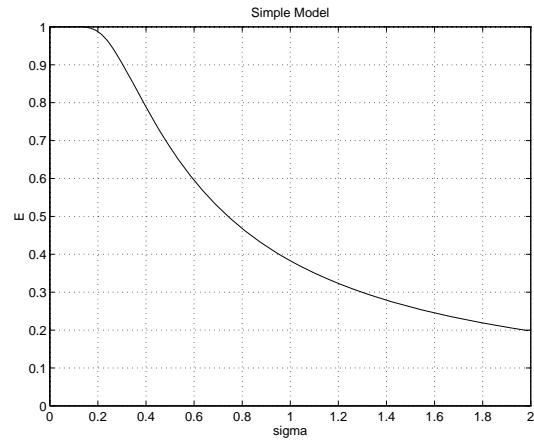


Figure 5.13 The simple model.

The conclusion on this section is that a peak is generated in the discrete parameter domain corresponding to each line in the image, and the value of the peak will be lowered corresponding to the noise level, which in this case is due to the wiggles. In Section 5.3 additive image noise will also be considered, and it could be of interest to combine the analysis of these two problems.

5.2 A “Fuzzy” Radon Transform

Instead of using the pixel positions designated by the discrete Radon transform, a search strategy could be used to incorporate wiggly lines into the curve detection algorithm. Assuming that the lines have wiggles as shown in Section 5.1, one strategy to incorporate this prior knowledge could be to allow the discrete Radon transform to search for the maximum image value within a fixed window.

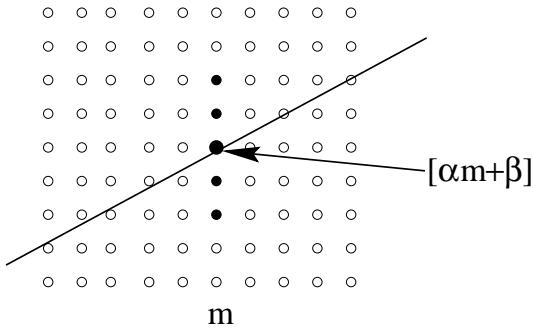


Figure 5.14 Instead of adding the nearest vertical pixel to the sum the maximum is used within a window of here five pixels.

The strategy is demonstrated for discrete slant stacking, but can be adapted to, e.g., the normal Radon transform or the discrete generalized Radon transform. Assuming that the curve amplitude is positive and additive noise is corrupting the image, Fig. 5.14 shows how a search window of five pixels is used for each value m on the horizontal axis. Within the window one apparent strategy is to update the sum in the discrete Radon transform by the maximum of the

five pixels. Besides that this strategy implies a big increase in the processing time compared to the use of discrete slant stacking, it is not optimal due to the fact that, this strategy is identical to filtering the input image with a median type filter for each value of m . The filter should return the maximum of the pixels within the window, hence the strategy is equivalent to an initial image domain filtering and then ordinary discrete slant stacking. This demonstrates that the alteration should not be used, but the local filtering should only be used, if the maximum filter can select the correct pixel within the window.

The following six figures illustrate the problem with the strategy. Fig. 5.15 is a noisy image containing two intersecting lines, and Fig. 5.16 shows the corresponding discrete Radon transform (nearest neighbour approximation). Two peaks are very clearly marked. Next Fig. 5.17 shows the image, where a maximum filter with window width of 3 pixels has been used for each value of m . Fig. 5.18 shows the corresponding discrete Radon transform, and it is seen that the peaks are still detectable, but the resolution has become worse. Finally Figs. 5.19 and 5.20 uses a filter window of length 5, and the result is more blurring of the peaks, and that the background level in the discrete parameter domain has gone up.

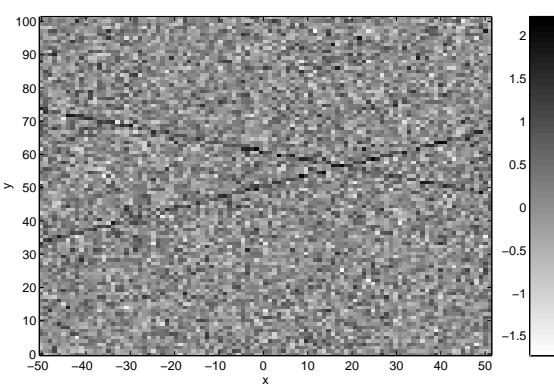


Figure 5.15 The original image containing two intersecting lines covered with additive noise.

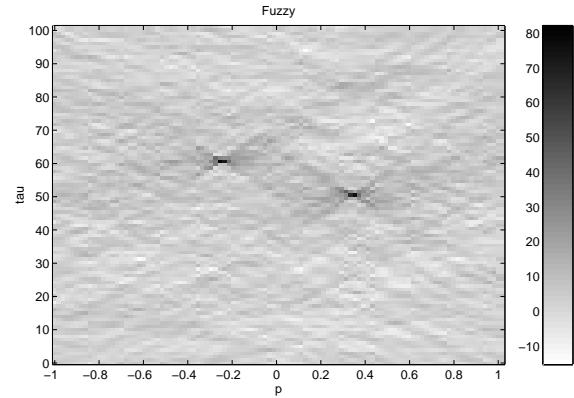


Figure 5.16 The corresponding discrete parameter domain.

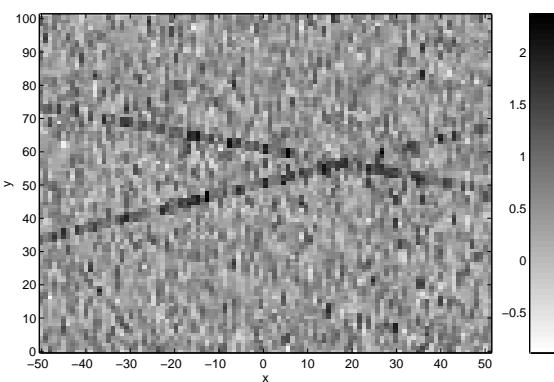


Figure 5.17 The filtered image containing two intersecting lines covered with additive noise.

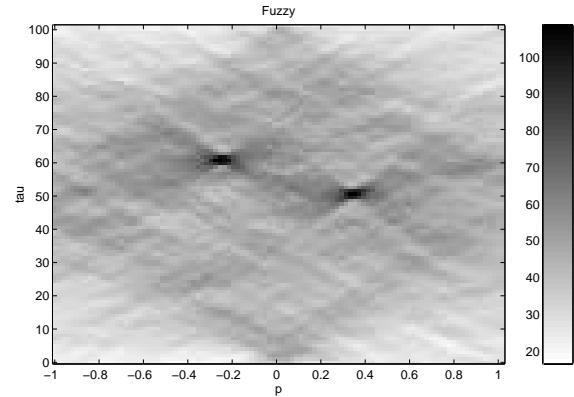


Figure 5.18 The corresponding discrete parameter domain.

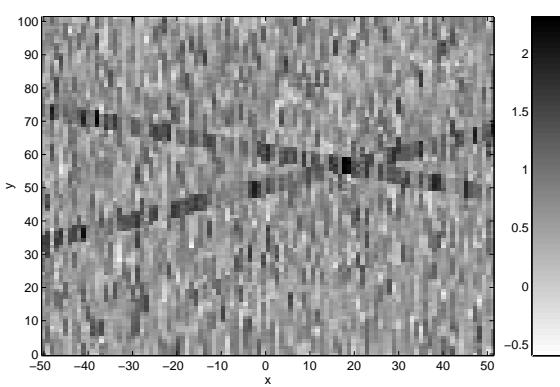


Figure 5.19 The filtered image containing two intersecting lines covered with additive noise.

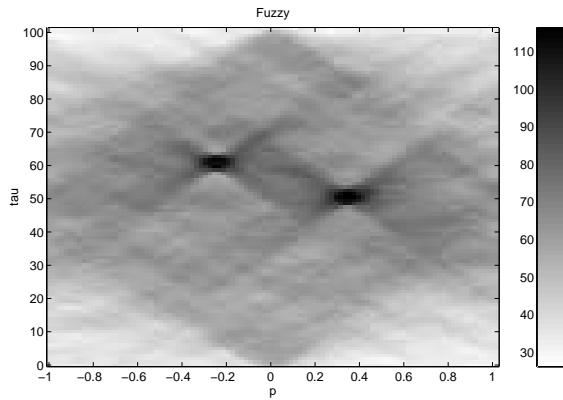


Figure 5.20 The corresponding discrete parameter domain.

5.3 Detection of Curves in Noisy Images

A natural expansion of the Radon transform is the (discrete) generalized Radon transform (GRT) [36, 1, 2]. As demonstrated in Chapter 4 a major advantage of the GRT is that curves are allowed to intersect. Another major advantage that will be demonstrated in this Section, is that the GRT is very robust to noise.

In this section a probabilistic approach is used to show that the GRT can be used for curve detection if the noise in the image is below a certain level compared to the signal values on the curves. An analytical expression is given quantifying the limits of using GRT for curve detection especially with respect to very noisy images. If noise is added to an image containing curves, the problem is that peaks in the parameter domain may or may not correspond to actual curve parameters. A threshold level, based on the noise level, is derived and applied for separation of noise and curve information in the parameter domain.

A numerical example is provided to illustrate the presented theory. Note that the theory can be used to reestimate the image containing curves [56].

5.3.1 The Generalized Radon Transform

The generalized Radon transform, GRT, of a digital image can be defined in many ways. One way is

$$\check{g}(\mathbf{j}) = \sum_{l=0}^{L-1} g(\phi_m(l, \mathbf{j}), \phi_n(l, \mathbf{j})) \quad (5.18)$$

where \check{g} denotes the GRT of the image $g(m, n)$ and \mathbf{j} is a multi dimensional vector containing the curve parameters. The two curve functions $\phi_m(l, \mathbf{j})$ and $\phi_n(l, \mathbf{j})$ define the curve type and are (in principle) arbitrary and an interpolation scheme is assumed implicitly; e.g., by rounding the functions $\phi_m(l, \mathbf{j})$ and $\phi_n(l, \mathbf{j})$ to the nearest sample point. A popular choice is the linear curve functions; e.g., normal parameters $\mathbf{j} = (\rho, \theta)$. Another frequent choice is the (τ, p) -parameters, cf. Chapter 1, where $\phi_m(l, \tau, p) = l$ and $\phi_n(l, \tau, p) = p l + \tau$. Choosing only $\phi_m(l, \mathbf{j}) = l$ the definition will match the one used in Chapter 4.

Even though the GRT can be applied to any given image, the main feature of the GRT is that an image, which contains a discrete curve matching the curve functions at one parameter vector \mathbf{j}^* implies that the parameter domain $\check{g}(\mathbf{j})$, will show a peak at that specific parameter vector $\mathbf{j} = \mathbf{j}^*$. The linearity of the GRT implies that each curve in the image will be transformed into

a peak in the parameter domain. In this section a curve in an image is defined by large image values of the same sign on the curve and otherwise (approximately) zero.

Initially only two values of the GRT will be considered. The first, $\check{g}(\mathbf{j}^*)$, corresponds to a curve in the image. Another, $\check{g}(\mathbf{j}^-)$, corresponds to a parameter vector, that does not match a curve in the image. It is assumed that $\check{g}(\mathbf{j}^*)$ is the sum of a mean signal value μ over all L samples, and $\check{g}(\mathbf{j}^-)$ covers (approximately) no samples of the curve(s) in the image. Both values of \check{g} are contaminated with noise, which is due to noise in the image. Assume the noise in the image is nearly uncorrelated with zero mean (e.g., by subtracting a DC-value from the image) and variance σ^2 .

5.3.2 Curve Detection using the Generalized Radon Transform

A classical curve detection algorithm is to determine the parameter vectors from the positions of peaks in the parameter space

$$\mathbf{j}^* = \arg \{ |\check{g}(\mathbf{j})| \geq L \mu^* \} \quad (5.19)$$

The reason for choosing the significance level in this way is that Eq. 5.18 consists of a summation over L samples, and μ^* is a lower positive bound on the mean signal level on the curve; e.g, found by estimation. The purpose of the following is to estimate whether curves having the signal level μ can be detected using Eq. 5.19, if the image is contaminated with the described noise.

Due to the linearity, \check{g} consists of a curve part and a noise part. If $L \gg 1$ the sum of the noise terms \check{g}_{noise} will approximately be Gaussian distributed with zero mean and variance $L\sigma^2$ due to the Central Limit Theorem. This implies that the two considered values of the GRT are distributed as $\check{g}(\mathbf{j}^*) = \mu L + \check{g}_{\text{noise}}^* \in \mathcal{N}(\mu L, L\sigma^2)$ and $\check{g}(\mathbf{j}^-) = \check{g}_{\text{noise}}^- \in \mathcal{N}(0, L\sigma^2)$. Since Eq. 5.19 selects the large values in the parameter domain, an important issue is the probability of detecting the correct parameter vector of the two considered

$$P_{\text{det } 2} \equiv P\{|\check{g}(\mathbf{j}^*)| > |\check{g}(\mathbf{j}^-)|\} \quad (5.20)$$

Assume two uncorrelated stochastic variables

$$A \in \mathcal{N}(\mu_A, \sigma^2) \quad \text{and} \quad B \in \mathcal{N}(\mu_B, \sigma^2) \quad (5.21)$$

hence the joint probability distribution function is given by

$$f(a, b) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}((a-\mu_A)^2+(b-\mu_B)^2)} \quad (5.22)$$

The probability $P\{|A| > |B|\}$ will now be found by rotating the coordinate system: $x = \frac{1}{\sqrt{2}}(a-b)$ and $y = \frac{1}{\sqrt{2}}(a+b)$. In the new coordinates the joint probability density function is

$$g(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}((x-\mu_x)^2+(y-\mu_y)^2)} \quad (5.23)$$

where $\mu_x = \frac{1}{\sqrt{2}}(\mu_A - \mu_B)$ and $\mu_y = \frac{1}{\sqrt{2}}(\mu_A + \mu_B)$.

$$P\{|A| > |B|\} = \int_{x=-\infty}^0 \int_{y=-\infty}^0 g(x, y) dy dx + \int_{x=0}^{\infty} \int_{y=0}^{\infty} g(x, y) dy dx \quad (5.24)$$

The two integrals can be separated and it is easily found that

$$P\{|A| > |B|\} = \frac{1}{2} \left(1 + \operatorname{erf}\left(\frac{\mu_A - \mu_B}{2\sigma}\right) \operatorname{erf}\left(\frac{\mu_A + \mu_B}{2\sigma}\right) \right) \quad (5.25)$$

where $\text{erf}(x)$ is the error function. Inserting $\mu_A = \mu L$, $\mu_B = 0$, and replacing σ with $\sigma\sqrt{L}$ gives

$$P_{\text{det } 2} = \frac{1}{2} \left(1 + \left(\text{erf} \left(\frac{\lambda}{2} \right) \right)^2 \right) \quad \text{and} \quad \lambda = \frac{\mu\sqrt{L}}{\sigma} \quad (5.26)$$

Note that in this case $P_{\text{det } 2}$, shown in Fig. 5.21, only depends on one parameter λ . Note that $|\lambda| > 4$ gives an almost certain detection. This is the case if $\sigma \rightarrow 0$ or $\mu \rightarrow \infty$.

When using the GRT to detect curves then the discrete parameter domain will not only have two, but \mathcal{J} different parameter vectors, where \mathcal{J} is the number of samples in the parameter domain. It is assumed that all the noise sources in the parameter domain are independent and in the following, the detection of a single curve is analyzed. Selecting the position of the highest peak in the parameter domain, the probability of the selected parameter vector being correct, can be approximated by

$$P_{\text{det all}} \cong \prod_{i=2}^{\mathcal{J}} P_{\text{det } 2} \cong P_{\text{det } 2}^{\mathcal{J}-1} = \left(\frac{1}{2} \left(1 + \left(\text{erf} \left(\frac{\lambda}{2} \right) \right)^2 \right) \right)^{\mathcal{J}-1} \cong 1 - \frac{2\mathcal{J}}{\lambda\sqrt{\pi}} e^{-\lambda^2/4} \quad (5.27)$$

The last simple approximation is valid if the detection probability is close to 1 as seen from Fig. 5.22. Several characteristics can be noted: The figure shows a narrow transition from low to high detection probability as a function of λ , and \mathcal{J} does not change the shape of $P_{\text{det all}}$ significantly. If demanding a high detection probability $P_{\text{det all}}$ then Eq. 5.27 and Fig. 5.22 demonstrate that \mathcal{J} should be held low; i.e., by reducing the number of samples in the parameter domain to a minimum. It should be noted that this will involve a compromise on the range of possible parameter vectors.

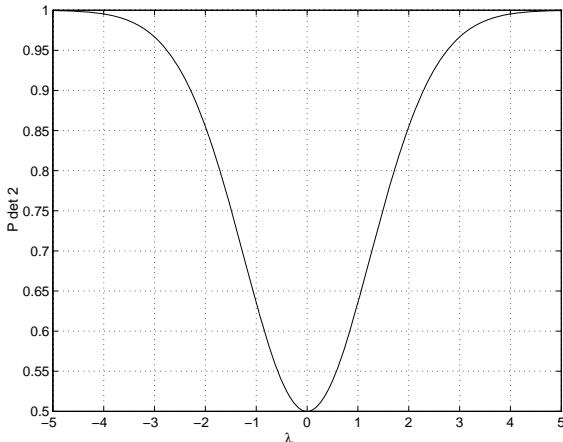


Figure 5.21 The probability of detecting the right curve parameters of two possible as a function of λ .

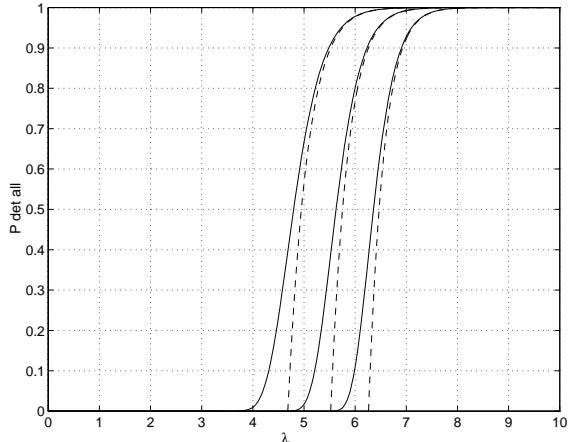


Figure 5.22 Solid lines show the probability of detecting the right curve parameter as a function of λ , and dashed lines show the simple approximation. From left to right $J=1000$, $J=10000$ and rightmost $J=100000$.

Eq. 5.27 can also be used to set requirements on, e.g., the absolute mean signal level μ^* of the curve(s) to be detected. Demanding a detection probability $P_{\text{det all}}$ greater than P^* implies that $\mu^* = \sigma / (\lambda^* \sqrt{L})$, where λ^* can be found by from Eq. 5.27 with a given detection probability, summation length L , number of samples in the parameter domain \mathcal{J} , and the standard deviation σ (e.g., by found by estimation). Any $|\mu|$ less than the threshold level, μ^* , can be considered as noise. In this way it is possible to give a statistically based estimate on the thresholding level in Eq. 5.19.

5.3.3 Discussion

Even though the above theory is developed by analyzing one curve in the image, the theory can be used if the image contains few curves. Instead of having one peak in the parameter domain representing one curve in the image, each of the Z curves in the image, where $Z \ll J$, will give a peak in the parameter domain, even if the curves cross each other. With Z curves each of corresponding Z \tilde{g} -values must be larger than the rest in the parameter domain. If only a few curves are present, the rest of the parameter domain is dominated by noise, and the probability of detection for each of the Z curves can be found from Eq. 5.27.

The theory used to derive Eq. 5.27 is somewhat pessimistic in estimation of the influence of the noise. This is partly due to the assumption that all the GRT-values include summing noise over L samples. Normally some of the GRT-values will require summing up over a curve partially outside the image, where the image must be assumed equal to zero. Furthermore some correlation must be expected in the parameter domain, especially if the number of dimensions in the parameter domain is higher than two. Depending on the sampling parameters, this implies that an effective J (less than the number of samples in the parameter domain) must be used in Eq. 5.27.

5.3.4 An Example of Line Detection in a very Noisy Image

A noise free image containing eight lines with limited slope is created. The image, shown in Fig. 5.23, has $101 * 101$ samples. The curve sampling functions are chosen to $\phi_m = l - 50$ and $\phi_n = p(l - 50) + \tau$ and L is set to 101. The offset is made in order to lower the sampling requirements in the parameter domain, cf. Subsection 1.4.4. The sampling distances in the parameter domain is set to $\Delta\tau = 1$ and $\Delta p = 0.01$. The line parameters are listed in Table 5.1.

No	p	τ	μ	No	p	τ	μ
1	0.10	60	1.5	5	0.00	30	1.0
2	0.25	50	-1.5	6	0.40	45	-1.0
3	-0.10	80	-1.0	7	-0.36	52	-1.0
4	-0.25	30	1.0	8	0.10	80	0.5

Table 5.1 Line parameters. p is the slope, τ is the offset, and μ is the curve amplitude.

To illustrate the potential of the GRT a very noisy image is generated, by adding Gaussian noise to the noise free image with zero mean and standard deviation $\sigma = 1$. It can be seen from Fig. 5.24, that the lines are hard to identify. Choosing $P_{\text{det all}} = 0.95$, Eq. 5.27 gives $\lambda^* = 6.53$; i.e., only lines with $|\mu| > 0.65$ should be detectable. This implies that all but line number eight should be detectable. The absolute value of the parameter domain obtained by the use of the GRT to the noisy image is shown in Fig. 5.25.

Since the noisy image contains few lines with absolute curve amplitude $|\mu|$ being of the same order of magnitude as σ and has approximately zero mean, σ was estimated from the image using the ordinary central variance estimator, which gave $\hat{\sigma} = 1.04$. Setting $P_{\text{det all}}$ to 0.95, Eq. 5.27 results in $L \mu^* = 65.7$. This is used for thresholding of the parameter domain as shown in Fig. 5.26. Seven of the eight line parameters are found despite the poor signal to noise ratio in the image. Note that some of the lines will be represented by a few neighbour parameter vectors, which can be corrected by clustering neighbour parameter vectors. The error is due to the sampling of the parameter domain and the finite image size.

The theory predicted that only seven lines could be detected. The eighth line can be detected if the curve length can be increased or the noise variance can be reduced. If the theory is used

with P_{det} all very low, $L\mu^*$ get lower and noise peaks will appear in the parameter domain along with parameters of the eighth line. In Fig. 5.27 the threshold level has been reduced to, e.g., $0.7L\mu^* = 46.0$. As it can be seen, noise will now give parameter vectors which do not represent a curve. As seen from Fig. 5.28 a further reduction of the threshold level to, e.g., $0.5L\mu^* = 32.9$ gives a parameter domain, where all eight lines are present. Due to the noise level many false parameter vectors can also be observed.

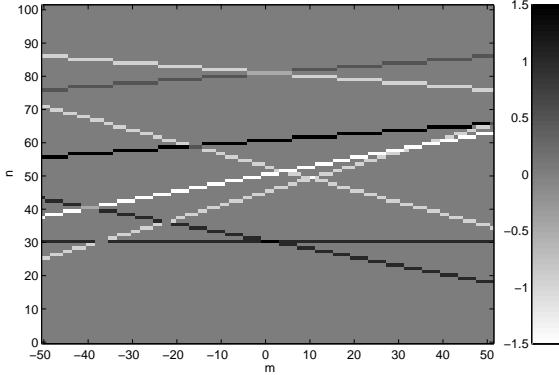


Figure 5.23 Noise free image with eight lines.

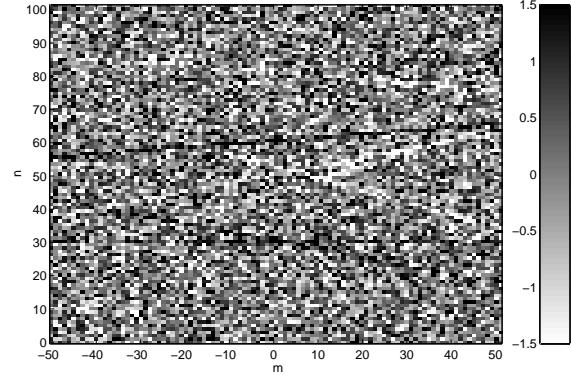


Figure 5.24 The same image contaminated with additive Gaussian noise.

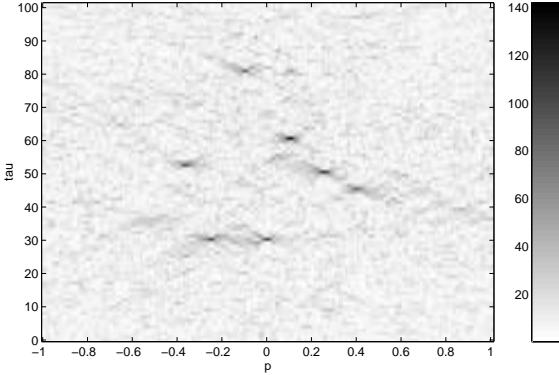


Figure 5.25 The absolute GRT of the noisy image. Note the peaks corresponding to the curves.

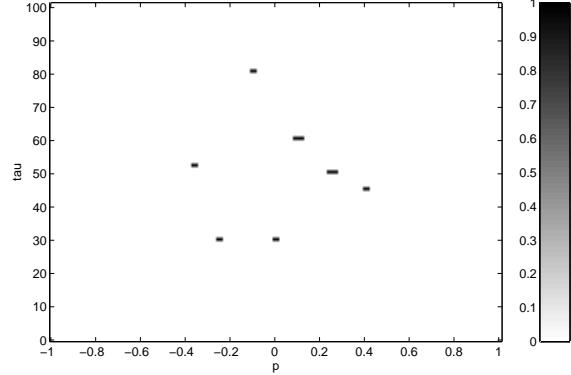


Figure 5.26 Threshold of the absolute GRT using the estimated threshold level.

5.4 Summary

This chapter demonstrated the possibilities and limitations of using discrete Radon transform with respect to noise. In Section 5.1 detection of wiggly lines has been analyzed, and analytical expressions have been derived that models the behavior of the discrete Radon transform with respect to wiggly lines. It was shown in Section 5.2 that the simple incorporation of wiggles can be perceived as a maximum filter used in the image domain.

Finally Section 5.3 presented a statistically based noise analysis of the generalized Radon transform. The analysis was used to derive a threshold level in order to separate curve information and noise in the parameter domain. A numerical example was provided to illustrate the theory.

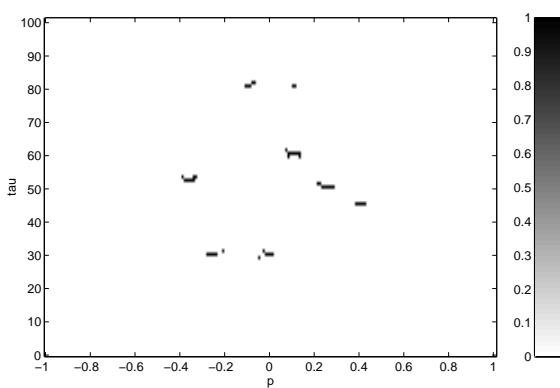


Figure 5.27 Threshold of the absolute GRT using 0.7 times the estimated threshold level.

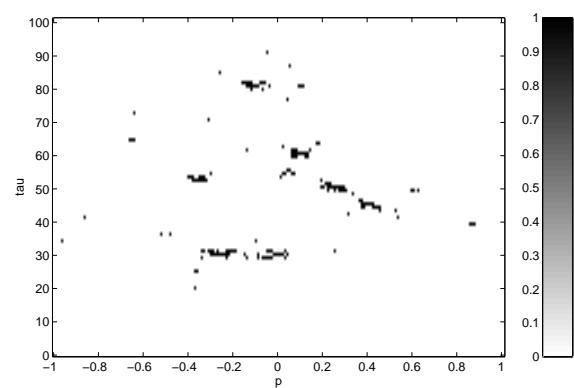


Figure 5.28 Threshold of the absolute GRT using 0.5 times the estimated threshold level.

Part II

The Inverse Radon Transform and PET

Chapter 6

Introduction to Computerized Tomography

One of the major inventions in this century is the CT-scanner (Computerized Tomography). It can be mentioned that Cormack and Hounsfield got the Nobel-prize in Medicine 1979 for their work with computed axial tomography. The CT-scanner can be used for reconstruction of the X-ray absorption in the interior of structures, such as patients or machine parts with possible internal fractures.

Over the years several types of scanners have emerged such as the MR-scanner (MR - Magnetic Resonance) for measuring features related to the contents of water. This type of scanner is common in larger hospitals and it only uses magnetic fields, no X-rays or radioactive tracers are involved. Two other emerging techniques, both used to monitor the concentration of radioactive tracers in the interior of an object, are SPECT (Single Photon Emission Computed Tomography) and PET (Positron Emission Tomography). If the radioactive tracer is attached to, e.g., glucose [57], then the interesting possibility of measuring the activity of the brain arises. This possibility of dynamic measurements of the activity of the brain is now also appearing with one of the newest technologies fMRI (functional MRI).

What is common for the development of all types of scanners is that they, to some extent, have been based on inversion of Radon transform, but it should be mentioned that the MRI scanners mostly use inverse Fourier transformation. The most direct use of inverse Radon transform is found with the CT-scanner, and Section 6.1 describes the fundamental theory of physics along with the motivation of modelling the problem by use of the Radon transform. Next, Section 6.2 goes into the fundamentals of PET-imaging and some of the problems in practical scanning. An introduction to the various scanners mentioned above is given in [58].

Besides Section 7.6, which goes through the major direct inversion methods for the (p, τ) Radon transform (slant stacking, cf. Chapter 1), this part of the thesis will go into reconstruction from line integrals using normal parameters. First, in Chapter 7 some of the well-known direct 2D Radon inversion formulas are derived [14]. A large software package that includes all of the inversion methods mentioned in Chapters 7-8 has been developed [59, 60, 61]. Together with the package for generating images and their Radon transform [22] described in Appendix C.1, efficient and easy tools have been made for testing new algorithms and doing 2D reconstruction with the built-in algorithms.

During the very first years of work with the CT-scanner Hounsfield and his colleagues used an iterative reconstruction scheme based on linear algebra for reconstructing images. This strategy had a number of problems, and was abandoned when it was realized that the reconstruction problem could be modelled by the Radon transform. From the moment this was realized, most

reconstruction algorithms in practical use have been based on direct reconstruction methods, especially the so called Filtered Backprojection algorithm. But in recent years, iterative reconstruction schemes have again become a very popular area of research. In Chapter 9 the basis of the iterative reconstruction algorithms are provided, and some of the more widely used methods are reviewed. A very fast implementation of these algorithms in 2D has been made [60, 5, 6]. It turns out that the methods used for iterative reconstruction of the 2D images directly can also be applied for reconstruction of 3D volumes, hence the methods both cover 2D and 3D reconstruction algorithms.

One of the newer and emerging techniques is 3D direct reconstruction from plane integrals [62] or line integrals relevant for 3D CT-scanners and 3D PET medical scanners respectively [63, 64, 65, 66, 67, 68, 69, 70, 71]. Some of the direct reconstruction methods, problems, and possibilities are shown in Chapter 10. Aspects of implementation including the use of multi processor hardware are also considered. A software packages has been developed for generating volumes and the corresponding Radon transform along with a package for reconstruction of 3D volumes using direct and iterative methods.

6.1 Fundamental Theory of the CT-Scanner

A CT-scanner can consist of a ring with one X-ray emitter and a large number of detectors positioned opposite to the emitter as shown in Fig. 6.1. Rotating the emitter and the detector array around the patient makes it possible to cover all parts of the brain. Note that the brain could be substituted by any structure with an interesting non-homogeneous interior hidden behind the surface.

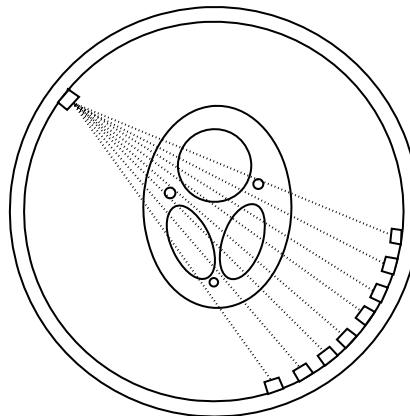


Figure 6.1 The CT Scanner.

It can be assumed that the X-rays travel in straight lines, and the beam is attenuated along that line with the attenuation coefficient μ . After the scan the attenuation coefficient is to be found by reconstruction in each point in the plane. It will be shown that this can be based on inversion of the Radon transform.

Assume a planar (2D) CT-scanner with only one slice of the brain measured, then the coordinate system can be chosen so that the slice is the $z = 0$ plane. Each line going through the scanned object is parameterized with two parameters: The normal distance from $(0, 0)$ to the line ρ , and the angle relative to the first axis θ . These two parameters are shown in Fig. 6.2.

It must be assumed, that the scanned object is non-homogeneous, hence the attenuation coefficient is a function of x and y , i.e., $\mu(x, y)$. Assume that the emitter and one of the detectors define the line (ρ, θ) . In Fig. 6.3 three different objects are shown (seen along the line).

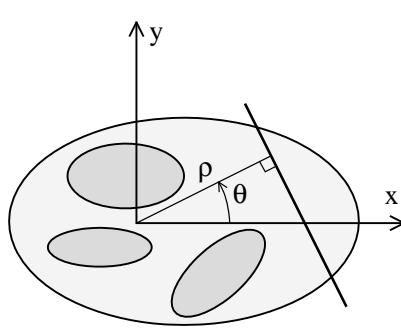


Figure 6.2 Parameters used to describe the lines.

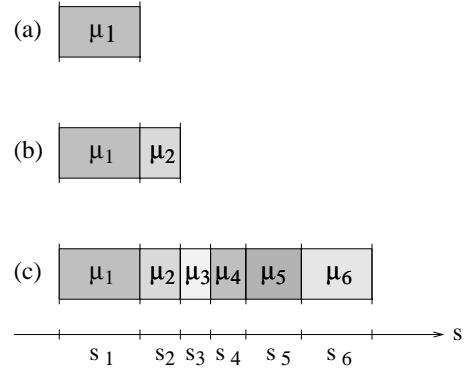


Figure 6.3 Three different media, with attenuation coefficients μ_i and widths s_i .

In case (a) the received intensity will follow the ordinary exponential decay

$$I(\rho, \theta) = I_0 e^{-\mu_1 s_1} \quad (6.1)$$

If two different homogeneous media are penetrated as shown at (b) the received intensity is

$$I(\rho, \theta) = I_0 e^{-\mu_1 s_1 - \mu_2 s_2} \quad (6.2)$$

If several homogeneous media are present, like in case (c), the result is simply that additional exponential factors are included for each layer

$$I(\rho, \theta) = I_0 e^{-\sum_i \mu_i s_i} \quad (6.3)$$

In the general form, where the attenuation coefficient is a function of the line path, the sum in Eq. 6.3 will become an integral and s_i becomes the line element ds .

$$I(\rho, \theta) = I_0 e^{-\int \mu(x, y) ds} \quad (6.4)$$

In this equation the parameter I_0 is the intensity of the emitter, and s denotes the parameter in the normal form of the line, where (x, y) lies on the line defined by (ρ, θ) . Note that the exponential factor can be perceived as the probability of a single photon getting through the absorbing medium.

From Eq. 6.4 the projection $\mathcal{P}(\rho, \theta)$ is defined

$$\mathcal{P}(\rho, \theta) = \log \left(\frac{I_0}{I(\rho, \theta)} \right) = \int \mu(x, y) ds = \int_{-\infty}^{\infty} \mu(\rho \cos \theta - s \sin \theta, \rho \sin \theta + s \cos \theta) ds \quad (6.5)$$

It can be recognized that $\mathcal{P}(\rho, \theta)$ is the Radon transform of $\mu(x, y)$, cf. Eq. 2.8. Using the Dirac delta function the projections can be rewritten, cf. Eq. 2.8

$$\mathcal{P}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mu(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (6.6)$$

The result is, that the attenuation coefficient $\mu(x, y)$ can be found from the projections using the inverse of the Radon transform.

In the literature [14, 72, 73] direct inversion schemes can be found. One class is based on the Fourier Slice Theorem, which is implemented using 1D Fast Fourier Transformation, 2D Fast Fourier transformation and 2D interpolation. This will normally result in a very fast algorithm, but numerical artifacts due to the interpolation can decrease the performance. Another famous inversion scheme is the Filtered Backprojection which is a combination of filtering and integration. This inversion scheme is by far the most common and used in medical scanners due to good numerical stability, but the algorithm will normally be somewhat slower. These aspects are covered in the following chapters.

In conclusion, it has been shown that the CT-scanner gives a map of the attenuation coefficient. The CT-scanners have been used several years, and one of the newer trends is to perform helical CT, where the emitter and detector ring rotate around the patient and make a uniform motion along the z -axis, as shown in Fig. 6.4. In this way a 3D-image, i.e., a volume of the interior can be obtained with a better resolution on the z -axis, compared to a large number of plane scans placed along the z -axis.

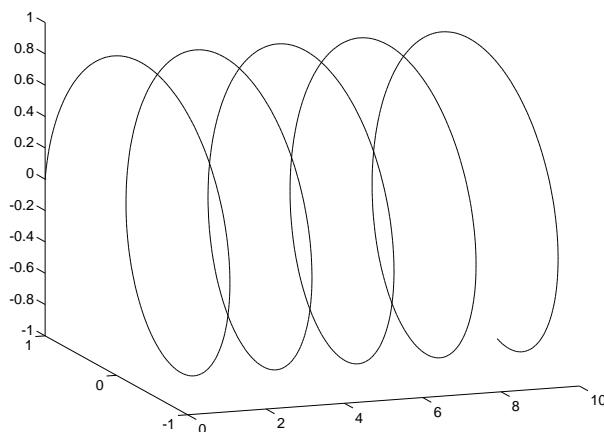


Figure 6.4 Motion of the emitter when using helical CT scanning.

It has been demonstrated that the attenuation coefficient can be measured indirectly, and the reconstruction of images can be done by use of the inverse Radon transform. In the following section another emerging technique, PET-scanning, is presented which have some similarities to the CT-scanner, but a major difference is the number of features, which can be measured, e.g., the glucose metabolism in the brain.

6.2 The PET Scanner

The PET scanner (Positron Emission Tomography) is based on another type of measurement. A small dosage of a radioactive β^+ (positron) tracer, such as F-18, O-15, or C-11, produced by use of a cyclotron is build into a larger molecule such as glucose. The β^+ -labeled glucose is injected into the patient. The β^+ tracer will then be circulated in the tissue by the blood flow. If for instance the brain is to be scanned then the glucose metabolism in the individual regions of the brain will correspond to the local brain activity. In the regions with high brain activity the glucose metabolism will be high, and a corresponding number of radioactive nucleons will decay

under emission of a positron. Within a few millimeters the positron will then interact with an electron and annihilate under emission of two 511 keV photons. The two photons will travel in (nearly) opposite directions outwards where the photons can be detected at detectors placed in a ring in the PET scanner, as illustrated in Fig. 6.5. Note that here only detectors are needed, as the emitter is placed within the patient.

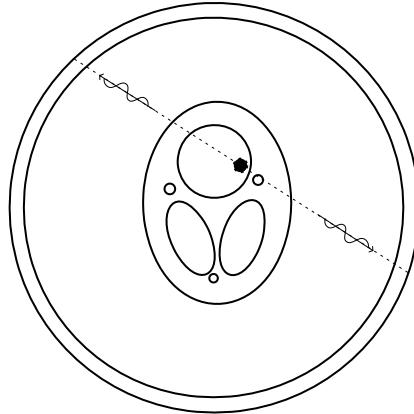


Figure 6.5 Emission of two photons from the place of decay.

In Fig. 6.6 a real PET scanner is shown - the General Electric Advance PET scanner. The photograph was taken at the PET center at the National University Hospital in Copenhagen.



Figure 6.6 The GE Advance PET scanner.

In Fig. 6.7 is shown the same scanner with the front opened. Note the detector blocks are visible. Within a single block a number of detectors are attached, and in this particular scanner 18 rings of detector elements are found, which makes it possible to record 35 slices in 2D (number of planes in the scan) at the same time. The 35 slices is found because the scanner in 2D recording mode allows measuring photon pairs detected in the same ring or two adjacent rings.

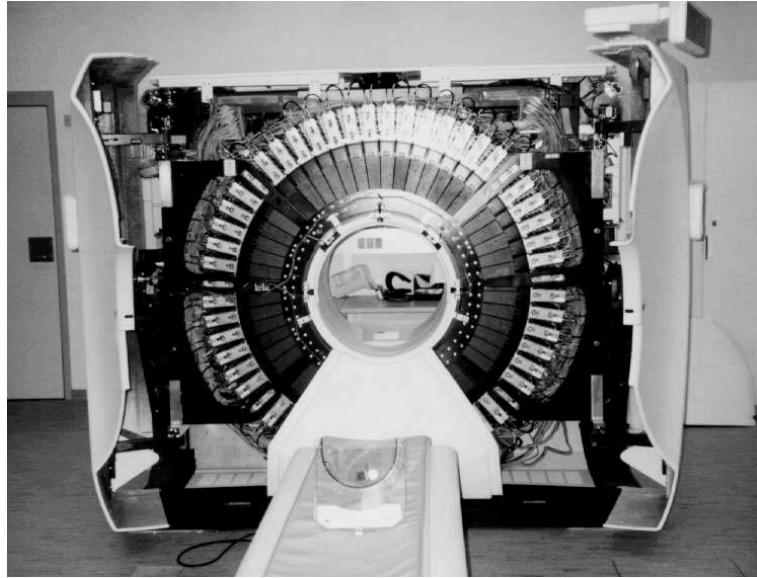


Figure 6.7 The GE Advance PET scanner opened.

The PET hardware will sort the arrival times of the photons, so only two photons that arrived (almost) at the same time at the detector ring are taken into account. The photons travel with the speed of light, and the small difference in arrival time (within approximately 2 nanoseconds) has so far only been used in the Time-of-Flight PET scanners or TOF-PET [74]. Only a very limited number of these scanners have been built. In the following the small time difference is neglected.

A two dimensional matrix of the possible detector versus detector combinations is created, and all values are initialized to zero. The number of possible detector combinations corresponds to a finite set of possible line parameters, (ρ, θ) . Assume that two photons have been detected and the line between the two detectors have line parameters $(\rho, \theta) = (\rho_0, \theta_0)$, then the array is incremented with the value one at that position in the array. This is because the only obtainable information from the two photons is that the photon emission took place somewhere along that line. Depending on the radioactive dosage given to the patient, many decays take place each second in each volume element, and after a PET recording, e.g., 10^7 photon pairs can have been recorded per slice. A PET-scanning can last up to one hour or more, depending on the task of the scanning and the half-life of the tracer. Note that the majority of photon pairs are never detected, due to the limited size, and hence spherical coverage of one detector ring.

After the recording is terminated an array of emissions has been recorded. The individual emission recordings are no longer stored. For a single position or bin in the matrix at $(\rho, \theta) = (\rho_0, \theta_0)$ the array has been developed by several β^+ emissions which took place along the line. The obtained array of emissions, known as the sinogram, are in each position approximately proportional to the total emission intensity along that particular line (times the total recording time, denoted by T_e). The obtained matrix divided by T_e is in mathematical terms an approximation to the integral of the emission intensity $A(x, y)$ over that particular line. If the measured emission sinogram is named $\mathcal{E}(\rho, \theta)$ then

$$\mathcal{E}(\rho, \theta) \approx \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} A(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy T_e \quad (6.7)$$

Again it can be recognized that the recorded array $\mathcal{E}(\rho, \theta)$ is the Radon transform of the emission

intensity $A(x, y)$. Note that this intensity will not be a constant in time as it varies with the metabolism of the brain. This may limit the allowable recording time. Note also that Eq. 6.7 is an approximation, and the approximation is not always very good, due to a very limited number of counts in the individual sinogram bin $\mathcal{E}(\rho, \theta)$. Normally the limited number of counts imply that noise is found in the reconstructed images of the activity of the patient. Nevertheless the approximation is used very often in order to apply the inverse Radon transform for reconstructing the activity pattern, even if this approach can limit the performance of the reconstruction algorithm. Later, in Chapter 9, the ML-EM algorithm is presented, which tries to incorporate the statistical nature of the emission scanning.

So far the absorption of photons in the tissue has been neglected, thus the inversion process must be modified to take this into account. The activity of the brain can be split up in a sum of point sources. Assume a point source is located in a fixed place $(x, y) = (x_0, y_0)$. Assume in the time window T_e that $A(x_0, y_0)$ pairs of photons are produced and emitted along on the line $(\rho, \theta) = (\rho_0, \theta_0)$. Assume the realistic situation that the attenuation cannot be neglected and that attenuation projections $\mathcal{P}(\rho, \theta)$ have been measured using the transmission tomography as discussed in Section 6.1. It can be mentioned that the inverse attenuation coefficient, i.e., μ^{-1} , for 511 keV photons in water, similar to tissue, is about 10 cm, and in a brain with a diameter of 20 cm approximately 86% of all photons are lost due to absorption.

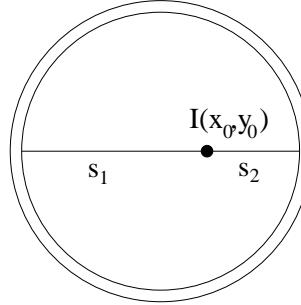


Figure 6.8 Emission of two photons from (x_0, y_0) along the line (ρ, θ) .

Now consider two photons traveling from (x_0, y_0) , as shown in Fig. 6.8. The photon traveling along the line piece s_1 will reach the detector ring with the probability P_1 , given by

$$P_1 = e^{- \int_{s_1} \mu(x, y) ds} \quad (6.8)$$

which is the result found in Eq. 6.4. The other photon traveling in the opposite direction along s_2 will reach the opposite detector with the probability P_2 .

$$P_2 = e^{- \int_{s_2} \mu(x, y) ds} \quad (6.9)$$

The two photons travel independently, which implies that the probability of detecting both photons is the product of P_1 and P_2 . This means that the measured matrix of emissions $\mathcal{E}(\rho, \theta)$ from the point source is given by

$$\mathcal{E}(\rho, \theta) = A(x_0, y_0) T_e P_1 P_2 \quad (6.10)$$

$$= A(x_0, y_0) T_e e^{- \int_{s_1} \mu(x, y) ds} e^{- \int_{s_2} \mu(x, y) ds} \quad (6.11)$$

$$= A(x_0, y_0) T_e e^{- \int_s \mu(x, y) ds} \quad (6.12)$$

where the integral in the last line is the integral of the attenuation coefficient along the line (ρ, θ) , which contains (x_0, y_0) . The last equation illustrates a very important conclusion, namely that attenuation correction in PET does not depend on the initial point of interest (x_0, y_0) , but only on the line parameters (ρ, θ) . If all point sources in the brain are considered, then the measured matrix becomes

$$\mathcal{E}(\rho, \theta) = \int_s A(x_0, y_0) T_e e^{- \int_s \mu(x, y) ds} ds_0 \quad (6.13)$$

$$= e^{- \int_s \mu(x, y) ds} \int_s A(x, y) T_e ds \quad (6.14)$$

$$= e^{-\mathcal{P}(\rho, \theta)} \int_s A(x, y) ds T_e \Rightarrow \quad (6.15)$$

$$\check{A}(\rho, \theta) = \int_s A(x, y) ds \quad (6.16)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} A(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (6.17)$$

$$= \frac{\mathcal{E}(\rho, \theta)}{T_e} e^{\mathcal{P}(\rho, \theta)} \quad (6.18)$$

where s_0 denotes integration with respect to (x_0, y_0) along the line (ρ, θ) . Note that the attenuation factor becomes a constant (along that line), and it can be moved out through the s_0 -integral. Note also that the transmission correction is used in the Radon domain, thus no inversion of $\mathcal{P}(\rho, \theta)$ is needed. The important result is that the attenuation coefficient only produces a multiplicative factor in the Radon domain. This implies that the essential part of producing brain images still can be based on inversion of the Radon transform.

Here it can be mentioned that another related emission scanner, the SPECT-scanner (Single Photon Emission Computed Tomography) operates by measuring one emission photon only and here the attenuation correction is a function of position and line parameters, and not as simple as in PET.

6.2.1 Correction for Attenuation in PET

In order to obtain an estimate of $\mathcal{P}(\rho, \theta)$, the normalization factor I_0 is needed. Up to this point this factor has been assumed to be a constant. This is not true, but this problem is easily corrected by measuring a total of three scans. First an emission scan, and then two scans to correct for attenuation. A blank scan is normally made prior to the two other scans by measuring the transmitted photon intensity or rather the total number of photons transmitted along the line (ρ, θ) for T_b seconds, which gives the blank scan sinogram $\mathcal{B}(\rho, \theta)$. After the patient is placed in the scanner a transmission scan is recorded for T_t seconds (with the same setup used for the blank scan). This gives the transmission sinogram $\mathcal{T}(\rho, \theta)$. Assume that in both scans $I_0(\rho, \theta, x_r, y_r)$ is transmitted per second from the radioactive source along the line with parameters (ρ, θ) to the detector at position (x_r, y_r) . Neglecting the absorption in air, then the measured matrix \mathcal{B} is given by

$$\mathcal{B}(\rho, \theta) = T_b I_0(\rho, \theta, x_r, y_r) e^{- \int \mu_{\text{hardware}} ds} \quad (6.19)$$

where the label ‘hardware’ indicates that some parts of the scanner cannot be avoided when measuring the blank scan along the particular line. This could be the cabinet and the bed for the patient.

With the patient in the scanner, the transmission matrix \mathcal{T} only differs by the absorption in the patient.

$$\mathcal{T}(\rho, \theta) = T_t I_0(\rho, \theta, x_r, y_r) e^{-\int \mu_{\text{hardware}} ds} e^{-\int \mu_{\text{patient}} ds} \quad (6.20)$$

From Eq. 6.19 and 6.20 the attenuation factor is easily found

$$e^{\int \mu_{\text{patient}} ds} = \frac{\mathcal{B}(\rho, \theta) T_t}{\mathcal{T}(\rho, \theta) T_b} \quad (6.21)$$

This implies that the Radon transform of the attenuation correction of the activity measurements, cf. Eq. 6.18, can be carried out in the Radon domain and the very important correction formula is given by

$$\check{A}(\rho, \theta) = \frac{\mathcal{E}(\rho, \theta) \mathcal{B}(\rho, \theta)}{\mathcal{T}(\rho, \theta)} \frac{T_t}{T_e T_b} \quad (6.22)$$

which reveals some potential problems. The sinogram corrected for attenuation \check{A} will be highly dependent on the statistical properties of the three individual scans. Practical aspects of this issue is further discussed in Chapter 11.

6.3 Summary

Some of the fundamental properties of the CT-scanner and the PET-scanner have been presented. It was shown that the CT-scanner can measure projections of the attenuation coefficient, and the measured projections can be modelled as the Radon transform of the attenuation coefficient. This explains why the inverse Radon transform now for many years has been an important tool in medical imaging for reconstructing images of the interior of a function from a set of external measurements. In Chapter 1 of [14] it is shown that several other fields of science, such as astronomy and microscopy also can use exterior measurements and reconstruct the interior function by use of the inverse Radon transform.

Note that the description given in this chapter is somewhat simplified on several points. In practical PET several problems are common such as, scattered photons (scatter), random detection of two photons not originating from the same annihilation process (randoms), geometrical factors, penetration of photons through several detectors before detection (radial elongation), e.g., the chapter about PET in [58].

Chapter 7

Inversion of the Radon Transform

In this chapter several direct inversion schemes are presented for the Radon transform [75]. It can be noted that other inversion schemes are available but perhaps not that used, e.g., [76, 77].

7.1 The Fourier Slice Theorem

Inversion of the Radon transform can be done in several ways. One standard algorithm is based on the Fourier Slice Theorem also known as the Central Slice Theorem [78, 79]. The Radon transform $\check{g}(\rho, \theta)$ is to be inverted into $g(x, y)$, where the inversion is based on the Fourier transform.

First the 2D Fourier Transform of $g(x, y)$ is needed.

$$G(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) e^{-j2\pi(k_x x + k_y y)} dx dy \quad (7.1)$$

and the inverse transform is given by

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.2)$$

Introducing polar frequency parameters

$$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = \nu \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} \quad (7.3)$$

and inserting in Eq. 7.3 into Eq. 7.1 gives

$$\begin{aligned} G(\nu \cos \theta, \nu \sin \theta) &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) e^{-j2\pi\rho\nu} dx dy \right] d\rho \\ &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \right] e^{-j2\pi\rho\nu} d\rho \\ &= \int_{-\infty}^{\infty} \check{g}(\rho, \theta) e^{-j2\pi\rho\nu} d\rho \end{aligned} \quad (7.4)$$

Thus, a one-dimensional Fourier Transform of the Fourier Transform gives the spectrum, which subsequently gives $g(x, y)$, cf. Eq. 7.2. The Fourier Slice Theorem can be summarized as

$$G(\nu \cos \theta, \nu \sin \theta) = \int_{-\infty}^{\infty} \check{g}(\rho, \theta) e^{-j2\pi\rho\nu} d\rho \quad (7.5)$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.6)$$

The Fourier Slice Theorem makes it possible to invert the Radon transform by use of 1D Fourier transforms and 2D Fourier transforms. Note, that the theorem can be used in reverse order to calculate the Radon transform, if the signal $g(x, y)$ is given.

7.2 Filtered Backprojection

Another very famous inversion scheme is the Filtered Backprojection [80, 81, 14, 82] or as it more correctly should have been named Backprojection of Filtered Projections method [14]. It is derived from Eq. 7.2, by introducing polar coordinates

$$\begin{aligned} g(x, y) &= \int_0^{2\pi} \int_0^\infty \nu G(\nu \cos \theta, \nu \sin \theta) e^{j2\pi\nu(x \cos \theta + y \sin \theta)} d\nu d\theta \\ &= \int_0^\pi \int_{-\infty}^\infty |\nu| G(\nu \cos \theta, \nu \sin \theta) e^{j2\pi\nu(x \cos \theta + y \sin \theta)} d\nu d\theta \\ &= \int_0^\pi \int_{-\infty}^\infty |\nu| \left(\int_{-\infty}^\infty \check{g}(\tilde{\rho}, \theta) e^{-j2\pi\tilde{\rho}\nu} d\tilde{\rho} \right) e^{j2\pi\nu(x \cos \theta + y \sin \theta)} d\nu d\theta \end{aligned} \quad (7.7)$$

Eq. 7.7 is commonly written in two parts: A filtering part and an integration part.

$$\check{g}(\rho, \theta) = \int_{-\infty}^\infty |\nu| \left(\int_{-\infty}^\infty \check{g}(\tilde{\rho}, \theta) e^{-j2\pi\tilde{\rho}\nu} d\tilde{\rho} \right) e^{j2\pi\rho\nu} d\nu \quad (7.8)$$

$$= \text{IFT}_{\nu \rightarrow \rho} \{ |\nu| \text{FT}_{\tilde{\rho} \rightarrow \nu} \{ \check{g}(\tilde{\rho}, \theta) \} \} \quad (7.9)$$

$$g(x, y) = \int_0^\pi \check{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (7.10)$$

$$= \int_0^\pi \int_{-\infty}^\infty \check{g}(\rho, \theta) \delta(\rho - x \cos \theta - y \sin \theta) d\rho d\theta \quad (7.11)$$

Note, the similarity of Eq. 2.4 to the forward Radon transform in Eq. 7.10.

The operation shown in Eq. 7.10 is named backprojection and it can be shown, page 120 of [14] that the backprojection operator is nearly linked to the adjoint Radon transform. To be more specific, the adjoint Radon transform is two times the backprojection operator. The concept of the adjoint Radon transform will be used in Chapter 9.

The filtering part in Eq. 7.8 is a forward Fourier transform for each of the angles. In the Fourier domain the signal is high pass filtered with the filter $|\nu|$. Finally an inverse 1D-Fourier transform is used to get the filtered Radon domain $\check{g}(\rho, \theta)$. The backprojection part is merely an integration along a sine-curve in the filtered Radon domain. Eqs. 7.8 and 7.10 form the Filtered Backprojection inversion scheme.

Filtered Backprojection can also be expressed without using filtering in the frequency domain. The filter $|\nu|$ is expressed as a product.

$$|\nu| = \frac{-j \text{ sign}(\nu)}{2\pi} j2\pi\nu \Rightarrow \quad (7.12)$$

$$\check{g}(\rho, \theta) = \text{IFT}_{\nu \rightarrow \rho} \left\{ \frac{-j \text{ sign}(\nu)}{2\pi} j2\pi\nu \text{FT}_{\tilde{\rho} \rightarrow \nu} \{ \check{g}(\tilde{\rho}, \theta) \} \right\} = \frac{1}{2\pi^2 \rho} * \frac{\partial \check{g}(\rho, \theta)}{\partial \rho} \quad (7.13)$$

where $*$ denotes a convolution in the parameter ρ . It should be noted that $|\nu|$ does not fulfill the ordinary requirements for having an inverse Fourier transform, but here the Cauchy principal value has been used in the inverse Fourier integral. Next, the Hilbert transform of a function is introduced.

$$\mathcal{H}g(t) = \frac{1}{\pi} \int_{-\infty}^\infty \frac{g(t')}{t - t'} dt' \Rightarrow \check{g}(\rho, \theta) = -\frac{1}{2\pi} \mathcal{H} \frac{\partial \check{g}(\rho, \theta)}{\partial \rho} \quad (7.14)$$

In total, filtered backprojection can be rewritten on a very compact form resembling the one used by Radon [13].

$$g(x, y) = -\frac{1}{2\pi} \int_0^\pi \left(\mathcal{H} \frac{\partial \check{g}(\rho, \theta)}{\partial \rho} \right)_{\rho=x \cos \theta + y \sin \theta} d\theta \quad (7.15)$$

Direct implementation from Eq. 7.15 is somewhat problematic, because of the Hilbert transformation. The Hilbert transform cannot be implemented easily without using Fourier transformation. If this is the case then the reconstruction formula matches the one given in Eq. 7.8 and 7.10.

7.3 Filtering after Backprojection

It is also possible to make the backprojection, i.e., the adjoint Radon transform before the filtering [14, 73]. In this case another filter must be used. Rewriting $g(x, y)$ using the delta function, cf. Eq. 2.16, gives

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(y - y^*) dx^* dy^* \Rightarrow \quad (7.16)$$

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\rho - x^* \cos \theta - y^* \sin \theta) dx^* dy^* \quad (7.17)$$

Eq. 7.17 shows that each point (x^*, y^*) is transformed into a sine curve in the Radon domain. The Radon transform of $g(x, y)$ is the sum (integral) of all the sine curves in the Radon domain. In the following the integrations over x^* and y^* are omitted. This can be done because only linear transforms are used. Now assume a point source is given and the adjoint Radon transform is applied before a filtering.

$$g(x, y) = \delta(x - x^*) \delta(y - y^*) \Rightarrow \quad (7.18)$$

$$\check{g}(\rho, \theta) = \delta(\rho - x^* \cos \theta - y^* \sin \theta) \quad (7.19)$$

$$\hat{g}(x, y) = \int_0^\pi \check{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (7.20)$$

$$= \frac{1}{|(x - x^*) \sin \theta - (y - y^*) \cos \theta|} \Big|_{(x-x^*) \cos \theta + (y-y^*) \sin \theta = 0} \quad (7.21)$$

$$= \frac{1}{|-(x - x^*) \sin \arctan \left(\frac{x-x^*}{y-y^*} \right) - (y - y^*) \cos \arctan \left(\frac{x-x^*}{y-y^*} \right)|} \quad (7.22)$$

$$= \frac{\sqrt{1 + \left(\frac{x-x^*}{y-y^*} \right)^2}}{\frac{(x-x^*)^2}{y-y^*} + (y - y^*)} = \frac{1}{\sqrt{(x - x^*)^2 + (y - y^*)^2}} \quad (7.23)$$

It can be recognized that this is a two dimensional convolution.

$$\hat{g}(x, y) = g(x, y) * h(x, y) \text{ where } h(x, y) = \frac{1}{\sqrt{x^2 + y^2}} \quad (7.24)$$

Using the technique shown in Eq. 7.17 it can be seen that Eq. 7.24 is valid for any given $g(x, y)$, Eq. 7.24 thus enables another inversion scheme. A two dimensional Fourier transform of Eq. 7.24 gives

$$\hat{G}(k_x, k_y) = G(k_x, k_y) H(k_x, k_y) \Rightarrow \quad (7.25)$$

$$G(k_x, k_y) = \frac{\hat{G}(k_x, k_y)}{H(k_x, k_y)} \quad (7.26)$$

From standard 2D Fourier transform tables it can be found that

$$h(x, y) = \frac{1}{\sqrt{x^2 + y^2}} \leftrightarrow H(k_x, k_y) = \frac{1}{\sqrt{k_x^2 + k_y^2}} \quad (7.27)$$

This means that $g(x, y)$ can be found as

$$\hat{g}(x, y) = \int_0^\pi \check{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (7.28)$$

$$\hat{G}(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \hat{g}(x, y) e^{-j2\pi(k_x x + k_y y)} dx dy \quad (7.29)$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sqrt{k_x^2 + k_y^2} \hat{G}(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.30)$$

The reconstruction method presented is here called Filtering after Backprojection, which makes use of an integration succeeded by a two dimensional high pass filtering. This is an inversion algorithm very similar to Filtered Backprojection.

7.4 Calculation using Operators

A number of textbooks, e.g. [14, 83], define operators. In this section relevant operators are defined, which will enable an easy way to write the inversion algorithms. In the following x , y , k_x , and k_y denotes continuous variables. To identify the continuous operators calligraphic letters are used, e.g., \mathcal{R} .

$$\mathcal{R}g(x, y) = \check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (7.31)$$

$$\mathcal{B}h(\rho, \theta) = \int_0^\pi h(x \cos \theta + y \sin \theta, \theta) d\theta \quad (7.32)$$

$$\mathcal{F}_{p1}^- h(\rho, \theta) = \int_{-\infty}^{\infty} h(\tilde{\rho}, \theta) e^{-j2\pi\tilde{\rho}\nu} d\tilde{\rho} \quad (7.33)$$

$$\mathcal{F}_{p1}^+ H(\nu, \theta) = \int_{-\infty}^{\infty} H(\nu, \theta) e^{j2\pi\rho\nu} d\nu \quad (7.34)$$

$$\mathcal{F}_{r2}^- h(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x, y) e^{-j2\pi(k_x x + k_y y)} dx dy \quad (7.35)$$

$$\mathcal{F}_{r2}^+ H(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} H(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.36)$$

$$\mathcal{C}h(\rho, \theta) = \int_{-\infty}^{\infty} |\nu| \left(\int_{-\infty}^{\infty} h(\tilde{\rho}, \theta) e^{-j2\pi\tilde{\rho}\nu} d\tilde{\rho} \right) e^{j2\pi\rho\nu} d\nu \quad (7.37)$$

$$\mathcal{D}h(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sqrt{k_x^2 + k_y^2} \left(\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(\tilde{x}, \tilde{y}) e^{-j2\pi(k_x \tilde{x} + k_y \tilde{y})} d\tilde{x} d\tilde{y} \right) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.38)$$

Using the symbolic operators it is easy to express the presented inversion algorithms in a short form

Filtered Backprojection

$$\mathcal{B}\mathcal{F}_{p1}^+ |\nu| \mathcal{F}_{p1}^- \check{g} = \mathcal{BC} \check{g} = g(x, y) \Leftrightarrow \mathcal{B}\mathcal{F}_{p1}^+ |\nu| \mathcal{F}_{p1}^- \mathcal{R} = \mathcal{BC}\mathcal{R} = \mathcal{I} \quad (7.39)$$

where \mathcal{I} is the unity operator, i.e., $\mathcal{I}g = g$.

Filtering after Backprojection

$$\mathcal{F}_{2r}^+ \sqrt{k_x^2 + k_y^2} \mathcal{F}_{2r}^- \mathcal{B} \check{g} = \mathcal{DB} \check{g} = g(x, y) \Leftrightarrow \quad (7.40)$$

$$\mathcal{F}_{2r}^+ \sqrt{k_x^2 + k_y^2} \mathcal{F}_{2r}^- \mathcal{B} \mathcal{R} = \mathcal{DB}\mathcal{R} = \mathcal{I} \quad (7.41)$$

Fourier Slice Theorem

$$\mathcal{F}_{r2}^+ \mathcal{F}_{p1}^- \check{g} = g(x, y) \Leftrightarrow \mathcal{F}_{r2}^+ \mathcal{F}_{p1}^- \mathcal{R} = \mathcal{I} \quad (7.42)$$

where a conversion between polar and rectangular frequency coordinates is implied.

7.4.1 The Zero Frequency Problem

In the previous sections a perfect inversion is assumed, e.g., Eq. 7.41, but the backprojection algorithms cannot invert perfectly, due to the filters used in the inversion. Using Filtered Backprojection it is seen that the polar spectrum at zero frequency ($\nu = 0$) is multiplied with zero. This implies that a non-zero mean-value in the sinogram is set to zero in *all* cases, hence the mean value of the reconstructed image need not be correct. Using Backprojection after Filtering, it is obvious that the zero frequency value of the inverted image $g(x, y)$ must become zero. The Fourier Slice Theorem also shows that problem when considering the discrete implementation.

7.5 Sampling Considerations

In the previous sections it was shown that direct reconstruction formulas use the adjoint Radon transform, in form of the backprojection operator, and/or the 1D/2D Fourier transform. This suggests that the presented direct reconstruction schemes can be implemented using a sum approximation for the adjoint Radon transform and FFT/IFFT for the Fourier transforms, but it calls for proper discretization of the continuous formulas and careful selection of the sampling parameters.

In most implementations a linear sampling of all variables is used.

$$\begin{aligned} x &= x_m = x_{\min} + m\Delta x \quad m = 0, 1, \dots, M - 1 \\ y &= y_n = y_{\min} + n\Delta y \quad n = 0, 1, \dots, N - 1 \\ \rho &= \rho_r = \rho_{\min} + r\Delta\rho \quad r = 0, 1, \dots, R - 1 \\ \theta &= \theta_t = \theta_{\min} + t\Delta\theta \quad t = 0, 1, \dots, T - 1 \end{aligned} \quad (7.43)$$

and a centered square image will be assumed

$$\Delta x = \Delta y \quad (7.44)$$

$$M = N \quad (7.45)$$

$$x_{\min} = -x_{\max} = -\frac{(M - 1)}{2}\Delta x \quad (7.46)$$

Note, that this choice will in general not give optimum numerical stability, but implies fast and fairly easy implementations. See [72] concerning a clever use of non-linear sampling of the parameter domain.

Eq. 2.10 gives a choice when choosing the limits of the parameter domain. It is possible to choose $\rho \geq 0$, which normally implies that $0 \leq \theta < 2\pi$. It is also possible to choose $0 \leq \theta < \pi$ and ρ both positive and negative. This choice must be done considering the specific implementation. In the following it is chosen to use $0 \leq \theta < \pi$ and $\theta_{\min} = 0$, which implies that $\Delta\theta = \frac{\pi}{T}$, because this choice matches the limits used in, e.g., Filtered Backprojection and those used in Chapter 2.

When implementing a Radon (or inverse Radon) transform algorithm several things must be fulfilled, to ensure a reasonable performance. Firstly sampling must be adequate in all parameters. This will imply bounds on the sampling intervals. Secondly it is assumed that the fundamental function $g(x, y)$ to be reconstructed have compact support, or more precisely is zero if $\sqrt{x^2 + y^2} > |\rho_{max}|$. This demand will ensure that $\check{g}(\rho, \theta) = 0$, if $|\rho| > |\rho_{max}|$. If this cannot be fulfilled, numerical problems must be expected, because then a numerical implementation does not have all the non-zero information necessary for reconstructing the function.

Assuming that $g(x, y)$ has compact support, then $(x, y) = (0, 0)$ should be placed to minimize $|\rho_{max}|$. This will reduce the size of the data array (with respect to ρ) used for the discrete Radon transform. Normally ρ_r is placed symmetrically around $\rho = 0$, and r is chosen to be odd. In this case

$$\rho_{min} = -\Delta\rho \frac{R-1}{2} \quad (7.47)$$

is used. This choice implies that $\rho_{min} = -\rho_{max}$, like it was chosen in Subsection 2.2.1. The parameter M can then be chosen to fulfill that the whole sinogram is confined in the reconstructed image, in order to reconstruct an image from all of the non-zero parts of the sinogram.

$$\max_{x_m, y_n} \sqrt{x^2 + y^2} = x_{min} \frac{\sqrt{2}}{\Delta x} > \rho_{max} \Rightarrow M > \left\lceil \sqrt{2} \frac{|\rho_{min}|}{\Delta x} \right\rceil + 1 \quad (7.48)$$

From a given sinogram the only parameter still needed to be determined is the sampling distance Δx . Unfortunately the sinc-interpolation strategy used in Subsection 2.2.1 cannot be used. It seems to be difficult to make a series expansion from a finite set in the Radon domain and reconstruct the set using one of the direct reconstruction schemes presented in previous sections in this chapter. Furthermore the optimal choice of image sampling distance Δx will also be determined by additional filters used in the discrete implementation. In general a lower bound is $\Delta\rho = \Delta x$, but if a low-pass filter is used on the sinogram or the structures being scanned are large, then a coarser can be used.

7.6 Inversion of the (p, τ) Radon Transform

In this section inversion formulas are derived for slant stacking. They all are equivalents forms of the ones derived for the (ρ, θ) Radon transform. This section has been included to support Chapter 1, but will not be supported further. The rest of the thesis will focus on the normal Radon transform. For further reading on the use of the inverse (p, τ) Radon transform see [36, 84, 85, 44]. It can be mentioned that the numerical implementation of the inversion schemes will resemble closely the ones presented in Chapter 8.

In this section the fundamental function is $g(x, y)$. Within the seismic literature the function in question normally depend on a distance parameter x and a time t , and the 2D Fourier transform has physical interpretation as a plane wave-expansion. Here the parameters have been chosen according to the rest of the thesis. The Radon transform using parameters (p, τ) , i.e., the slant stack of a function $g(x, y)$ is assumed

$$\check{g}(p, \tau) = \int_{-\infty}^{\infty} g(x, px + \tau) dx \quad (7.49)$$

7.6.1 Fourier Slice Theorem

The derivation of the inversion formula is based on the two dimensional Fourier transform of $g(x, y)$

$$G(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) e^{-j2\pi(k_x x + k_y y)} dx dy \quad (7.50)$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.51)$$

Now two new parameters are introduced. Firstly, the negative ratio between the two frequencies k_x and k_y is denoted p , and secondly the offset τ , given by

$$p = -\frac{k_x}{k_y} \quad (7.52)$$

$$\tau = y - px \quad (7.53)$$

If these are inserted into Eq. 7.50 it is found that

$$G(-pk_y, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, px + \tau) e^{-j2\pi k_y \tau} dx dy \quad (7.54)$$

$$= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} g(x, px + \tau) dx \right] e^{-j2\pi k_y \tau} dy \quad (7.55)$$

$$= \int_{-\infty}^{\infty} \check{g}(p, \tau) e^{-j2\pi k_y \tau} dy \quad (7.56)$$

The last equation can be recognized as the 1D Fourier transform of Radon transform, hence an inversion formula is found by combining Eqs. 7.56 and 7.51

$$G(-pk_y, k_y) = \int_{-\infty}^{\infty} \check{g}(p, \tau) e^{-j2\pi k_y \tau} dy$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (7.57)$$

This Fourier Slice Theorem shows that the function $g(x, y)$ can be reconstructed by mapping the 1D Fourier transform of the Radon transform into the 2D Fourier spectrum of $g(x, y)$. Again the problem regarding high slopes, pointed out in Section 1.7, reappears. If the Radon transform only is given in a certain interval, e.g., between -1 and 1, then only a bow tie shaped part of the 2D Fourier domain domain can be provided from the 1D Fourier transform of the Radon transform, which is shown in Fig. 7.1.

This implies that the function $g(x, y)$ only can be reconstructed if the main Fourier components are concentrated at the k_y -axis. This demand can again be translated into the image domain, demanding that the function cannot have line-like objects with a slope higher than the one used when computing the Radon transform, i.e., if the limiting slope in the parameter domain is chosen to small, then the obvious result is that the function cannot be recovered from its Radon transform.

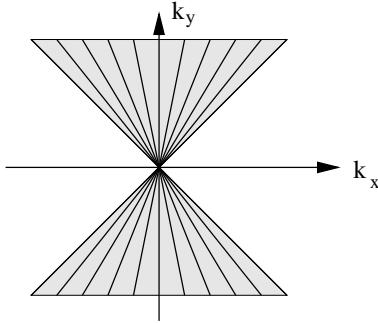


Figure 7.1 From the Radon transform only a bow tie shaped part of the The 2D Fourier domain can be covered.

7.6.2 Filtered Backprojection

Eq. 7.57 can also be combined with Eq. 7.53 in order to get a Filtered Backprojection inversion formula for slant stacking

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(-pk_y, k_y) e^{j2\pi k_y \tau} |k_y| dp dk_y \Big|_{\tau=y-px} \quad (7.58)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |k_y| G(-pk_y, k_y) e^{j2\pi k_y \tau} dk_y dp \quad (7.59)$$

$$= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} |k_y| \left(\int_{-\infty}^{\infty} \check{g}(p, \tilde{\tau}) e^{-j2\pi k_y \tilde{\tau}} d\tilde{\tau} \right) e^{j2\pi k_y \tau} dk_y \right)_{\tau=y-px} dp \quad (7.60)$$

For sake of clarity, Eq. 7.60 is written in two parts; a filtering part and a backprojection part

$$\check{g}(p, \tau) = \int_{-\infty}^{\infty} |k_y| \left(\int_{-\infty}^{\infty} \check{g}(p, \tilde{\tau}) e^{-j2\pi k_y \tilde{\tau}} d\tilde{\tau} \right) e^{j2\pi k_y \tau} dk_y \quad (7.61)$$

$$g(x, y) = \int_{-\infty}^{\infty} \check{g}(p, y - px) dp \quad (7.62)$$

where the last equation also is called the backprojection operator.

This Filtered Backprojection formula for computing the inverse Radon transform implies that the parameter domain is filtered with the absolute frequency in the y -direction for all values of x , and then the backprojection part integrates up along a line. Note that the linear expression in the backprojection integral is like the one found with the point source, i.e., the point sources have by the Radon transform been spread into the parameter domain, and by the backprojection part, the filtered parameter domain is collected again.

7.6.3 An Inversion Formula using the Hilbert Transform

The next inversion formula mentioned here is on the same form as Radon derives in [13]. Eq. 7.60 is now written using that the Fourier transform of a product becomes a convolution (here marked by $*$)

$$\begin{aligned} g(x, y) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |k_y| G(-pk_y, k_y) e^{j2\pi k_y \tau} dp dk_y \text{ where } \tau = y - px \\ &= \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} |k_y| e^{j2\pi k_y \tau} dk_y * \int_{-\infty}^{\infty} G(-pk_y, k_y) e^{j2\pi k_y \tau} dk_y \right] dp \\ &= \int_{-\infty}^{\infty} \frac{-1}{2\pi^2 \tau^2} * \check{g}(p, \tau) dp \end{aligned} \quad (7.63)$$

where it again should be noted that $|k_y|$ does not fulfill the ordinary requirements for having an inverse Fourier transform, but here the Cauchy principal value has been used in the inverse Fourier integral. The result can be found in, e.g., [86] (page 13.2 F26). Now the expressions are rearranged furthermore by inserting the convolution

$$g(x, y) = \int_{-\infty}^{\infty} \frac{-1}{2\pi^2 \tau^2} * \check{g}(p, \tau) dp \text{ where } \tau = y - px \quad (7.64)$$

$$= \frac{-1}{2\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\check{g}(p, \tilde{\tau})}{(\tau - \tilde{\tau})^2} d\tilde{\tau} dp \quad (7.65)$$

$$= \frac{1}{2\pi^2} \frac{d}{d\tau} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\check{g}(p, \tilde{\tau})}{\tau - \tilde{\tau}} d\tilde{\tau} dp \quad (7.66)$$

$$= \frac{1}{2\pi^2} \frac{d}{dt} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\check{g}(p, \tilde{\tau})}{\tau - \tilde{\tau}} d\tilde{\tau} dp \quad (7.67)$$

$$= \frac{1}{2\pi^2} \frac{d}{dy} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\check{g}(p, \tilde{\tau})}{y - px - \tilde{\tau}} d\tilde{\tau} dp \quad (7.68)$$

$$= \frac{1}{2\pi^2} \frac{d}{dy} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\check{g}(p, \tilde{y} - px)}{y - \tilde{y}} dy dp \quad (7.69)$$

The last equation can be written on a shorter form by use of the Hilbert transform

$$\mathcal{H}\{f(x, y)\} = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{f(x, \tilde{y})}{y - \tilde{y}} d\tilde{y} \Rightarrow \quad (7.70)$$

$$g(x, y) = \frac{1}{2\pi} \frac{d}{dy} \mathcal{H} \left\{ \int_{-\infty}^{\infty} \check{g}(p, \tilde{y} - px) dp \right\} \quad (7.71)$$

This equation is very concise, but not that easy to implement on a discrete form. The inner part of the inversion formula can be discretized by use of Section 1.4. The remaining part requires a discrete approximation of the Hilbert transform and the derivative operator. Note, that the Hilbert transform has a singularity which makes the discrete implementation difficult.

7.6.4 Filtering after Backprojection

The last inversion form is equivalent to Eq. 7.71. A way to circumvent the implementation problems with the Hilbert transform is to note that Eq. 7.71 can be written in two stages

$$\check{g}(x, y) = \int_{-\infty}^{\infty} \check{g}(p, y - px) dp \quad (7.72)$$

$$g(x, y) = \frac{1}{2\pi} \frac{d}{dy} \mathcal{H} \{ \check{g}(x, y) \} \quad (7.73)$$

and using that the last equation is a convolution, then the inversion scheme becomes a filter in the Fourier domain.

$$\begin{aligned} \text{The Hilbert transform: } & \frac{1}{\pi y} \leftrightarrow -j \operatorname{sign}(k_y) \\ \text{The derivative operator: } & \frac{dq(y)}{dy} \leftrightarrow j2\pi k_y G(k_y) \end{aligned} \} \Rightarrow \quad (7.74)$$

$$g(x, y) = \int_{-\infty}^{\infty} |k_y| \left[\int_{-\infty}^{\infty} \check{g}(x, y) e^{-j2\pi k_y y} dy \right] e^{j2\pi k_y \tau} dk_y \quad (7.75)$$

where $\operatorname{sign}(\cdot)$ is the sign function. The conclusion is that the inversion formula proposed by Radon can be interpreted as a backprojection followed by a filtering operation. A very important feature is that the reconstructed zero frequency component value becomes zero for any fixed value of x .

7.7 Summary

In this section several formulas for computing the inverse Radon transform have been derived. It has been shown that they can be based on Fourier techniques and integral transformations. Formulas are provided both using the normal Radon transform and for the slant stack definition. Some remarks have been given, regarding that the inverse Radon transform can not recover the mean value correctly.

Considerations concerning the sampling of the continuous parameters are given, which will be used in the following chapters.

Chapter 8

Numerical Implementation of Direct Reconstruction Algorithms

Several authors discuss implementations of the direct reconstruction schemes presented in Chapter 7, e.g., [81, 78, 79, 16]. In this chapter the implementation of Filtered Backprojection, Filtering after Backprojection and the Fourier Slice Theorem are shown. All of these algorithms can be based on the Fourier transform, which is reviewed in the next section. This theory might be considered basic, but it appears that applying the DFT as a discrete approximation to the Fourier transform is requires special care with phase and normalization.

8.1 Using the DFT to Approximate the Fourier Transformation

All of the presented direct reconstruction schemes use Fourier transformation, either for filtering or for re-mapping of spectra (Fourier Slice Theorem). For digital signals the discrete Fourier transform (DFT) can be used to estimate the spectrum, and in practice the spectrum is estimated by the Fast Fourier Transform (FFT).

Assume that the one dimensional function, of a continuous variable t , denoted $g(t)$ is sampled uniformly, i.e., $g_s(n) = g(n\Delta t)$. Furthermore, assume that only N values are non-zero. In this case the Fourier transform can be approximated by the DFT

$$G(f) = \int_{-\infty}^{\infty} g(t) e^{-j2\pi ft} dt \approx \Delta t \sum_{n=0}^{N-1} g_s(n) e^{-j2\pi mn/N} = G_s(m) \quad (8.1)$$

hence, $G_s(m)$ will approximate the continuous spectrum

$$\Delta t G_s(m) \approx G\left(\frac{m}{N\Delta t}\right) \quad (8.2)$$

The discrete spectrum is in Eq. 8.1 computed using the DFT of $g_s(n)$. With the same approach the inverse Fourier transform is commonly approximated by

$$g_a(t) = \int_{-\infty}^{\infty} G_a(f) e^{j2\pi ft} df \approx \frac{1}{N\Delta t} \sum_{m=0}^{N-1} G_d(m) e^{j2\pi mn/N} = g_d(n) \quad (8.3)$$

This technique can easily be generalized to two or more dimensions. Note that the inverse Fourier transform can also be implemented using the FFT [87].

When approximating the Fourier transform by the DFT it should be noted that the spectrum is available only in discrete samples and the samples represents the Fourier transform of a periodical repetition of the discrete signal $g_s(n)$ with period N.

Furthermore a very important factor is that the discrete spectrum as a function of m also is periodical with period N, i.e.,

$$G_s(m) = G_s(m + N) \quad (8.4)$$

This implies that the maximum absolute frequency corresponds to $m = N/2 \Rightarrow f_u = \frac{1}{2\Delta t}$, i.e., the upper frequency f_u equals half of the sampling frequency. Due to the periodical behavior of the discrete spectrum, the last half of the digital spectrum, i.e., from $m = N/2$ to $m = N - 1$ will correspond to negative frequencies.

8.1.1 The FFT Applied for Filtering

Now the practical use of the DFT for implementation of the operator \mathcal{F}_{p1}^- will be discussed. It is very important to note that the DFT uses an array of signal values $g(n)$, $n = 0, 1, \dots, N - 1$ where the last half of the array corresponds to negative continuous variable. In this case the relevant signal is the discrete samples of the sinogram for a certain value of θ_t , i.e., $h(r) = \check{g}(\theta_t, \rho_r)$, where the values of ρ lies symmetrically around 0, cf. Eqs. 7.43 and 7.47. In order to get the phase of the spectrum correct, the array $g(n)$ used for the DFT must be filled as shown in Fig. 8.1. Note the wrapping so negative values of ρ are filled into the last half of the array. It is also assumed that the DFT transformation length is a power of two in order to use one of the fast radix-2 FFT algorithms. For the unknown entries in the middle of the array of $g(n)$ zeros must be filled in. This operation is called zero padding and this action will affect the spectrum.

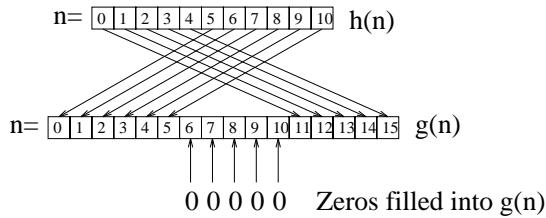


Figure 8.1 Filling an array $h(n)$ into a larger array $g(n)$, when using radix-2 FFT.

A large number of zeros gives a more smooth spectrum (due to a denser sampling in frequency domain), which can be desired if interpolation in the spectrum is required. Thus, the extra zeros padded can help to improve the numerical stability. In Fig. 8.2 a square signal is shown in the left upper corner, and the corresponding spectrum in the right upper corner. Zeros have been padded, and again the spectrum has been found. This is shown on the lower part of the figure. Note, that the new spectrum is more smooth.

As it briefly has been described, the use of DFT to approximate the Fourier transform also implies that the signal becomes cyclical with the period of the transformation length. This can lead to problems, as it later will be demonstrated in Subsection 8.5.1. A common strategy to reduce this cyclical influence is to multiply the signal with a window, i.e., a weight function attenuating the edges of the signal.

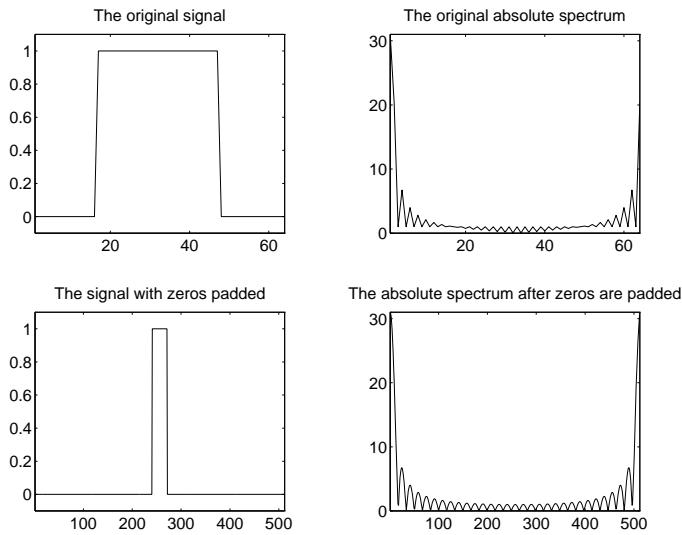


Figure 8.2 Upper left shows a square and the upper right shows the corresponding absolute spectrum. The frequency ranges from 0 to the sampling frequency (last half is the negative frequencies). Lower left shows the square where zeros have been padded, and lower right shows the corresponding absolute spectrum, which appears more smooth.

A property, which can be used to reduce the computational cost is based on real (non-complex) valued signals. Assume a discrete cyclical real valued signal $g(n)$

$$G(m) = G(-m)^* = G(N - m)^* \quad (8.5)$$

where $*$ denotes the complex conjugate and $g(n) = g(n + N)$. This symmetry is easily proved from the DFT definition shown in Eq. 8.1, and it should be used if only a real signal is provided, such as a sinogram which should be filtered (without complex values). In this way the length of the FFT needed can be reduced by a factor of two, or two real valued signals can be transformed with the same FFT [88, 89].

Numerical algorithms are available in virtually any numerical package for efficient calculation of the FFT. Several packages furthermore also includes functions for calculating the DFT of a real valued sequence $g(n)$ of length $N = 2^p$, e.g., Numerical Recipes [90]. Often the FFT-algorithm is a radix-2 algorithm. This restriction is for reconstruction purposes not harsh, but must be remembered when zeros are padded.

In the implementation of Filtered Backprojection a 1D filtering of the sinogram is needed, where the filter is the absolute of the frequency parameter, i.e., $|\nu|$, cf. Eq. 7.8. This filter is approximated in many ways, but the structure of the FFT based filtering is the same, as shown in Algorithm 8.1. The algorithm does not demonstrate the implementation of the extra speedup gained by exploiting Eq. 8.5, hence the array $g(n)$ has complex entries. Note also that the algorithm does not split the signal values as it was shown in Fig. 8.1, because the filtering does not use the actual phase of the spectrum. What matters, is that the filtered result is extracted from the same positions in the array, and is returned in the original sinogram ($g_radon(t,r)$), which is done for sake of memory efficiency.

The filter calculated in line three of Algorithm 8.1 (sampling of Eq. 8.2) is called the ramp filter or Ram-Lak filter. Often it is multiplied with a weight function or simply another function in order to get a better signal to noise ratio. The only item all of the filters have in common is that they approximate $|\nu|$ at low frequencies, and the difference is pronounced as the frequency ν approaches half of the sampling frequency, denoted by $\nu_u = \frac{1}{2\Delta\rho}$.

ALGORITHM 8.1 : FILTERING USING DFT

```

Set P to upper power of two ( $P \geq R$ )           //Assuming radix-2 FFT
For r = 1 to P/2                                  //Initialize Filter
   $f(r) = r / (\Delta\rho * P)$                    //Approximate filter
End
For t = 0 to T-1                                  //For all angular samples
  For r = 0 to R-1                                //For the radial samples
     $g(r) = g_{\text{radon}}(t, r)$                  //Move the sinogram values
  For r = R to P-1                                //For padding
     $g(r) = 0$                                  //Pad with zeros
  End
  Compute FFT of  $g(n)$                          //Using radix-2 FFT
   $g(0) = 0$                                  //Handle zero freq. specially
   $g(P/2) = g(P/2) * f(P/2)$                   //Handle half sampling freq.
  For r = 1 to P/2-1                            //Multiply with filter
     $g(r) = g(r) * f(r)$                       //Positive frequency
     $g(P-r) = g(P-r) * f(r)$                 //Negative frequency
  End
  Compute Inverse FFT of  $g(r)$                 //Using radix-2 FFT
  For r = 0 to R-1                                //For the radial samples
     $g_{\text{radon}}(t, r) = \text{real}(g(r))$  //Move the real-part
  End
End

```

Some of the filters reported in the literature [73, 58] are given below, where it is only the the part below a certain limit frequency $\nu_l \leq \nu_u$, which is sampled and used. The reason for using these weight functions also called windows or apodizing functions is to suppress the influence of noise. It is obvious that the filter $|\nu|$ is a high pass filter and it will attenuate any noise present in the sinogram. Examples of this property will be given in Subsection 8.5.5. Many windows can be presented, but here only four examples are given

The Cropped Ram-Lak/Ramp filter Sample the filter $|\nu|$ until ν_l , i.e., the filter is

$$H(\nu)_{\text{Ram-Lak}} = |\nu| \quad (8.6)$$

and for $\nu_l < |\nu| < \nu_u$ the filter is set to zero. The ramp filter is widely used but will amplify noise if ν_l is chosen too high.

The Shepp-Logan filter A sinc window is multiplied to the ramp filter

$$H(\nu)_{\text{Shepp-Logan}} = |\nu| \frac{1}{2} \frac{\sin\left(\frac{\pi\nu}{2\nu_l}\right)}{\left(\frac{\pi\nu}{2\nu_l}\right)} \quad (8.7)$$

and for $\nu_l < |\nu| < \nu_u$ the filter is set to zero.

The Hann filter A Hann window is multiplied to the filter

$$H(\nu)_{\text{Hann}} = |\nu| \left(1 + \cos\left(\frac{\pi\nu}{\nu_l}\right)\right) \quad (8.8)$$

and for $\nu_l < |\nu| < \nu_u$ the filter is set to zero.

The Generalized Hamming filter

$$H(\nu)_{\text{Generalized Hamming}} = |\nu|(\alpha + (1 - \alpha)\cos(\pi\nu/\nu_u)) \quad (8.9)$$

where typical values of α are $0.5 - 0.54$, cf. page rec-22 of [58]. Again for $\nu_l < |\nu| < \nu_u$ the filter is set to zero.

Stochastic Filters In Section 10.8 of [73], Jain derives a parameterized filter, which is shaped to the actual noise level.

Fig. 8.3 shows three of the filters. They can all be written as a product of the theoretical derived ramp filter and an apodizing window, which will influence the performance in presence of noise. In general, if the apodizing window have a low cutoff frequency the resolution gets worse, but the noise suppression can be improved.

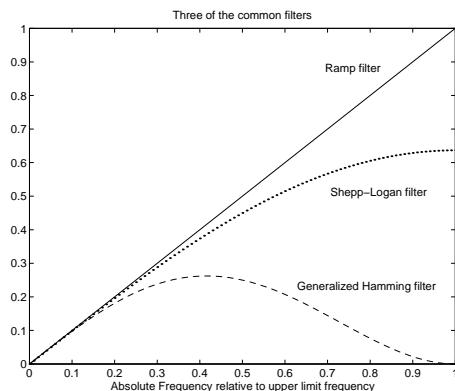


Figure 8.3 The amplitude of the Ram-Lak/Lamp filter, the Shepp-Logan filter, and the generalized Hamming filter using $\alpha = 0.5$, all three as a function of frequency normalized to the upper limit frequency ν_u .

In the top part Fig. 8.4 is shown the impulse response from the ramp filter. Note that the lower sub-figure uses logarithmic scale. The figure shows that the impulse response has long tails which implies that a number of zeros, in principle an infinite number, have to be padded to the signal or else the cyclical behavior of the DFT can influence the filtered signal.

Fig. 8.5 shows filtering of a sinogram corresponding to a circular disc in the image domain. Here a Hann window has been multiplied to the ramp filter. This filtered sinogram will later, in Fig. 8.7, be backprojected to demonstrate the full reconstruction algorithm of Filtering after Backprojection. From Fig. 8.5 it can be seen that the spectrum is very localized around zero frequency, and the high-pass filter will amplify the edges and from the last sub-figure, it can be seen that significant negative values are found. A small remark is that the data representation in these algorithms should include a sign bit.

It should be mentioned that the complexity of filtering the sinogram is the number of angular samples times the complexity of the FFT operations (plus some lower order terms)

$$\mathcal{O}_{\text{Filtering}} = \mathcal{O}(T R \log R) \quad (8.10)$$

where R should be replaced by the smallest power of two larger or equal to R if a radix-2 FFT is used.

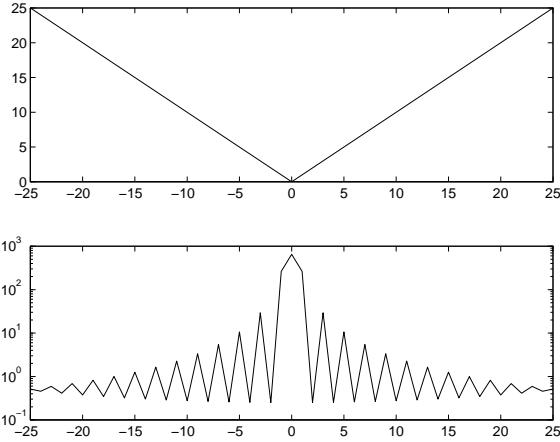


Figure 8.4 Upper shows 51 samples of a ramp filter as a function of frequency. Upper right shows the absolute value of the corresponding spectrum, found from a DFT of the same length. Here no windowing has been used to reduce the cyclical behaviour of the DFT.

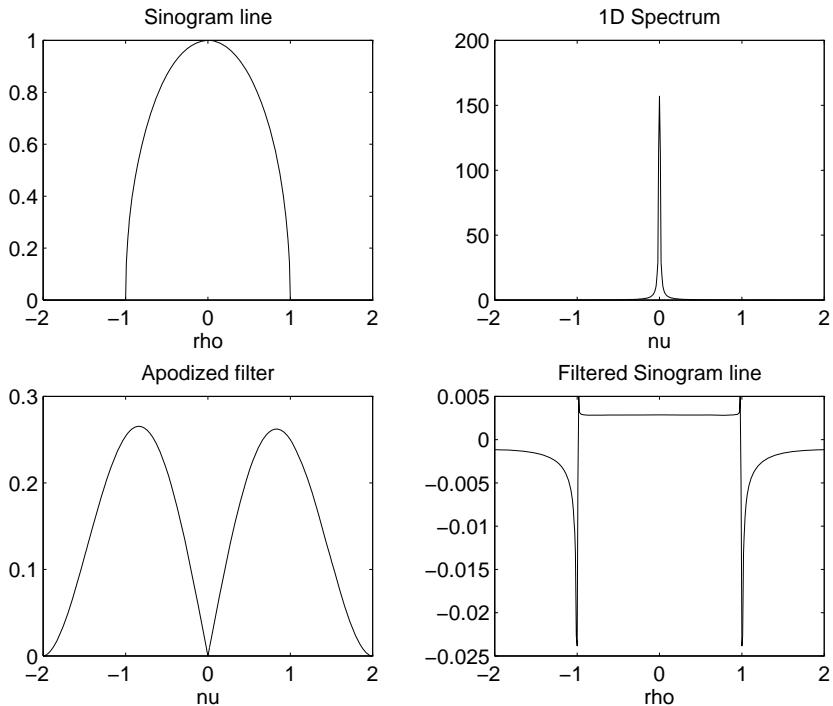


Figure 8.5 Upper left shows the sinogram for a fixed value of θ and varying ρ . Upper right shows the corresponding discrete spectrum, and it can be noted that there is a heavy low frequency dominance. Lower left shows the filter, which here is the ramp multiplied with a Hann window. Lower right shows the filtered sinogram part.

The 1D filtering could also have been done by convolving the sinogram with the proper impulse response, which is infinitely long as indicated in Fig. 8.4, hence windowing is needed. This approach is fast if the impulse response is truncated into a short signal, which will somewhat sacrifice the performance in the frequency domain. This implementation of filtering of the sinogram followed by backprojection is known as convolution backprojection.

8.2 Discrete Implementation of Backprojection

Filtered Backprojection is the easiest inversion scheme to implement. An algorithm based on Filtered Backprojection will have two parts: A filtering part and an integration part. The filtering part has been covered in Subsection 8.1.1 and in this section the implementation of the backprojection part is described.

The backprojection operator was found in Eq. 7.10

$$g(x, y) = \int_0^\pi \bar{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (8.11)$$

where \bar{g} is the filtered sinogram in case of Filtered Backprojection and the original sinogram in Filtering after Backprojection.

A commonly used approximation of Eq. 8.11 is

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} \bar{g}(x_m \cos \theta_t + y_n \sin \theta_t, \theta_t) \quad (8.12)$$

where a one-dimensional interpolation must be used in the ρ -direction. Normally either a nearest neighbour approximation or a linear interpolation is incorporated. Now the discrete indices of the sinogram are used.

Nearest neighbour approximation

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} \bar{g}([r^*], t), \text{ where } r^*(m, n; t) = \frac{x_m \cos \theta_t + y_n \sin \theta_t - \rho_{min}}{\Delta\rho} \quad (8.13)$$

Linear Interpolation

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} (1-w)\bar{g}(r_l, t) + w\bar{g}(r_l + 1, t) \quad (8.14)$$

$$\text{where } r^*(m, n; t) = \frac{x_m \cos \theta_t + y_n \sin \theta_t - \rho_{min}}{\Delta\rho} \quad (8.15)$$

$$r_l = \lfloor r^*(m, n; t) \rfloor \text{ and } w = r^* - r_l \quad (8.16)$$

Higher order interpolation is seldom used, due to the increased accuracy will normally not match the increase in computational cost. See also the discussions in Chapters 1 and 2 concerning interpolation. The complexity of the backprojection is given by the image size times the number of angular samples

$$\mathcal{O}_{\text{backprojection}} = \mathcal{O}(M^2 T) \quad (8.17)$$

traditionally the backprojection part has been considered to be far more computationally expensive than the 1D filtering, but the backprojection part can be optimized so the two operations use similar amount of time for relevant sized sinograms. This will naturally also depend on the actual hardware used for reconstruction.

From Eqs. 8.13 and 8.14 it can be seen, that the evaluation of the function $r^*(m, n; t)$ should be optimized, which can reduce the computational cost drastically. Note also the similarity to (ρ, θ) implementation of the Hough transform found in Section 3.2, where a single signal value was distributed along the sinusoid, here all the signal values along the same sinusoid are collected, and it is relevant to the results found in Subsection 3.2.2. Table 3.4, where a full image has to

be transformed, showed that time consuming initialization of one- and two-dimensional arrays can be tolerated, if the mapping procedure is very fast. Using the optimization technique shown in Case 4 in Subsection 3.2.2 implies that the backprojection operator can be implemented effectively as shown in Algorithms 8.2 and 8.3. Here the pseudo code has been split in the initialization part and the true backprojection part in order to get a better overview, but also to indicate that the initialization part for a given parameter setup can be computed once and only the backprojection part shown in Algorithm 8.3 reused many times in many images with the same parameter setup should be reconstructed. This is the case in multi slice 2D reconstruction of PET images.

ALGORITHM 8.2 : INITIALIZATION BEFORE BACKPROJECTION

```

rhooff=rho_min/Delta_rho          //Compute offset
For t = 0 to T-1                  //For all values of theta
    theta = t*Delta_theta          //theta is computed
    costheta(t) = cos(theta)       //cos(theta) is stored
    sintheta(t) = sin(theta)       //sin(theta) is stored
End
For m = 0 to M-1                  //For all values of x
    xrel = (x_min + m*Delta_x)/Delta_rho //compute x
    For t = 0 to T-1                  //For all values of theta
        xc(m,t)=xrel*costheta(t)      //Store x times cos theta
        ys(m,t)=xrel*sintheta(t)-rhooff //Store y times sin theta
    End
End

```

ALGORITHM 8.3 : FAST BACKPROJECTION

```

For m = 0 to M-1                  //For all values of x
    For n = 0 to M-1              //For all values of y
        sum = 0                   //Initialize simple variable
        For t = 0 to T-1           //For all values of theta
            rm = xc(m,t)+ys(n,t)  //Compute non-integer index
            rl = floor(rm)         //Find lower integer
            w = (rm-rl)             //Compute weight
            sum = sum +(1-w)*g_radon(t,rl)+w*g_radon(t,rl+1)
        End                         //Linear interpolation finished
        g(m,n)=sum*Delta_theta
    End
End

```

Note that in Algorithm 8.3 it is not checked whether the value of rl corresponds to pixels in the image or not, i.e., $0 \leq rl < R - 1$. This operation is very time consuming as it is evaluated in the most inner core of the loop. Checking can be avoided, if the initial sinogram is expanded in size in the ρ -direction by padding zeros, in order to fulfill

$$0 < \frac{x_m \cos \theta_t + y_n \sin \theta_t - \rho_{min}^*}{\Delta \rho} < R - 1 \quad \forall (x_m, y_n, \theta_t) \Rightarrow R^* > \sqrt{2}(M - 1) \frac{\Delta x}{\Delta \rho} + 1 \quad (8.18)$$

where R^* is the number of samples required in the new sinogram, when using the same sampling

interval $\Delta\rho$, and also adjusting the value of ρ_{min} to maintain a symmetrical sampling of ρ

$$\rho_{min}^* = -\Delta\rho \frac{R^* - 1}{2} \quad (8.19)$$

The given implementation shown in Algorithms 8.2 and 8.3 will require two matrices of size MT and the computational load is lowered significantly, cf. Table 3.4 for the implementation of the Hough transform.

The discrete implementation of Filtered Backprojection is schematically shown in Fig. 8.6 using first filtering and then backprojection.

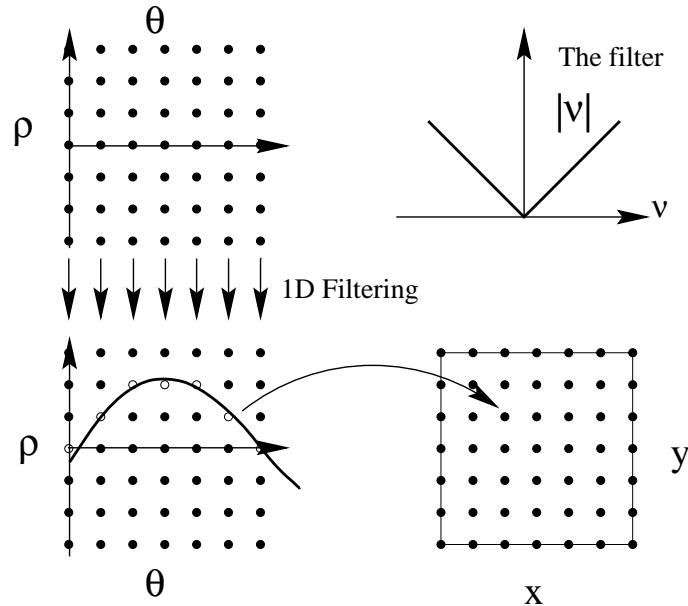


Figure 8.6 The discrete implementation of Filtered Backprojection. At the left the sinogram is filtered and then the filtered sinogram is by backprojection mapped into the reconstructed image.

Now a set of images are shown where a synthetic sinogram, corresponding to a circular disc in the image domain, must be reconstructed using Filtered Backprojection with an increasing number of angular samples. Fig. 8.5 showed the filtering of the sinogram, and here the (rotational symmetrical) disc is placed in the middle of the coordinate system, hence each of one-dimensional the filtered sinogram does not depend on θ_t , cf. Eq. B.29. From 5 angular samples in the sinogram, i.e., $T = 5$, Fig. 8.7 demonstrates the reconstructed image using Filtered Backprojection. The figure clearly shows how the 5 sinogram parts are backprojected into the image, and more angular samples are obviously needed. In Fig. 8.8, only 10 angular samples was used, and the reconstructed shape of the disc is much better recovered, but large artifacts are still very visible outside the disc. Increasing to 20 angular samples, as shown in Fig. 8.9, the artifacts are reduced in amplitude compared to Fig. 8.8, and with 100 angular samples, shown in Fig. 8.10, the disc is nearly recovered perfectly.

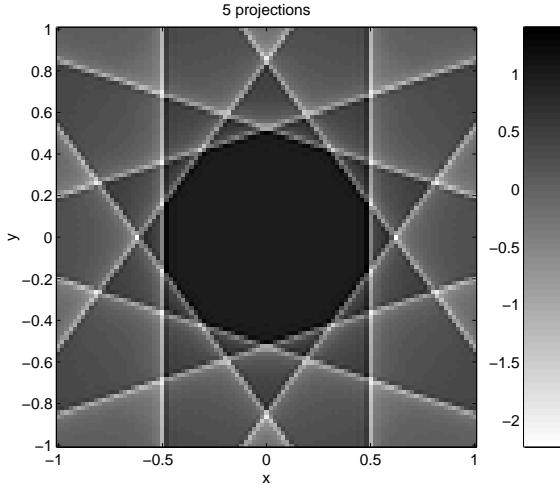


Figure 8.7 Filtered Backprojection from a sinogram with 5 angular samples ($T=5$).

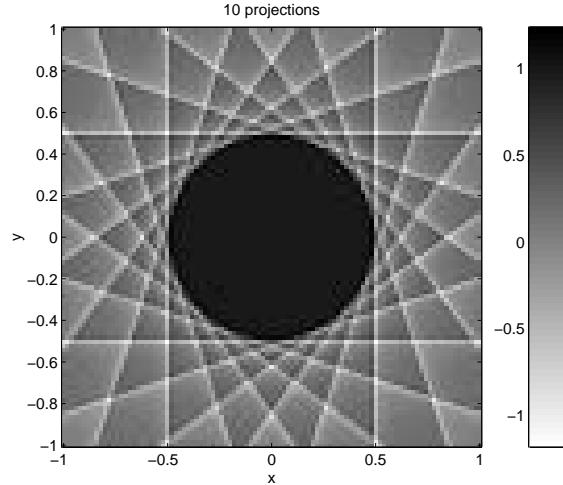


Figure 8.8 Filtered Backprojection from a sinogram with 10 angular samples ($T=10$).

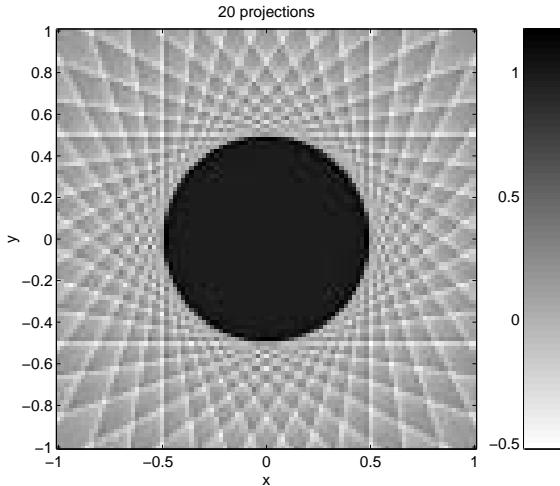


Figure 8.9 Filtered Backprojection from a sinogram with 20 angular samples ($T=20$).

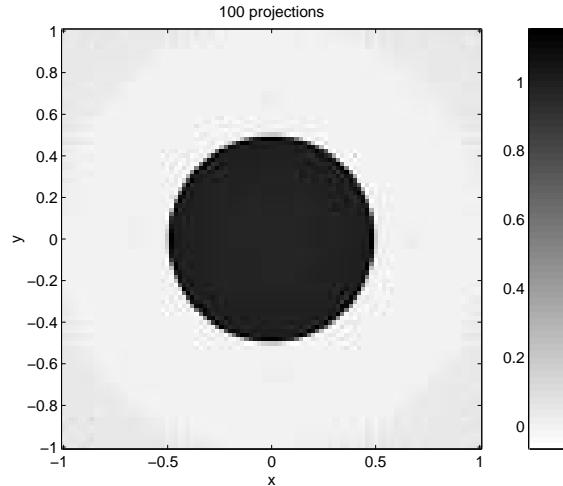


Figure 8.10 Filtered Backprojection from a sinogram with 100 angular samples ($T=100$).

8.3 Implementation of Filtering after Backprojection

Implementation of Filtering after Backprojection requires a discrete implementation of the backprojection operator, as shown in Section 8.2. After the backprojection the matrix $g(m, n)$ must be high pass filtered cf. Eq. 7.30. The implementation resembles the one shown in Subsection 8.1.1, but is extended to two dimensions.

The spectrum of an image can be obtained in two ways. Either by using one of the multidimensional FFT algorithms, e.g., [90], or by using a one dimensional FFT on first all the rows and then all the columns of the image, which is possible because the discrete spectrum can be written

$$G(u, v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) e^{-j2\pi(mu/M+nv/N)} \quad (8.20)$$

$$= \sum_{m=0}^{M-1} \left[\sum_{n=0}^{N-1} g(m, n) e^{-j2\pi mu/M} \right] e^{-j2\pi nv/N} \quad (8.21)$$

where both image sizes M and N must be powers of two, if using a radix-2 FFT. If this is not true extra pad image samples at the edges. For filtering the additional samples should not be can be padded with zeros. Assume that the reconstructed image should look like a disc. Then the backprojected sinogram (into the image domain) will have large non-zero values away from the disc, due to the convolution shown in Eq. 7.24. This implies that the filtering in the image domain will meet problem with the cyclical behavior of the DFT (or FFT). These edge problem must be solved by backprojecting onto a larger image than necessary (if using radix-2 FFT often to the nearest upper power of 2), and then filter the expanded image, and cropping values of corresponding to the specified image size.

Note that the image is considered periodical and the spectrum will have a complex conjugate symmetry for a real valued signal, as shown in Fig. 8.11.

$$g(m, n) = g(m + M, n) = g(m, n + N) \quad (8.22)$$

$$G(u, v) = G(m + M, n) = G(m, n + N) \quad (8.23)$$

$$g(m, n) = g(m, n)^* \Rightarrow G(u, v) = G(-u, -v)^* = G(M - u, N - v)^* \quad (8.24)$$

The symmetry of the spectrum can be exploited for faster computation of the spectrum, but the symmetry can not easily be exploited for reducing the memory requirements due to a rather odd storage strategy needed.

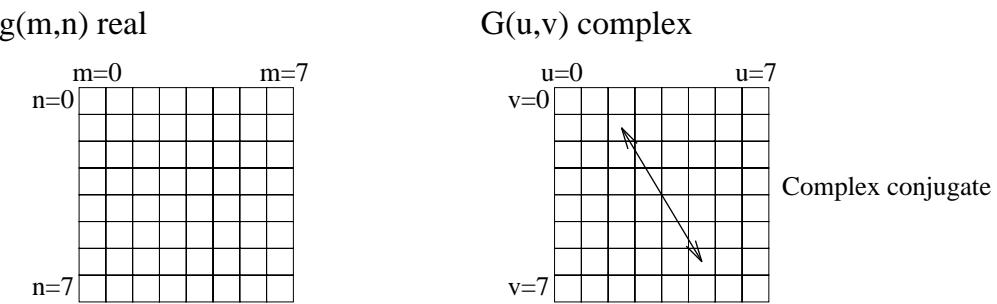


Figure 8.11 A real valued image gives a spectrum with complex conjugate pairs.

The 2D-filtering is very easy if the spectrum is calculated properly. The complex spectrum can easily be multiplied by a sampled version of the filter $\sqrt{k_x^2 + k_y^2}$.

$$k_x \rightarrow \frac{u}{M\Delta x} \text{ and } k_y \rightarrow \frac{v}{N\Delta x} \quad (8.25)$$

$$\sqrt{k_x^2 + k_y^2} \rightarrow \sqrt{\left(\frac{u}{M\Delta x}\right)^2 + \left(\frac{v}{N\Delta x}\right)^2} \quad (8.26)$$

If addressing the negative frequencies $k_x < 0$, u must be replaced by $u - M$ and if $k_y < 0$, v must be replaced by $v - N$. Symmetry can be used so approximately half of the complex spectrum is multiplied with the filter and the other half is duplicated from the first due to the complex conjugate symmetry. After the multiplication of the filter, the inverse two dimensional DFT (or rather FFT) is used, and the real part of the result is extracted, and the imaginary part should be zero.

The 2D DFT implementation of the high-pass filter should also incorporate multiplication by an apodizing window, in order to reduce the edge effects, due to the periodical behavior of the spectrum. Of the huge amount of windows available, two relevant choices of windows should be mentioned.

Cropped 2D Ramp filter Here the theoretically derived 2D ramp filter is cropped at a certain frequency k_l

$$H(k_x, k_y) = \begin{cases} \sqrt{k_x^2 + k_y^2} & \text{if } \sqrt{k_x^2 + k_y^2} < k_l \\ 0 & \text{else} \end{cases} \quad (8.27)$$

where k_l is set below the upper limit frequency in one of the directions, i.e., $k_l < \frac{1}{2\Delta x}$.

Hanning Window The 2D ramp filter could also be multiplied by a Hanning window

$$H(k_x, k_y) = \begin{cases} \frac{1}{2}\sqrt{k_x^2 + k_y^2} \left(1 + \cos\left(\pi \frac{\sqrt{k_x^2 + k_y^2}}{k_l}\right) \right) & \text{if } \sqrt{k_x^2 + k_y^2} < k_l \\ 0 & \text{else} \end{cases} \quad (8.28)$$

where k_l again is set below the upper limit frequency in one of the directions, i.e., $k_l < \frac{1}{2\Delta x}$.

The two windows both use a cutoff frequency k_l , which can be varied, depending on the noise-level. A high noise level might call for a low value for k_l , which implies that the reconstructed image will be somewhat blurred.

Note again that the mean value of the reconstructed image will always be set to zero. This value might be estimated in an area of the reconstructed image where some prior knowledge implies that the value should be, e.g., zero. In brain tomography the relevant area could be outside the brain.

8.4 Implementation of The Fourier Slice Theorem

The basis of the Fourier Slice Theorem is given in Section 7.1. In the implementation, the discrete spectrum of the sinogram is calculated for each of the angular samples like it was shown in Sub-section 8.1.1. Note that here the phase is important, hence the shifting shown in Fig. 8.1 must be considered. This spectrum is considered as polar samples, and must be mapped onto a quadratic frequency grid, as shown in Fig. 8.12. This operation calls for two-dimensional interpolation in the frequency domain, an item to be discussed furthermore. Finally, the two-dimensional quadratic spectrum can be inverted using 2D inverse FFT in order to get the reconstructed image.

The complexity of this implementation, where the FFT is used to computing the spectra is given by

$$\mathcal{O}_{\text{Forward 1D FFT of sinogram}} = \mathcal{O}(T R \log R) \quad (8.29)$$

$$\mathcal{O}_{\text{Inverse 2D FFT of spectrum}} = \mathcal{O}(M^2 \log M) \Rightarrow \quad (8.30)$$

$$\mathcal{O}_{\text{Fourier Slice Theorem}} \approx \mathcal{O}(M^2 \log M) \quad (8.31)$$

where the time used to map the polar spectrum onto the quadratic spectrum has not been considered, and it will actually be negligible if using nearest neighbour interpolation. In the last equation it has been assumed that $T \approx R \approx M$. In all three equations the values of R and M should correspond to the expanded sinogram and image (powers of two), if using radix-2 FFT. In conclusion, Eq. 8.31 indicates that the implementation of the Fourier Slice Theorem is of a lower order than Filtered Backprojection and Filtering after Backprojection.

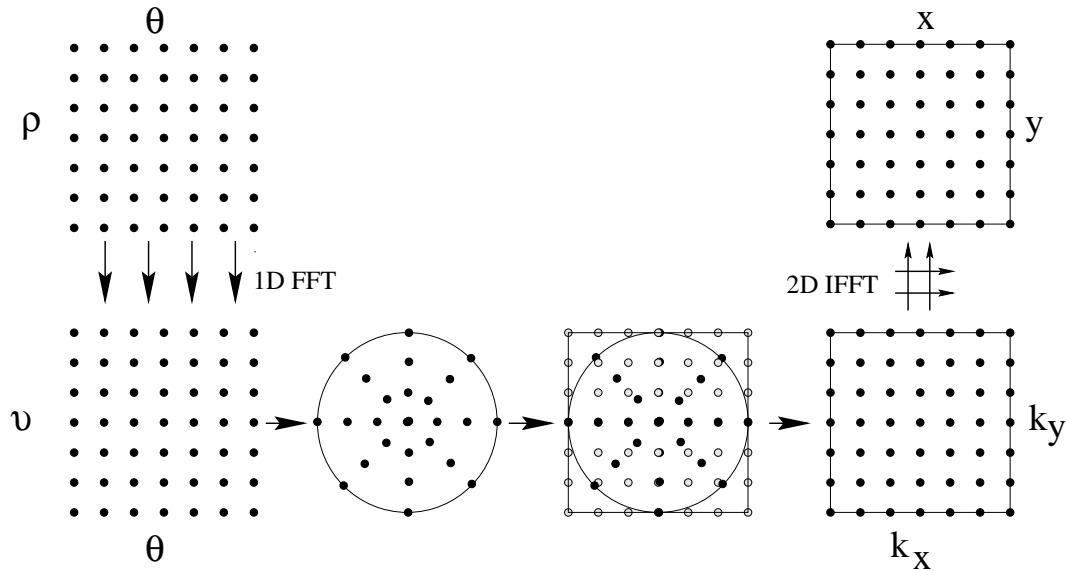


Figure 8.12 Following from upper left. At first the discrete spectrum is computed. The spectrum is then considered to be polar, and mapped onto a quadratic grid in the frequency domain using two-dimensional interpolation. Finally the 2D spectrum is inverted into the reconstructed image using 2D inverse FFT.

Next some of the problems with this implementation are discussed. If omitting the important shifting problems, illustrated in Fig. 8.1, the polar and the quadratic spectrum can match in three different ways as shown in Fig. 8.13. The boundary corresponds to the maximum frequencies (half of the sampling frequencies). Note that a square and centered reconstructed image is still assumed $M = N$, $x_{min} = y_{min}$ and $\Delta x = \Delta y$.

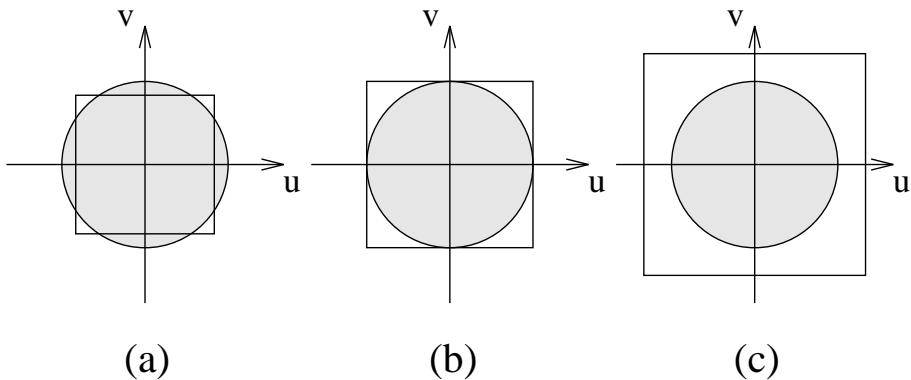


Figure 8.13 Three ways that the polar and the quadratic spectrum can match

In the first case (a) the quadratic spectrum is too small. Some parts of the polar spectrum is not mapped onto the quadratic spectrum. This is a very bad situation, and the result is unreliable when

$$\max |k_x| = \frac{1}{2\Delta x} < \nu_{max} = \frac{1}{2\Delta\rho} \Leftrightarrow \Delta x > \Delta\rho \quad (8.32)$$

In the second case (b) all of the polar spectrum is mapped onto the quadratic spectrum. This will happen when $\Delta x = \Delta\rho$. In the final case (c), where the polar spectrum is fully covered by the

quadratic spectrum. This means that the output image in principle uses all of the spectrum, but the reconstructed image will appear as it was low pass filtered, because the polar spectrum must be assumed to be equal to zero for frequencies higher than half of the polar sampling frequency.

Concerning the 2D interpolation, nearest neighbour interpolation is very fast, but the cost is that artifacts must be expected in the reconstructed image. Another common choice is instead to use the slower but more stable bilinear interpolation as shown in the following equations. Fig. 8.14 illustrates, that the value of each sample in the quadratic grid is a weighted sum of the four nearest neighbours in the polar grid. First of all the frequencies in the quadratic grid are expressed in polar coordinates.

$$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = \tilde{\nu} \begin{pmatrix} \cos \tilde{\theta} \\ \sin \tilde{\theta} \end{pmatrix} \quad (8.33)$$

Then the four nearest neighbours are found.

$$\text{Find the integer } t = \left\lfloor \frac{\tilde{\theta}}{\Delta\theta} \right\rfloor \Rightarrow \theta_t \leq \tilde{\theta} < \theta_{t+1} \text{ and set } w_\theta = \frac{\tilde{\theta} - \theta_t}{\Delta\theta} \quad (8.34)$$

$$\text{Find the integer } r = \left\lfloor \frac{\tilde{\nu}}{\Delta\nu} \right\rfloor \Rightarrow \nu_r \leq \tilde{\nu} < \nu_{r+1} \text{ and set } w_\nu = \frac{\tilde{\nu} - \nu_r}{\Delta\nu} = \Delta\rho(\tilde{\nu} - \nu_r) \quad (8.35)$$

where the last formula only is valid for positive frequencies. In case of negative frequencies, the periodical behavior of the spectrum must be considered and proper shifting must be used.

Finally, a bilinear interpolation in the polar coordinates of the four values are used

$$G(k_x, k_y) = (1 - w_\theta)((1 - w_\nu)G(\nu_r, \theta_t) + w_\nu G(\nu_{r+1}, \theta_t)) + w_\theta((1 - w_\nu)G(\nu_r, \theta_{t+1}) + w_\nu G(\nu_{r+1}, \theta_{t+1})) \quad (8.36)$$

The sampling of the output image must furthermore be sufficiently dense to adjust to the level of information in the polar spectrum, but due to the distribution of polar samples, with an increased density towards $(0,0)$, a general problem is that the quadratic spectrum does not exploit the high number of samples near $(0,0)$ leading to aliasing artifacts. On the other hand for high polar frequencies, the density is low, so the quadratic spectrum samples the polar spectrum faster than necessary. Note, the angular distance between samples in the polar grid is $\Delta\theta$, and in the radial parameter ν the distance between samples is $\Delta\nu = \frac{1}{R\Delta\rho}$, cf. Eq. 8.2. The distance in each of the coordinates in the rectangular grid is $\Delta k_x = \Delta k_y = \frac{1}{M\Delta x}$. If choosing $\Delta\rho = \Delta x$, cf. Eq. 8.32, one criterion is that the sampling interval in the radial parameter ν must match the one in the rectangular grid, which is a reasonable tradeoff, i.e.,

$$\Delta k_x = \Delta k_y = \frac{1}{M\Delta x} = \Delta\nu = \frac{1}{R\Delta\rho} \Rightarrow M = R \quad (8.37)$$

where both M and R should correspond to the expanded values, i.e., being a power of two if using a radix-2 FFT.

The presented reconstruction algorithm will introduce more noise than accumulated in the backprojection algorithms. Especially, ringing problems are common. The problem can be reduced by use of higher-order interpolation and/or use of, e.g. a non-linear grid in the Radon domain. Note that higher-order filters can provide better numerical results, but the computational load might increase to an unacceptable level. Another method [91, 58] illustrated in Fig. 8.15 is to distribute each of the polar samples onto the rectangular map using proper weights, which implies that all of the polar samples will always be represented in the quadratic grid, but the cost of the uneven density of samples is that an additional filtering is required.

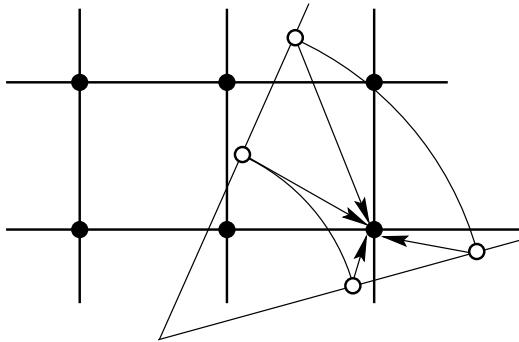


Figure 8.14 Strategy 1: A weighted sum of the four closest polar samples is used to estimate the spectrum on the quadratic grid.

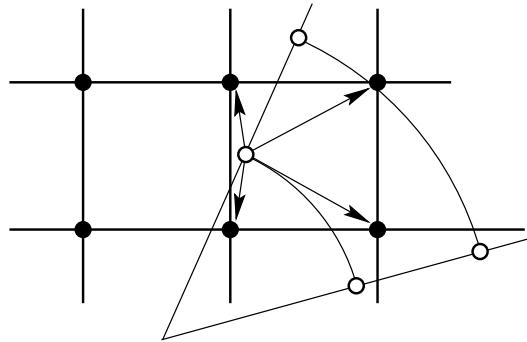


Figure 8.15 Strategy 2: Any of the samples on the polar grid are distributed with certain weights to the quadratic grid.

The 2D interpolation in the frequency domain described in this section is in general considered the major problem in implementations of the Fourier Slice Theorem, and the method has apparently found limited success in clinical use.

8.4.1 Non-linear sampling of the Radon domain

The problems by using two-dimensional interpolation in the frequency domain can be reduced by using a nonlinear grid in the Radon domain. The basic idea is to use one dimensional interpolation in the frequency domain and (perhaps) one dimensional interpolation in the Radon domain. This idea has been used in the Linogram method [92] and the method has attracted attention, because of the low complexity $M^2 \log M$, and the authors claim to provide an image quality as good as by use of Filtered Backprojection, in case of a noise-free sinogram. The Linogram method is also covered very thoroughly in [72], hence only the basic elements of the method is covered here. It can be noted that other reconstruction algorithms similar to the linogram method can be found in the literature, e.g., [93, 59].

The last half of the Fourier Slice Theorem, i.e., the operator \mathcal{F}_{2r}^+ is easily implemented using quadratic sampling of the frequency domain. Thus the spectrum $G(k_x, k_y)$ is approximated by the two dimensional DFT on a quadratic grid denoted $G(m, n)$, where the (positive) frequencies are given by

$$k_x = \frac{m}{M\Delta x} \quad (8.38)$$

$$k_y = \frac{n}{M\Delta x} \quad (8.39)$$

The polar spectrum $H(\nu, \theta)$ is as previous mapped onto the quadratic grid using Eq. 8.33, and linear sampling of θ is still applied, and the sampling of $\nu = r/(R\Delta\rho)$ can be chosen to vary with θ by allowing $\Delta\rho$ to be a function of θ . The frequency plane is divided into two sectors where the available values of k_x and k_y are used

- Sector 1

$$\sin \theta < \frac{1}{\sqrt{2}} : \begin{cases} k_x = \nu \cos \theta \Rightarrow \\ \nu = \frac{k_x}{\cos \theta} = \frac{m}{M\Delta x \cos \theta} \Rightarrow \\ \Delta\nu = \frac{1}{M\Delta x |\cos \theta|} \end{cases} \quad (8.40)$$

- Sector 2

$$\sin \theta \geq \frac{1}{\sqrt{2}} : \begin{cases} k_y = \nu \sin \theta \Rightarrow \\ \nu = \frac{k_y}{\sin \theta} \Rightarrow \\ \Delta \nu = \frac{1}{M \Delta x |\sin \theta|} \end{cases} \quad (8.41)$$

With this choice only one-dimensional interpolation of the spectrum is needed. The non-linear sampling of ν results in a non-linear sampling of ρ , i.e.

$$\sin \theta < \frac{1}{\sqrt{2}} \Rightarrow \Delta \rho = \frac{M}{R} \Delta x |\cos \theta| \quad (8.42)$$

$$\sin \theta \geq \frac{1}{\sqrt{2}} \Rightarrow \Delta \rho = \frac{M}{R} \Delta x |\sin \theta| \quad (8.43)$$

In Fig. 8.16 is shown the non-linear sampling of the frequency domain, and Fig. 8.17 shows the non-linear sampling of the Radon domain.

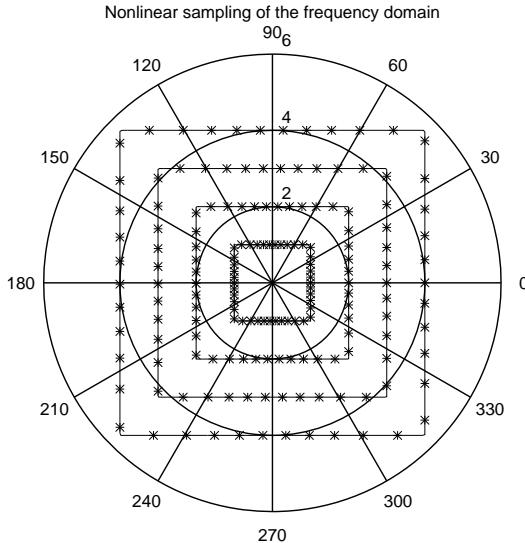


Figure 8.16 Non-linear sampling of the frequency domain.

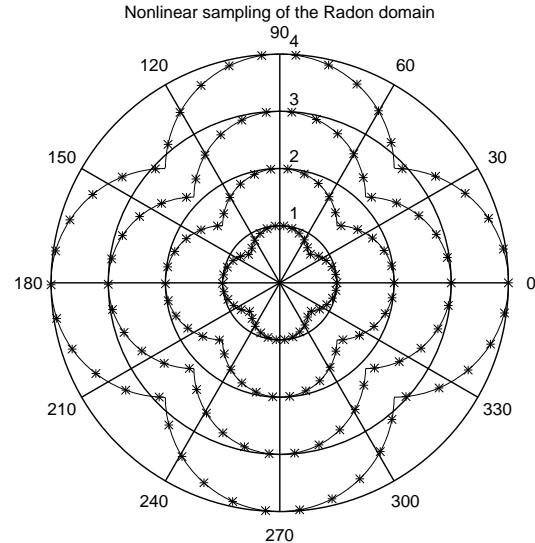


Figure 8.17 Non-linear sampling of the Radon domain. Here (ρ, θ) is used as polar coordinates.

If this technique is used, only interpolation with respect to k_x when $\frac{\pi}{4} < \theta < \frac{3\pi}{4}$ and only with respect to k_y is needed in the case $0 \leq \theta < \frac{\pi}{4} \vee \frac{3\pi}{4} \leq \theta < \pi$. The additional interpolation in the Radon domain is normally not that critical because the Radon transform has low pass characteristics, i.e., low-order interpolation is normally sufficient.

Another related strategy is to use the Chirp-z transform [94, 95] to transform directly from the sinogram (without interpolation) to the spectrum shaped as shown in Fig. 8.16. In [59] this method has been implemented and evaluated, but the conclusion was that even given that all of the involved step has complexity $\mathcal{O}(M^2 \log M)$, the method is not that fast for small sized images, e.g., 100*100 and certainly more difficult to implement compared to, e.g., the Fourier Slice Theorem.

8.5 Examples Using Direct Reconstruction Algorithms

A software package has been developed [59], where several direct reconstruction algorithms have been implemented, and lately additional features such as numerical forward projection (discrete Radon transform) has been implemented. The package called “iradon” is available from [10], and the usage of the programs are shown in Section C.2. All of the reconstruction examples in this chapter has been generated using “iradon”. Synthetic images and corresponding sinograms are generated using another developed package named “RadonAna”, described in Sections C.1 and B.3.

8.5.1 Reconstruction using Different Methods

Here reconstruction of a head phantom is examined. In Fig. 8.18 is shown the noise-free sinogram corresponding to the image shown in Fig. 8.19. Note that all of the following figures are color-scaled individually corresponding to the minimum and maximum value. First, Filtered Backprojection has been used to reconstruct the image as shown in Fig. 8.20. It can be seen that the mean value is displaced, and a ring is visible outside the head phantom with a radius corresponding to the extension of the sinogram in the ρ -direction. This is a good example showing that zeros in general must be padded to the sinogram in that direction in order to reduce the cyclical behavior of the FFT based filtering. If using Filtering after Backprojection, as shown in Fig. 8.21, the ring effect has disappeared, but otherwise the result appears to be just as good with respect to edge sharpness.

Finally, a nearest neighbour implementation of the Fourier Slice theorem has been used as shown in Fig. 8.22. The reconstructed image appears to be comparable to the two backprojection methods, but more “texture” can be found in the area outside of the head (and inside too).

The sampling parameters of the sinogram are $\Delta\rho = 0.01$, $R = 201$, $T = 200$, and for the image, $M = 201$ and $\Delta x = 0.01$ has been used.

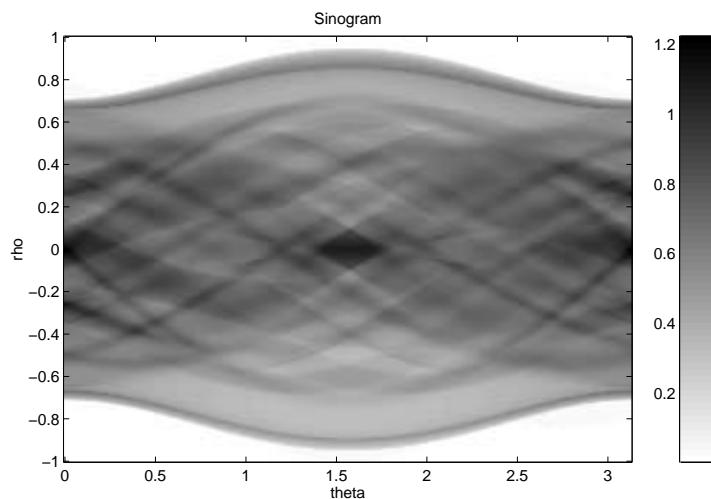


Figure 8.18 Sinogram of the phantom.

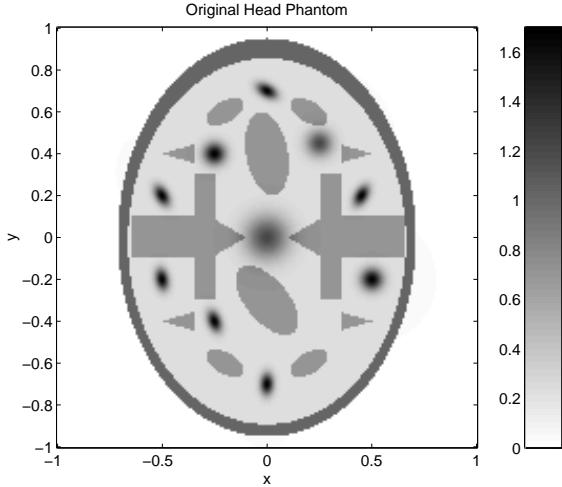


Figure 8.19 The original head phantom.

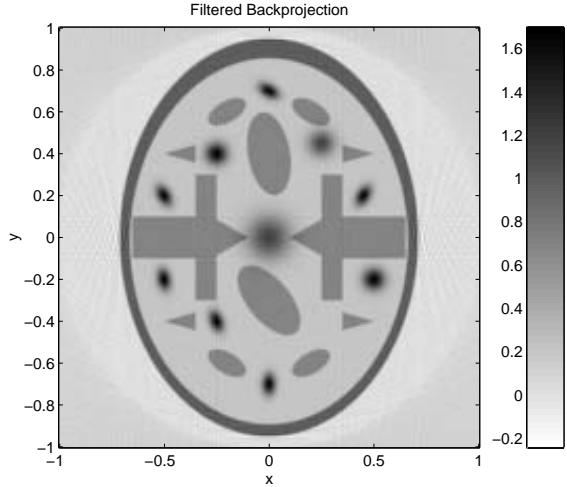


Figure 8.20 The reconstructed head phantom using Filtered Backprojection.

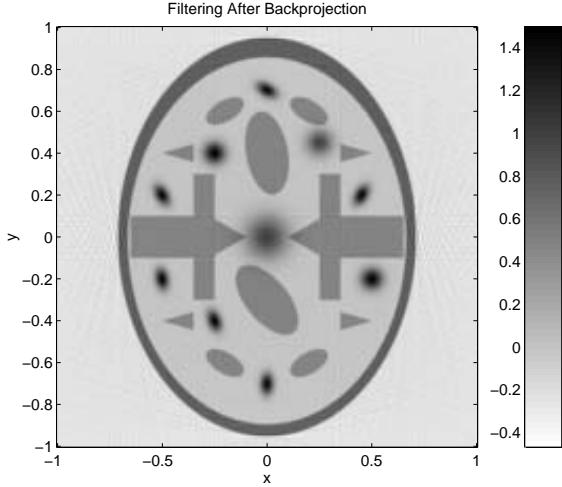


Figure 8.21 The reconstructed head phantom using Filtering After Backprojection.

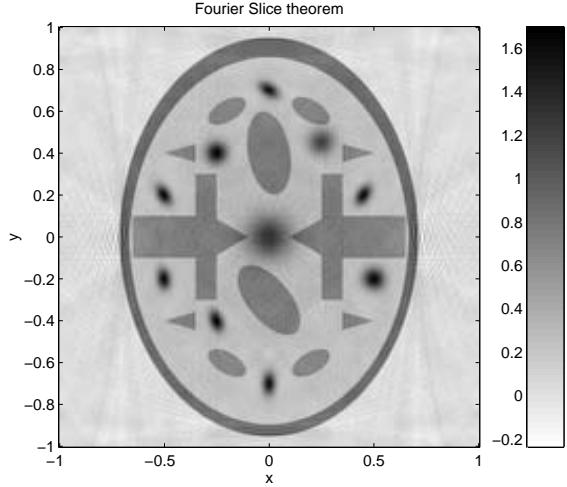


Figure 8.22 The reconstructed head phantom using Fourier Slice theorem.

8.5.2 Sinogram with very few Samples in the Angular Direction

A sinogram corresponding to a spiral of discs was created, and the number of samples in the angular direction, $T = 30$, is very small compared to $R = 151$ and $\Delta\rho = 0.2$. The sampling parameters of the reconstructed image is chosen to $M = 151$ and $\Delta x = 0.2$. First, Fig. 8.23 shows the reconstructed image using a nearest neighbour implementation of the Fourier Slice theorem. The few samples in the θ -direction can be seen as an angular blurring especially farthest away from the center of the image. Using Filtered backprojection (with a ramp filter) gives approximately the same visual impression, but without the angular blurring. Approximately the same number of discs can be identified in the image.

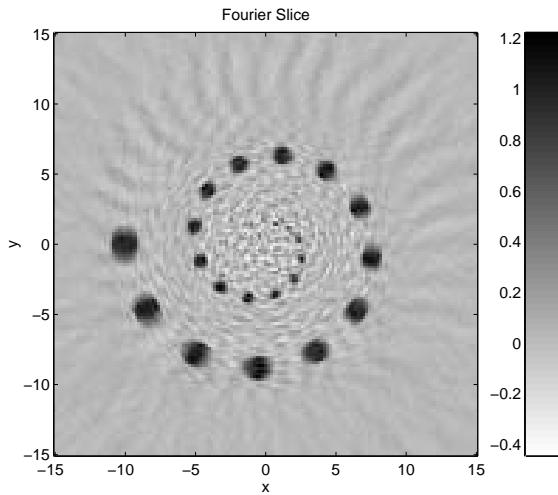


Figure 8.23 The reconstructed spiral using Fourier Slice Theorem.

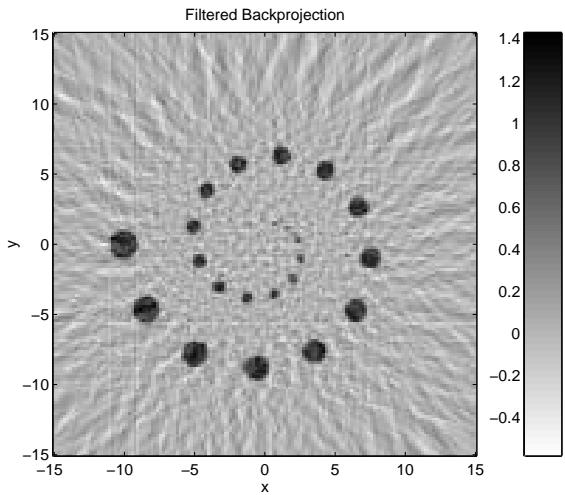


Figure 8.24 The reconstructed spiral using Filtered Backprojection.

8.5.3 Reconstruction with Varying Image Size

The next example addresses quantification of the reconstruction quality and artifacts when using a Fourier Slice algorithm, Filtered Backprojection, and Filtering after Backprojection. From the sinogram shown in Fig. 8.18, images have been reconstructed with increasing image size, going from $M = 51$ to $M = 401$ in steps of 50 samples (on each dimension of the reconstructed image), where the sampling distance has been changed in order to keep the image in focus, as shown in Fig. 8.19.

A modified L2-measure of misfit given in Eq. 8.44, has been used to quantify the error. The modification is that the DC-level does not alter the measure, due to the problems mentioned in Subsection 7.4.1.

$$L2 = \sqrt{\frac{\sum_{m,n} (g_{m,n} - \bar{g}_{m,n} - \bar{g} + \bar{g}^{ref})^2}{\sum_{m,n} (\bar{g}_{m,n} - \bar{g}^{ref})^2}} \quad (8.44)$$

where $\bar{g}_{m,n}^{ref}$ is the reference image (the original) and the bars indicate the average over all samples.

Fig. 8.25 shows the misfit in this case as a function of the image size M . It can be seen that here the errors limited, and the Fourier Slice method has the worst error-measure. These observations are naturally also depend on the image contents. For Filtered Backprojection Fig. 8.26 shows that the error has several reasons. It is obvious that the steep edges in the image are not reconstructed perfectly, due to the use of simple linear filters. Furthermore, lines are visible in the figure, which are due to aliasing problems. The sinogram should have been sampled more densely. Finally the ring also found in Fig. 8.20 will also add to the total error.

For a Pentium 120 MHz (Linux system) the time needed to reconstruct the images are shown in Fig. 8.27. It is clear that the radix-2 FFT used for this implementation implies that several steps can be seen in the Filtering after Backprojection and the Fourier Slice implementation. The reason that Filtering after Backprojection is much slower compared to Filtered Backprojection is that backprojection must be done into a larger number of samples (power of two) in order to reduce edge effects in the subsequent filtering. It can be seen that Filtering after Backprojection from image size 151 to 251 gets slightly faster. In this range the use of radix-2 FFT implies that a 256*256 image is generated in all three cases (151, 201, and 251) when backprojecting, and due to the implementation, the subsequent cropping becomes slightly faster here.

This example shows that Filtered Backprojection can be implemented efficiently on a PC and provide fast reconstruction, approximately as fast as the Fourier Slice theorem for the images size shown in this example.

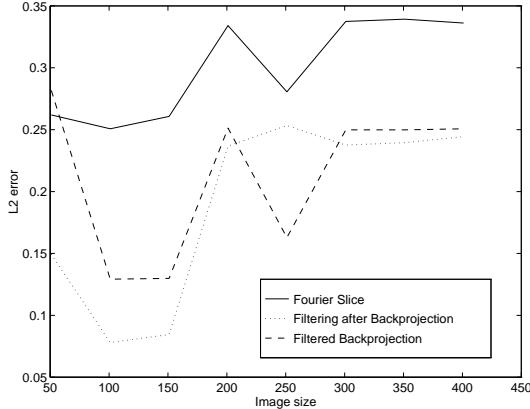


Figure 8.25 L2 error as a function of the reconstructed image size M for Filtered Backprojection, Filtering after Backprojection, and a Fourier Slice implementation.

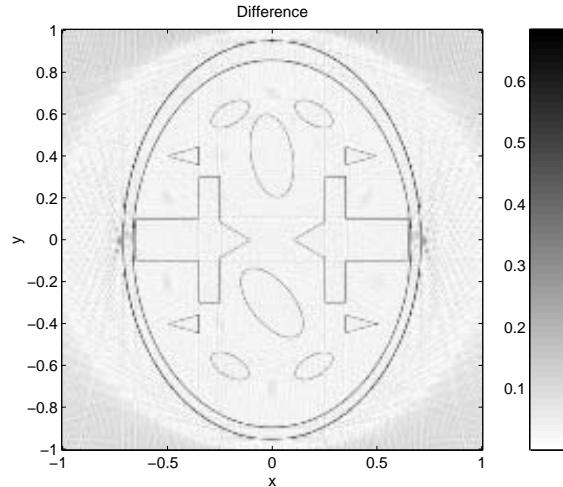


Figure 8.26 The absolute error between the original image and the reconstructed image using Filtered Backprojection.

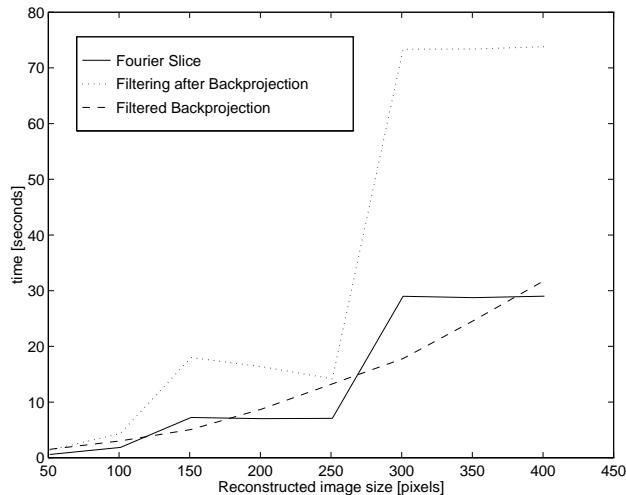


Figure 8.27 Time usage for reconstructing the sinogram in seconds as a function of the reconstructed image size M for Filtered Backprojection, Filtering after Backprojection, and a Fourier Slice Theorem implementation on a Pentium 120 MHz.

8.5.4 Reconstruction into a Oversampled Image

In order to illustrate the problems with the Fourier Slice theorem if $\Delta x > \Delta\rho$, (cf. Eq. 8.32) a sinogram with $R = 251$, $T = 200$, and $\Delta\rho = 0.0025$ has been created. The sinogram corresponding to a square in the image domain is shown in Fig. 8.28. Then two reconstruction methods have been used, namely the Fourier Slice Theorem in Fig. 8.29, and Filtering after Backprojection in Fig. 8.30. It can be seen that the filtering after backprojection has no problems, but the DC component is of course wrong. Using the Fourier Slice theorem severe artifacts can be seen, due to aliasing of the spectrum.

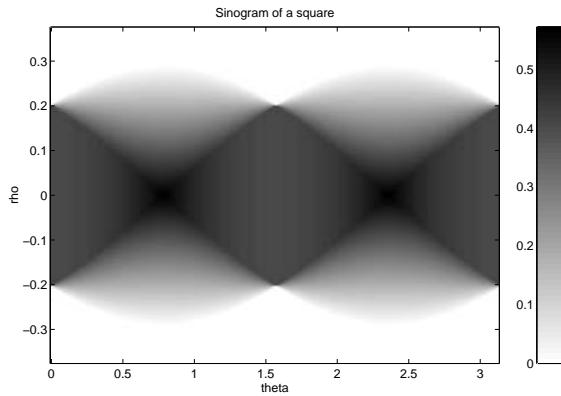


Figure 8.28 Sinogram of a square with $R = 251$, $T = 200$, and $\Delta\rho = 0.0025$.

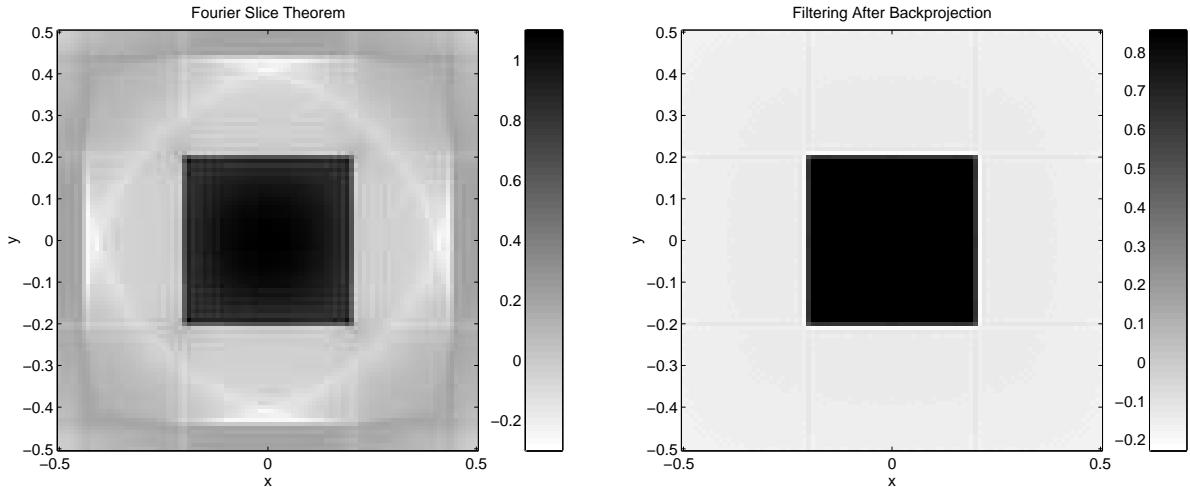


Figure 8.29 Reconstructed image with $M = 101$ and $\Delta x = 0.01$ using Fourier Slice theorem.

Figure 8.30 Reconstructed image with $M = 101$ and $\Delta x = 0.01$ using Filtering after Backprojection.

8.5.5 Noise in the Sinogram

One of the problems found in PET reconstruction, is that the sinograms are noisy. Assume that the noise is mainly due to finite measurement time of the emission sinogram, then each of the sinogram bins will be approximately Poisson distributed $P(\lambda_{r,t})$ with a parameter $\lambda_{r,t} = \mathcal{E}(\rho_r, \theta_t)T_e$, where \mathcal{E} denotes the mean and unknown emission activity in a bin of the sinogram, and T_e is the measurement time for the emission sinogram. This situation is often assumed in the literature, but it will be shown in Chapter 11 that especially the transmission scan might add much noise.

Here the sinogram is quantified by the total number of counts in the emission sinogram, i.e., the sum over all bins in the sinogram. For four values of the total number of counts, a sinogram corresponding to a disc is generated using a Poisson generator, [90]. The sinogram is then reconstructed using Filtered Backprojection.

First Fig. 8.31 shows the $101 * 100$ samples sinogram (corresponding to a disc in the image domain) with 10^5 counts, and Fig. 8.32 shows the corresponding reconstructed $151 * 151$ samples image. Likewise Fig. 8.33 has been reconstructed from 10^6 counts in the sinogram, and finally Fig. 8.34 from 10^7 counts. The sinogram size (10100 bins) corresponds to an average number of counts per sample in the range 10-1000. All three reconstructions have been made with Filtered Backprojection using a ramp filter.

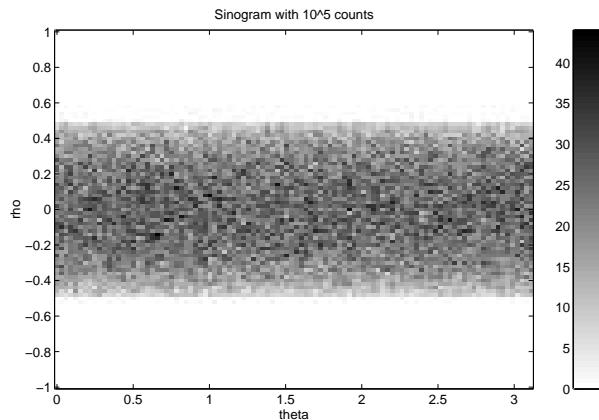


Figure 8.31 Sinogram with 10^5 counts.

From the reconstructed images it is clear that few counts implies a high noise level in the reconstructed images. As mentioned in Chapter 8 the ordinary cure is to multiply the ramp filter with an apodizing window. As an example a Hanning window will be multiplied to the ramp filter with a varying relative cutoff frequency ν_u , cf. Eq. 8.8. The L2 measure has been computed as a function of ν_l/ν_u , i.e., the cutoff frequency relative to half of the sampling frequency, and the result is shown in Fig. 8.35. In this case where the object of interest is very large a very low value of the optimal ν_l is found. Using $\nu = 0.2\nu_u$ the “optimal” reconstructed image is shown in Fig. 8.36. It can be seen that the high-frequency contents of the image has been removed, but the edges have also been blurred.

In general the choice of ν_u will always imply a tradeoff between resolution and noise suppression. To illustrate the tradeoff, the head phantom, previously shown in Fig. 8.19, has been reconstructed using a ramp filter apodized with a Hanning window with $\nu_u = 0.2\nu_u$. Here it can be seen that the value of ν_u is definitely too small, and most of the finer details have been removed by the low pass filtering.

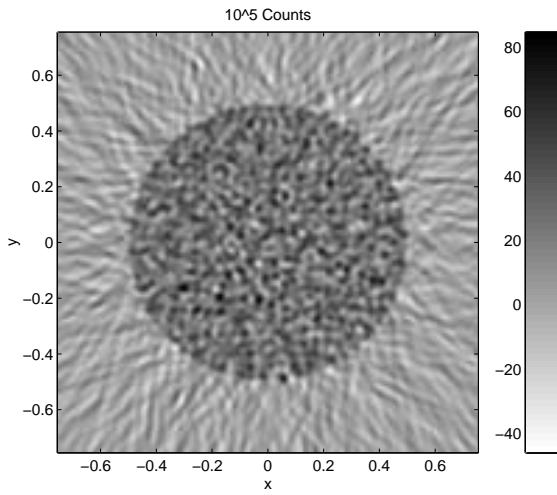


Figure 8.32 Reconstructed image from 10^5 counts in the sinogram.

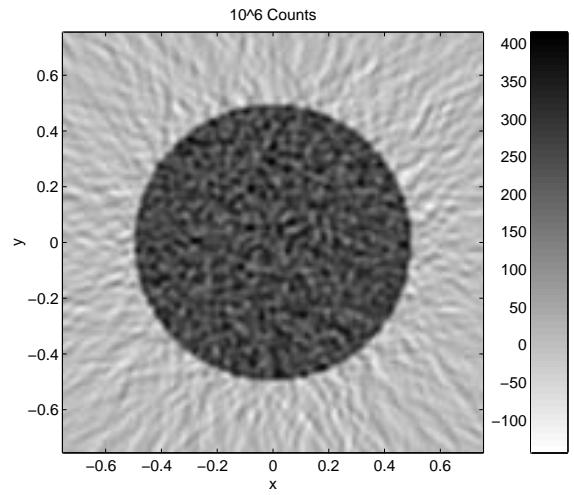


Figure 8.33 Reconstructed image from 10^6 counts in the sinogram.

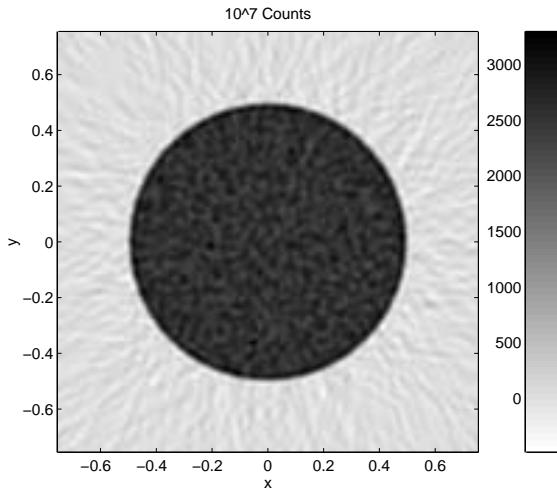


Figure 8.34 Reconstructed image from 10^7 counts in the sinogram.

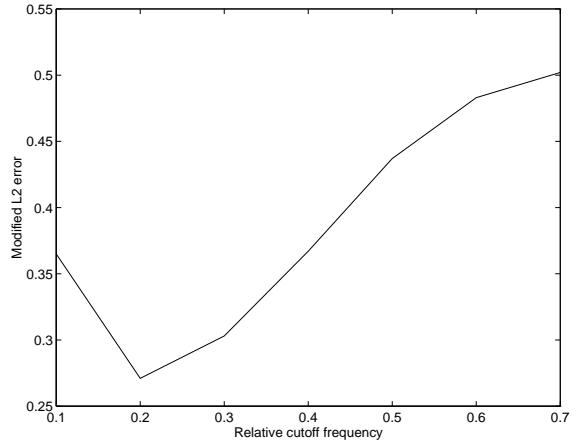


Figure 8.35 Modified L2 measure as a function of the relative cutoff frequency of the Hanning window.

8.6 Summary

In this chapter the implementation of several direct inversion schemes and hints for improving the speed have been presented. Filtered Backprojection is normally precise but somewhat slow, with complexity $\mathcal{O}(M^3)$, where M is the number of samples in one direction of the resulting image. Filtered Backprojection is the algorithm used in most scanners today. The Fourier Slice Theorem gives a fast algorithm, $\mathcal{O}(M^2 \log M)$, but does not have the same numerical stability, due to the two-dimensional interpolation. Another algorithm is a hybrid between the two inversion schemes: The Linogram method, see [72, 92]. This inversion method gives better numerical stability than implementations of the Fourier Slice Theorem and has complexity $\mathcal{O}(M^2 \log M)$. The algorithm uses the non-linear sampling of the frequency domain described above.

Finally, a set of examples using some of the direct reconstruction algorithms have been shown, demonstrating some of the possibilities and limitations with the direct reconstruction algorithms.

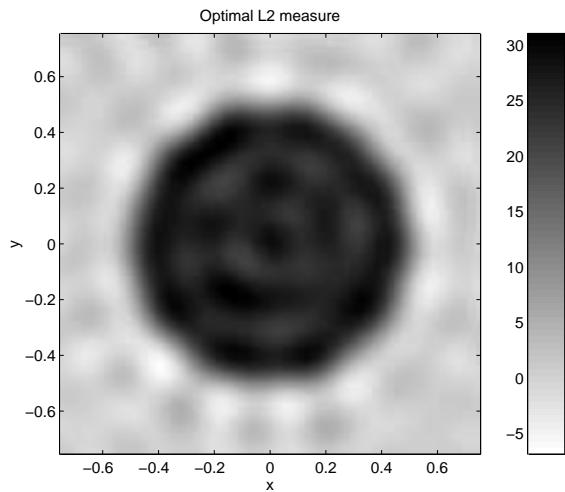


Figure 8.36 The reconstructed disc using $\nu_u = 0.2\nu_u$ from the sinogram with 10^5 counts.

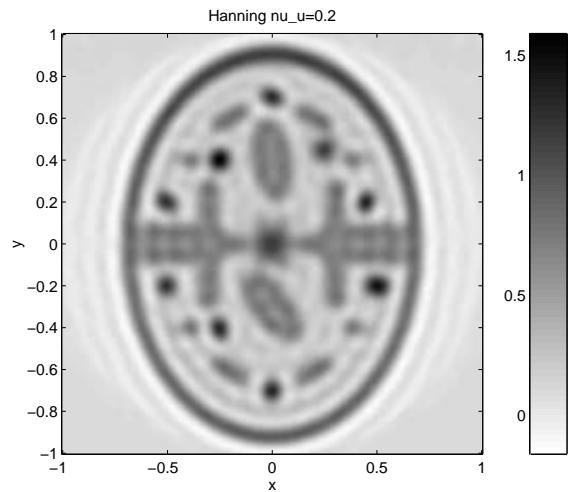


Figure 8.37 Reconstructed head phantom as shown in Fig. 8.19 but using a Hanning window with $\nu_u = 0.2\nu_u$. In this case the reconstructed images becomes very blurred.

Chapter 9

Reconstruction Algorithms Based on Linear Algebra

Inversion of the Radon transform need not be based on the direct inverse formulas as shown in Section 7.4. A different class of reconstruction schemes is based on linear algebra [96, 14, 97]. The chapter presents the linear algebra based reconstruction theory, and the modelling of the reconstruction problem. Several of the well known iterative reconstruction techniques are demonstrated, and a fast implementation of the iterative reconstruction algorithms is presented along with a set of examples.

9.1 From the Radon Transform to Linear Algebra based Reconstruction

The Radon transform is a linear transform, with respect to the function $g(x, y)$, and so is the discrete versions of the Radon transform, hence instead of considering the integral version of the Radon transform operators a matrix representation can be used

$$\begin{array}{ccc} \check{g}(\rho, \theta) & = & \mathcal{R} g(x, y) \\ \downarrow & & \downarrow \\ \mathbf{b} & = & \mathbf{A} \mathbf{x} \end{array} \quad (9.1)$$

Assume a discrete set of values for the Radon transform $\check{g}(\rho_r, \theta_t) = \check{g}_d(r, t)$, and a given sampling of wanted image $g(x_m, y_n) = g_d(m, n)$ used for the reconstruction. If the matrix $\check{g}(r, t)$ is rearranged into a vector, e.g.,

$$b_i = b_{rT+t} = \check{g}(r, t) \quad (9.2)$$

and the same technique is applied to the image, e.g.,

$$x_j = x_{nM+m} = g(m, n) \quad (9.3)$$

then using a series expansion shown in Eq. 9.5 the Radon transform can be written in the form shown in equation 9.11. The vector dimensions are $I = RT$ and $J = MN$, thus the transformation matrix \mathbf{A} have $RTMN$ elements, and will normally be very large.

$$g(x, y) = \sum_m \sum_n g(m, n) \phi(x - x_m, y - y_n) \Rightarrow \quad (9.4)$$

$$\check{g}(r, t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(\sum_m \sum_n g(m, n) \phi(x - x_m, y - y_n) \right) \delta(\rho_r - x \cos \theta_t - y \sin \theta_t) dx dy \quad (9.5)$$

$$= \sum_m \sum_n g(m, n) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \phi(x - x_m, y - y_n) \delta(\rho_r - x \cos \theta_t - y \sin \theta_t) dx dy \quad (9.6)$$

hence the matrix elements of the system matrix can be calculated as

$$a_{i,j} = a_{rT+t,nM+m} \quad (9.7)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \phi(x - x_m, y - y_n) \delta(\rho_r - x \cos \theta_t - y \sin \theta_t) dx dy \quad (9.8)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \phi(x, y) \delta((\rho_r - x_m \cos \theta_t - y_n \sin \theta_t) - x \cos \theta_t - y \sin \theta_t) dx dy \quad (9.9)$$

$$= \check{\phi}(\rho_r - x_m \cos \theta_t - y_n \sin \theta_t, \theta_t) \quad (9.10)$$

where the function $\phi(\cdot)$ is the expansion function in the image domain specifying how the pixel at (x_m, y_n) models the image domain with the continuous positions (x, y) . Eq. 9.10 shows that the matrix element is the Radon transform of the expansion function, where the ρ -parameter has been shifted according to the pixel position. Alternatives to the image pixel driven generation of matrix elements can be found in [98].

The methods to be presented all rely on a linear dependence between two vectors; the known I -dimensional vector \mathbf{b} ($I = RT$), containing the sinogram, and the unknown J -dimensional vector \mathbf{x} ($J = M^2$). For tomography the vector \mathbf{b} will contain the sinogram values wrapped into a vector using Eq. 9.2, and \mathbf{x} is the set of reconstructed pixels in the image formed as a vector using Eq. 9.3.

In the rest of this chapter a linear algebra formalism is used instead of deriving integrals formulations of the inverse Radon transform, which then is approximated in order to implement the algorithms. Now the reconstruction problem will be written in a matrix vector formulation

$$\mathbf{b} = \mathbf{Ax} \quad (9.11)$$

where $\mathbf{A} \in \mathbb{R}^{I \times J}$ is called the system matrix containing the weight factors between each of the image pixels and each of the line orientations from the sinogram, as illustrated in Fig. 9.1.

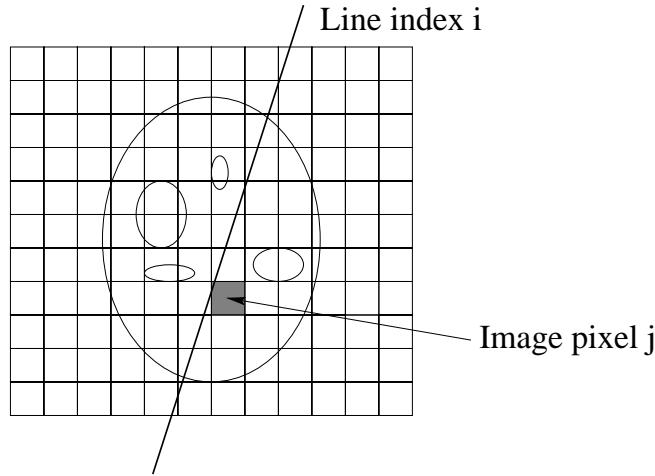


Figure 9.1 The matrix element $a_{i,j}$ can be considered as the weight factor between a certain sinogram value numbered by i and the image pixel j .

Compared to the Radon based direct reconstruction methods shown in Chapter 7, several new possibilities and problems arise

- Linear algebra is very strongly supported in mathematics, and the formalism can be used for both 2D and 3D PET reconstruction.
- An irregular scanner geometry with, e.g., limited line orientation is very bad for the direct methods, e.g., see [99], and with the linear algebra approach, problems with missing data in the sinogram can easily be incorporated into the matrix formalism.
- A finite, i.e., non-zero detector size can be modelled into the system, hence the model need no be based on a ray approximation, and varying detector sensibility can in principle be modelled into the system of equations, hence better modelling of the physical scanner setup is possible.
- The system matrix \mathbf{A} is in general not quadratic ($I \neq J$), which limit the number of techniques applicable if not forming the normal equations, which will be shown in Eq. 9.20.
- The system matrix \mathbf{A} will be (near) singular, i.e., have very small singular values, i.e., that reconstruction written in the linear algebra formalism is an ill-conditioned problem. Some of the image valued can be under determined and others very over determined due to the uneven coverage of the projections (sinogram lines) in the image domain. This problem must then be controlled with constraints and/or regularization.
- It turns out that \mathbf{A} does not has a simple structure, such as a band matrix, hence the inversion of \mathbf{A} have to use rather slow methods. The reason that \mathbf{A} does not have a simple structure is partly due to the easy sorting schemes shown in Eqs. 9.2 and 9.3. Another reason is that line parameters (ρ, θ) does not have a very simple relation to the image coordinates (x, y) .
- The matrix \mathbf{A} is normally very large, e.g., $256^2 \times 256^2$ elements, thus calculation of a generalized inverse of \mathbf{A} is extremely costly both in time and memory.
- The system matrix will be sparse due to the fact that only approximately M of the $M \times M$ image pixels adds weight to a certain bin in the sinogram.

9.2 The Calculation of Matrix Elements

The matrix \mathbf{A} can be estimated in several ways. One very common approach is to use a nearest neighbour approximation [96]. But a first order approximation better interpolation can also be used. As mentioned previous, alternatives to the image pixel driven generation of matrix elements can be found in [98].

Even though that some of the following interpolation schemes are very simple, they can be attractive if they can be computed fast, due to the fact that the large matrices normally are not stored in memory but calculated many times in the iterative reconstruction schemes and a good interpolation scheme takes longer time compared to the coarse interpolation schemes.

9.2.1 Pixel Oriented Nearest Neighbour Approximation

Around each pixel (x_m, y_m) is placed a square $\Delta x * \Delta x$ wide. If the line with parameters (ρ_r, θ_t) crosses the square then the matrix element a_{ij} is set to Δx , and else to 0. From Fig. 9.2 it is easy to see that the test criterion can written as

$$\rho' = \rho_r - x_m \cos \theta_t - y_m \sin \theta_t \quad (9.12)$$

$$|\rho' \cos \theta_t| < \frac{\Delta x}{2} \text{ and } |\rho' \sin \theta_t| < \frac{\Delta x}{2} \Rightarrow a_{rT+t,nM+m} = \Delta x \quad (9.13)$$

Note that the index $rT + t$ can easily be inverted to give r and t using truncation to lower integer and the modulus operator, and likewise with the index $nM + m$. This is relevant when generating the matrix, e.g., column-wise or row-wise. Often the matrix elements are set to 1 in case of the line crosses the pixel [96] and zero otherwise. This will imply a general scaling of the solution \mathbf{x} .

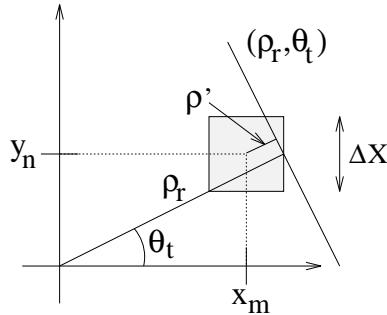


Figure 9.2 The line with index i , corresponding to (ρ_r, θ_t) , crosses the square pixel centered at (x_m, y_n) .

9.2.2 Discrete Radon Transform

Another interpolation scheme is to use the discrete Radon transform to approximate the forward matrix multiplication. This can be done using Algorithm 2.1, either once (requires storage of the system matrix) or every times it is needed. If multiplication with the transpose of the matrix is needed, then the discrete backprojection operator (multiplied with a factor of two) can be used, cf. Eq. 9.19 and Algorithm 8.3.

9.2.3 First Order Pixel Oriented Interpolation Strategy

Alternatively a ray-tracing strategy can be used, by assuming that the pixel at position (x_m, y_m) is a square $\Delta x * \Delta x$ wide with constant amplitude of 1. Now the length through the pixel with line orientation given by (ρ_r, θ_t) is used as the matrix element. By use of the rules for translation and scaling given in Eqs. B.6 and B.22 the result can be obtained from the Radon transform of a basic square \check{g}_{sq} . If the square is centered around $(0, 0)$ with side lengths $2 * 2$, then the results found in Subsection B.3.2 can used directly

$$a_{rT+t,nM+m} = \frac{\Delta x}{2} \check{g}_{sq} \left(2 \frac{\rho_r - x_m \cos \theta_t - y_m \sin \theta_t}{\Delta x}, \theta_t \right) \quad (9.14)$$

A variation of this scheme is to use several rays and average the result from each of the rays. The rays could represent some symmetrically placed spots in the parameter domain, lying within a square corresponding to the sampling distances, which is indicated in Fig. 9.3. In this way the finite size of pixel resolution and detector size (width of lines) can be modelled.

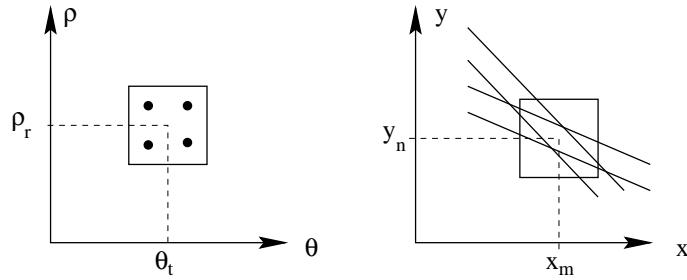


Figure 9.3 Left: A pixel in the discrete parameter domain, which here is subdivided into four center values. Right: For each of the center values the length though the square image pixel is averaged and used as the matrix value.

9.2.4 The Sinc Interpolation Strategy

Another approach is to use Eq. 2.42, which is based on a sinc interpolation scheme. This expression gives the elements of the \mathbf{A}

$$a_{rT+t,nM+m} = \frac{\Delta x}{\Psi} \sin \left(\Psi \min \left\{ \frac{1}{|\sin \theta_t|}, \frac{1}{|\cos \theta_t|} \right\} \right) \quad (9.15)$$

$$\Psi = \frac{\pi}{\Delta x} (\rho_r - x_m \cos \theta_t - y_n \sin \theta_t) \quad (9.16)$$

Note that this way of generating the matrix implies that some of the matrix elements will be negative, which definitely is bad from a physical point of view. Assuming that only one pixel is non-zero in the image, then negative counts will be measured for some line orientations. Nevertheless Eq. 9.15 represent a better interpolation from a signal processing point of view. Another drawback is that it is very costly to generate the matrix in this way compared to the other methods shown above.

9.3 Duality between Matrix Operations and the Radon Transform

Using the Radon transform scheme along with the matrix formalism as shown in Eq. 9.1 implies ordinary matrix operations have equivalent Radon transform operations. Eq. 9.1 is the Radon transform of discrete image $g(m, n)$ into the full discrete parameter domain $\check{g}(r, t)$.

$$\mathbf{b} = \mathbf{Ax} \leftrightarrow \text{Radon transform to full size parameter domain} \quad (9.17)$$

Iterative algorithms, such as ART and MART, described in the Sections 9.6 and 9.7, use the scalar product between row number i of \mathbf{A} , i.e., \mathbf{a}_i and the current reconstructed image \mathbf{x} , i.e., $\mathbf{a}_i^T \mathbf{x}$, which is the Radon transform of the image \mathbf{x} into one specific sample in the parameter domain.

$$b_i = \mathbf{a}_i^T \mathbf{x} \leftrightarrow \text{Radon transform to one sample in the parameter domain} \quad (9.18)$$

Another common operator in iterative algorithms is the transpose of the matrix. The operation $\tilde{\mathbf{x}} = \mathbf{A}^T \mathbf{b}$ is the backprojected discrete parameter domain into the image domain.

$$\tilde{\mathbf{x}} = \mathbf{A}^T \mathbf{b} \leftrightarrow \text{Adjoint Radon transform of the sinogram into the image domain} \quad (9.19)$$

This fact is often not recognized in the literature, but it is a direct consequence from the matrix formalism. The transpose of a matrix (without complex values) is the adjoint operator, and in

in this case the adjoint Radon transform is two times the backprojection operator, cf. page 134 of [14], and it is equivalent to the transpose matrix, cf. Eq. 7.10.

One well known approach to solve set of equations with a non-square system matrix is to form the normal equations.

$$\mathbf{A}^T \mathbf{b} = (\mathbf{A}^T \mathbf{A}) \mathbf{x} \quad (9.20)$$

An interesting analysis shown in [100] is that the solution to the normal equations

$$\mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \quad (9.21)$$

can be interpreted in formalism of the direct reconstruction methods. The matrix \mathbf{A}^T represent the backprojection operator, cf. Eq. 9.19, and then $(\mathbf{A}^T \mathbf{A})^{-1}$ represents the filter in Eq. 7.30. Likewise can it be shown that Filtered Backprojection can interpreted from Eq. 9.21 if assuming full rank of \mathbf{A}

$$\mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \quad (9.22)$$

$$= (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T (\mathbf{A} \mathbf{A}^T) (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b} \quad (9.23)$$

$$= (\mathbf{A}^T \mathbf{A})^{-1} (\mathbf{A}^T \mathbf{A}) \mathbf{A}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b} \quad (9.24)$$

$$= \mathbf{A}^T (\mathbf{A} \mathbf{A}^T)^{-1} \mathbf{b} \quad (9.25)$$

where $(\mathbf{A} \mathbf{A}^T)^{-1}$ represents the $|\nu|$ filter in Eq. 7.8 and \mathbf{A}^T (again) represents the backprojection operator. Note that the assumption of full rank is not quite valid, due to the problems mentioned in Subsection 7.4.1 with recovering the DC level when using Filtering after Backprojection or Filtered Backprojection.

9.4 Regularization and Constraints

In linear algebra regularization or constraints of an ill-conditioned set of equations is used to stabilize the solution. The problem is that the solution to Eq. 9.11 in a two-norm sense as shown in Eq. 9.26 often is useless, due to large undesired fluctuations in the solution [101].

$$\mathbf{x} = \operatorname{argmin} \left\{ \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 \right\} \quad (9.26)$$

These problems are nearly related to the conditional number of the system matrix. One way to deal with this problem is to use truncated SVD, which will be described in Section 9.5. Note that Eq. 9.26 is very general least-squares fit and is not based the noise actual noise statistics found in PET sinograms.

A very simple approach to stabilize the solution is to use constraints on the solution in iterative reconstruction algorithms. If, e.g., the activity of the brain is to be reconstructed, then a non-negativity constraint can be imposed, and perhaps a upper limit on the activity also can be imposed, then the solution can be truncated to the desired values simply by setting values outside of the interval to the upper/lower limit. It seems like an ad hoc approach, but it is often used in practice [96] and constraints can definitely improve the stability of iterative reconstruction algorithms, but it seems to be difficult to analyze the implications of constraints from a theoretical point of view.

A very common way of stabilizing iterative methods is to use regularization. This can be done by adding another term to Eq. 9.26.

$$\mathbf{x} = \operatorname{argmin} \left\{ \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda^2 \|\mathbf{L}(\mathbf{x} - \mathbf{x}^*)\|_2^2 \right\} \quad (9.27)$$

where \mathbf{x}^* is an estimate of the solution, and if none is available it can be set to zero. The operator \mathbf{L} is a matrix which can be selected to the identity matrix (restricts large values) or to approximate the first or second order derivative of the solution \mathbf{x} (restricts fast variations). The parameter λ controls how much weight the regularization term should have. Methods to estimate the optimal value of λ can be found in [101].

A simple way to include regularization terms in the reconstruction algorithms is to expand the set of equation by adding a set of regularization rows to the original set of equation

$$\begin{pmatrix} \mathbf{A} \\ \lambda \mathbf{L} \end{pmatrix} \mathbf{x} = \begin{pmatrix} \mathbf{b} \\ \lambda \mathbf{Lx}^* \end{pmatrix} \quad (9.28)$$

But it should be noted that the noise properties of the original and the expanded set of equations is normally very different, and it is not easy to predict theoretically how it will influence on a particular iterative reconstruction algorithm.

9.5 Singular Value Decomposition

The system matrix \mathbf{A} can be analyzed by use of the SVD (Singular Value Decomposition) [102, 73, 63]. The basic idea of the SVD is to decompose the matrix $\mathbf{A} \in \mathbb{R}^{I \times J}$ as

$$\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T \quad (9.29)$$

where the matrix $\Sigma \in \mathbb{R}^{I \times J}$ is a diagonal matrix with the singular values of \mathbf{A} on the diagonal, $\Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_J)$ and the matrices $\mathbf{U} \in \mathbb{R}^{I \times I}$ and $\mathbf{V} \in \mathbb{R}^{J \times J}$ are orthogonal matrices. The singular values are normally ordered in a non-increasing order, $\sigma_i \geq \sigma_{i+1}$, and the conditional number of \mathbf{A} is the ratio between the largest and the smallest singular value.

$$\text{cond}(\mathbf{A}) = \frac{\sigma_1}{\sigma_J} \quad J \leq I \quad (9.30)$$

To illustrate the properties of the system matrix, \mathbf{A} has been calculated from Eq. 9.15, cf. Section 9.2. The sampling parameters are chosen as shown in Eqs. 9.31-9.34, and the corresponding system matrix shown in Fig. 9.4.

Note that this example is a toy example, as it only concerns a very small 21×21 image reconstructed from a 21×20 sinogram, but it illustrates some of the basic properties of the system matrix. In this example the SVD could be computed in approximately 80 sec on a Pentium 120 MHz using the SVD-routine in Matlab, and the decomposition required approximately additionally 4 MBytes of memory.

$$x_m = -1 + m \cdot 0.1, \quad m = 0, 1, \dots, 20 \quad (9.31)$$

$$y_n = -1 + n \cdot 0.1, \quad n = 0, 1, \dots, 20 \quad (9.32)$$

$$\rho_r = -1 + r \cdot 0.1, \quad r = 0, 1, \dots, 20 \quad (9.33)$$

$$\theta_t = t \cdot \frac{\pi}{20}, \quad t = 0, 1, \dots, 19 \quad (9.34)$$

The singular values is shown in Fig. 9.5, and it can be seen that approximately 20 large singular values are succeeded by approximately 255 slowly decaying ones, and here a dramatic change occurs and then the singular values are decaying rapidly. The 20 large singular values corresponds to the size of the system matrix and the actual sampling. It can be seen that the

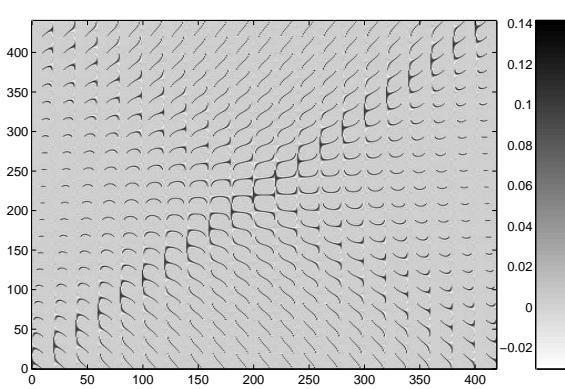


Figure 9.4 The system matrix \mathbf{A} .

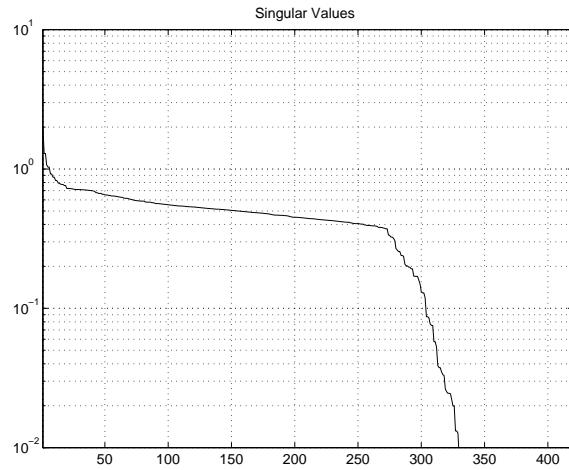


Figure 9.5 The singular values of the system matrix \mathbf{A} .

problem is ill-conditioned. This is normally mended by restricting the variations of \mathbf{x} , by use of regularization [101].

One way to invert an ill-conditioned problem is to separate the matrix Σ in a part with large singular values Σ_1 and a part with small singular values Σ_2 .

$$\Sigma = \begin{bmatrix} \Sigma_1 & \mathbf{0} \\ \mathbf{0} & \Sigma_2 \end{bmatrix} \quad (9.35)$$

i.e., the singular values in Σ_2 are insignificant. Choosing a rank p approximation, i.e. $\Sigma_1 \in \mathbb{R}^{p \times p}$. The generalized inverse of Σ is defined as

$$\Sigma^+ = \begin{bmatrix} \Sigma_1^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (9.36)$$

Note that this inversion is easily computed. The generalized inverse can be used for inversion of Eq. 9.11.

$$\mathbf{x} = \mathbf{V}\Sigma^+\mathbf{U}^T\mathbf{b} \quad (9.37)$$

This solution is a minimum norm solution to the least square problem $\min_{\mathbf{x}} \|\mathbf{b} - \mathbf{Ax}\|_2$.

From Fig. 9.5 it could be suggested to separate large and small singular values at singular value number 275. In Fig. 9.6 is shown the generalized inverse matrix using the first 275 singular values. A very stable structure is found looking something like the one found in Fig. 9.4, which could be expected due to the underlying line transformation model. Next, Fig. 9.7 shows the generalized inverse matrix using the first 320 singular values. Very little structure can be seen and here the inverted matrix is dominated by fast oscillations.

The suggested inversion strategy is not viable for varying geometries because SVD has complexity $\mathcal{O}(4I^2J + 8IJ^2)$, cf. page 248 of [102], where I and J is the number of rows and columns of the matrix \mathbf{A} respectively. This implies that the SVD used for inversion of the Radon transform has complexity approximately $\mathcal{O}(M^6)$, where M still is the number of pixels on each of the axes in the image domain. Here it should be noted that for a fixed scanner geometry the SVD only has to be computed once. After the SVD decomposition of the system matrix the minimum norm solution can easily be found using matrix multiplications, and a varying rank or filtering the singular values more softly by use of, e.g., the popular Tikhonov regularization, e.g., [101, 103] is

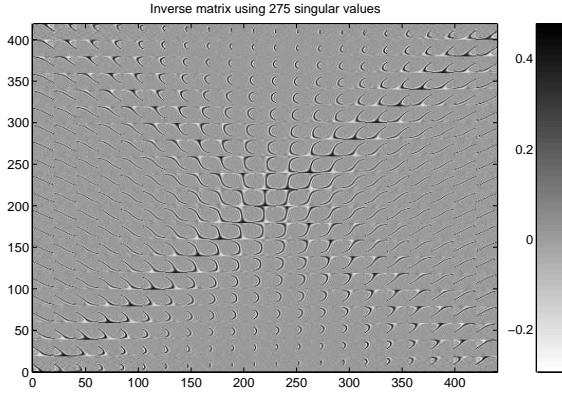


Figure 9.6 The generalized inverse of the system matrix using 275 singular values. A structure resembling the transpose of the system matrix can be seen.

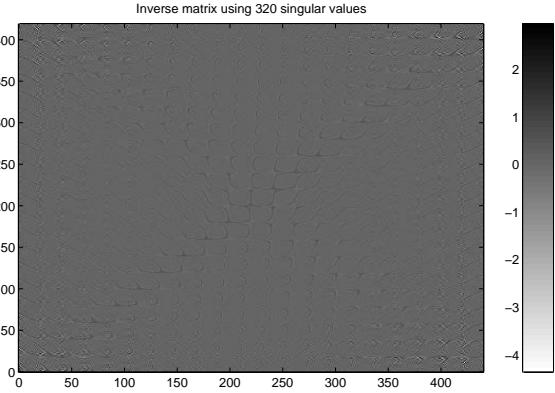


Figure 9.7 The generalized inverse of the system matrix using 320 singular values. The structure has vanished and note the span in amplitude compared to Fig. 9.6.

easily incorporated in the inversion scheme. Despite the high computational complexity, the SVD has been used for reconstruction of SPECT images [104, 105], and it has been reported that the SVD of matrices with size 7080×4436 can be computed in approximately 35 hours on a SUN Sparc 10/40. For 3D reconstruction with a significantly higher number of sinogram values and voxels, the SVD based reconstruction requires a change in computing technology.

9.6 Iterative Reconstruction using ART

A well-known way to solve Eq. 9.11 is named ART, which stands for Algebraic Reconstruction Technique. Many articles, e.g., [96, 106, 73] demonstrate the algorithm. ART was published in the biomedical literature in 1970, and Cormack and Hounsfield used ART for reconstructing the very first tomography images. They probably did not know of the work of Johann Radon at that point, and later it was discovered that ART is a reinvention of the algorithm introduced by Kaczmarz [107] in back in 1937.

The basis operator required in ART is the scalar product between to certain row i of the system matrix, \mathbf{a}_i and a solution vector $\tilde{\mathbf{x}}$

$$\tilde{b}_i = \sum_{j=1}^J a_{ij} \tilde{x}_j = \mathbf{a}_i^T \tilde{\mathbf{x}} \quad (9.38)$$

ART is formulated as an iterative reconstruction algorithm, where the solution vector in iteration k is updated by adding a scaled version of row i of the system matrix.

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \frac{b_i - \mathbf{a}_i^T \mathbf{x}^{(k)}}{\mathbf{a}_i^T \mathbf{a}_i} \mathbf{a}_i \quad (9.39)$$

The main idea of ART is that the equation i is fulfilled in iteration k , which easily is shown:

$$\mathbf{a}_i^T \mathbf{x}^{(k)} = \mathbf{a}_i^T \mathbf{x}^{(k-1)} + \frac{b_i - \mathbf{a}_i^T \mathbf{x}^{(k-1)}}{\mathbf{a}_i^T \mathbf{a}_i} \mathbf{a}_i^T \mathbf{a}_i = b_i \quad (9.40)$$

If using the Radon transformation terminology, then ART will modify the reconstructed image in iteration k , in order to give the correct Radon transform at (ρ_r, θ_t) , where (r, t) is found from

the sinogram sorting scheme shown in Eq. 9.2. In this way the reconstruction quality of ART in a given area can be altered by choosing that i matches the lines passing through an interesting area frequently.

Often [73] i as a function of k is chosen to $i = k \text{ MOD } I$, where MOD is the modulus operator, but this choice is not very good [106, 108]. Another very common and easy strategy is to choose i randomly using a uniform probability density function. Initially $\mathbf{x}^{(0)}$ can be chosen to zero or some good guess on the solution, e.g., in 2D from a fast algorithm like one based on Fourier Slice Theorem. Yet another strategy is to initialize all solution values with a constant, as it will be shown in Subsection 9.6.2.

The ART algorithm can also be interpreted from a geometrical point of view. Fig. 9.8 shows the use of ART in a two parameter estimation problem with two projection lines. As shown in Eq. 9.40 the current solution in iteration k , i.e., $\mathbf{x}^{(k)}$ is projected perpendicularly (along the direction of \mathbf{a}_i^T) onto the hyperplane (in this case a line) determined by $\mathbf{b}_{i(k)} = \mathbf{a}_i^T \mathbf{x}^{(k)}$. As seen from Fig. 9.8 the speed of convergence is very dependent on the angle between the hyperplanes.

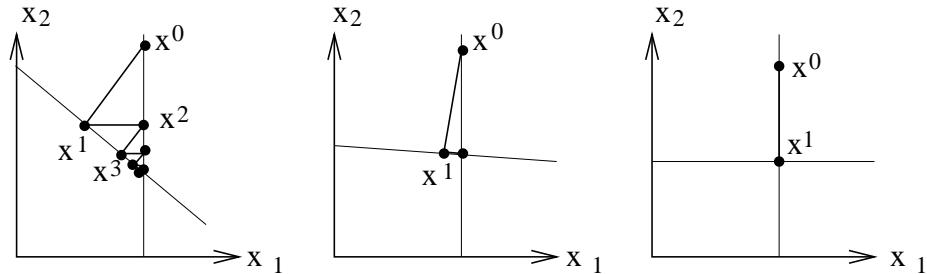


Figure 9.8 Three different cases of iteration in a two parameter problem using ART. Depending of the orthogonality of the hyperplanes (here lines) the convergence can be slow (left most figure) or fast (right most figure).

Note that ART in each iteration only requires the scalar product between \mathbf{a}_i and the current solution $\mathbf{x}^{(k)}$, which can be compared to calculating one value in the parameter domain using discrete Radon transform, hence each iteration is very fast but the quality enhancement gained from one iteration is in general very limited. Often when comparing ART to other iterative algorithms that requires computation of a full discrete parameter domain, the iteration number used for ART is a full loop through all I rows, i.e., I times the actual number of iterations in ART.

A common alteration of ART is to introduce a relaxation parameter in form of a weight factor as shown in Eq. 9.41.

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \lambda_k \frac{\mathbf{b}_i - \mathbf{a}_i^T \mathbf{x}^{(k-1)}}{\mathbf{a}_i^T \mathbf{a}_i} \mathbf{a}_i^T \quad (9.41)$$

where λ_k can be set as a simple function of k , such as a linear function or a exponential decay. Even given the result Eq. 9.40 it has experimentally been proven [109], that setting λ_k to values different from one can improve the speed of convergence. It should be noted that the optimum value of λ_k is a function of k , the sinogram values, and the sampling parameters of the reconstructed image. In [106] it is shown that optimizing the value of λ_k in each iteration and careful selection of row index i as a function of k results in a reconstructed image quality as good as using the EM-algorithm, at a order of magnitude less computational cost (The EM-algorithm is presented in Section 9.8).

In the literature, authors have optimized ART and/or EM, and for some years it was not clear

whether the one was better than the other. Now it seems as if ART is loosing popularity compared to EM for 2D reconstruction, but for 3D reconstruction the opposite seems to be the case [110] and Section 10.6.

As shown in Algorithms 9.1 and 9.2 ART is very easy to implement. Only a scalar product is really needed. In the algorithm the matrix elements are used over and over again, and note for a huge problem which cannot be stored in memory all at one time each matrix element has been calculated in a function like it was shown in Section 9.2. In Algorithm 9.1 the denominator values $\mathbf{a}_i^T \mathbf{a}_i$ are computed once as a function of i . The reason for showing two algorithms is that the first part can be computed once, and if several sinograms with the same geometry should be reconstructed, then it is only Algorithm 9.2 which should be used several times.

ALGORITHM 9.1 : INITIALIZATION OF THE ART ALGORITHM

```

For i = 0 to I-1                                //For all values of i
    sum = 0                                         //Calculate the denominator
    For j = 0 to J-1                               //For all values of j
        sum = sum + a(i,j)*a(i,j)                  //Accumulate value
    End
    anorm(i)=sum                                    //Store denominator
End
For j = 0 to J-1                                //For all values of j
    x(j) = c                                       //Initialize with a constant
End

```

ALGORITHM 9.2 : THE ART ALGORITHM

```

For k = 0 to K-1                                //For K iterations of ART
    Set i as a function of k                      //Choose row index in iteration k
    sum = 0                                         //Initialize scalar product
    For j = 0 to J-1                               //For all values of j
        sum = sum + a(i,j)*x(j)                   //Update scalar product
    End
    w = lambda(k)*(b(i)-sum)/anorm(i)            //Calculate common weight
    For j = 0 to J-1                               //For all values of j
        x(j) = x(j) + w*a(i,j)                   //Project solution
    End
End

```

9.6.1 ART with Constraints

It is not guaranteed, that ART will provide a non-negative solution, as required by the physical meaning of the solution; in PET emission activity and in CT absorption coefficient. As mentioned in Section 9.4 a crude, but very easily implemented strategy is to restrict the solution after some iterations from an upper limit estimate on the solution in each pixel, i.e., vector element or maybe just using an upper limit constant in each iteration. A non-negativity constraint can be imposed as shown in Algorithm 9.3, where the initialization part shown in Algorithm 9.1 has been removed. Other constraining schemes for ART can be found in [96].

ALGORITHM 9.3 : THE ART ALGORITHM WITH CONSTRAINTS

```

For k = 0 to K-1                                //For K iterations of ART
    Set i as a function of k
    sum = 0                                         //Choose row index in iteration k
    For j = 0 to J-1                               //Initialize scalar product
        sum = sum + a(i,j)*x(j)
    End                                              //For all values of j
    w = lambda(k)*(b(i)-sum)/anorm(i)            //Update scalar product
    For j = 0 to J-1                               //Calculate common weight
        x(j) = x(j) + w*a(i,j)                    //For all values of j
    End                                              //Project solution
    If k MOD kc = 0                                //Every kc iterations use constraints
        For j = 0 to J-1                            //For all values of j
            If x(j) < 0                           //Negative solution is found
                x(j) = 0                           //which is set to zero
            End
            If x(j) > Maxx(j)                   //Too large solution is found
                x(j) = Maxx(j)                  //which is set to max limit
            End
        End
    End
End

```

9.6.2 Initialization

Using iterative algorithms also implies that the solution vector \mathbf{x} should be initialized with a constant value or a good start guess, which need not be a constant. One possibility is to use fast direct algorithms, such as Fourier Slice based methods, for producing a start guess. If the start guess is good the iterative algorithms in general will converge faster, but it also implies that the behavior of the algorithm will be biased by the direct method. Another very common starting guess is to initialize the solution with a constant [111]. For the ART algorithm no restrictions are made concerning the initial value, but for other iterative algorithms such as MART (Section 9.7) and EM (Section 9.8) the initial value has to be positive. Assuming that the solution is a constant, then by averaging in Eq. 9.11 the value should be

$$x_j^0 = \frac{\sum_{i=1}^I b_i}{\sum_{j=1}^J \sum_{i=1}^I a_{i,j}} \quad \forall j \quad (9.42)$$

This initialization requires that the system matrix is computed an additional time or can be combined with the first part of Algorithm 9.1. A faster scheme is to use the approximation that for a certain value of i a line approximately crosses M pixels each with a value of approximately Δx , hence

$$\sum_{j=1}^J \sum_{i=1}^I a_{i,j} \approx IM\Delta x \Rightarrow x_j^0 = \frac{1}{IM\Delta x} \sum_{i=1}^I b_i \quad \forall j \quad (9.43)$$

9.7 Multiplicative ART

Another technique is MART (Multiplicative Algebraic Reconstruction Technique), that can be found in the literature, but MART has apparently not had a major impact in practical reconstruction applications. The basis of MART is to maximize the entropy of the solution

$$-\sum_{j=1}^J x_j \log x_j \quad (9.44)$$

under the constraints that

$$b_i = \mathbf{a}_i^T \mathbf{x}^{(k)} \text{ and } x_j \geq 0 \forall j \quad (9.45)$$

Like ART, the algorithm is very easy. At first all image values are initialized to the constant value

$$x^0 = e^{-1} \quad (9.46)$$

and then the general iteration step looks like

$$\mathbf{x}_j^{(k+1)} = \left(\frac{b_i}{\mathbf{a}_i^T \cdot \mathbf{x}^{(k)}} \right)^{\lambda_k a_{i,j}} \mathbf{x}_j^{(k)} \quad (9.47)$$

where λ_k is a positive tunable relaxation parameter. Just like ART will MART require a scheme for selecting the row index i as a function of the iteration index k . Again two strategies are commonly used; cyclical ($i = k \text{ MOD } I$) or randomly with a uniform probability distribution over all indices.

According to [96] the behavior of the algorithm is not properly known, especially with respect to the influence of noise in the sinogram. Another problem is that the denominator in Eq. 9.47 can become zero, which indicates that problems with instability can be expected.

9.8 The EM algorithm

So far the reconstruction methods have modelled the projections as line integrals, which was reconstructed by use of discretization of the inverse Radon transform. Especially in emission tomography with limited counts in each sinogram bin, the statistical noise can dominate the reconstructed images when using direct reconstruction methods, cf. Subsection 8.5.5. This lead many scientists to consider statistical approaches to derive reconstruction algorithms.

One of the most prominent iterative reconstruction methods for emission tomography is Maximum Likelihood Reconstruction using the EM algorithm, which stands for Expectation Maximization. In the very famous articles [80] by Shepp and Vardi and [112] by Vardi, Shepp, and Kaufman a statistical framework for reconstruction is given. It is assumed that the measurements originate from uncorrelated Poisson generators, which ideally model the underlying physics, but problems like attenuation correction are not modelled in this framework. Another feature of the EM-algorithm is that the model includes a non-negativity constraint.

The key idea of the EM-algorithm is to maximize the Likelihood

$$L(\mathbf{x}) = P(\mathbf{b}|\mathbf{x}) = \prod_{i=1}^I \frac{(b_i^*)^{b_i}}{b_i!} e^{-b_i^*} \quad (9.48)$$

where \mathbf{b}^* contains the unknown mean values of \mathbf{b} , i.e., the true sinogram without noise, and it is assumed that \mathbf{b}^* fit the solution perfectly

$$\mathbf{b}^* = \mathbf{A}\mathbf{x} \quad (9.49)$$

where the coefficients of the system matrix are considered as transition probabilities normalized as

$$1 = \sum_{i=1}^I a_{i,j} \quad (9.50)$$

in PET meaning that a photon pair at detector pair i originates from one of the regions in the brain (pixels in the image) with the probability of one.

Now the expectation of the Likelihood is maximized from the log Likelihood $l(\mathbf{x})$ by setting the derivatives of $l(\mathbf{x})$ to zero

$$\frac{\partial l(\mathbf{x})}{\partial x_j} = - \sum_{i=1}^I a_{i,j} + \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}} = -1 + \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}} = 0 \quad (9.51)$$

In the literature an additional non-negativity constraint is imposed and it can be shown that this results in the Kuhn-Tucker conditions,

$$x_j \frac{\partial l(\mathbf{x})}{\partial x_j} = 0 \quad \forall j \text{ where } x_j > 0 \quad (9.52)$$

$$\frac{\partial l(\mathbf{x})}{\partial x_j} \leq 0 \quad \forall j \text{ where } x_j = 0 \quad (9.53)$$

where the first equation is used to formulate an iterative mapping scheme

$$0 = x_j \frac{\partial l(\mathbf{x})}{\partial x_j} = x_j \left(-1 + \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}} \right) \Rightarrow \quad (9.54)$$

$$x_j^{(k)} = x_j^{(k-1)} \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}^{(k-1)}} \quad (9.55)$$

This equation is very often found in statistical reconstruction literature, but it should be noted that it requires that the elements of the system matrix is properly normalized. Here another version of the EM algorithm proposed in [113], and found to give very good results in [114], is used. It does not require the assumption shown in Eq. 9.50, and works fine with the normalization used in Section 9.2.

$$x_j^{(k)} = \frac{x_j^{(k-1)}}{\sum_{i'=1}^I a_{i',j}} \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}^{(k-1)}} \quad (9.56)$$

Eq. 9.56 is a very compact form to show the EM-algorithm, but it does not show that each iteration consists of four steps

$$\mathbf{b}^f = \mathbf{A}\mathbf{x}^{(k-1)} \quad (9.57)$$

$$b_i^q = \frac{b_i}{b_i^f} \quad (9.58)$$

$$\mathbf{x}^b = \mathbf{A}^T \mathbf{b}^q \quad (9.59)$$

$$x_j^{(k)} = \frac{x_j^{(k-1)} x_j^b}{s_j} \quad (9.60)$$

where $s_j = \sum_{i=1}^I a_{i,j}$, forms a normalization vector that can be calculated once for all.

In Subsection 9.11.4 it will be demonstrated, that Eq. 9.58 can lead to instability. Assume a PET imaging situation where a certain i corresponds to a line not entering regions of activity, hence the forward projected value b_i^f becomes zero, thus the denominator in Eq. 9.58 is a potential stability problem.

The EM algorithm is computationally demanding. Eq. 9.57 shows that each iteration requires a forward projection (Radon transform) of the current solution, cf. Eq. 9.17. The quotient in each point between the measured sinogram b_i and the forward projected solution b_i^f , i.e., \mathbf{b}^q is then backprojected into the image domain. Finally, in Eq. 9.60, the next estimate of the solution is generated by multiplying for each index j , the current estimate of the solution $x_j^{(k-1)}$ with the backprojected solution x_j^b weighted with s_j . This shows that each iteration of EM actually costs more than using, e.g., Filtered Backprojection. Note also that the EM algorithm is nonlinear, hence additive noise in the sinogram will not automatically lead to an additive noise term in the reconstructed images, as it has been the case for all the previous methods. For further reading on the EM algorithm for Maximum Likelihood iterative reconstruction methods [80, 112, 115, 116, 111, 113] are recommended.

In Algorithms 9.4 and 9.5 are shown the implementation of the EM-algorithm. It has been split into two parts indicating that the first part has to be calculated once, and the last part can then be used to reconstruct several images with the same geometry.

ALGORITHM 9.4 : INITIALIZATION OF THE EM ALGORITHM

```

For j = 0 to J-1                                //For all values of j
    sum = 0                                     //Calculate the values of s(j)
    For i = 0 to I-1                            //For all values of i
        sum = sum + a(i,j)                      //Accumulate value
    End
    s(j)=sum                                     //Store value
End
For j = 0 to J-1                                //For all values of j
    x(j) = c                                     //Initialize with a constant
End

```

Note that the initialization of the EM-algorithm requires that the system matrix is generated once in the initialization part, but this can be avoided by initializing the solution vector as as shown in Algorithm 9.6, and then use Algorithm 9.5 for the remaining iterations. Note that in the first line of the Algorithm 9.5 the counter should then be **For k = 1 to K-1**.

ALGORITHM 9.5 : THE EM ALGORITHM

```

For k = 0 to K-1                                //For K iterations of EM
  For i = 0 to l-1                               //For all values of row index
    sum = 0                                         //Initialize scalar product
    For j = 0 to J-1                               //For all values of j
      sum = sum + a(i,j)*x(j)                      //Update scalar product
    End
    bq(i) = b(i)/sum                             //Store scaled forward projection
  End
  For j = 0 to J-1                               //For all values of column index
    sum=0                                         //Initialize backprojection sum
    For i = 0 to l-1                               //For all values of row index
      sum = sum + a(i,j)*bq(i)                     //Accumulate sum
    End
    x(j) = x(j)*sum/s(j)                         //Update solution
  End
End

```

ALGORITHM 9.6 : THE FIRST ITERATION OF THE EM ALGORITHM

```

For i = 0 to l-1                                //For all values of row index
  sum = 0                                         //Initialize scalar product
  For j = 0 to J-1                               //For all values of j
    sum = sum + a(i,j)*x(j)                      //Update scalar product
  End
  bq(i) = b(i)/sum                             //Store scaled forward projection
End
For j = 0 to J-1                                //For all values of column index
  s(j) = 0                                         //Initialize s-values
  sum=0                                         //Initialize backprojection sum
  For i = 0 to l-1                               //For all values of row index
    la = a(i,j)                                    //Store value in simple variable
    sum = sum + la*bq(i)                         //Accumulate sum
    s(j) = s(j) + la                            //Increment s
  End
  x(j) = x(j)*sum/s(j)                         //Update solution
End

```

9.9 The Conjugate Gradient Method

The Conjugate Gradient algorithm (or CG for short) is a well-documented technique for iteratively solving equation systems with a symmetrical positive definite system matrix [101, 117]. For reconstruction purposes the Conjugate Gradient algorithm is not well suited, due to the restrictions on the system matrix, but reconstruction can form the normal equations, shown in Eq. 9.20, which gives the least squares solution to the original set of equations, shown in Eq. 9.11. Using normal equations the effective system matrix $\mathbf{A}^T \mathbf{A}$ is symmetrical and the singular values are now squared compared to the singular values of \mathbf{A} .

Given that the system matrix is sparse, which can be exploited for effective storage, the system matrix of the normal equations $\mathbf{A}^T \mathbf{A}$ should not be generated, because it in general will not be sparse, thus it requires a lot of storage space. In several textbooks, e.g., [117] an algorithm is found implementing the Least Squares Conjugate Gradient method (LSCG) without computing $\mathbf{A}^T \mathbf{A}$.

$$\text{Initialize } \mathbf{x}^0 \quad (9.61)$$

$$\mathbf{s}^0 = \mathbf{b} - \mathbf{A}\mathbf{x}^0 \quad (9.62)$$

$$\mathbf{r}^0 = \mathbf{p}^0 = \mathbf{A}^T \mathbf{s}^0 \quad (9.63)$$

$$\mathbf{q}^0 = \mathbf{A}\mathbf{p}^0 \quad (9.64)$$

Then for each iteration the LSCG algorithm on the normal equations becomes

$$\alpha = \frac{\|\mathbf{r}^{k-1}\|_2^2}{\|\mathbf{q}^{k-1}\|_2^2} \quad (9.65)$$

$$\mathbf{x}^k = \mathbf{x}^{k-1} + \alpha \mathbf{p}^{k-1} \quad (9.66)$$

$$\mathbf{r}^k = \mathbf{A}^T \mathbf{s}^{k-1} \quad (9.67)$$

$$\mathbf{r}^k = \mathbf{r}^{k-1} - \alpha \mathbf{q}^{k-1} \quad (9.68)$$

$$\beta = \frac{\|\mathbf{r}^k\|_2^2}{\|\mathbf{r}^{k-1}\|_2^2} \quad (9.69)$$

$$\mathbf{p}^k = \mathbf{r}^k + \beta \mathbf{p}^{k-1} \quad (9.70)$$

$$\mathbf{q}^k = \mathbf{A}\mathbf{p}^k \quad (9.71)$$

Note that each step of the iteration requires several vector updates and scalar products, by the bulk of the computational load lies in the forward projection and the backprojection operation. So the computational complexity per iteration resembles that of the EM-algorithm. No algorithm is given here, due to the fact that Eqs. 9.61-9.71 only includes simple operations, but many, hence the pseudo code will be long.

Note that the LSCG algorithm estimates a solution for the equation system shown in Eq. 9.20, which implies that the effective system matrix $\mathbf{A}^T \mathbf{A}$ squares the singular values, and a bad conditional number gets squared. According to [101] the CG method has an inherent regularization and in [118] the regularization properties are examined.

In [100], it is demonstrated that for the LSCG Method, it really does not matter, whether a constant (here zeros) or an image generated by use of Filtered Backprojection is used.

9.10 Accelerated Iterative Reconstruction

From an implementation point of view the iterative algorithms shown in the previous sections, so far have been treated as if all the values in the system matrix $a_{i,j}$ are available in general. One problem is that the system matrix is normally huge. A 2D sinogram from, e.g., a GE Advance PET scanner contains $I = 281 * 336$ values, and reconstructed into a $J = 301 * 301$ grid, i.e., the system matrix has approximately 8.5 billion elements, requiring over 34 GBytes of memory, when using 4 bytes per matrix element. This is a lot of memory, even looking some years into the future. Besides, it would not be wise to store all that data, due to the fact that approximately 98% of the matrix entries will be zeros. This knowledge should be incorporated into the reconstruction schemes.

Assuming that memory is not available for storing the full system matrix, one possibility is to compute the individual matrix elements in each iteration when needed. This can be done by use of the discrete Radon transform, as described in Section 9.2 or other modelling schemes for the scanner. This approach is viable, is rather easily implemented, and the storage requirements are reduced to a minimum. Memory is only required for the sinogram (\mathbf{b}) and the current solution (\mathbf{x}), and perhaps some additional temporary variables of the same size or smaller, but no system matrix is stored in memory. It will be demonstrated that this implementation has a major drawback in speed, since the system matrix will be computed many times during an iterative reconstruction. Each time with the same high computational cost.

Here a new hybrid solution is proposed [5, 6] for accelerating the iterative reconstruction algorithms, but requiring as much memory as modern workstations are currently equipped with, or will be soon. The idea is to only store the non-zero elements of the system matrix in the main memory using sparse matrix techniques. In this way the core of the reconstruction algorithms, highly based on matrix vector multiplications, can be accelerated significantly, and thereby removing one of the major drawbacks of the iterative methods.

It is proposed that the system matrix \mathbf{A} is calculated one time only using all the modifications found for the actual scanner setup. If no specific scanner model is provided then the system matrix can be modelled and generated using the Radon transform or other simpler schemes as shown in Section 9.2. From the system matrix the very small values are truncated to zero.

$$\tilde{a}_{i,j} = \begin{cases} a_{i,j} & \text{if } a_{i,j} > \gamma \\ 0 & \text{Otherwise} \end{cases} \quad (9.72)$$

where the threshold γ can be chosen to a small fraction of the maximum matrix value, e.g., $\gamma = 0.05 \max_{i,j} \{a_{i,j}\}$. If γ is chosen sufficiently low, a good compromise between resolution and the sparseness of the matrix can be reached, and normally this does not alter the behavior of the algorithms, and for some of the schemes used to estimate the matrix values the value of γ can be set to 0, due to a finite interpolation width in, e.g., Eq. 9.13.

The sparse structure of \mathbf{A} can be exploited by only storing non-zero values in the fast memory. For a certain row, number i , all of the matrix elements are calculated, stored, and truncated using Eq. 9.72. Hereby will the number of non-zero elements in the row, denoted by Z_i , be much smaller than the image size $J = M^2$. The values of Z_i are stored in a simple one dimensional vector. Two vectors of length Z_i , indexed by an integer z , can then be allocated and stored containing the non-zero matrix value a_z and the corresponding column index j_z . The procedure is repeated for all rows.

Assuming a nearest neighbour approximation with one pixel for each point along the integration lines and using 4 bytes for storing each of the vector elements, then the total storage requirement is reduced to approximately $8 \sum_{i=1}^I z_i \approx 8IM$ bytes. In the example shown above

approximately 100 MBytes of memory is required. Assuming that amount of memory is present most iterative algorithms can be implemented from three basic operations: Matrix vector multiplication $\mathbf{A}\tilde{\mathbf{x}}$, scalar product between the i 'th row of the system matrix and a vector $\mathbf{a}_i^T\tilde{\mathbf{x}}$, and finally multiplication with the transpose of the matrix $\mathbf{A}^T\tilde{\mathbf{b}}$.

In the following, the implementations of the scalar product (Algorithm 9.7), the matrix vector multiplication (Algorithm 9.8), and the multiplication with the transpose the system matrix (Algorithm 9.9) are shown.

ALGORITHM 9.7 : $\mathbf{b}_i^f = \mathbf{a}_i^T\tilde{\mathbf{x}}$

```

Set a and j to correct row          //Use pointer technique
sum = 0                            //Initialize
For z=0 to Z(i)-1                 //For row i
    sum=sum+a(z)*x(j(z))          //Increment sum
End
bt=sum                             //Store value

```

ALGORITHM 9.8 : $\mathbf{b}^f = \mathbf{A}\mathbf{x}$

```

For i = 0 to l-1                  //For all rows
    sum = 0                        //Initialize
    Set a and j to correct row     //Use pointers
    For z=0 to Z(i)-1             //For row i
        sum=sum+a(z)*x(j(z))      //Increment sum
    End
    bt(i)=sum                      //Store value
End

```

ALGORITHM 9.9 : $\mathbf{x}^b = \mathbf{A}^T\mathbf{b}$

```

For j=0 to J-1                    //For all columns
    xb(j)=0                       //Initialize
End
For i=0 to l-1                  //For all rows
    Set a and j to correct row    //Using pointers
    For z=0 to Z(i)-1             //Compute sum
        xb(j(z))=xb(j(z))+a(z)*b(i) //Update sum
    End
End

```

9.10.1 A Fast 2D Iterative Reconstruction Package

A software package, named “it” has been written in the C-language. A short description is provided in Section C.3. The package includes the proper structures for manipulating sparse matrices and vectors, along with optimized code for computing matrix vector like products, well suited for iterative reconstruction algorithms. The software package are provided for free from [10].

In the package ART, EM, and LSCG are implemented both in a fast version using sparse matrix storage of the system matrix and in a slow version where the system matrix is not stored and required matrix entries are computed in each part of the iterative steps. In this way the package can be used to reconstruct any size of images, and if enough memory is available the reconstruction is very fast.

The package also features constraining of the solution using either a constant minimum and a constant maximum for each of the solution values, or an image can be supplied masking the ROI (region of interest), used to limit the solutions differently in several areas.

Currently one regularization option is found in the package, namely a discrete version of the Laplace operator, shown in Table 9.1, used in the reconstructed image. The operator and the image are convolved, and for each pixel in the image the regularization term will add one line to the system matrix, cf. Eq. 9.28. Likewise is the \mathbf{b} -vector expanded with zeros match the extra number of rows of the system matrix.

0	-1	0
-1	4	-1
0	-1	0

Table 9.1 Discrete approximation to the Laplace operator, used in the image domain.

In the software package “it” it has been done by appending extra rows to the system matrix at runtime, after the system matrix has been generated or read from the disk. Due to the use of sparse storage techniques and the small support of the operator, the appended part to the system matrix can be generated very fast and will not require much memory, hence the regularization factor λ can be supplied at runtime and the software package can be used number of times with different regularization factors and use the same system matrix stored on the disk.

Several interpolation methods have been implemented for generation of the system matrix.

- Binary estimation of the matrix matrix, cf. Eq. 9.13.
- Nearest Neighbour interpolation based on the Radon transform, cf. Algorithm 2.1.
- Linear interpolation based on the Radon transform.
- Analytical Radon transform of a square, i.e., the length through a quadratic pixel is used, cf. Subsection B.3.2.
- The Radon transform of a sinc expansion in the image domain, cf. Eq. 9.15.

9.11 Examples Using Iterative Reconstruction Algorithms

In this section a set of examples are presented where iterative reconstruction methods have been used. The examples have been generated with the package “it”. The artificial sinograms used for some examples is generated using “RadonAna”. Both packages are described in Appendix C.

The program has been used on two types of machines. A Linux machine with a 120 MHz Pentium processor and an Silicon Graphics (SGI) Onyx equipped with four 200 MHz R4400 processors, where the program was running on one processor.

9.11.1 A Small Reconstruction Example

In the first example the (synthetic) sinogram has $125 * 101$ samples and the reconstructed image has $101 * 101$ samples. In Table 9.2 the reconstruction times on both machines are shown for the fast and the slow method as well as the ratio between the execution times (slow/fast).

Times are measured for ART, EM, and the LSCG-method, when EM and LSCG were running (arbitrarily) 20 iterations, and ART 20 full iterations, i.e., 20 times the number of rows (chosen randomly), which is $20*125*101$ iterations in Eq. 9.39. Note that all times only correspond to the actual iterations. For the fast versions of the iterative reconstruction algorithms, the time to generate the system matrix once should be added if changing the system matrix, e.g., when changing the sampling parameters of the reconstructed image.

For this example the system matrix was modelled using discrete Radon transformation with linear interpolation, where the threshold γ was chosen to zero, hence the slow and the fast methods give exactly the same results. Note that the large difference in speedup between ART and EM/LSCG is due to the implementation of the discrete Radon transform is more efficient than the multiplication with the transpose of the system matrix. The slow methods can be accelerated some by implementing multiplication with the transpose of the system matrix (adjoint operator) as a backprojection integral, but note that this implies that the approximation of the system matrix will be different in the forward and the backprojection part. The sparse system matrix for this transformation geometry required approximately 13 MBytes of memory, and each iteration requires approximately one second.

Machine	Type	ART	EM	LSCG
Pentium	Fast	26 sec	17 sec	17 sec
	Slow	2218 sec	5776 sec	5250 sec
	Ratio	85	340	309
SGI Onyx	Fast	16 sec	16 sec	16 sec
	Slow	1306 sec	2722 sec	2715 sec
	Ratio	82	170	170

Table 9.2 Time usage for 20 iterations of EM and LSCG. For ART the time is for 20 full iterations, i.e., $20*125*101$ of the iterations used in Eq. 9.39. The time measurements are for a sinogram with $125 * 101$ samples reconstructed into a $101 * 101$ samples image.

9.11.2 A Larger Reconstruction Example

In Fig. 9.9 a 2D sinogram with $281*336$ samples is shown, which was measured on a GE Advance PET scanner. The sinogram is reconstructed into a (large) image with $301*301$ samples. The system matrix has $94416*90601$ elements of which 0.23% are non-zero when modelling the system matrix using the Radon transform of a square. On the Onyx it required 1466 seconds to generate sparse matrix requiring 157 MBytes of memory, and each iteration of EM required 15 sec in the fast version and 11678 sec in the slow implementation. After 10 iterations the algorithm was stopped. The reconstructed image is shown in Fig. 9.10. For sake of comparison, the same sinogram reconstructed image using Filtered Backprojection is shown in Fig. 9.11, and this reconstruction used 6 seconds on the Onyx.

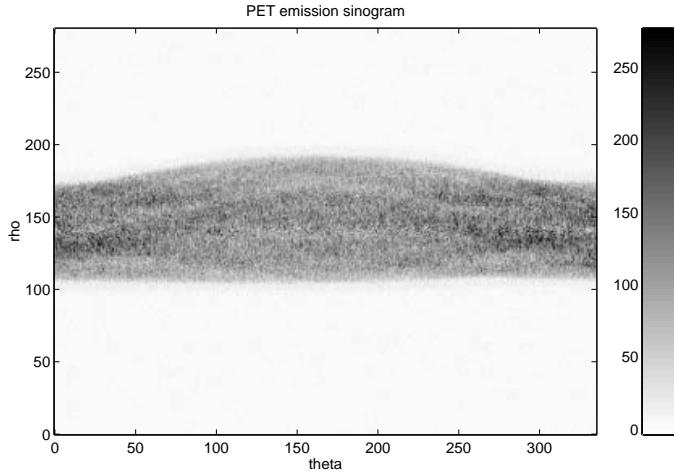


Figure 9.9 A slice of a 2D PET sinogram of a human brain.

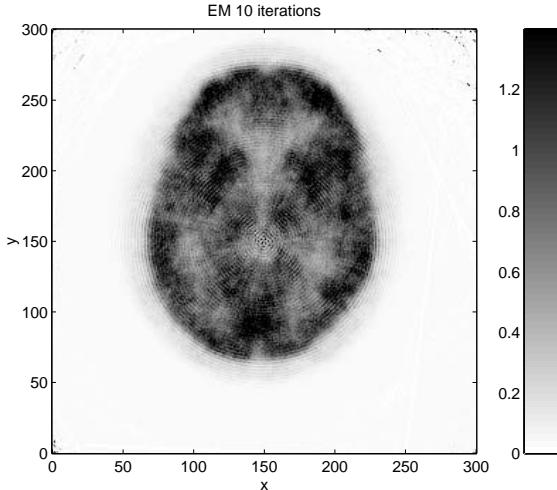


Figure 9.10 The reconstructed image after 10 iterations of EM.

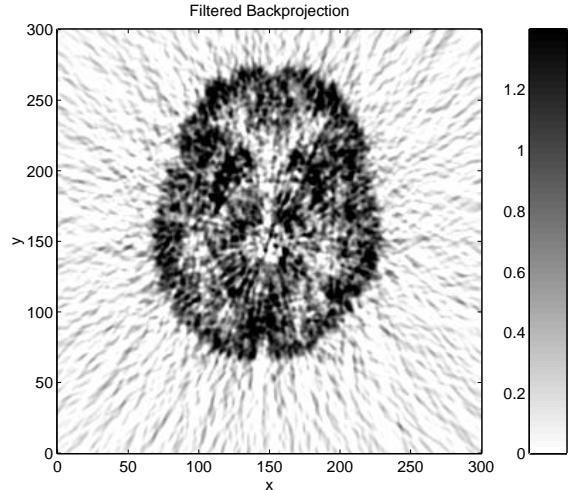


Figure 9.11 The reconstructed image using Filtered Backprojection with a ramp filter.

The front page of the thesis is generated from the reconstruction of 35 slices of 2D sinograms, as the one shown in Fig. 9.9. The reconstructed slices have been put into a single volume. The volume has been segmented using a marching cubes based program [119], which makes a iso-surface corresponding to a chosen threshold level. A 3D model of the reconstructed brain is available from [120].

9.11.3 Comparison with Different Noise Levels

In this subsection an (synthetic) image with 101×101 pixels shown in Fig. 9.12 should be reconstructed from the noisy sinogram shown in Fig. 9.13. The sinogram has been generated by use of the program "RadonAna", described in Section C.1, where an additive uncorrelated Gaussian noise ($\sigma = 1$) term was added. Likewise has a sinogram been generated without the noise term, and one where the Gaussian deviance is 5 ($\sigma = 5$). It can very well be argued, that this noise term is very unphysical, due to (negative) sinogram values clearly lying outside of the head, but some of the correction algorithms used in practice can also produce artifacts like this.

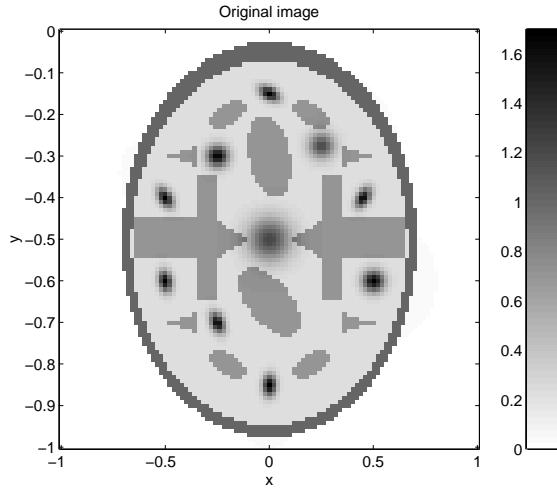


Figure 9.12 The original head phantom containing 101×101 pixels.

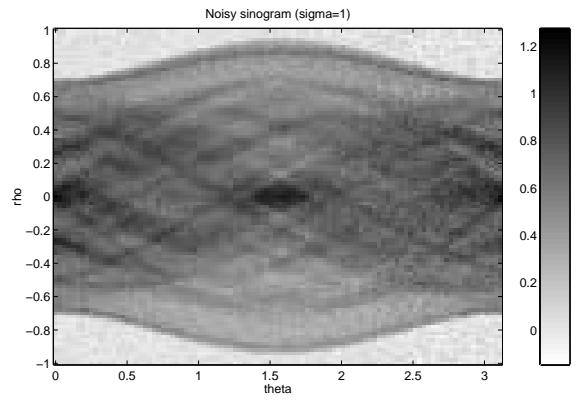


Figure 9.13 Noise corrupted sinogram with 100×101 pixels.

In order to quantify the quality of images here it is chosen to use the L2 measure of misfit.

$$L2 = \frac{\|\mathbf{x}^* - \mathbf{x}_{ref}\|_2}{\|\mathbf{x}_{ref}\|_2} \quad (9.73)$$

where \mathbf{x}^* is the estimated solution and \mathbf{x}_{ref} is the true solution. It should also be mentioned that constraints have been used to generate the following figures. After each iteration the solution has been limited to the interval between 0 and 10.

In Fig. 9.14 ART, EM, and LSCG has been used for reconstruction of the sinogram without noise and the L2 measures of misfit have been shown. For ART, both cyclical and random selection of row index are shown. It can be seen that ART is very fast, and using random selection of row index ends up with the lowest error after three full iterations. The figure also shows that EM approximately gets to the same level, but so much slower. Here the LSCG algorithm does not do a good job, and more constraining or regularization is needed here. It can be mentioned that Filtered Backprojection gets an L2-measure of 0.266 (and only requiring less than one second).

Fig. 9.15 has the same setup as Fig. 9.14 for the noise corrupted sinogram with $\sigma = 1$. Here ART and LSCG reaches approximately the same error level (but ART is by far the fastest here), and EM clearly demonstrates a better performance. Note also that EM has a very flat plateau at the lowest error, which is very nice when designing criteria for selection of the number of iterations. For the same sinogram Filtered Backprojection gives L2=0.437, so EM is the better algorithm in this case, even though the error is not Poisson distributed.

Finally for the high noise level $\sigma = 5$, Fig. 9.16 shows a very different behavior for ART. It clearly diverges, but the constraints implies that the solution stays limited (but does not approximate the true solution). Both LSCG and EM converges in the first nine iterations, and ends up with L2=0.62 and L2=0.57 respectively and in this case Filtered Backprojection ends up with L2=1.75. But note that the comparison is somewhat harsh, because that the filter in Filtered Backprojection is a ramp filter amplifying the noise, and in general the sinogram could be low-pass filtered with lowered cutoff at high noise levels.

For the previous three figures the system matrix has been modelled using the Radon transform of a square, cf. Subsection B.3.2. Now the results are compared to another choice of model for the noisy sinogram $\sigma = 1$. In Fig. 9.17 the binary model found in Eq. 9.13 and the Radon transform

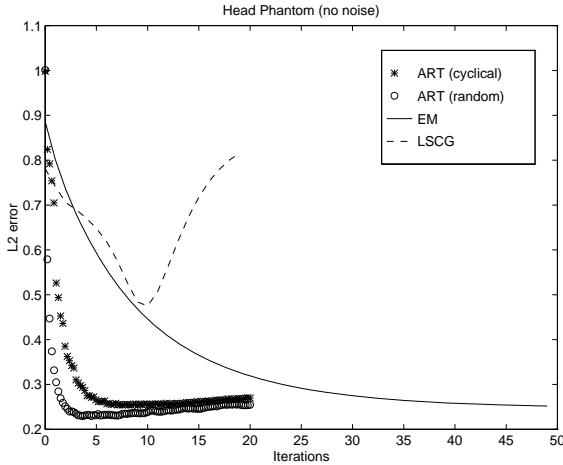


Figure 9.14 L2 measure of misfit as a function of number of full iterations when using ART, EM, and LSCG to reconstruct from the sinogram without noise. For ART both cyclical and random selection of row index are shown.

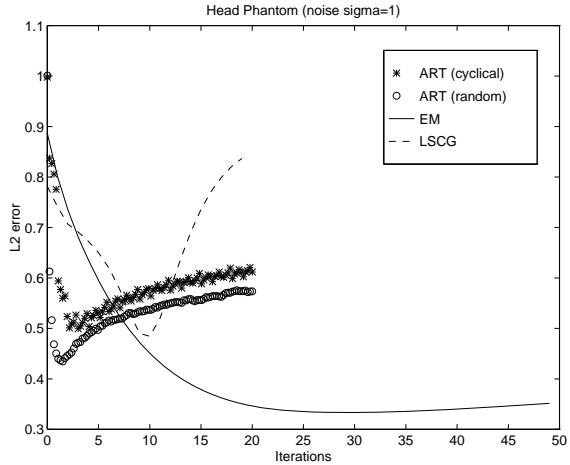


Figure 9.15 L2 measure of misfit as a function of number of full iterations when using ART, EM, and LSCG to reconstruct from the noisy sinogram ($\sigma = 1$).

of a square has been used to model the system matrix, and for ART is apparently the simple binary model which provides the lowest error, but not much difference is found.

For EM, three models have been tried. The binary model, the Radon transform of a square, and the sinc-interpolation based model, cf. Eq. 9.15. For the sinc model a cutoff value $\gamma = 0.2\Delta x$ was used to limit the size of the sparse stored system matrix to 20 MBytes (lowering γ with an additional factor of two implied twice as much memory was needed). Fig. 9.18 shows that EM gets to a lower error when the model is improved, and the sinc interpolation scheme and the Radon transform of a square gives the same convergence until iteration 16, where the truncation of the matrix values implies that the two models split up in performance. For LSCG shown in Fig. 9.19 an odd convergence is found. The two models ends up with the same error, but after a very different number of iterations, but it is clear that LSCG can and should be improved.

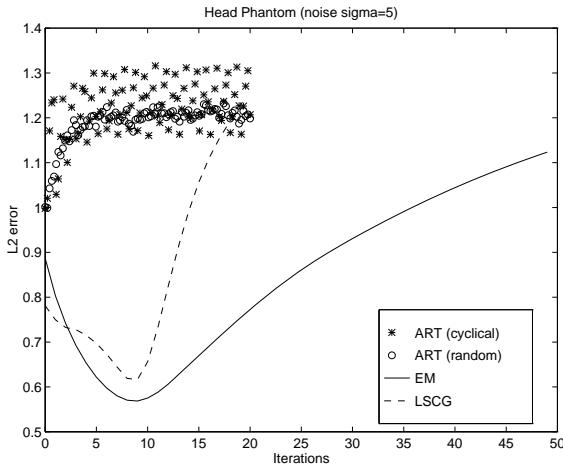


Figure 9.16 L2 as a function of iterations when using ART, EM, and LSCG ($\sigma = 5$).

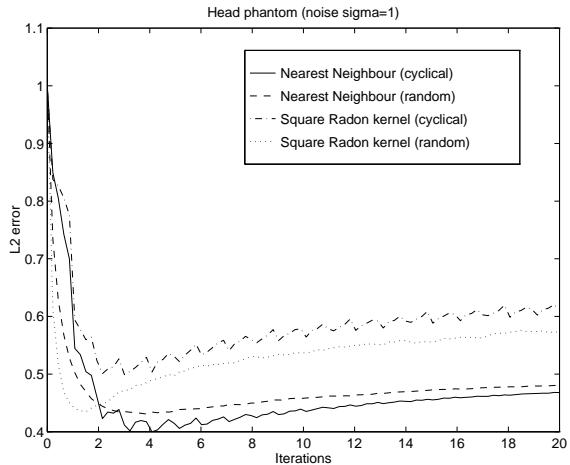


Figure 9.17 L2 as a function of iterations for ART for two models of the system matrix.

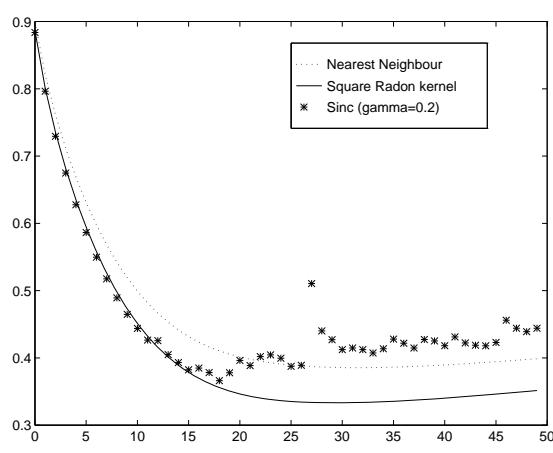


Figure 9.18 L2 as a function of iterations for EM for three models of the system matrix.

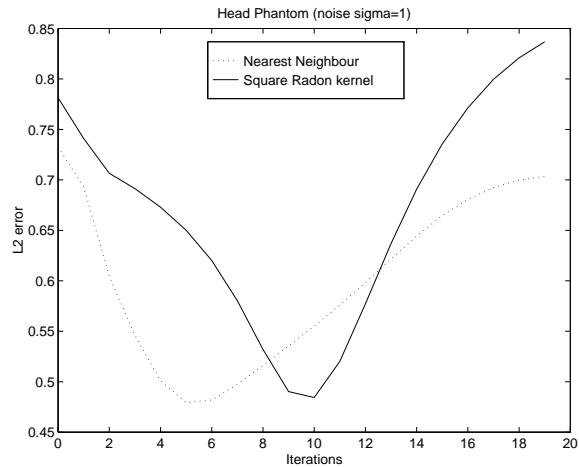


Figure 9.19 L2 as a function of iterations for LSCG for two models of the system matrix.

9.11.4 Reconstruction Using Constraints

In this example the EM and the LSCG algorithm has been used to reconstruct images of 201×201 samples from the sinogram shown in Fig. 9.9.

Figs. 9.20-9.25 first show the reconstructed images after 5, 10, and 15 iterations using EM with and without a simple constraints. In this case the constraints are implemented by placing an ellipse around the reconstructed head. After each iteration the values will be truncated to zero outside of the head. This is a very simple prior, which actually works out fine. The following figures use the same color scaling in a row, but not in a column. The color scaling has been truncated to see a reasonable image of the brain. Already after 5 iterations of EM is instability found in a single pixel originating from pixels and lines outside of the head. Fig. 9.22 uses a truncated color scaling, but a line can be seen to the right of the head, and Fig. 9.24 many artifacts are found in the image, which at some places have extremely high values. In the figures to the right the ellipse is clearly marked and no artifacts are found until after 10 iterations. The reconstructed images appear to be very nice and far better than the one shown in Fig. 9.11 where Filtered Backprojection was used. The intrinsic noise in the sinogram is not dominating in the reconstructed image, and edges appear to be more sharp, which is a feature that often is reported in the reconstruction literature.

Next reconstructed images using LSCG are shown in Figs. 9.26-9.31. Note that Figs. 9.26 and 9.27 show a very different result than Figs. 9.20 and 9.21, but after 10 iterations the reconstructed images from EM and LSCG look almost identical in the area of the brain. After 15 iterations LSCG shows signs of instability: Low-frequency components dominate the area outside of the brain, which is major source for the bad performance of LSCG showed in Section 9.11.3. But it is probably even worse that the values in the interior of the brain (the so called the white matter) are forced very close to zero. With even more iterations the reconstructed images gets worse.

This example clearly demonstrates the potential of the iterative methods compared to the (classical) direct reconstruction methods, but it is not a simple task to choose the number of iterations in a real-world case due to the numerous noise sources.

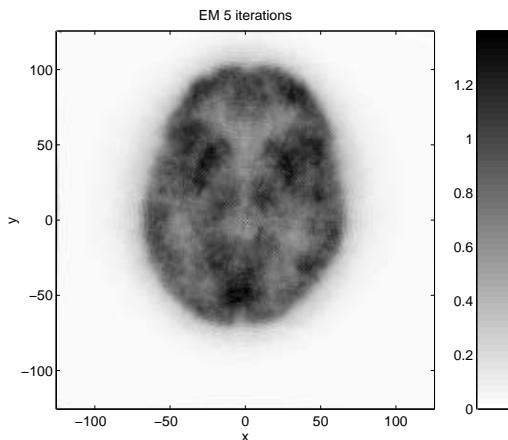


Figure 9.20 EM reconstructed image after 5 iterations.

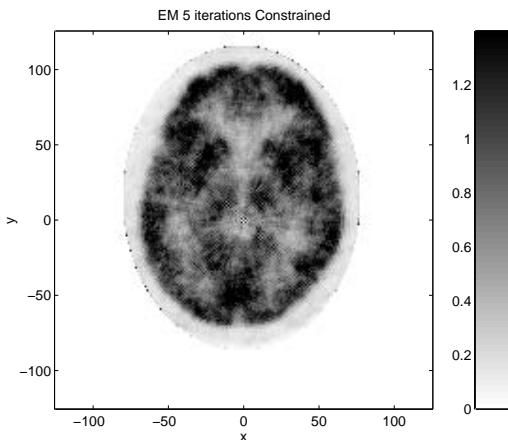


Figure 9.21 EM reconstructed image after 5 iteration with constraints.

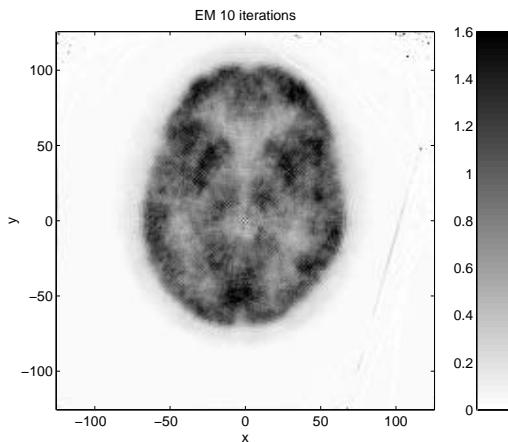


Figure 9.22 EM reconstructed image after 10 iterations.

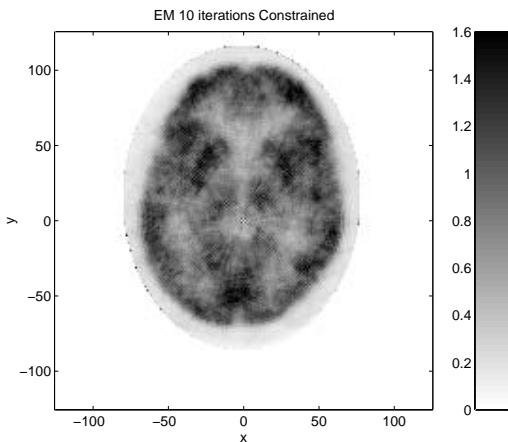


Figure 9.23 EM reconstructed image after 10 iteration with constraints.

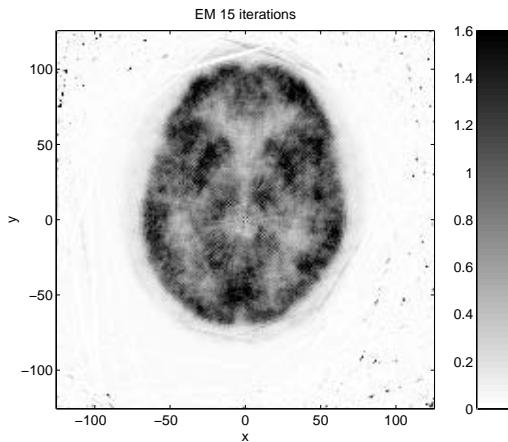


Figure 9.24 EM reconstructed image after 15 iterations.

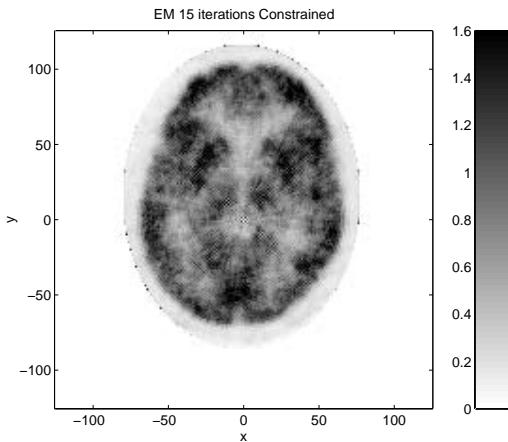


Figure 9.25 EM reconstructed image after 15 iteration with constraints.

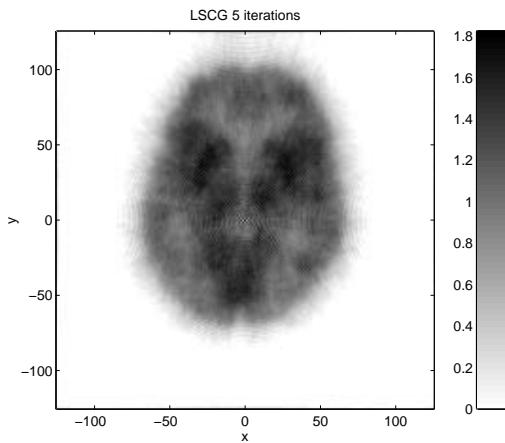


Figure 9.26 LSCG reconstructed image after 5 iterations.

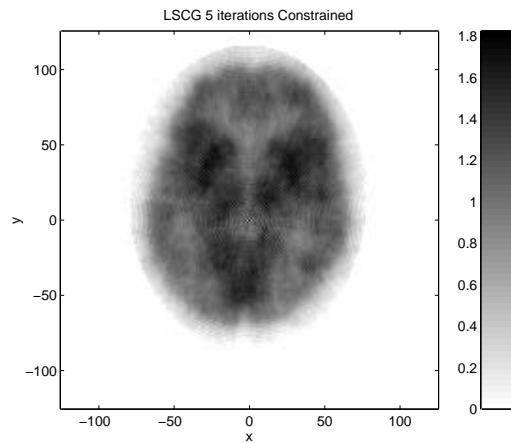


Figure 9.27 LSCG reconstructed image after 5 iteration with constraints.

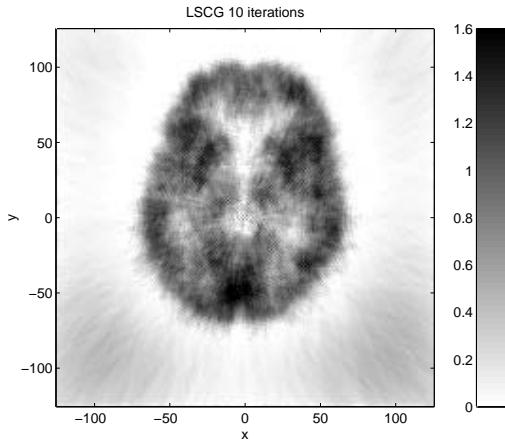


Figure 9.28 LSCG reconstructed image after 10 iterations.

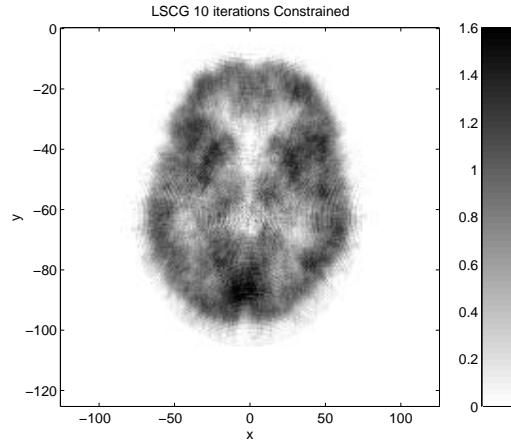


Figure 9.29 LSCG reconstructed image after 10 iteration with constraints.

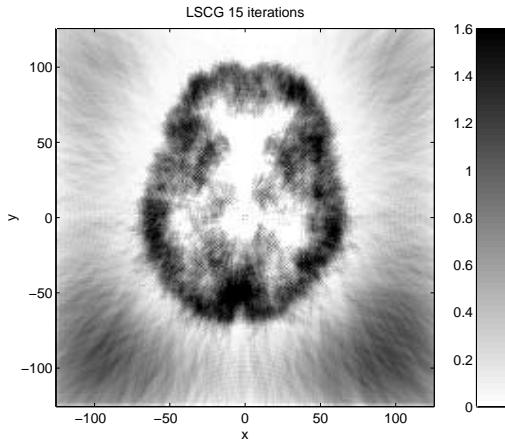


Figure 9.30 LSCG reconstructed image after 15 iterations.

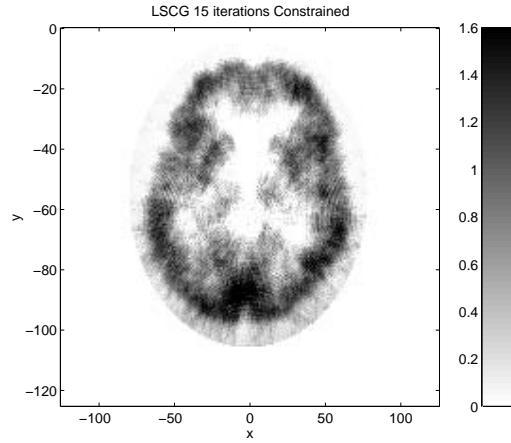


Figure 9.31 LSCG reconstructed image after 15 iteration with constraints.

9.11.5 Reconstruction Using Regularization

For the EM and the LSCG methods tests have been conducted using the Laplace operator as a regularizing operator L . For ART it can be found that the regularization term will add some (fixed) stability, but the regularization parameter λ is has no effect, due to the update scheme

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \frac{0 - \lambda \mathbf{l}_i^T \mathbf{x}^{(k)}}{\lambda \mathbf{l}_i^T \lambda \mathbf{l}_i} \lambda \mathbf{l}_i^T = \mathbf{x}^{(k)} - \frac{\mathbf{l}_i^T \mathbf{x}^{(k)}}{\mathbf{l}_i^T \mathbf{l}_i} \mathbf{l}_i^T \quad (9.74)$$

where \mathbf{l}_i^T is certain row of the regularization matrix \mathbf{L} . The conclusion is that regularization of ART will have to be devised with another scheme.

In Fig. 9.32 is shown that EM responds very negatively to the extra regularization rows in the (expanded) system matrix. The sinogram is the same as used in Section 9.11.3 with the noise amplitude $\sigma = 2$. From the figure is can be seen that a non-zero λ actually implies a faster divergence and a larger error. This is due to the fact that the extra rows of the system matrix have a very different statistical nature compared to the original system matrix.

For LSCG with the same sinogram, as shown in Fig. 9.33, the regularization term will stabilize the solution, i.e., the change in error per iteration is minimized, but it does not lead to a better solution in terms of L2 error.

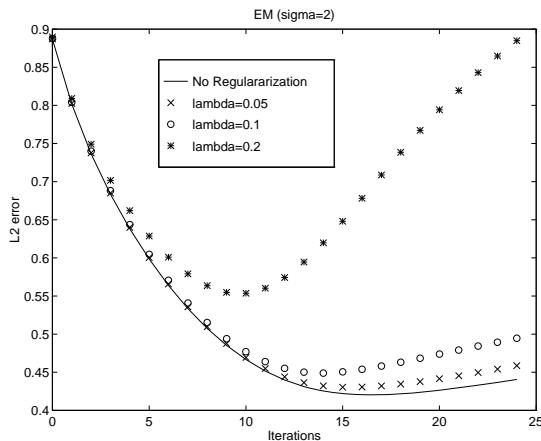


Figure 9.32 L2 measure as a function of the iteration number when reconstructing a noisy sinogram with EM and four different values of the regularization parameter λ .

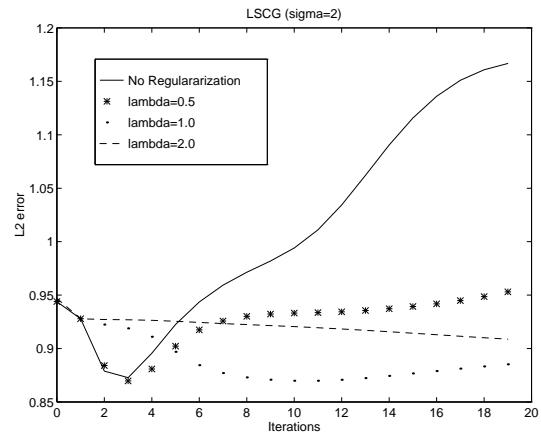


Figure 9.33 L2 measure as a function of the iteration number when reconstructing a noisy sinogram with LSCG and four different values of the regularization parameter λ .

9.12 Summary

This chapter demonstrated reconstruction methods based on linear algebra. It was shown that the system matrix can be modelled in different ways, here mostly shown from the Radon transform point of view.

The SVD was then used to indicate the structure of the singular values of the system matrix, but the SVD is very computationally demanding, hence it has currently a limited value for practical reconstruction purposes.

Some of the popular iterative reconstruction algorithms have been presented, and a very fast implementation of iterative reconstruction algorithms, comparable in speed to direct reconstruction methods, has been presented with theory and examples. The implementation is based on

storing of the system matrix in fast memory using sparse techniques. The approach is mainly applicable to 2D reconstruction, due to the requirements of a sufficient amount of memory, but in principle the method can also be applied for 3D reconstruction.

Examples have been shown demonstrating how different modelling schemes of the system matrix influences on the solution, and the use of constraints and regularization have been shown in a number of examples. It is also clear the many open ends exist in this field, which the number of publications at conferences and relevant journals clearly demonstrate.

Chapter 10

The 3D Radon Transform for Lines

So far the reconstruction techniques have been related to 2D reconstruction. One of the problems with 2D PET scanners is the very low photon efficiency. The problem is that only a small fraction of photon pairs are actually detected on the same detector ring. One obvious possibility is to use a multi ring detector setup where photons pairs can be detected at different rings, as shown in Fig. 10.1. The basic physics is as described in Section 6.2, but now the photon pairs can form a line with (in principle) arbitrary orientation, which leads to a much higher rate of pairs of photons.

This chapter will first describe lines in 3D, which then is used to generalize the Radon transform for lines in 3D. Some direct reconstruction methods are also shown. The implementation of some of the elements in direct and iterative reconstruction algorithms will be described. A software package has been developed providing both direct and iterative reconstruction methods. The chapter will also present some examples to illustrate 3D reconstruction.

In recent years commercial 3D PET scanners have become available, such as the GE Advance PET scanner and the Siemens/CTI Ecat Exact scanners. Some theoretical issues are still not fully resolved [67] and some practical, such as attenuation correction, can still be improved in 3D scanners. Currently the attenuation correction is often based on 2D measurements.

In Chapter 2 it was shown that the 2D Radon transform used lines are the fundamental shape. The Radon transform is also defined in higher dimensions [14], and in 3D the fundamental shape is a plane (and not a line in the 3D space), hence the ordinary 3D Radon transform cannot be used. Appendix D describes some of the fundamental properties, along with some reconstruction methods, and a method for using these techniques to reconstruct volumes from line integrals is shown.

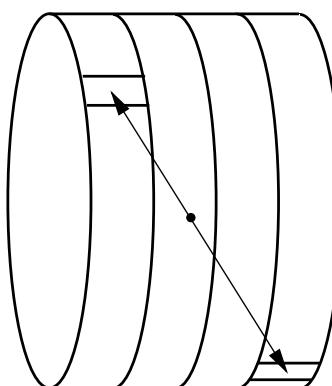


Figure 10.1 A multi ring PET scanner where pairs of photons can be detected at different rings.

10.1 Lines in a Three Dimensional Space

Lines in a three dimensional space can not be written in a single normal form as in the two dimensional case. Instead a parameter description can be used. In the following a line description using four parameters is shown. This description is the basis of direct inversion schemes. In general a line can be described by

$$\mathbf{r} = \mathbf{r}_0 + s\boldsymbol{\tau} \quad (10.1)$$

where s is the free parameter, and \mathbf{r}_0 is an offset vector. The vector $\boldsymbol{\tau}$ is a directional vector which can be normalized to unit length, i.e., $|\boldsymbol{\tau}| = 1$. The vector can, e.g., be described by

$$\boldsymbol{\tau} = \begin{pmatrix} \cos \theta \cos \phi \\ \sin \theta \cos \phi \\ \sin \phi \end{pmatrix} \quad (10.2)$$

The base point vector (or offset vector) \mathbf{r}_0 is described by two parameters u and v using two directional vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, both normalized to unit length.

$$\mathbf{r}_0 = u\boldsymbol{\alpha} + v\boldsymbol{\beta} \quad (10.3)$$

The coordinate system is shown in Fig. 10.2

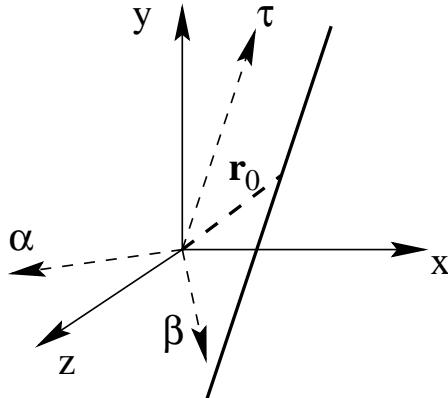


Figure 10.2 The (x, y, z) coordinate system and a line lying along the $\boldsymbol{\tau}$ -axis. The base point vector \mathbf{r}_0 can be written as a linear combination of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

It can be chosen that the three vectors $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ form an orthogonal basis, and the last rotational degree of freedom can be used for specifying that the z -component of $\boldsymbol{\alpha}$ is zero. In [68], $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are defined as

$$\boldsymbol{\alpha} = \begin{pmatrix} -\sin \theta \\ \cos \theta \\ 0 \end{pmatrix} \text{ and } \boldsymbol{\beta} = \begin{pmatrix} -\cos \theta \sin \phi \\ -\sin \theta \sin \phi \\ \cos \phi \end{pmatrix} \quad (10.4)$$

It should be mentioned that other symbols of the vectors can be found in the literature, e.g., [64, 65, 63].

Following [67], line integrals through a three dimensional space can be expressed by the symbols defined in Eqs. 10.1 and 10.3

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta}) \, ds \quad (10.5)$$

where (θ, ϕ, u, v) is the four-dimensional parameter domain and in PET the (measured) values are still named the sinogram, like in 2D.

The mapping from the (x, y, z) domain to (s, u, v) is very useful

$$\boldsymbol{r} = s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta} \Rightarrow \quad (10.6)$$

$$\boldsymbol{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos \theta \cos \phi & -\sin \theta & -\cos \theta \sin \phi \\ \sin \theta \cos \phi & \cos \theta & -\sin \theta \sin \phi \\ \sin \phi & 0 & \cos \phi \end{pmatrix} \begin{pmatrix} s \\ u \\ v \end{pmatrix} = \boldsymbol{Q}\boldsymbol{p} \quad (10.7)$$

Using that the base vectors are orthogonal and normalized to unit length, hence the rotation matrix \boldsymbol{Q} is unitary, i.e.,

$$\boldsymbol{p} = \boldsymbol{Q}^{-1}\boldsymbol{r} = \boldsymbol{Q}^T\boldsymbol{r} \quad (10.8)$$

One feature of the matrix \boldsymbol{Q} is that only two angles are used compared to a general rotation matrix which uses three degrees of freedom, i.e., three angles. The definition of the line integrals in Eq. 10.5 implies that a three dimensional function $g(x, y, z)$ is transformed into a four dimensional parameter domain $\check{g}(\theta, \phi, u, v)$. This change in dimensions will naturally impose some inherent difficulties to be analyzed later.

It has been chosen to denote the line integrals of $g(\boldsymbol{r})$ with the symbol \check{g} . It can be argued that Eq. 10.5 is not a Radon transform of the function $g(x, y, z)$. It is not, cf. Eq. D.1, a three dimensional Radon transform nor a four dimensional, but it can be seen as a hybrid or generalized Radon transform for lines through a three dimensional space.

In the following, Eq. 10.5 is called the 3D line Radon transform or simply the Radon transform. In the rest of this chapter only the 3D line Radon definition will be considered, and the actual parameters will (as previously) uniquely determine which type of transformation is used. From the definition of the 3D line Radon transform some basic rules are derived and shown in Appendix E. Additionally, the Radon transform of simple geometrical functions are given in the same appendix.

The 3D line Radon transform can be perceived as a convolution between a function $g(\boldsymbol{r})$ and a kernel h expressed in the coordinates \boldsymbol{p} for a given combination of the angles θ and ϕ .

$$h(\boldsymbol{r}) = h(\boldsymbol{p}) = h(s, u, v) = \delta(u)\delta(v) \quad (10.9)$$

which can be seen from the following, where $\ast\ast\ast$ implies a three dimensional convolution in the parameters (s, u, v) .

$$\check{g}(\theta, \phi, u, v) = g(\boldsymbol{r}) \ast\ast\ast \delta(u)\delta(v) \quad (10.10)$$

$$\begin{aligned} &= \int_{v'=-\infty}^{\infty} \int_{u'=-\infty}^{\infty} \int_{s'=-\infty}^{\infty} g(s'\boldsymbol{\tau} + u'\boldsymbol{\alpha} + v'\boldsymbol{\beta}) \delta(u - u')\delta(v - v') \, ds' \, du' \, dv' \\ &= \int_{s'=-\infty}^{\infty} g(s'\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta}) \, ds' \end{aligned} \quad (10.11)$$

Besides the new integration variable s' , the result matches Eq. 10.5.

At this time it could be appropriate to look back to the two dimensional results. This can be done by choosing $\phi = 0$, $v = z_0$ and $u = \rho$, thus the line integral reduces to

$$\check{g}(\rho, 0, \theta) = \int_{-\infty}^{\infty} g(s \cos \theta - \rho \sin \theta, s \sin \theta + \rho \cos \theta, z_0) ds \quad (10.12)$$

From Eq. 2.8, this can be recognized as the ordinary two dimensional Radon transform of the function in the plane $z = z_0$, where the angular parameter θ has been rotated 90 degrees compared to the definition of the two dimensional Radon transform.

10.1.1 Limiting the 3D line parameters

The angular parameters θ and ϕ control the orientation of the line and it is obvious that they can be bounded as normal spherical angles, i.e., $0 \leq \theta < 2\pi$ and $-\pi/2 \leq \phi < \pi/2$. This corresponds to a full angular coverage, but the interval of the θ -parameter can further be restricted to $0 \leq \theta < \pi$, due to Eqs. 10.3 and 10.4

$$\check{g}(\theta, \phi, u, v) = \check{g}(\theta + \pi, -\phi, -u, v) \quad (10.13)$$

This little trick will enable a reduction of the parameter domain with a factor of two. The trick is not used in [67] and [66], hence the filters found in these papers will differ with a constant (a factor of two) from those presented in the following. This problem relates to the difference between the adjoint and the backprojection operator, which for 2D was mentioned in Section 9.3.

The reconstruction geometry will now be based on truncated elliptical tubes, relevant to multi ring PET scanners, where the geometry Ω_ψ is defined as

$$\Omega_\psi = \{0 \leq \theta < \pi \vee |\phi| \leq \psi\} \quad (10.14)$$

where ψ is called the axial acceptance angle. The geometry, where $\psi = 0$, i.e., Ω_0 corresponds to the two dimensional case treated in the previous chapters (see also Eq. 10.12), and $\Omega_{\pi/2}$ is full angular coverage. Actually Ω_ψ is of interest due to the possibility to derive direct reconstruction formulas, but actually 3D PET scanners do not measure in this geometry - some of the line orientations are missing. This aspect will be covered in Section 10.6.

The two remaining parameters u and v are translation parameters controlling the position of the line in a plane perpendicular to the directional vector τ . Theoretically u and v , can only be bounded if it can be assumed that the scanned object is finite. The origin of the three dimensional coordinate system is placed (approximatively) at the center of the object. This implies that the distance from the origin to the outer edge of the object is minimized to r_{\max} , or in mathematical terms

$$g(x, y, z) = 0 \text{ for } x^2 + y^2 + z^2 > r_{\max}^2 \quad (10.15)$$

If r_{\max} is found then u and v can be restricted, because the distance from the origin to the volume element at $\mathbf{r} = s\tau + u\alpha + v\beta$ are $|\mathbf{r}| = \sqrt{s^2 + u^2 + v^2}$. This is due to othonormality of the three expansion vectors. Assuming that $u^2 + v^2 > r_{\max}^2$, implies that the integral is zero, because the function g is zero along the line. This result implies that each of the translation parameters can be bounded as

$$-r_{\max} \leq u \leq r_{\max} \quad \text{and} \quad -r_{\max} \leq v \leq r_{\max} \quad (10.16)$$

10.2 Fourier Slice Reconstruction in 3D

From the line integrals defined in Eq. 10.5, the function $g(\mathbf{r})$ can be recovered using Fourier techniques, and the Fourier Slice Theorem is now derived for 3D line integrals by applying the two dimensional Fourier transform to each of the (u, v) -planes in Eq. 10.5.

$$\check{G}(\theta, \phi, \nu_u, \nu_v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \check{g}(\theta, \phi, u, v) e^{-j2\pi(u\nu_u + v\nu_v)} du dv \quad (10.17)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta}) e^{-j2\pi(u\nu_u + v\nu_v)} du dv ds \quad (10.18)$$

which indicates a close connection to the three dimensional Fourier transform of the function $g(\mathbf{r})$.

$$G(\boldsymbol{\nu}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\mathbf{r}) e^{-j2\pi\mathbf{r}\cdot\boldsymbol{\nu}} d\mathbf{r} \quad (10.19)$$

$$g(\mathbf{r}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(\boldsymbol{\nu}) e^{j2\pi\mathbf{r}\cdot\boldsymbol{\nu}} d\boldsymbol{\nu} \quad (10.20)$$

where $\boldsymbol{\nu}$ is the three dimensional frequency vector.

Now the integration parameters in Eq. 10.18 is changed into x, y and z , i.e., \mathbf{r} . It is used that $\boldsymbol{\tau}, \boldsymbol{\alpha}$, and $\boldsymbol{\beta}$ are orthogonal.

$$\mathbf{r} = s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta} \Rightarrow \quad (10.21)$$

$$u = \boldsymbol{\alpha} \cdot \mathbf{r} \text{ and } v = \boldsymbol{\beta} \cdot \mathbf{r} \Rightarrow \quad (10.22)$$

$$u\nu_u + v\nu_v = \mathbf{r} \cdot (\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta}) \Rightarrow \quad (10.23)$$

$$\check{G}(\theta, \phi, \nu_u, \nu_v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\mathbf{r}) e^{-j2\pi\mathbf{r}\cdot(\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta})} d\mathbf{r} \quad (10.24)$$

This interesting result shows that the two dimensional Fourier transform of the line integrals is the three dimensional Fourier transform of the function to be reconstructed [67].

$$\check{G}(\theta, \phi, \nu_u, \nu_v) = G(\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta}) \quad (10.25)$$

which is a Fourier Slice Theorem for line integrals in a three dimensional space. It implies that the function $g(\mathbf{r})$ can be recovered by applying a two dimensional Fourier transform to the sinogram for all values of (θ, ϕ) , followed by a mapping of the spectrum, and finally recovering the desired volume by using a three dimensional inverse Fourier transform.

$$G(\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \check{g}(\theta, \phi, u, v) e^{-j2\pi(u\nu_u + v\nu_v)} du dv \quad (10.26)$$

$$g(\mathbf{r}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(\boldsymbol{\nu}) e^{j2\pi\mathbf{r}\cdot\boldsymbol{\nu}} d\boldsymbol{\nu} \quad (10.27)$$

A non-trivial problem arises when implementing the mapping of the spectrum. Each frequency point $\boldsymbol{\nu}$ is mapped into $\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta}$, but $\boldsymbol{\nu}$ is a three-dimensional vector and $\nu_u\boldsymbol{\alpha} + \nu_v\boldsymbol{\beta}$ has four degrees of freedom. This implies that each $\boldsymbol{\nu}$ matches an infinite set of parameters (θ, ϕ, u, v) . One possible solution is to use weighted averages of the possible 4D frequency vectors for each value of $\boldsymbol{\nu}$, and this problem is still an area of active research, though in [69] several aspects have been covered.

10.3 Backprojection Based Inversion of Line Integrals in 3D

Analogous to the 2D Filtered Backprojection, it is possible to filter the line projections and then make a backprojection of the sinogram into the volume domain (x, y, z) [67]. The backprojection operator in 3D is now analyzed from the transformation of a point source. Again, it is used that any function can be resolved into a weighted sum (integral) of delta functions.

$$g(\mathbf{r}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\mathbf{r}_0) \delta(x - x_0) \delta(y - y_0) \delta(z - z_0) dx_0 dy_0 dz_0 \quad (10.28)$$

$$\equiv \int_{-\infty}^{\infty} g(\mathbf{r}_0) \delta(\mathbf{r} - \mathbf{r}_0) d\mathbf{r}_0 \quad (10.29)$$

This implies that the corresponding Radon transform is given by

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(\mathbf{r}_0) \left(\int_{s=-\infty}^{\infty} \delta(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta} - \mathbf{r}_0) ds \right) d\mathbf{r}_0 \quad (10.30)$$

This shows that any function $g(\mathbf{r})$ can be Radon transformed, if the point source can be transformed. Now this will be done.

$$g_p(\mathbf{r}) = \delta(\mathbf{r} - \mathbf{r}_0) \Rightarrow \quad (10.31)$$

$$\check{g}_p(\theta, \phi, u, v) = \int_{-\infty}^{\infty} \delta(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta} - \mathbf{r}_0) ds \quad (10.32)$$

$$= \int_{-\infty}^{\infty} \delta((s - s_0)\boldsymbol{\tau} + (u - u_0)\boldsymbol{\alpha} + (v - v_0)\boldsymbol{\beta}) ds \quad (10.33)$$

$$= \int_{-\infty}^{\infty} \delta(\tilde{s}\boldsymbol{\tau} + (u - u_0)\boldsymbol{\alpha} + (v - v_0)\boldsymbol{\beta}) d\tilde{s} \quad (10.34)$$

Here the unambiguous substitution $\mathbf{r}_0 = s_0\boldsymbol{\tau} + u_0\boldsymbol{\alpha} + v_0\boldsymbol{\beta}$ has been used. Using that the delta function will be non-zero if and only if the argument is a zero length vector implies that the result will be non-zero only if $u = u_0 = \mathbf{r}_0 \cdot \boldsymbol{\alpha}$ and $v = v_0 = \mathbf{r}_0 \cdot \boldsymbol{\beta}$, and because the base vectors are orthogonal the result can be expressed using the delta function.

$$\check{g}_p(\theta, \phi, u, v) = \delta(u - u_0) \delta(v - v_0) = \delta(u - \mathbf{r}_0 \cdot \boldsymbol{\alpha}) \delta(v - \mathbf{r}_0 \cdot \boldsymbol{\beta}) \quad (10.35)$$

This can also be found directly from Eq. 10.10. It is a very important result which can be used to formulate a backprojection operator in 3D [67].

$$\hat{g} = \int_{\Omega} \check{g}(\theta, \phi, u = \mathbf{r} \cdot \boldsymbol{\alpha}, v = \mathbf{r} \cdot \boldsymbol{\beta}) d\Omega = \int_{\theta=0}^{\pi} \int_{\phi=-\psi}^{\psi} \check{g}(\theta, \phi, u = \mathbf{r} \cdot \boldsymbol{\alpha}, v = \mathbf{r} \cdot \boldsymbol{\beta}) \cos \phi d\phi d\theta \quad (10.36)$$

For convenience the integration over angles is written as a single integral with index Ω . If the geometry is, e.g., Ω_{ψ} the integration becomes the last part of Eq. 10.36, where the term $\cos \phi$ is the Jacobian, found when converting to spherical coordinates.

10.4 Filtering after Backprojection of Line Integrals in 3D

Like in two dimensional backprojection algorithms, a high pass filter is needed either before or after the backprojection [67]. In this section the aim is to find a condition for the filter used after backprojection, i.e., on the volume. It can be derived by backprojecting the Radon transform of a function g . Inserting Eq. 10.5 in 10.36 gives

$$\hat{g}(\mathbf{r}) = \int_{\Omega} \int_{s=-\infty}^{\infty} g((\mathbf{r} \cdot \boldsymbol{\alpha})\boldsymbol{\alpha} + (\mathbf{r} \cdot \boldsymbol{\beta})\boldsymbol{\beta} + s\boldsymbol{\tau}) ds d\Omega \quad (10.37)$$

Now it is utilized that the s -integration can be shifted along the $\boldsymbol{\tau}$ -axis, like it was done from Eq. 10.33 to 10.34. Here the offset is chosen to $\mathbf{r} \cdot \boldsymbol{\tau}$.

$$\hat{g}(\mathbf{r}) = \int_{\Omega} \int_{s=-\infty}^{\infty} g((\mathbf{r} \cdot \boldsymbol{\alpha})\boldsymbol{\alpha} + (\mathbf{r} \cdot \boldsymbol{\beta})\boldsymbol{\beta} + (\mathbf{r} \cdot \boldsymbol{\tau})\boldsymbol{\tau} + s\boldsymbol{\tau}) ds d\Omega \quad (10.38)$$

$$= \int_{\Omega} \int_{s=-\infty}^{\infty} g(\mathbf{r} + s\boldsymbol{\tau}) ds d\Omega \quad (10.39)$$

which can be recognized to be a convolution, thus a 3D Fourier transform can be applied on both sides.

$$\tilde{G}(\boldsymbol{\nu}) = \int_{\Omega} \int_{s=-\infty}^{\infty} \int_{\mathbf{r}=-\infty}^{\infty} g(\mathbf{r} + s\boldsymbol{\tau}) e^{-j2\pi\boldsymbol{\nu} \cdot \mathbf{r}} d\mathbf{r} ds d\Omega \quad (10.40)$$

$$= \int_{\Omega} \int_{s=-\infty}^{\infty} \int_{\tilde{\mathbf{r}}=-\infty}^{\infty} g(\tilde{\mathbf{r}}) e^{-j2\pi\boldsymbol{\nu} \cdot (\tilde{\mathbf{r}} - s\boldsymbol{\tau})} d\tilde{\mathbf{r}} ds d\Omega \quad (10.41)$$

$$= \int_{\Omega} \int_{s=-\infty}^{\infty} G(\boldsymbol{\nu}) e^{j2\pi s\boldsymbol{\nu} \cdot \boldsymbol{\tau}} ds d\Omega \quad (10.42)$$

$$= \int_{\Omega} G(\boldsymbol{\nu}) \left(\int_{s=-\infty}^{\infty} e^{j2\pi s\boldsymbol{\nu} \cdot \boldsymbol{\tau}} ds \right) d\Omega \quad (10.43)$$

$$= G(\boldsymbol{\nu}) \int_{\Omega} \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega \equiv \frac{G(\boldsymbol{\nu})}{H_a(\boldsymbol{\nu})} \quad (10.44)$$

This result is a 3D Filtering after Backprojection reconstruction method. The spectrum of the backprojected sinogram is multiplied with the filter $H_a(\boldsymbol{\nu})$, shown in Eq. 10.45. Finally Eq. 10.20 is used recover the desired volume.

$$H_a(\boldsymbol{\nu}) = \frac{1}{\int_{\Omega} \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega} \quad (10.45)$$

In the geometry Ω_ψ , the filter can also be found on an analytical form. Several papers, e.g., [66, 67], have shown different filters, which again have been shown to only differ with a normalization constant. Due to the circular symmetry around the z -axis the filter can be calculated, with ν_y , i.e., the y -component of the frequency vector, set to zero.

$$\int_{\Omega} \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega = \int_{\phi=-\psi}^{\psi} \int_{\theta=0}^{\pi} \delta(\nu_x \cos \theta \cos \phi + \nu_z \sin \phi) \cos \phi d\theta d\phi \quad (10.46)$$

$$= 2 \int_{\phi=-\psi}^{\psi} \int_{\theta=0}^{\pi/2} \delta(\nu_x \cos \theta + \nu_z \tan \phi) d\theta d\phi \quad (10.47)$$

$$= 2 \int_{\phi=-\gamma}^{\gamma} \frac{1}{|\nu_x \sin \theta_0|} d\phi \text{ where } \begin{cases} \gamma = \min\{\psi, |\arctan(\nu_x/\nu_z)|\} \\ \nu_x \cos \theta_0 + \nu_z \tan \phi = 0 \end{cases} \quad (10.48)$$

$$= 2 \int_{\phi=-\gamma}^{\gamma} \frac{\cos \phi}{\sqrt{\nu_x^2 - (\nu_x^2 + \nu_z^2) \sin^2 \phi}} d\phi \quad (10.49)$$

The symmetry of the integral is used again to get the general result, i.e., in the last line $|\nu_x|$ is substituted back to $\sqrt{\nu_x^2 + \nu_y^2}$, and the result is

$$H_a(\boldsymbol{\nu}) = \begin{cases} \frac{\boldsymbol{\nu}}{\pi} & \text{if } \sqrt{\nu_x^2 + \nu_y^2} < |\boldsymbol{\nu}| \sin \psi \\ \boldsymbol{\nu} \left(2 \arcsin \left(\frac{|\boldsymbol{\nu}|}{\sqrt{\nu_x^2 + \nu_y^2}} \sin \psi \right) \right)^{-1} & \text{otherwise} \end{cases} \quad (10.50)$$

Two things can be noted in Eq. 10.50. First, letting $\psi \rightarrow \pi/2$ implies full angular coverage and Eq. 10.50 becomes very easy, i.e., $H_a(\boldsymbol{\nu}) = \boldsymbol{\nu}/\pi$. On the other hand letting $\psi \rightarrow 0$ implies that the filter becomes close to $H_a(\boldsymbol{\nu}) = \sqrt{\nu_x^2 + \nu_y^2}/(2\psi)$. If neglecting the normalization constant 2ψ the filter can be recognized as the filter used in 2D Filtering after Backprojection, cf. Eq. 7.24. In Fig. 10.3 is shown the angular dependence of the filter from Eq. 10.50. The filter has been multiplied by $\psi/|\boldsymbol{\nu}|$ and on the first axis is shown the angle $\phi_{\boldsymbol{\nu}} = \arcsin(\nu_z/|\boldsymbol{\nu}|)$.

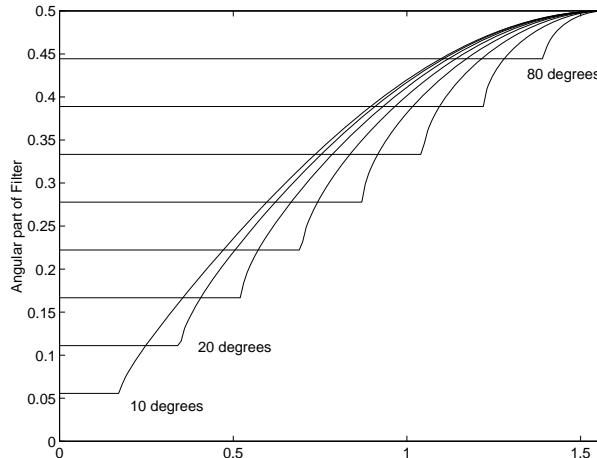


Figure 10.3 Normalized angular part of the filter as a function of $\phi_{\boldsymbol{\nu}} = \arcsin(\nu_z/|\boldsymbol{\nu}|)$ for $\psi = 10^\circ, 20^\circ, \dots, 80^\circ$.

10.5 Filtered Backprojection of Line Integrals in 3D

Analogous to the two dimensional Filtered Backprojection method, the filtering can be done before the backprojection of the line integrals in 3D. Here a two dimensional filter $h_b(\theta, \phi, u, v)$ is convolved in the (u, v) -plane of the sinogram for each value of (θ, ϕ) . After the filtering shown in Eq. 10.51, the backprojection operator is used as shown in Eq. 10.53.

$$\bar{g}(\theta, \phi, u, v) = h_b(\theta, \phi, u, v) * \check{g}(\theta, \phi, u, v) \quad (10.51)$$

$$= \int_{u'=-\infty}^{\infty} \int_{v'=-\infty}^{\infty} h_b(\theta, \phi, u - u', v - v') \check{g}(\theta, \phi, u', v') du' dv' \quad (10.52)$$

$$g(\mathbf{r}) = \int_{\Omega} \bar{g}(\theta, \phi, \mathbf{r} \cdot \boldsymbol{\alpha}, \mathbf{r} \cdot \boldsymbol{\beta}) d\Omega \quad (10.53)$$

The criterion that the filter in 3D Filtered Backprojection will have to satisfy can be derived by choosing $g(\mathbf{r})$ as a point source and requiring that Eqs. 10.51 and 10.53 are self-consistent, i.e.,

$$g(\mathbf{r}) = \delta(\mathbf{r}) \Rightarrow \check{g}(\theta, \phi, u, v) = \delta(u)\delta(v) \Rightarrow \quad (10.54)$$

$$\delta(\mathbf{r}) = \int_{\Omega} h_b(\theta, \phi, \mathbf{r} \cdot \boldsymbol{\alpha}, \mathbf{r} \cdot \boldsymbol{\beta}) d\Omega \quad (10.55)$$

The last equation can also be viewed in the 3D Fourier domain

$$1 = \int_{\Omega} \left(\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_b(\theta, \phi, \mathbf{r} \cdot \boldsymbol{\alpha}, \mathbf{r} \cdot \boldsymbol{\beta}) e^{-j2\pi\boldsymbol{\nu} \cdot \mathbf{r}} d\mathbf{r} \right) d\Omega \quad (10.56)$$

$$= \int_{\Omega} \left(\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_b(\theta, \phi, u, v) e^{-j2\pi(s\boldsymbol{\nu} \cdot \boldsymbol{\tau} + u\boldsymbol{\nu} \cdot \boldsymbol{\alpha} + v\boldsymbol{\nu} \cdot \boldsymbol{\beta})} d\mathbf{p} \right) d\Omega \quad (10.57)$$

$$= \int_{\Omega} H_b(\theta, \phi, \boldsymbol{\nu} \cdot \boldsymbol{\alpha}, \boldsymbol{\nu} \cdot \boldsymbol{\beta}) \int_{-\infty}^{\infty} e^{-j2\pi s\boldsymbol{\nu} \cdot \boldsymbol{\tau}} ds d\Omega \quad (10.58)$$

$$= \int_{\Omega} H_b(\theta, \phi, \boldsymbol{\nu} \cdot \boldsymbol{\alpha}, \boldsymbol{\nu} \cdot \boldsymbol{\beta}) \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega \quad (10.59)$$

where the 2D Fourier transform of the filter has been used

$$H_b(\theta, \phi, \nu_u, \nu_v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_b(\theta, \phi, u, v) e^{-j2\pi(u\nu_u + v\nu_v)} du dv \quad (10.60)$$

In a given geometry, several filters are valid [67], due to the 4D to 3D transformation during reconstruction. In general, part of the filters belong to a null-space, which could be used to improve noise performance without altering the signal reconstruction, though more research is needed to derive appropriate filters.

In [67] a criterion is given which makes Filtered Backprojection in a sense equivalent to Filtering after Backprojection.

$$H_b(\nu_u, \nu_v) = H_a(\nu_u \boldsymbol{\alpha} + \nu_v \boldsymbol{\beta}) \quad (10.61)$$

Defining $\nu_x = \nu_u \alpha_x + \nu_v \beta_x$ and $\nu_y = \nu_u \alpha_y + \nu_v \beta_y$, then Eqs. 10.50 and 10.61 imply that one valid filter is

$$H_b(\nu_u, \nu_v) = \begin{cases} \frac{\sqrt{\nu_u^2 + \nu_v^2}}{\pi} & \text{if } \sqrt{\nu_x^2 + \nu_y^2} < |\boldsymbol{\nu}| \sin \psi \\ \frac{\sqrt{\nu_u^2 + \nu_v^2}}{2 \arcsin \left(\frac{\sqrt{\nu_u^2 + \nu_v^2}}{\sqrt{\nu_x^2 + \nu_y^2}} \sin \psi \right)} & \text{otherwise} \end{cases} \quad (10.62)$$

In the geometry $\Omega_{\pi/2}$, the filter is very simple

$$H_b(\theta, \phi, \nu_u, \nu_v) = \frac{1}{\pi} \sqrt{\nu_u^2 + \nu_v^2} \quad (10.63)$$

The $\Omega_{\pi/2}$ -filter can be validated by inserting Eq. 10.63 in 10.59.

$$\int_{\Omega_{\pi/2}} H_b(\theta, \phi, \boldsymbol{\nu} \cdot \boldsymbol{\alpha}, \boldsymbol{\nu} \cdot \boldsymbol{\beta}) \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega \quad (10.64)$$

$$= \frac{1}{\pi} \int_{\Omega_{\pi/2}} \sqrt{(\boldsymbol{\nu} \cdot \boldsymbol{\alpha})^2 + (\boldsymbol{\nu} \cdot \boldsymbol{\beta})^2} \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega \quad (10.65)$$

$$= \frac{1}{\pi} \int_{\Omega_{\pi/2}} \sqrt{(\boldsymbol{\nu} \cdot \boldsymbol{\alpha})^2 + (\boldsymbol{\nu} \cdot \boldsymbol{\beta})^2 + (\boldsymbol{\nu} \cdot \boldsymbol{\tau})^2} \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega \quad (10.66)$$

$$= \frac{1}{\pi} \int_{\Omega_{\pi/2}} |\boldsymbol{\nu}| \delta(\boldsymbol{\nu} \cdot \boldsymbol{\tau}) d\Omega = 1 \quad (10.67)$$

In the last line the result from Eqs. 10.46-10.50 are used, with $\psi = \pi/2$.

10.6 Reconstruction Scheme for 3D Multi Ring PET Scanners

One of the fundamental problems in 3D PET reconstruction is that the Ω_ψ geometry defined in Eq. 10.14 does not perfectly match the measuring situation with multi ring PET scanners. In order to detect many events (detection of paired photons) a large scanner is needed, which leads to a high value of ψ , but it can be shown, e.g., [70] that a large value of ψ implies that scanners will lack some areas of the sinogram. The relative importance of this problem increases with ψ .

In the very famous article [70] by Kinahan, Harrop and Rogers, a three-stage process is proposed, namely that a crude volume, actually a set of axial slices, is reconstructed from the events lying (approximately) in the same axial plane, cf. Eq. 10.12, using 2D Filtered Backprojection. From this volume of reconstructed slices, the missing parts of the four-dimensional sinogram can be generated using the Radon transform, Eq. 10.5. Finally, the complete sinogram are used to reconstruct the volume using either Section 10.4 or 10.5. An implementation is reviewed in [71], where it is shown that the approach is viable and it is shown that an improvement of the volumes quality is obtained, compared to 2D reconstruction of the axial planes from the sinogram. The implementation was done on a VAX 8650 where the total reconstruction algorithm requires 236 minutes for a volume of $100 * 100 * 50$ voxels and a sinogram with $9 * 60 * 100 * 50$ samples.

This technique has become very popular. Currently, almost all 3D reconstruction articles relate their results to this method and the Kinahan & Rogers reprojection technique (called 3DRP in [110]) is also used in commercial scanners.

It should be mentioned that other strategies exist to reconstruct volumes from line integrals. In [121] the SSRB method (Single Slice Rebinning) is presented. A simple approximation is used, namely that the four-dimensional sinogram first is rebinned into a set of 2D sinograms, by projecting onto the nearest axial slice. These 2D sinograms are then reconstructed using 2D techniques. This approach is much faster, and can be implemented using significantly less memory, compared to the “true” 3D reconstruction techniques. In [110] a comparison has been made showing that for a PET scanner with $\psi = 9^\circ$ the SSRB method can essentially be as accurate as the 3DRP method, and for the HEAD PENN-PET scanner with $\psi = 27^\circ$ (developed at the University of Pennsylvania) a more accurate reconstruction is found using 3DRP.

Another strategy has recently been developed by Defrise [122]. Basically the idea is like the SSRB, i.e., to map the 4D sinogram onto a set of 2D sinograms, but here the mathematical development is based on a frequency-distance relation in the sinogram. In this method the Fourier transform of the 4D sinogram is mapped onto a set of 2D sinograms in the Fourier domain. For practical PET usage this method will also need additional parts of the sinogram found by use of, e.g., the Kinahan & Rogers technique.

As stated in Section 9.1, one of the benefits of using iterative reconstruction methods is that an irregular geometry is (in principle) easily supported, hence the reprojection step described above can be avoided. In [110] 3D iterative reconstruction algorithms (ART and EM) have been tested along with the 3DRP method. The conclusion was that ART gave the highest Signal to Noise ratio, closely followed by 3DRP. Another conclusion was that the EM method apparently performed worse. In their example the reconstruction required 2-10 hours for ART, 3.3 hours for 3DRP, and 22-64 hours for EM. These measurements correspond to a SUN Sparc 10.

10.7 A 3D Reconstruction Package

A software package for 3D reconstruction of volumes from line integrals has been developed. The package has been written in C for Unix systems, and is available for free at [10]. The package has been tested on Linux systems, and on an Onyx-computer from Silicon Graphics (SGI).

Going from 2D to 3D reconstruction one major difference is the size of the inversion problem, as the examples mentioned in Section 10.6 indicated. A practical reconstruction program (direct or iterative) will include several steps, of which filtering, Radon transform, and the backprojection is the most time consuming (the actual elements depend on the algorithm). In the implementation these central elements have been written to run efficiently on a single processor system, as well as in parallel on an Onyx using the Iris Power-C compiler. Compiler options has been added to the code to enable parallel execution, and one of the nice features of the SGI implementation is that the code still can be compiled on any other Unix machine, e.g., a Linux box, where the additional parallel compiler options will be ignored.

The reconstruction program “Recon3D” is intended for testing of different 3D reconstruction algorithms, and no Kinahan & Rogers reprojection step is currently included, in order not to mix 2D and 3D algorithms, but all of the tools for implementing the reprojection step are available.

The program will require a uniformly sampled sinogram, as it will shown in Eq. 10.68, based on the Ω_ψ -geometry and the reconstructed volumes will also be sampled uniformly. The value of ψ is defined by the user. The package also includes the program “3D_RadonAna”, which has been developed for generating a volume and the corresponding four-dimensional sinogram from a set of scaled, translated, and rotated primitives. This program is based on the properties shown in Appendix E and especially Section E.2. The usage of the program is shown in Appendix F. In the following section the implementation of the Radon transform operator and the backprojection operator are shown, which cover two of the central elements of the software package.

10.8 Implementation of the 3D Reconstruction Methods

It has been shown that reconstruction of volumes from 4D line integrals can be done by filtering, either the projections, cf. Eq. 10.62, or the backprojected volume, cf. Eq. 10.50. The implementation is a straight forward generalization of the 2D reconstruction implementation discussed in Chapter 8, but the filters are not as simple, if the limiting angle $\psi < \frac{\pi}{2}$. Again, apodizing windows should be multiplied to the filters in order to limit the influence of noise.

Here the reconstruction algorithms are based on the geometry Ω_ψ , and the parameters in the parameter domain are sampled uniformly

$$\begin{aligned} \phi &= \phi_p = -\psi + p \frac{2\psi}{P-1}, & p &= 0, 1, \dots, P-1 \\ \theta &= \theta_t = t \frac{\pi}{T}, & t &= 0, 1, \dots, T-1 \\ u &= u_i = u_{min} + i \Delta u, & i &= 0, 1, \dots, I-1 \\ v &= v_j = v_{min} + j \Delta v, & j &= 0, 1, \dots, J-1 \end{aligned} \tag{10.68}$$

where $\Delta\phi = \frac{2\psi}{P-1}$, $\Delta\theta = \frac{\pi}{T}$, $\theta_{min} = 0$, and $\phi_{min} = -\psi$ have been inserted.

The parameters of the reconstructed volume are also samples uniformly

$$\begin{aligned} x = x_k &= x_{min} + k\Delta x, \quad k = 0, 1, \dots, K-1 \\ y = y_l &= y_{min} + l\Delta y, \quad l = 0, 1, \dots, L-1 \\ z = z_m &= z_{min} + m\Delta z, \quad m = 0, 1, \dots, M-1 \end{aligned} \quad (10.69)$$

Note that a measured sinogram must be resampled (rebinned) into this geometry and missing parts of the sinogram can be estimated using the Kinahan & Rogers reprojeciton technique.

10.8.1 Implementation of the Backprojection Operator

The backprojection operator can be approximated by a sum, here shown using a nearest neighbour approximation

$$\hat{g}(\mathbf{r}_{k,l,m}) = \int_{\theta=0}^{\pi} \int_{\phi=-\psi}^{\psi} \check{g}(\theta, \phi, u = \mathbf{r} \cdot \boldsymbol{\alpha}, v = \mathbf{r} \cdot \boldsymbol{\beta}) \cos \phi \, d\phi \, d\theta \quad (10.70)$$

$$\approx \Delta\theta \Delta\phi \sum_{p=0}^{P-1} \cos \phi_p \sum_{t=0}^{T-1} \check{g}\left(t, p, \left[\frac{\mathbf{r}_{k,l,m} \cdot \boldsymbol{\alpha}_{p,t} - u_{min}}{\Delta u}\right], \left[\frac{\mathbf{r}_{k,l,m} \cdot \boldsymbol{\beta}_{p,t} - v_{min}}{\Delta v}\right]\right) \quad (10.71)$$

In Algorithm 10.1 the implementation of the backprojection operator is shown, when using nearest neighbour interpolation. Better interpolation schemes, such as bilinear interpolation in the u and v parameters, can easily be included.

It has been shown that all of the reconstruction methods are heavily based on projections of the directional vectors $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\tau}$ onto the volume coordinates \mathbf{r} . In order to speed up the reconstruction algorithms, the vectors $\boldsymbol{\alpha} = (\alpha_x, \alpha_y, \alpha_z)^T$, $\boldsymbol{\beta} = (\beta_x, \beta_y, \beta_z)^T$, and $\boldsymbol{\tau} = (\tau_x, \tau_y, \tau_z)^T$, should be computed once, for all values of (p, t) . This can be done rather efficiently and will only require six matrices. In Algorithm 10.1 the arrays are assumed given, e.g., $\text{alpha_x}(t, p) = \alpha_x(\theta_t, \phi_p)$. For further optimization, the double loop of all possible angles (θ, ϕ) can be combined into one loop, and the values of $x(k)$, $y(l)$, and $z(m)$ should be moved to simple variables before entering the inner loops. In this way many of the array calculations can be avoided. Additional optimization techniques have been shown in Chapter 1 for the 2D Radon transform.

For each value of (t, p) , Eq. 10.71 requires that the u - and the v -values lies within the bounds shown in Eq. 10.68. One scheme to avoid this time-consuming testing is to expand the sinogram in the u and v direction (by padding with zeros at the edges), though this might not be desirable, due to the memory requirements (a measured sinogram from a GE Advance PET scanner can require about 65 MBytes of memory). For the expansion, it can easily be shown that

$$\left\lfloor \frac{-\max |\mathbf{r}_{k,l,m}| - u_{min}}{\Delta u} \right\rfloor \leq i \leq \left\lceil \frac{\max |\mathbf{r}_{k,l,m}| - u_{min}}{\Delta u} \right\rceil \quad (10.72)$$

$$\left\lfloor \frac{-\max |\mathbf{r}_{k,l,m}| - v_{min}}{\Delta v} \right\rfloor \leq j \leq \left\lceil \frac{\max |\mathbf{r}_{k,l,m}| - v_{min}}{\Delta v} \right\rceil \quad (10.73)$$

Another scheme to avoid the index testing, could be to compute the intervals (p, t) , which will give legal values of u and v , cf. Eq. 10.68. In principle this is viable, but might require more computations compared to the reduction being offered by this alteration.

ALGORITHM 10.1 : BACKPROJECTION OPERATOR IN 3D

```

For k=0 to K-1                                //For all values of x
  For l=0 to L-1                               //For all values of y
    For m=0 to M-1                             //For all values of z
      sum=0                                     //Initialize sum
      For p=0 to P-1                           //For all values of phi
        sump=0                                  //Initialize sump
        For t=0 to T-1                           //For all values of theta
          u=x(k)*alpha_x(t,p)+y(l)*alpha_y(t,p)+z(m)*alpha_z(t,p)
          i=round((u-u_min)/Delta_u)             //Calculate i index
          If 0≤i<I                            //Check if index is valid
            v=x(k)*beta_x(t,p)+y(l)*beta_y(t,p)+z(m)*beta_z(t,p)
            j=round((v-v_min)/Delta_v)           //Calculate j index
            If 0≤j<J                          //Check if index is valid
              sum=sum+g_radon(t,p,i,j)         //Update sum
            End
          End
        End
        sum=sum+sump*cosphi(p)                  //Update sump
      End
      g_backproject(k,l,m)=sum*Delta_rho*Delta_phi //Store result
    End
  End
End

```

The number of loops shown in Algorithm 10.1 illustrates that the complexity of backprojection is high, namely

$$\mathcal{O}_{\text{3D Backprojection}} = \mathcal{O}(KLMPT) \quad (10.74)$$

Hence, the complexity increases with the number of voxels in the reconstructed volume times the number of angular samples in the sinogram, and it indicates that the backprojection operator is a rather demanding operation.

One nice feature is that Algorithm 10.1 can be parallelized very easily. On the Onyx-computer using Iris Power-C, it was concluded that the parallel chunks of code should be as large as possible, and in Section 10.9 it will be shown that splitting the outer-most loop of the algorithm, i.e., the k -loop on four processors leads to a very good performance.

10.8.2 Implementation of the Radon Transform Operator

An implementation of the Radon transform defined in Eq. 10.5 will now be shown. This operator is needed for the Kinahan & Rogers reprojection methods and for any of the iterative methods presented in Chapter 9.

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\tau + \mathbf{r}_0) ds, \quad \text{where } \mathbf{r}_0 = (x_0, y_0, z_0)^T = u\boldsymbol{\alpha} + v\boldsymbol{\beta} \quad (10.75)$$

$$= \int_{-\infty}^{\infty} g(s\tau_x + x_0, s\tau_y + y_0, s\tau_z + z_0) ds \quad (10.76)$$

In order to avoid the problems with too high slopes, analogous to the problems pointed out in Section 1.7, the line integral is projected onto the axis with maximum absolute component of the

directional vector relative to its sampling interval. It can be mentioned that this strategy also was applied for the 2D Radon transform in Section 2.2. Assume that the projection is made onto the x -axis

$$\frac{|\tau_x|}{\Delta x} > \frac{|\tau_y|}{\Delta y} \quad \text{and} \quad \frac{|\tau_x|}{\Delta x} > \frac{|\tau_z|}{\Delta z} \quad (10.77)$$

Then the 3D Radon transform can be rewritten

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} \frac{1}{|\tau_x|} g\left(x, xy^{(x)} + y_0^{(x)}, xz^{(x)} + z_0^{(x)}\right) dz \quad (10.78)$$

where

$$y^{(x)} = \frac{\tau_y}{\tau_x} \quad \text{and} \quad y_0^{(x)} = y_0 - y^{(x)} x_0 \quad (10.79)$$

$$z^{(x)} = \frac{\tau_z}{\tau_x} \quad \text{and} \quad z_0^{(x)} = z_0 - z^{(x)} x_0 \quad (10.80)$$

where the exponent (x) merely means with respect to x .

A simple discretization, which will require few calculations each time, can now be derived from Eqs. 10.69, 10.76, 10.79, and 10.80.

$$\check{g}(\theta, \phi, u, v) \approx \Delta x \sum_{k=0}^{K-1} g\left(k, \left[\frac{x_k y^{(x)} + y_0^{(x)} - y_{min}}{\Delta y}\right], \left[\frac{x_k z^{(x)} + z_0^{(x)} - x_{min}}{\Delta z}\right]\right) \quad (10.81)$$

$$= \Delta x \sum_{k=0}^{K-1} g\left(k, [a_y^{(x)} k + b_y^{(x)}], [a_z^{(x)} k + b_z^{(x)}]\right) \quad (10.82)$$

$$\text{where } a_y^{(x)} = \frac{\tau_y}{\tau_x} \frac{\Delta x}{\Delta y} \quad \text{and} \quad b_y^{(x)} = \frac{\tau_y}{\tau_x} \frac{(x_{min} - x_0)}{\Delta y} + \frac{y_0 - y_{min}}{\Delta y} \quad (10.83)$$

$$a_z^{(x)} = \frac{\tau_z}{\tau_x} \frac{\Delta x}{\Delta z} \quad \text{and} \quad b_z^{(x)} = \frac{\tau_z}{\tau_x} \frac{(x_{min} - x_0)}{\Delta z} + \frac{z_0 - z_{min}}{\Delta z} \quad (10.84)$$

With respect to the high slope problem, Eq. 10.77 implies that $|a_y^{(x)}| \leq 1$ and $|a_z^{(x)}| \leq 1$, i.e., a step from k to $k+1$ in the sum shown in Eq. 10.82, will not lead to a step in l or m greater than 1.

The discrete implementation of the Radon transform is also quite demanding, due to the complexity

$$\mathcal{O}_{3D \text{ Radon transform}} = \mathcal{O}(PTIJK) \quad (10.85)$$

which accounts for the number of times where the projection is made onto the x -axis. For projection onto the y - and z -axis similar expressions are found, with K substituted by L and M , respectively.

If either the absolute y - or z -component of τ is the greatest, then it is very easy to derive almost identical formulas, and the formulas will only require swapping of symbols compared to the ones shown above. In Algorithm 10.2 the implementation of the discrete 3D Radon transform is shown using a nearest neighbour approximation, but, e.g., bilinear interpolation in the y and z coordinate can easily be incorporated. The algorithm only shows the case where Eq. 10.77 is fulfilled. The algorithm uses the discrete variables $a_y = a_y^{(x)}$, $a_z = a_z^{(x)}$, $b_y = b_y^{(x)}$, and $b_z = b_z^{(x)}$. Two very similar algorithms are needed to cover projection onto the y -axis and the z -axis, respectively.

ALGORITHM 10.2 : DISCRETE 3D RADON TRANSFORM

```

For p=0 to P-1                                //For all values of phi
  For t=0 to T-1                               //For all values of theta
    For i=0 to I-1                             //For all values of u
      For j=0 to J-1                           //For all values of v
        Assuming Eq. 10.77 is fulfilled
        Set a_y, b_y, a_z, and b_z from Eqs. 10.83 and 10.84
        sum=0                                     //Initialize sum
        For k=0 to K-1                           //For all values of x
          l=round(a_y*k+b_y)                     //Calculate y-index
          If 0≤l<L                            //Check if index is valid
            m=round(a_z*k+b_z)                   //Calculate z-index
            If 0≤m<M                            //Check if index is valid
              sum=sum+g(k,l,m)                  //Update sum
            End
          End
        End
        g_radon(p,t,i,j)=sum*Delta_x           //Update Radon domain
      End
    End
  End
End

```

10.9 Examples of 3D Reconstructed Volumes

In this section a few examples of 3D reconstruction using “Recon3D” are given. The 3D visualization has been done using either the commercial package IDL, or combination of the “Polyr” program [119] and the excellent Geomview program, available for free at [123]. Polyr can convert volumes into a mesh of triangles. The mesh is a iso-surface though the volume for a certain user-defined threshold level. The mesh can be imported directly into Geomview for interactively inspection of the 3D object.

10.9.1 Reconstruction of a Ball

The first example concern reconstruction of a homogeneous ball, cf. Subsection E.2.1. In this case the axial limiting angle ψ was set to 90° . Here the reconstructed volume has $51 * 51 * 51$ voxels and the (noise free) sinogram uses $P = 12$, $T = 10$, and $51 * 51$ samples in each of the (u, v) planes. Both the volume and the sinogram requires roughly 0.5 MBytes of memory.

In Fig. 10.4, 3D Filtered backprojection has been used to reconstruct the ball, and in Fig. 10.5 3D Filtering after Backprojection. In both cases an iso-surface has been generated using IDL, and a slice has been inserted to show the volume values in that plane. The figures indicate that the ball has been recovered well, though the edges of the ball is somewhat blurred. This could be expected, due to the low number of samples in both domains. The figures are not intended for quantitatively evaluation of the reconstruction results, but illustrates which kind of tools can be used to visualize 3D reconstructed volumes.

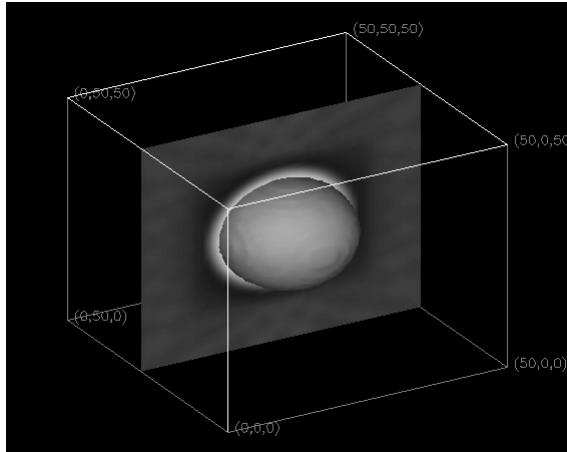


Figure 10.4 Reconstructed ball using 3D Filtered Backprojection.

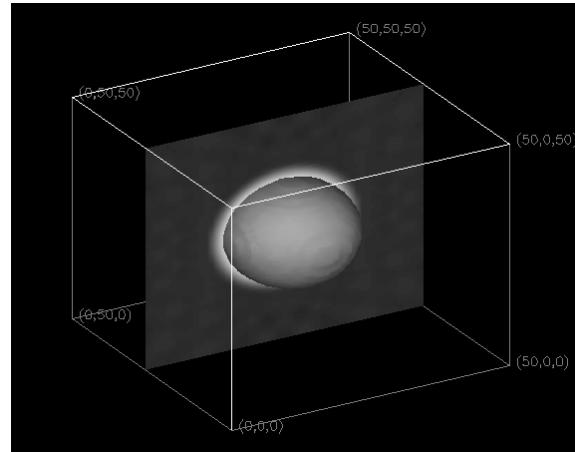


Figure 10.5 Reconstructed ball using 3D Filtering after Backprojection.

Next the iterative reconstruction methods ART and EM have been used to reconstruct the same ball. For ART some artifacts were found, while EM is very successful. The values outside of the ball are very close to zero. Here figures like Fig. 10.4 has been omitted, due to the high similarity to the one shown above. For ART the sinogram entries were chosen randomly, though with a large axial acceptance angle ψ , it was experimentally found useful to use a weight function favoring the line orientations perpendicular to the z -axis, i.e., favoring a small value of ϕ . Assume $\phi = 90^\circ$, and a uniform random weighing scheme to select the sinogram index in ART, cf. Section 9.6. In that case many of the lines orientations through the scanned object will be close, and ART will converge very slowly or produce artifacts, cf. Fig. 9.8.

For the Onyx, Table 10.1 shows the times needed to reconstruct the ball, obtained with and without parallel options enabled. When running in parallel, the program will use four 200 MHz MIPS R4400 processors.

For Filtering Backprojection running in parallel the filtering required 0.24 sec and the backprojection required 3.95 sec. The same figures for one processor are 0.62 sec and 13.91 sec. In the software package the efforts were concentrated on the backprojection, and here a speedup factor of 3.5 was obtained on four processors. Filtering after Backprojection gets slower because the sinogram is being backprojected into a larger volume in order to avoid edge effects, and the speedup factor is somewhat lower due to a larger part of the total time is assigned to 3D filtering, based on a multi-dimensional FFT (currently) running on a single processor. For reference, the much cheaper Pentium 120 MHz running Linux can reconstruct the same volume in 28 sec, when using Filtered Backprojection.

From Table 10.1 it can be seen that ART actually gets slower when running in parallel mode. In each iteration of ART, the part of the program running in parallel is the Radon transform of the volume for one single parameter vector. The size of code assigned to each processor is here very small, and most of the time the four processors are waiting to synchronize their operation.

10.9.2 Reconstruction of the Mickey Phantom

In this example the axial acceptance angle has been reduced to merely $\psi = 9^\circ$. The volume was centered around $(0, 0, 0)$ and sampled with $\Delta x = \Delta y = \Delta z = 0.5$, and $K = L = M = 71$, which requires 1.4 MBytes of memory. In the sinogram $T = 90$, $P = 11$, $I = J = 61$, and $\Delta u = \Delta v = 1$ were used, and the sinogram requires 14.7 MBytes of memory.

Algorithm	FB	FAB	ART	EM
Time in seconds on one processor	14.5	26	6	6
Time in seconds running parallel	4.2	10	39	18
Speedup factor	3.5	2.6	0.15	3

Table 10.1 Reconstruction time for reconstruction of the ball using Filtered Backprojection (FB), Filtering after Backprojection (FAB), one full iteration of ART, and finally one iteration of EM. All times are measured in seconds on an Onyx.

Using 3D Filtering after Backprojection, the reconstructed volume has been visualized using Polyr and Geomview. Fig. 10.6 shows the result, which looks like the original Mickey Mouse phantom. In Figs. 10.7 and 10.8 the reconstructed central (x, y) and (y, z) planes are shown. The following figures use individual color scale according to the minimal and maximal value. This phantom has the value 0.2 inside the “skull” and a small “tumor”-ball with radius of 1 and value 0.4 placed at $(0, 1, 0)$. The tumor is visible but blurred, but it is clear that the general structures have been recovered well. The reconstructing required 6754 sec on a 120 MHz Pentium, and here the sinogram was backprojected onto a $128 \times 128 \times 128$ volume in order to use radix-2 FFT filtering. No apodizing windows have been used, and is probably the reason for the ripples outside of the phantom.

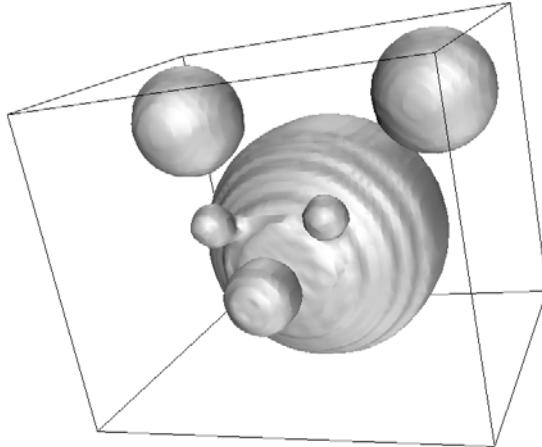


Figure 10.6 The Mickey phantom.

Next Filtered Backprojection has been used to reconstruct the same phantom. The total reconstruction used 1189 sec on the 120 MHz Pentium, divided in 75 sec for the filtering, 1111 sec for the backprojection and 3 sec for additional minor operations. Figs. 10.9 and 10.10 show the central (x, y) and (y, z) planes for Filtered Backprojection.

Finally, five iterations using the EM algorithm has been used. Fig. 10.11 and 10.12 again show the (x, y) and (y, z) planes. It is obvious that additional iterations are needed, and the images look blurred.

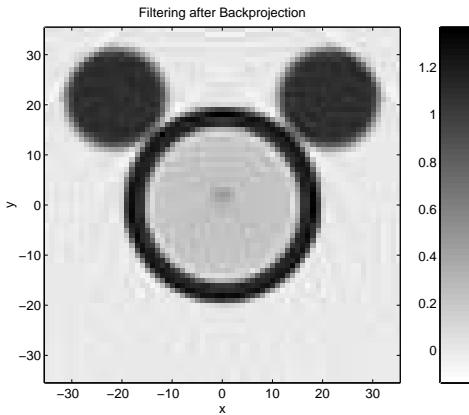


Figure 10.7 Reconstructed phantom in the central (x, y) -plane using Filtering after bacprojection.

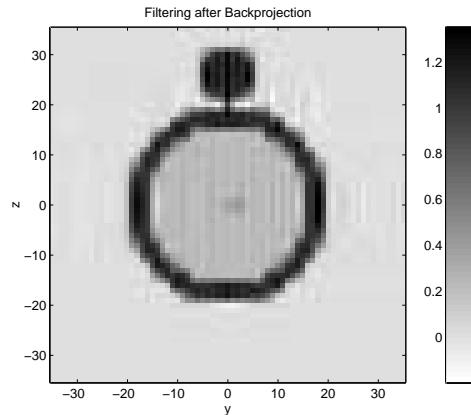


Figure 10.8 Reconstructed phantom in the central (y, z) -plane using Filtering after bacprojection.

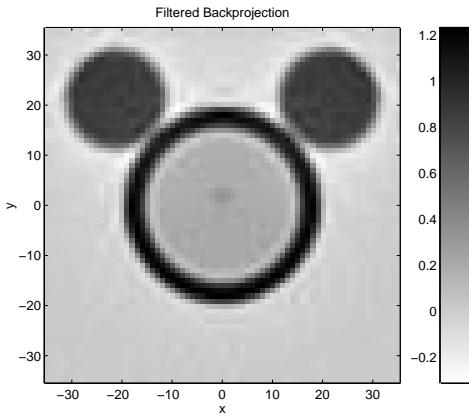


Figure 10.9 Reconstructed phantom in the central (x, y) -plane using Filtered Backprojection.

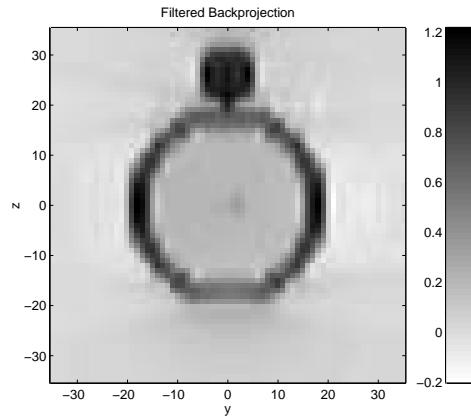


Figure 10.10 Reconstructed phantom in the central (y, z) -plane using Filtered Backprojection.

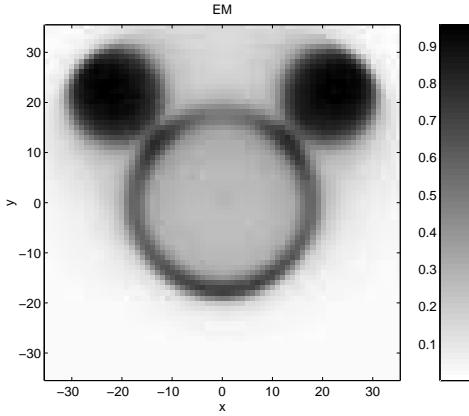


Figure 10.11 Reconstructed phantom in the central (x, y) -plane using five iterations of EM.

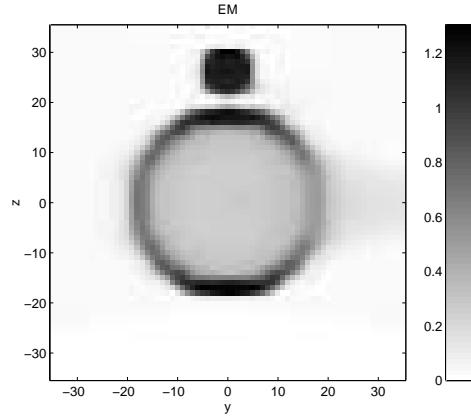


Figure 10.12 Reconstructed phantom in the central (y, z) -plane using five iterations of EM.

10.10 Summary

In this chapter several methods have been described for reconstruction of volumes from their line integrals, which is very relevant for 3D PET scanners. It has been shown that the measured sinogram in this case becomes four dimensional, and the reconstructed volumes are three dimensional. The problems with this change in dimensions have been outlined, but more research is needed in this area.

The implementation of two of the central elements of reconstruction algorithms have been shown, and aspects of parallel implementation are given. The theory has been implemented in a software package, where both direct and iterative reconstruction methods are available. The package also includes a program for generation of synthetic sinograms and the corresponding volume. This is very useful when testing the reconstruction methods.

The software package is working, but some parts of the package can be improved with respect to the tradeoff between the degree of approximation and the time needed to reconstruct the volumes. Especially the use of apodizing windows in the algorithms could be investigated, and this will definitely alter how the reconstruction algorithms are influenced by noise. That is why only a few examples have been presented with sparse information on the quantitatively performance of the individual algorithms.

Chapter 11

Noise Contributions from Blank, Transmission and Emission Scans in PET

This chapter describes quantification of noise contributions in PET, and is more practically oriented than the previous chapters. This work has been carried out at the National University Hospital in Copenhagen (Rigshospitalet) [7, 8].

11.1 Introduction

PET scans require correction for attenuation in order to be quantitative, cf. Eq. 6.22. Even in brain activation studies where absolute units are often replaced by relative values (i.e., normalized to a mean value), inter-individual comparisons still assume that values across an image reflect the local tracer concentration. Corrections can be applied, e.g., by assuming uniform values of attenuation within geometrically defined boundaries or in segmented areas of the image, but a more accurate description of the attenuation requires measurement by a transmission scan.

Unfortunately the attenuation correction also adds noise to the image and adds to the total procedure time. In some cases, mainly in body scanning, the increase in noise is immediately noticed and may even be so high that quantification must be sacrificed in order to provide a reasonable visual impression. However, in other situations, effects that are invisible for the human observer might still be of importance when examining the subtle differences of brain activation by statistical methods. The purpose of this work is to establish guidelines for the relative importance of the noise contributions from blank, transmission, and emission scans in typical imaging situations.

Although much information can be deduced theoretically from existing knowledge of reconstruction algorithms [124, 125, 126], here a mainly experimental approach has been applied. The limitations in generality caused by the obvious difficulty of varying all of the many possible parameters of acquisition and reconstruction are outweighed by the advantage of the close reproduction of actual scan setups, including some human studies. The following section describes the model used for fitting to the observed data. It should not be considered as a rigorous derivation; it merely lists the different structures of the datasets included in the study comprising homogeneous as well as inhomogeneous objects.

11.2 Theory

Due to the linearity of the reconstruction process when using direct reconstruction methods, such as Filtered Backprojection, the PET images (volumes) denoted x_i , can be described as a weighted sum of the sinogram values s_j , with weight factors ϕ_{ij} corresponding to the linear reconstruction algorithm, cf. 9.25.

$$x_i = \sum_j \phi_{ij} s_j \quad (11.1)$$

where the actual image pixel (voxel) is denoted by i , and j denotes the pixel position in the sinogram.

In order to compensate for attenuation the sinogram used for reconstruction is calculated from an emission sinogram e_j , a transmission sinogram t_j , and a blank scan b_j . In Eq. 6.18 it is shown that calculation basically amounts to

$$s_j = \frac{e_j b_j}{t_j} \quad (11.2)$$

although this in practice usually is qualified by a filtering of the factor $\frac{b_j}{t_j}$.

All three types of sinograms in principle inherit their statistical properties from the Poisson statistics of the radioactive decay. A number of corrections that are performed before the stage of reconstruction described here, however, adds to the raw counts' noise. The overall effect of correction for randoms and scatter can be described as a decrease in effective counts, the resulting figure being widely known as Noise Equivalent Counts (NEC) [127, 126]. A practical formulation used in this work is:

$$\text{NEC} = \mathcal{T} \frac{(1 - \text{SF})^2}{1 + 2f \frac{\mathcal{R}}{\mathcal{T}}} \quad (11.3)$$

where \mathcal{T} is true counts (including scatter), \mathcal{R} is random counts from the full field-of-view, f is the fraction of the sinogram covered by the object under investigation, and SF is the scatter fraction. In all applications of NEC in this study, the value of SF has been set to zero.

While NEC as a global measure may be adequate for the noise description in the case of homogeneous phantoms extending the full axial field-of-view (AFOV), objects of limited extension may be better characterized by a value per slice. In the following, all NEC values quoted correspond to a single slice.

11.2.1 One Emission and One Transmission Scan

In the simplest case where the image noise is estimated from a single image set, using a Taylor expansion and assuming no noise correlation between pixels in the reconstructed images implies that the pixel noise variance $V\{x_i\}$ can be estimated by

$$V\{x_i\} \simeq \sum_j \phi_{ij}^2 V\left\{\frac{e_j b_j}{t_j}\right\} \simeq \sum_j \phi_{ij}^2 E^2\{s_j\} \left(\frac{V\{e_j\}}{E^2\{e_j\}} + \frac{V\{b_j\}}{E^2\{b_j\}} + \frac{V\{t_j\}}{E^2\{t_j\}} \right) \quad (11.4)$$

Due to the Poisson statistics the variance in the emission sinogram relative to the squared mean value $\frac{V\{e_j\}}{E^2\{e_j\}}$ is inversely proportional to the mean of an infinite amount of experiments $E\{e_j\}$, i.e., inversely proportional to number of counts NEC in the emission sinogram. Similar arguments can be used for the transmission and blank scan terms, hence the total pixel variance normalized with the mean of the image can be modelled by

$$\frac{V\{x_i\}}{E^2\{x_i\}} = \frac{a}{\text{NEC}} + b + \frac{c}{T_t} + \frac{d}{T_b} \quad (11.5)$$

where T_t is the transmission scan time in seconds, T_b the blank scan time in seconds (for the given geometry and source strength). In the model a base term b is included to accommodate effects in the reconstruction process, not explained by the other terms, e.g., varying sensitivity of the detectors, not properly corrected for by normalization. For fixed values of (either) T_t or T_b the corresponding terms may also be thought of as part of this constant term. The model applies to regions with limited structural variation, hence measuring variance in a heterogeneous object like the brain or the thorax is problematic.

11.2.2 Two Emission Scans

To overcome the problem with heterogeneous structures the difference between two PET images, corresponding to two equivalent but independent emission scans $e_j^{(1)}$ and $e_j^{(2)}$, can be calculated. Due to Eqs. 11.1 and 11.2 the difference image is given by

$$x_i = \sum_j \phi_{ij} (e_j^{(2)} - e_j^{(1)}) \frac{b_j}{t_j} \quad (11.6)$$

Using a Taylor expansion, and using that the two emission scan have the same mean values $E\{e_j\}$ and variance $V\{e_j\}$ implies that

$$x_i \simeq \sum_j \phi_{ij} ((e_j^{(2)} - E\{e_j\}) - (e_j^{(1)} - E\{e_j\})) \frac{E\{b_j\}}{E\{t_j\}} \quad (11.7)$$

Thus the variance $V\{x_i\}$ is given by

$$V_{\text{two e-scans}}\{x_i\} \simeq 2 \sum_j \phi_{ij}^2 E^2\{s_j\} \frac{V\{e_j\}}{E^2\{e_j\}} \quad (11.8)$$

Normalized with the squared mean of the heterogeneous structure and using the same definition of constants as in Eq. 11.5 gives that

$$\frac{1}{2} \frac{V_{\text{two e-scans}}\{x_i\}}{E^2\{x_i\}} \simeq \frac{a}{\text{NEC}} \quad (11.9)$$

Thus only the emission term of the noise variance can be estimated from two emission scan and one transmission scan. This result somewhat surprising, but is also found in the measurements. Since all structural information is removed by using differences, no constant term (b) can appear.

If instead two transmission scans (and one emission scan) are available, it is easy to show that two times the transmission variance can be estimated (and only that) from a normalized difference image.

11.2.3 Two Emission and Two Transmission Scans

Using the same Taylor technique with two sets of emission scans and two transmission scans, it can be shown that

$$\frac{1}{2} \frac{V_{\text{two e and t-scans}}\{x_i\}}{E^2\{x_i\}} \simeq \frac{a}{\text{NEC}} + \frac{c}{T_t} \quad (11.10)$$

This implies that both the emission and transmission term can be estimated in the inhomogeneous case if a double set of independent and equivalent emission and transmission scans are measured and reconstructed pairwise. Note that any normalized variance shown in the following has been divided by two if measured from two scans, cf. Eqs. 11.5, 11.9 and 11.10.

11.2.4 Zeroes in the Transmission Sinogram

A common problem especially in body scanning is zeros in the transmission sinogram due to huge attenuation leading to few counts and poor statistics in the transmission sinogram. One option is to replace the zeros with a small number T_0 , so the attenuation correction can be done using Eq. 11.2. Although overflow is avoided this may imply that reconstructed images contain strong lines causing a significant change in the noise structure. Let \mathcal{Z} be the set of j , where the measured t_j is zero. Assume that the variance of the blank scan is negligible and further that only the terms with zeros effectively contribute to the noise. In this case

$$V\{x_i\} \simeq \sum_{j \in \mathcal{Z}} \phi_{ij}^2 V\left\{\frac{e_j b_j}{T_0}\right\} \simeq \sum_{j \in \mathcal{Z}} \phi_{ij}^2 V\{e_j\} \frac{E^2\{b_j\}}{T_0^2} \quad (11.11)$$

This indicates that the number of zeros (through the sum) and the variance on the emission scan determines the noise level. Hence the variance normalized with the squared mean is approximately proportional to the number of zeros and inversely proportional to NEC. In the limit where this kind of noise is dominating, however, the reconstructed images consist of many lines and become useless anyway.

11.2.5 Limitations

Using either of Subsections 11.2.1, 11.2.2, or 11.2.3 to estimate the noise terms, we estimate the noise variance from a chosen ROI (region of interest) and normalize with the squared mean of the ROI. Due to the global features in the reconstruction process the noise tends to have only a very small correlation with the underlying structure and often it is approximately evenly distributed across the ROI's used. This implies that the normalized noise variance of the ROI depends strongly on the selected ROI, through the squared mean.

11.3 Overview of the Measurements

Measurements were made on the GE Advance PET scanner [128, 129, 130] with 3D acquisition and reconstruction capability. Its 9 GBytes raw data disk can hold about 180 3D frames (recordings with 35 slices), suitable for phantom decay studies. Reconstruction in our configuration (with 10 i860 processors) takes approximately 8 minutes per frame [131]. For blank and transmission scanning, which is always performed in 2D, the scanner applies two pin sources. During the initial experiments the pin source activities were 389 MBq and 134 MBq, respectively. The blank scan sinogram count rate was 0.91 Mcps, corresponding to an average count rate per sinogram element of ≈ 0.23 cps (counts per sec) in the center. On later phantom and human studies, blank and transmission scan times have been scaled in accordance with the currently observed blank scan count rates and the values quoted in seconds are therefore directly comparable.

11.3.1 Phantom studies

Three different water-filled phantoms have been used: the 20 cm standard (NEMA) cylinder, a torso-like ellipse with axes 18 and 35 cm (shown in Fig. 11.1), and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm, as shown in Fig. 11.2.

Using initially the two homogeneous phantoms in accordance with Eq. 11.5, emission scans were made as decay series with F-18 in 2D (total of 22 measurements) and 3D (limited to 10 time frames). Transmission scans were made starting from $T_t=64$ seconds and doubling the time

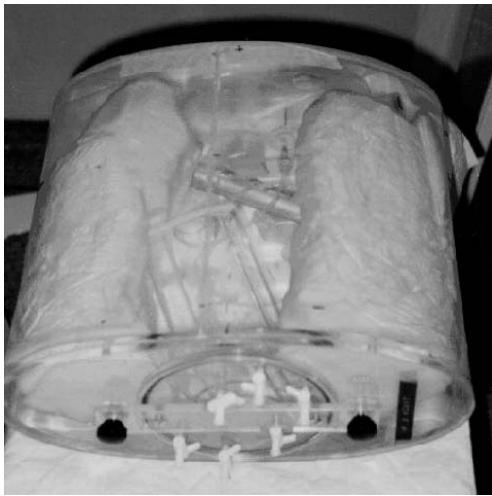


Figure 11.1 The torso-like phantom, here shown with lung-phantoms.



Figure 11.2 The elliptical brain phantom (Capintec) with axes 15 and 20 cm.

for each step up to 32768 seconds. Blank scans were made from $T_b=64$ to 4096 seconds and complemented with one 100000 seconds scan.

Reconstructions were made for all relevant combinations of emission and transmission scans using the 2048 seconds blank scan. All emission scans were further reconstructed with a calculated attenuation correction (CAC) using a circular or elliptical contour respectively. For the 2D cylinder case the emission image with highest count was reconstructed with the longest transmission scan and all blank scans. The cylinder was reconstructed in a 128^2 matrix with pixel size 2.0 mm, and a 4 mm Hann filter. For the ellipse, a 128^2 matrix of pixel size 4.0 mm was used with a 8 mm Hann filter. The 3D axial filter (Hann) was also set to its minimum value of 8.5 mm. The attenuation data were preprocessed with a 8 mm Gaussian filter for the cylinder case, and 10 mm for the ellipse. All available corrections were applied, including detector normalization, randoms subtraction, and scatter correction. In all slices of all reconstructed images (order of 25,000 images) the mean and variance was calculated from an ROI extending 70% of the diameter or ellipse axes. In 2D, the data were averaged over 31 (of 35) slices avoiding edge effects but ignoring the minor differences between direct and cross slices. In 3D only the central 15 slices having almost identical noise were included in the average. For each reconstructed dataset the noise was represented by the normalized variance, i.e., variance/mean², in subsequent plots and fitting. From the total rates curves a Noise Equivalent Count (NEC) value per slice was calculated for each scan frame. The rate dependent correction between trues and NECs due to randoms did not exceed 15%.

The brain phantom has two separate chambers ('grey' and 'white' matter, respectively). It was filled to resemble the usually quoted ratio of 4:1 between these two substances for flow or metabolism. Two different sets of measurements have been performed. One set (repeated in 2D and 3D) was originally designed to address count rate performance [132, 133]. The phantoms were loaded with C-11 carbonate well above the expected saturation limit of the scanner and pairs of (60:62) seconds scans were performed every 10 minutes for about 5 hours (15 half lives). The maximum number of slice counts observed in this study was limited to 10^6 which would not allow a sufficiently clear distinction between the transmission scan times. One more decay series was therefore prepared in 3D with F-18 measuring for 12 half lives, each split in two frames with a ratio of (0.415:0.585) yielding approximately equal NEC. The C-11 series were reconstructed using an 'infinite' (32768 sec) transmission scan. For the F-18 series, a set of 6 pairs of transmission scans,

64 - 16384 sec, were obtained. Reconstructions were made to provide datasets with independent transmission and emission noise according to Eq. 11.10 by combining the E-T frames as odd-odd, even-even. Reconstruction parameters for the brain phantom were identical to those for the cylinder except that the axial Hann filter was replaced by a Ramp filter. From both the C-11 series and the F-18 series difference images were calculated, and ROI's placed over grey matter (central and peripheral), white matter, and whole brain in the original as well as the difference images for mean and standard deviation calculation respectively.

11.3.2 Human studies

For one person (case KL) included in a count rate performance and dose optimization study with O-15 [132], the usual 90 second integration time (starting 20 seconds after bolus injection) was supplemented by 2 short frames (20-22 sec). The study comprised of injections with 25-800 MBq in 2D and 50-800 MBq in 3D. Reconstruction and analysis was performed as above, although only with one 12 minutes transmission scan. The number of NECs in the short scans were about 16% of the corresponding 90 seconds frames normally used.

One person (case MN) injected with 240 MBq of FDG had a double set of emission scans (8-512 sec) of the brain starting 2 hours after injection, followed by a double set of transmission scans (64-512 sec) yielding a total of 28 data points in (E,T)-plane. Reconstruction and data analysis was made to match the brain phantom F-18 series.

One patient (case BB) undergoing a dynamic FDG scanning of the heart additionally had a series of 6 transmission scans (0.5 - 20 min). Pairs of emission scans (0.5,1,2,5 min) from different parts of the dynamic study were reconstructed with all the transmission scans, and subsequent data analysis made as described above (Eq. 11.9) in a large body circumferential ROI.

11.3.3 Fitting data

From the emission data (see Subsections 11.2.1, 11.2.2, and 11.2.3) the parameter a was estimated by linear regression in a log-log diagram of the normalized variance using the longest transmission scan available. Depending on the setup the transmission parameter c (and the base term b) was subsequently estimated by subtracting the estimated emission part of the normalized variance, a/NEC , and again applying linear regression. This approach is very simple and was considered adequate for the purpose. Non-linear data fitting was also tried and found to give similar results.

11.4 Results

In this section data and parameters are presented for the previously described experiments and compared to the model. Again it should be noted that any variance shown has been divided by two if measured from paired difference scans so that the results presented are comparable and represent noise in the single images.

Estimated model parameters from the Cylinder phantom in 2D and 3D modes, the Elliptical Body phantom in 2D and 3D, The Brain Phantom, and Case MN, are given in Table 11.1. This table give parameters valid for any practically used value of NEC and T_t .

In Fig. 11.3 the measured normalized variance in the Cylinder (2D) case is shown. It can be seen that the transmission scans will effectively start to add to the variance if NEC is larger than approximately 10^5 counts per slice and that they - for transmission scan lengths normally encountered - are dominating at 10^7 . Note also, that the emission noise model as judged from the

Case	Mode	a [k counts]	b []	c [sec]
Cylinder	2D	47.7	0.00188	6.40
Cylinder	3D	27.0	0.00035	1.64
Ellipse	2D	24.6	0.00167	8.85
Ellipse	3D	33.8	0.00154	4.16
Brain Phantom	3D	16.7		1.01
Case MN	3D	18.5		1.59

Table 11.1 Estimated model constants.

linearity in the log-log plot remains valid down to the level where the noise exceeds the mean by an order of magnitude, i.e., beyond any practical application of the images as such.

In the Cylinder (2D) case the blank scan parameter d has also been estimated from reconstructed images corresponding to varying blank scan length T_b , and the estimated value is $d = 0.96$. By comparison with the parameter c from Table 11.1 the transmission scan in the cylinder case (2D) is seen to contribute approximately 6.7 times more to the variance than a blank scan with the same duration, in accordance with the average attenuation of the central region of a 20 cm water-filled phantom.

In Fig. 11.4, corresponding to the Brain phantom scanned in 3D, a contour plot shows the normalized variance as a function of NEC and transmission length T_t . Shading is used to indicate the actual variance, which can be seen from the rightmost grey scale bar. Contours show approximate equidistant level of noise. Furthermore two lines are inserted to show where the transmission term equals 10% and 100% respectively of the emission term.

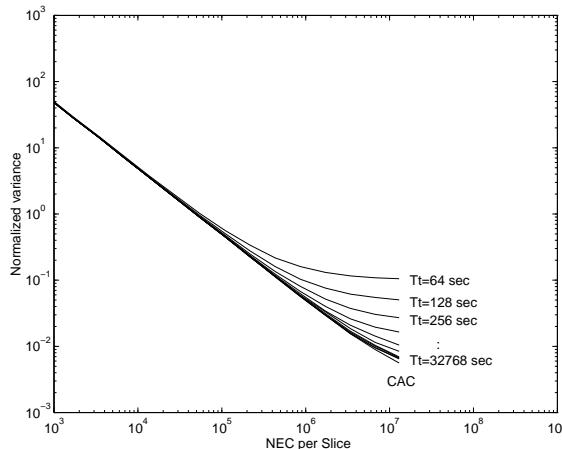


Figure 11.3 Normalized variance from 2D PET scan of the 20 cm Cylinder phantom as a function of NEC per slice and duration of transmission scan T_t . CAC means calculated attenuation correction.

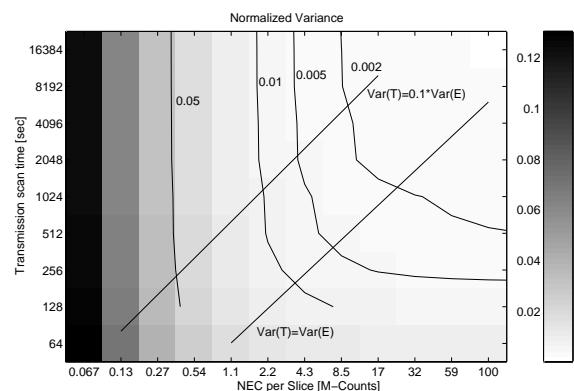


Figure 11.4 Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t . Contours show iso-noise curves with levels indicated. Lines show where the ratio of transmission to emission noise terms is unity and 10% respectively.

In the Brain phantom (3D) case and in case MN two emission and two transmission scans were measured, thus both emission and transmission parameters have been estimated. The parameters are listed in Table 11.1. Despite the approximations made for deriving the models Figs. 11.5 and 11.6 demonstrate an excellent match between the measured data and the model for many decades

of NEC. Note that the very low noise level, compared to Fig. 11.3, is due to the use of difference images made from paired emission and transmission scans, which eliminates the b -term.

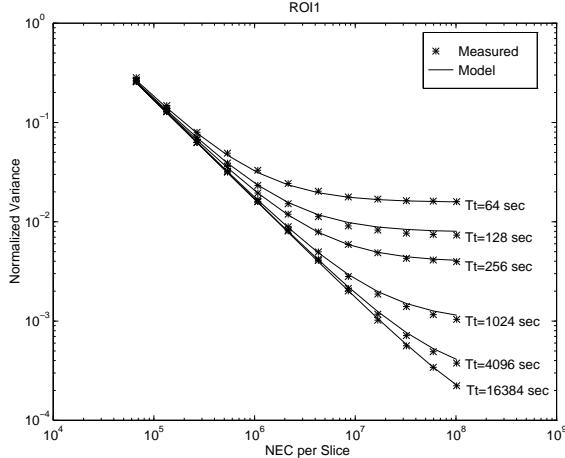


Figure 11.5 Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t . Stars show the measured data and lines the fitted model.

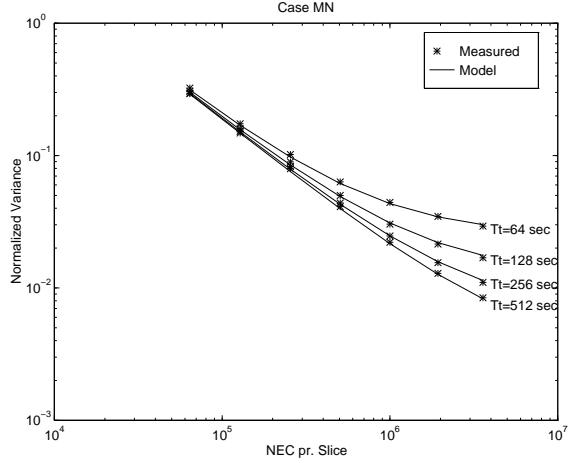


Figure 11.6 Normalized variance of case MN (3D) as a function of NEC per slice and duration of transmission scan T_t . Stars show the measured data and lines the fitted model.

Fig. 11.7 illustrates how well the emission data measured on human brains agree with the phantom studies. The figure shows case KL (2D/3D), case MN (2D/3D), and the Brain Phantom (2D/3D). In all cases data corresponding to only one transmission scan are shown.

Table 11.4 shows noise estimates obtained from different ROIs. The emission parameter a and the transmission parameter c have been estimated for the Brain Phantom (3D) in four different ROIs. The Head ROI is an ellipse just surrounding the activity of the brain, WM is a small region within white matter, GM_p and GM_c are two small grey matter regions, peripheral and central, respectively. The averages of the measured regions are normalized to the average of WM. Note the approximately constant ratio between a and c , indicating the major influence of the average on the coefficients.

ROI	a [k counts]	c [sec]	Average
Head	17	1.0	2.07
WM	72	3.7	1.00
GM _p	4.4	0.24	3.98
GM _c	7.2	0.49	4.05

Table 11.2 Brain phantom data (3D) with different ROIs.

Finally Fig. 11.8 shows data from body measurements. The 2D Elliptical Body phantom follows the previous noise model, but in case BB (large patient with arms in FOV, only a single set of transmission scans) the noise curves for the 1 and 2 minutes transmission scans are shifted upwards. Inspection of the reconstructed images reveals large number of lines, due to (clusters of) zeros in the transmission scan (cf. Eq. 11.11).

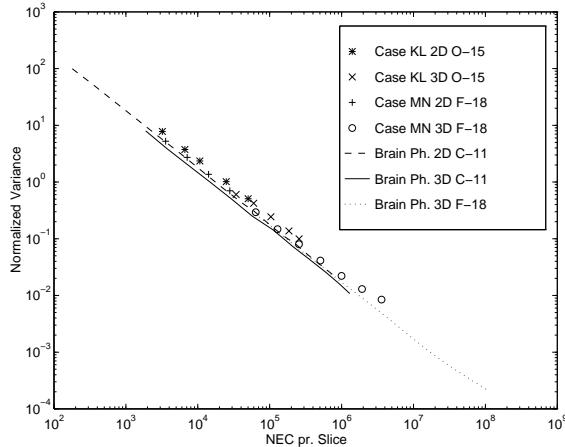


Figure 11.7 Normalized variance of Case KL (2D/3D), case MN (2D/3D), and Brain phantom (2D/3D) as a function of NEC per slice.

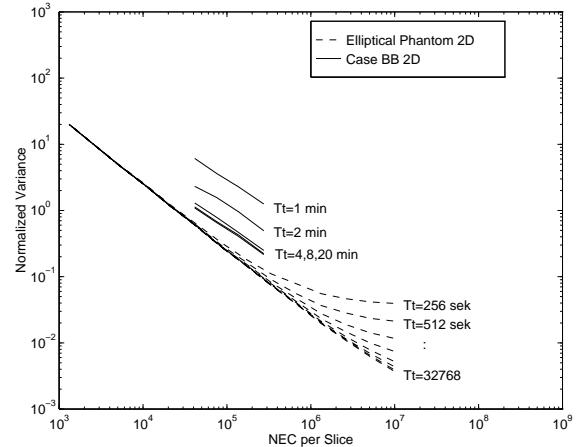


Figure 11.8 Normalized variance of body studies case BB (2D) and Elliptical phantom (2D) as a function of NEC per slice. Note that case BB (1,2 and 4 min) shows a different noise behavior.

11.5 Optimization

One application of the estimated noise parameters is optimization of the patient time used in the scanner [126]. Assume that the total time available for the examination of a patient is T seconds and that the transmission scan and the emission scan are distinct. The duration of the two scans are T_t and T_e respectively, and $T = T_t + T_e$. For simplicity assume that the rate of Noise Equivalent Counts is constant and equals R_{NEC} . This implies that the sum of the emission and transmission term are given by

$$\frac{V}{E^2} = \frac{a}{R_{\text{NEC}}(T - T_t)} + \frac{c}{T_t} \quad (11.12)$$

Thus the optimum duration of the transmission scan is given by

$$T_t = \frac{T}{1 + \sqrt{\frac{a}{c R_{\text{NEC}}}}} \quad (11.13)$$

Figs. 11.9 and 11.10 show the normalized variance of the individual noise terms and their sum as a function of transmission scan time T_t in two cases. Both correspond to the parameters found in Table 11.1 for the Brain Phantom (3D) and the sum of the transmission scan time and emission scan time is arbitrarily set to 1800 sec. In Fig. 11.9 it is assumed that $R_{\text{NEC}} = 1000$ counts per slice per sec, which gives an optimized normalized variance of 0.014 when $T_t = 354$ sec. Assuming $R_{\text{NEC}} = 10000$ counts per slice per sec as in Fig. 11.10 implies that the optimum is found at $T_t = 786$ sec and the normalized variance is 0.0029.

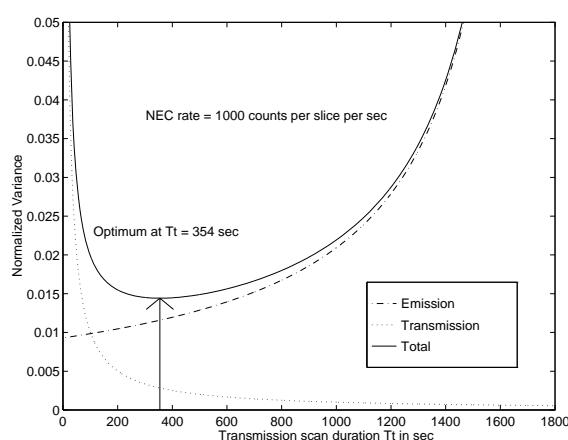


Figure 11.9 The transmission and emission noise terms and their sum as a function of the transmission scan duration, when the total (patient) scan time is 1800 sec. Parameters correspond to the Brain Phantom (3D). The rate of Noise Equivalent counts is 1000 counts per slice per sec.

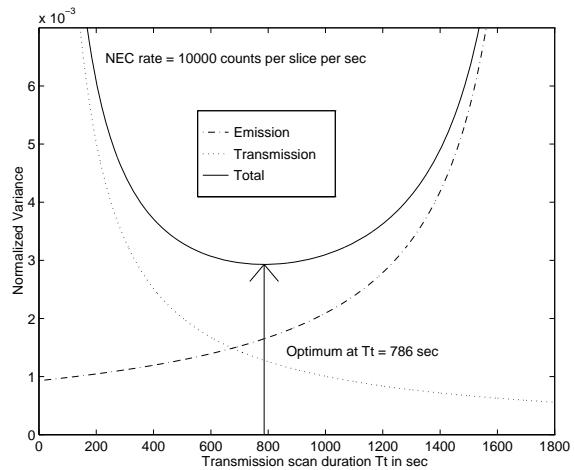


Figure 11.10 The transmission and emission noise terms and their sum as a function of the transmission scan duration, when the total (patient) scan time is 1800 sec. Parameters correspond to the Brain Phantom (3D). The rate of Noise Equivalent counts is 10000 counts per slice per sec.

11.6 Discussion of the Results

The general noise properties of reconstructive tomography are well described in the literature [124, 125]. Previous descriptions of the Advance scanner also include count rate dependence in phantoms and humans [129, 132, 133] but the noise characterization so far [134] focused on the emission noise. A theoretical derivation [126] using data from the GE 2048 scanner formed part of the basis for the design of the Advance scanner's 2-pin source transmission scanning system. In the present paper we have measured the relative importance of the noise contributions from blank, transmission, and emission scans for a range of imaging situations encountered. Transmission scan values are presented as scan durations in seconds for a given, specified set of sources. Optimal conditions with 2 sources of maximal strength (400 MBq) would almost double the transmission count rate. Given the current expensive Germanium-68 pin sources, the presented configuration represents a likely mean value over source life time. Results can be scaled according to the observed blank scan count rate, to accommodate for differences in number or activity of pin sources and their rotation radius.

An excellent fit to the empirical variance has been found both in 2D and 3D studies. The numerical values of the measure chosen for the noise examination is strongly dependent on reconstruction parameters, in part because of the neighbour pixel correlation disregarded in the theory section. Most parameters, however, will affect the emission and transmission contributions in the same way (through ϕ_{ij} only), and therefore the model and its output is considered adequate for the purpose of identifying areas in the E-T plane where one source is dominating. The analysis largely ignores the fact that the tomographic noise is non-stationary with higher values found towards the center as exemplified by the differently located ROIs in the brain phantom, but also in this respect is the ratio between the emission and the transmission term approximately constant.

From the estimated value of $d = 0.96$ it is seen that the blank scan contribution to the noise as expected is only a small fraction of the transmission noise for same duration, the ratio being well explained by the average attenuation of the (central part of) the 20 cm Cylinder phantom. For all practical purposes, therefore, the application of a 20 minutes blank scan will ensure that

the blank scan contribution is negligible since the cases where a longer transmission scan might be applied are those with a higher attenuation.

A typical 3D brain activation study with O-15 water in our setup will contain 0.5 Mcounts for a central slice, (Fig. 11.7,[132]). With a 10 minute transmission scan, the transmission variance calculated from the phantom data in Table 11.1 is 0.00167 compared to the emission contribution of 0.0334, i.e., the transmission adds (only) 5% to the final result. If difference images are made, the effect becomes even smaller. For the case MN the corresponding figures would be 0.00265 (transmission), 0.037 (emission) and 7%, respectively. This higher value is due to the lack of a skull for the Brain phantom.

A 10 minute, 2D FDG brain scan typically would also have 0.5-1 M counts yielding the same 5-10% ratio, while acquiring the same data in 3D would make the two contributions almost equal and therefore call for a more detailed analysis of timing. It should be noted that in this imaging condition no attribute has been made to the small additional noise from the emission correction of the transmission scan which is necessary if the transmission is performed with activity present.

Fig. 11.8 suggests that within the emission count range of a typical dynamic heart FDG scan, the emission noise contribution (when using a 20 minute transmission scan) is dominating. Data are, however, currently not available that can demonstrate the region in which the transmission noise curves split up. It should be emphasized that the observed shifts of the curves with short transmission time are due to line-artifacts caused by (clusters of) zeros in the transmission sinogram. This deteriorating effect is usually more important and calls for an improved methods of attenuation correction, e.g., by using image segmentation as described in [135].

11.7 Summary

The measured image noise variance from a GE Advance PET scanner has been modelled as a sum of terms corresponding to the noise in the emission scan, the transmission scan, and the blank scan. The weight parameters in this simple model have been determined from a large number of experiments in both 2D and 3D scan mode for the 20 cm standard (NEMA) cylinder, a torso-like ellipse with axes 18 and 35 cm, and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm. Furthermore, noise model parameters have been found from some human studies. For brain studies there is an excellent agreement between model and observation, and between phantoms and human studies. The estimated parameters have been used to optimize the durations of the emission and transmission scan under the constraint that their sum is constant. For studies of the heart or other body regions the model is still generally valid, but the parameters not as well documented, and the transmission noise is often dominated by line artifacts.

Conclusion and Topics for Further Research

This thesis has demonstrated that the Radon transform and its inverse can be implemented on ordinary computers and provide a stable and useful tool for image processing.

Two common parameter definitions have been considered with the linear Radon transform, and the central issues concerning sampling requirements have been covered thoroughly in Chapters 1 and 2. The literature often neglect to consider the implications of discretization, and a new sinc-interpolation strategy has been used to show that simple interpolation strategies in general are sufficient for line detection algorithms.

In Chapter 3 the relationship between the Radon and the Hough transform has been covered, and it was shown that the discrete forms of the two algorithms can give exactly the same parameter domain, if some restrictions are made on the sampling parameters. Many variations of the Hough transform can be found in the literature, and some of them are described in the last part of the chapter. From the literature, it is not clear how the Radon and the Hough transform are related. It has been demonstrated that the two methods act very differently, when changing the sampling distances in the parameter domain.

A new computationally efficient algorithm, named the FCE-algorithm, for fast curve parameter estimation has been presented in Chapter 4. The algorithm uses the Radon transform and the Hough transform in a generalized form, and exploits the best properties of the two. The algorithm initially uses the generalized Hough transform to estimate a coarsely sampled parameter domain, which is exploited to reduce the computational cost for the subsequently used generalized Radon transform. The algorithm has been validated in two examples, in which hyperbolas are detected from seismic images. The FCE-algorithm has been derived using a specific form of the generalized Radon transform, but the FCE-algorithm can be modified to cover virtually all parameterized curve shapes.

Novel approaches to analyze the influence of noise on the discrete Radon transform have been presented in Chapter 5. For the discrete Radon transform, it has been shown that parameterized curves can be detected even for extremely bad signal to noise ratios. It was also shown that the discrete Radon transform is robust to random fluctuations on the line pixels. Analytical expressions have been given in Chapter 5 to quantify the influence from these noise sources. It could also be of interest to make a combined analysis of these two noise sources.

A large software package has been developed for 2D Radon based reconstruction. Several Radon based reconstruction methods have been implemented, based on the theory presented in Chapters 7 and 8. The package also includes two other programs. The first is a program for generation of an image and the corresponding sinogram from a set of primitives. This program is based on the analytically derived Radon transform of a set of primitives, as shown in Section B.3. With the program it is simple to combine shifted, scaled, and rotated primitives to build a complex image and its sinogram. A noise model can also be added to the noise free sinogram,

hence the program is suitable for testing reconstruction algorithms.

The last program in the 2D reconstruction package contains several iterative reconstruction methods, implemented using a new and computationally efficient technique, shown in Chapter 9. A huge acceleration has been obtained with the last package compared to a traditional approach, where the individual values of the system matrix are not stored but calculated when needed.

Chapter 9 covered some parts of the linear algebra based reconstruction methods. Even though many aspects have been covered or mentioned in the thesis, several issues should be examined. A modification of the iterative methods that might aid performance is to incorporate different regularization techniques. This topic should be investigated in the future. Lately a very interesting article [118], based on the combination of the Least Squares Conjugate Gradient method and regularization techniques, showed interesting results.

One major question in all iterative methods, is the choice of a stopping criterion. Like it was assumed for the derivation of the EM algorithm, work has been done, where the fundamental assumption is that the values in the sinogram are generated in a Poisson process, but it seems doubtful that methods narrowly based on the Poisson assumption is the way to proceed. In practical PET reconstruction the noise has many sources, such as the noise from the finite length of the emission, transmission, and blank scan, cf. Chapter 11. As previously mentioned additional effects like scatter, randoms, and detector variations (Chapter PET in [58]) also contribute to a total noise model being far more complicated.

An important issue in iterative reconstruction theory is the modelling of the system matrix. In Chapter 9, examples mostly using Radon based approaches have been presented, but given that a reconstructed image is a result of both the sinogram, the system matrix, the number of iterations, iterative method chosen, it could also be of interest to take the interactions of the individual parts into account.

An interesting path to follow could be to incorporate prior knowledge of edges in the model of the reconstructed image and use mean field theory [136, 137] or [11, 12]. In the last two papers, shown in Appendix N, the model only operates on the reconstructed image. However, a combination of the reconstruction process and a more detailed model of the image seems to be a very interesting research area. This model is very relevant to PET reconstruction, where prior information could be included, e.g., found from a high resolution MR-scan. The MR-scan can be used to extract edges between localized anatomical structures, which can be used to stabilize the adaptive reconstruction methods for PET. Adaptive Finite Element methods [138] might also be a way to incorporate information about edges, in order to enhance and segment the volumes.

Chapter 10 was concerned with 3D reconstruction methods, where the volume of interest should be recovered from a set of line integrals through the volume. The derivation of several reconstruction methods has been shown. The methods are not contained within the normal Radon transform definition, but it has been shown that the 2D Radon based reconstruction methods can be generalized to the 3D case, and the 2D iterative methods are almost directly applicable for 3D reconstruction. For this purpose yet another software package has been developed, providing both direct and iterative reconstruction methods. The package also includes a program for generation of volumes and the corresponding sinogram from a set of primitives like in 2D.

The area of reconstruction methods is still under development, where new adaptive methods and very computer demanding methods will become feasible, due to the rapid development in computer performance. It has, e.g., been reported [139], that reconstruction methods can be implemented efficiently using texture mapping hardware. Techniques as VRML (Virtual Reality Modelling Language) and volume rendering, will soon add new dimensions to the interpretation of volume scans.

Part III

Appendices

Appendix A

The Dirac Delta Function

The Dirac delta function $\delta(\cdot)$ was introduced by Dirac [83]. The function is a part of a new class of functions known as generalized functions. A good reference for generalized functions is [140].

The delta function is special in the sense that it is defined from

$$\delta(x) = 0 \quad \text{for } x \neq 0 \tag{A.1}$$

$$\int_{-\infty}^{\infty} \delta(x) \, dx = 1 \tag{A.2}$$

The most important property of the delta function comes is that

$$\int_{-\infty}^{\infty} g(x) \delta(ax + b) \, dx = \frac{1}{|a|} \int_{-\infty}^{\infty} g\left(\frac{x - b}{a}\right) \delta(x) \, dx = \frac{1}{|a|} g\left(\frac{-b}{a}\right) \tag{A.3}$$

which also shows that the delta function is an even function of its argument $\delta(x) = \delta(-x)$.

If the delta function has a general function as argument is can be rewritten [73]

$$\delta(f(x)) = \sum_{i=1}^I \frac{\delta(x - x_i)}{|f'(x_i)|} \tag{A.4}$$

where x_i is the zeros of the function $f(x)$. Thus it is found that

$$\int_{-\infty}^{\infty} g(x) \delta(f(x)) \, dx = \sum_{i=1}^I \frac{g(x_i)}{|f'(x_i)|} \tag{A.5}$$

This result is very important with respect to the Radon transform and especially to the generalized Radon transform, where the argument of the delta function determines the curve shape.

The delta function is not a normal function and often approximations that converge to the delta function are quite useful, e.g.

$$\psi(x) = \frac{1}{\sqrt{\pi}a} e^{-\left(\frac{x}{a}\right)^2} \tag{A.6}$$

and in the limit

$$\lim_{a \rightarrow 0} \psi(x) = \delta(x) \tag{A.7}$$

The function $\psi(\cdot)$ can be very valuable, because it easily can be analyzed with conventional calculus and its derivatives can also be used to approximate the derivatives of the delta function.

$$\lim_{a \rightarrow 0} \psi^{(n)}(x) = \delta^{(n)}(x) \tag{A.8}$$

Appendix B

Properties of the Normal Radon Transformation

B.1 Basic Properties of the Radon Transform

In this section several basic properties of the normal Radon transform are shown. All of the following based on the properties of the δ -function reviewed in Appendix A.

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (\text{B.1})$$

B.1.1 Linearity

The first property is that the normal Radon transform is linearity. If α_q is an array of constants, then

$$g(x, y) = \sum_q \alpha_q g_q(x, y) \Rightarrow \check{g}(\rho, \theta) = \sum_q \alpha_q \check{g}_q(\rho, \theta) \quad (\text{B.2})$$

B.1.2 Shifting

Assume that a function $g(x, y)$ is shifted

$$h(x, y) = g(x - x_0, y - y_0) \Rightarrow \quad (\text{B.3})$$

$$\check{h}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x - x_0, y - y_0) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (\text{B.4})$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\tilde{x}, \tilde{y}) \delta((\rho - x_0 \cos \theta - y_0 \sin \theta) - \tilde{x} \cos \theta - \tilde{y} \sin \theta) d\tilde{x} d\tilde{y} \quad (\text{B.5})$$

$$= \check{g}(\rho - x_0 \cos \theta - y_0 \sin \theta, \theta) \quad (\text{B.6})$$

Note that only the ρ -coordinate is changed.

B.1.3 Rotation

Here $g(x, y)$ is expressed in polar form, i.e., $g(x, y) = g(r, \phi)$. In this case rotation is fairly easy

$$h(r, \phi) = g(r, \phi - \phi_0) \quad (\text{B.7})$$

$$\check{h}(\rho, \theta) = \int_{-\infty}^{\infty} \int_0^{\pi} g(r, \phi - \phi_0) \delta(\rho - r \cos \phi \cos \theta - r \sin \phi \sin \theta) |r| d\phi dr \quad (\text{B.8})$$

$$= \int_{-\infty}^{\infty} \int_0^{\pi} \check{g}(r, \tilde{\phi}) \delta(\rho - r \cos(\theta - \tilde{\phi} - \phi_0)) |r| d\tilde{\phi} dr \quad (\text{B.9})$$

$$= \check{g}(\rho, \theta - \phi_0) \quad (\text{B.10})$$

This is quite obvious. If the coordinate system (x, y) is turned ϕ_0 , then the Radon transform is also turned ϕ_0 .

B.1.4 Scaling

Assume a scaling in both coordinates

$$h(x, y) = g\left(\frac{x}{a}, \frac{y}{b}\right), \quad a > 0 \text{ and } b > 0 \quad (\text{B.11})$$

In this case the Radon transform can be rewritten

$$\check{h}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g\left(\frac{x}{a}, \frac{y}{b}\right) \delta(\rho - x \cos \theta - y \sin \theta) dx dy \quad (\text{B.12})$$

$$= ab \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\tilde{x}, \tilde{y}) \delta(\rho - a\tilde{x} \cos \theta - b\tilde{y} \sin \theta) d\tilde{x} d\tilde{y} \quad (\text{B.13})$$

$$= \frac{ab}{|\gamma|} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\tilde{x}, \tilde{y}) \delta\left(\frac{\rho}{\gamma} - \tilde{x} \frac{a \cos \theta}{\gamma} - \tilde{y} \frac{b \sin \theta}{\gamma}\right) d\tilde{x} d\tilde{y} \quad (\text{B.14})$$

$$\equiv \frac{ab}{|\gamma|} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\tilde{x}, \tilde{y}) \delta(\tilde{\rho} - \tilde{x} \cos \tilde{\theta} - \tilde{y} \sin \tilde{\theta}) d\tilde{x} d\tilde{y} \quad (\text{B.15})$$

$$= \frac{ab}{|\gamma|} \check{g}(\tilde{\rho}, \tilde{\theta}) \quad (\text{B.16})$$

Now $\tilde{\rho}$ and $\tilde{\theta}$ must be found as functions of ρ and θ . From the equations above it is found that

$$\cos \tilde{\theta} = \frac{a \cos \theta}{\gamma} \quad \text{and} \quad \sin \tilde{\theta} = \frac{b \sin \theta}{\gamma} \quad (\text{B.17})$$

$$\cos^2 \tilde{\theta} + \sin^2 \tilde{\theta} = 1 \Rightarrow \gamma = \sqrt{(a \cos \theta)^2 + (b \sin \theta)^2} \Rightarrow \quad (\text{B.18})$$

$$\tilde{\rho} = \frac{\rho}{\gamma} = \frac{\rho}{\sqrt{(a \cos \theta)^2 + (b \sin \theta)^2}} \quad (\text{B.19})$$

$$a \tan \tilde{\theta} = b \tan \theta \Rightarrow \tilde{\theta} = \arctan\left(\frac{b}{a} \tan \theta\right) \quad (\text{B.20})$$

To summarize the equations shown above

$$h(x, y) = g\left(\frac{x}{a}, \frac{y}{b}\right), \quad a > 0 \text{ and } b > 0 \Rightarrow \quad (\text{B.21})$$

$$\check{h}(\rho, \theta) = \frac{ab}{\gamma} \check{g}\left(\frac{\rho}{\gamma}, \arctan\left(\frac{b}{a} \tan \theta\right)\right) \quad (\text{B.22})$$

$$\gamma = \sqrt{(a \cos \theta)^2 + (b \sin \theta)^2} \quad (\text{B.23})$$

If using $0 \leq \theta < \pi$, then the arctan function is

$$\arctan\left(\frac{b}{a} \tan \theta\right) = \begin{cases} \text{Arctan}\left(\frac{b}{a} \tan \theta\right) & \text{if } 0 \leq \theta < \frac{\pi}{2} \\ \text{Arctan}\left(\frac{b}{a} \tan \theta\right) + \pi & \text{if } \frac{\pi}{2} \leq \theta < \pi \end{cases} \quad (\text{B.24})$$

B.1.5 Convolution

Assume the function $h(x, y)$ being a 2D convolution of $f(x, y)$ and $g(x, y)$.

$$h(x, y) = f(x, y) * g(x, y) = \int \int f(x_1, y_1) g(x - x_1, y - y_1) dx_1 dy_1 \quad (\text{B.25})$$

Then the Radon transform of $h(x, y)$ is given by

$$\begin{aligned} \check{h}(\rho, \theta) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, y_1) g(x - x_1, y - y_1) \delta(\rho - x \cos \theta - y \sin \theta) dx_1 dy_1 dx dy \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, y_1) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x - x_1, y - y_1) \delta(\rho - x \cos \theta - y \sin \theta) dx dy dx_1 dy_1 \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, y_1) \check{g}(\rho - x_1 \cos \theta - y_1 \sin \theta, \theta) dx_1 dy_1 \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, y_1) \int_{-\infty}^{\infty} \check{g}(\rho - \rho_1, \theta) \delta(\rho_1 - x_1 \cos \theta - y_1 \sin \theta, \theta) d\rho_1 dx_1 dy_1 \\ &= \int_{-\infty}^{\infty} \check{g}(\rho - \rho_1, \theta) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, y_1) \delta(\rho_1 - x_1 \cos \theta - y_1 \sin \theta, \theta) dx_1 dy_1 d\rho_1 \\ &= \int_{-\infty}^{\infty} \check{f}(\rho_1, \theta) \check{g}(\rho - \rho_1, \theta) d\rho_1 \quad (\text{B.26}) \\ &= \check{f}(\rho, \theta) * \check{g}(\rho, \theta) \quad (\text{B.27}) \end{aligned}$$

The result is that the Radon transform of a two dimensional convolution is a one dimensional convolution of the Radon transformed functions with respect to ρ . The result is quite obvious from the Fourier Slice Theorem shown in Eq. 7.6. The two dimensional Fourier transformation of a convolution becomes a product. The product is still a product expressed in polar coordinates. The inverse one dimensional Radon transform of the product becomes a convolution in ρ .

B.2 The Shepp-Logan Phantom Brain

A ‘famous’ model of a brain is the Shepp Logan Phantom. This is a model based on ellipses, which can be Radon transformed analytically. The phantom can be used to test the numerical algorithms and evaluate the numerical errors quantitatively. Using the properties shown in Section B.1 a sum of ellipses can be transformed analytically, if a circle with radius one can be transformed, as shown in Fig. B.1. A unit circle is here defined as

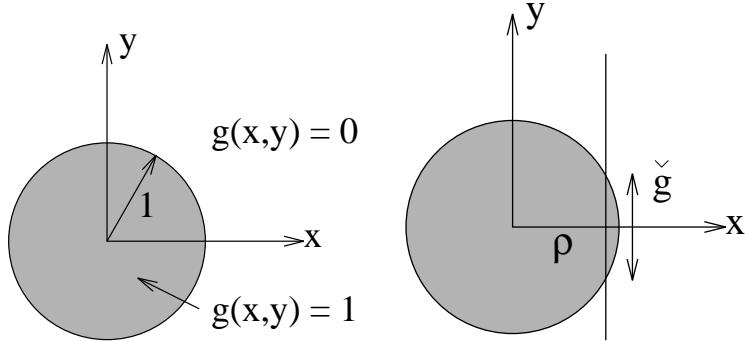
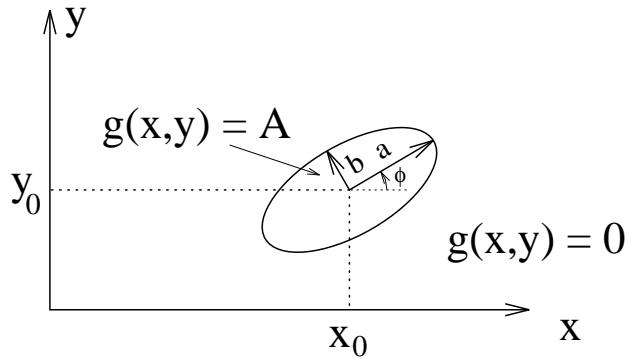
$$g(x, y) = \begin{cases} 1 & \text{if } x^2 + y^2 < 1 \\ 0 & \text{otherwise} \end{cases} \quad (\text{B.28})$$

In this case the function is invariant to rotation around the origin of the coordinate system, thus the Radon transform does not depend on θ . The Radon transform can, e.g., be calculated for $\theta = 0$, as shown in Fig. B.2. Due to the constant excitation on the circle, the Radon transform is merely the length of the line crossing the circle.

From Fig. B.2 it is found that

$$\check{g}(\rho, \theta) = \begin{cases} 2\sqrt{1 - \rho^2} & \text{if } \rho^2 < 1 \\ 0 & \text{otherwise} \end{cases} \quad (\text{B.29})$$

Using Eq. B.29 and the properties shown in appendix B, it is found that a sum of ellipses can be transformed analytically. If the unit circle is scaled, rotated and shifted an ellipse is formed, shown in Fig. B.3.

**Figure B.1** The unit circle**Figure B.2** Radon transformation for $\theta = 0$.**Figure B.3** An ellipse with the relevant parameters indicated

If the ellipse is called $g_{A,a,b,\phi,x_0,y_0}(x, y)$, then the Radon transform is given by

$$\check{g}_{A,a,b,\phi,x_0,y_0}(\rho, \theta) = \begin{cases} 2Aab \frac{\sqrt{\rho_0^2 - (\rho - x_0 \cos \theta - y_0 \sin \theta)^2}}{\rho_0^2} & \text{if } |\rho - x_0 \cos \theta - y_0 \sin \theta| < \rho_0 \\ 0 & \text{otherwise} \end{cases} \quad (\text{B.30})$$

$$\rho_0^2 = a^2 \cos^2(\theta - \phi) + b^2 \sin^2(\theta - \phi) \quad (\text{B.31})$$

If Q ellipses are given with a set of parameters, then due to the linearity, the Radon transform is a sum

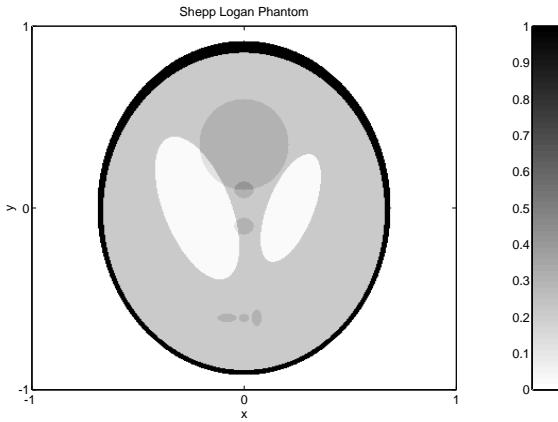
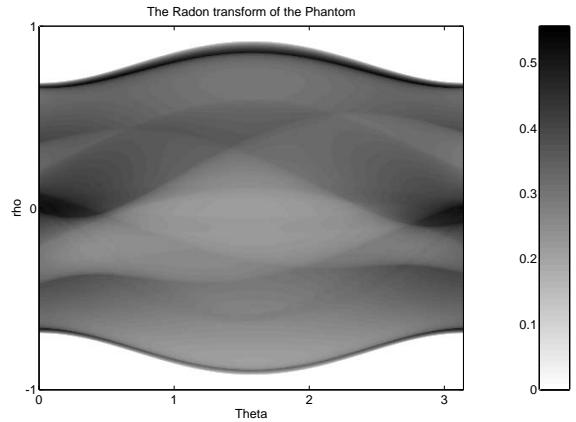
$$g(x, y) = \sum_{q=1}^Q g_{A_q, a_q, b_q, \phi_q, x_{0,q}, y_{0,q}}(x, y) \Rightarrow \quad (\text{B.32})$$

$$\check{g}(\rho, \theta) = \sum_{q=1}^Q \check{g}_{A_q, a_q, b_q, \phi_q, x_{0,q}, y_{0,q}}(\rho, \theta) \quad (\text{B.33})$$

The Shepp-Logan Phantom is a sum of ten ellipses with parameters, shown in Table B.1. Note that the ellipse amplitude A does not exactly match those of the original phantom. This choice gives a better contrast.

The resulting phantom brain is shown in Fig. B.4 and the corresponding Radon transform is shown in Fig. B.5.

A	a	b	x_0	y_0	ϕ
1.0	0.69	0.92	0.0	0.0	0°
-0.8	0.6624	0.874	0.0	-0.0184	0°
-0.2	0.11	0.31	0.22	0.0	-18°
-0.2	0.16	0.41	-0.22	0.0	18°
0.1	0.21	0.25	0.0	0.35	0°
0.1	0.046	0.046	0.0	0.1	0°
0.1	0.046	0.046	0.0	-0.1	0°
0.1	0.046	0.023	-0.08	-0.605	0°
0.1	0.023	0.023	0.0	-0.606	0°
0.1	0.023	0.046	0.06	-0.605	0°

Table B.1 Parameter settings for the Shepp Logan Phantom.**Figure B.4** The Shepp Logan Phantom brain.**Figure B.5** The corresponding Radon transform.

B.3 Analytical Radon Transform of Primitives

During the project the normal Radon transform of several functions were derived analytically. The results are given in the following subsections. Using the linearity of the Radon transform and basic rules about translation, rotation and scaling rather complicated two dimensional functions can be modelled and the corresponding Radon transform can be expressed on an analytical form. This provides a useful tool to generate test sets for testing of reconstruction algorithms. A program is available, where the theory is implemented and sampled versions of the functions and their Radon transform can be obtained. In [22] and Section C.1 are also shown how to use the program. The following paragraphs show which basic functions and their Radon transform, available in the program. The program is available from [10].

B.3.1 The Circular Disc

The normal Radon transform of the circular disc normal Radon transform was found in Eq. B.29. In Fig. B.6 the basic function is shown and Fig. B.7 shows the corresponding parameter domain. Due to the rotational symmetry of the disc, the Radon transform does not depend on the angle θ .

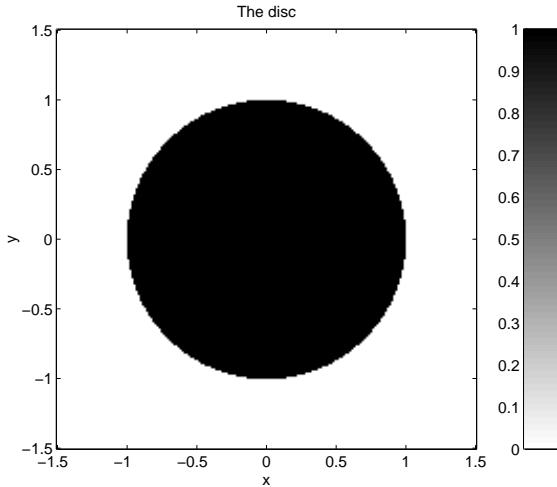


Figure B.6 The disc.

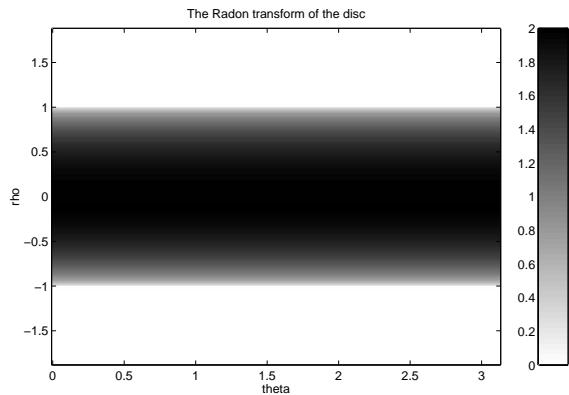


Figure B.7 The corresponding Radon transform.

B.3.2 The Square

Another very useful primitive is the square as shown in Fig. B.8. This function can be written as

$$g_s(x, y) = \mu(1 - |x|)\mu(1 - |y|) \quad (\text{B.34})$$

where $\mu(\cdot)$ is the Hamilton step function. It is easy to see that the Radon transform of this primitive must exhibit several symmetries.

$$g_s(x, y) = g_s(|x|, |y|) \quad (\text{B.35})$$

$$\check{g}_s(\rho, \theta) = \check{g}_s(\rho, \theta + p\frac{\pi}{2}) \quad (\text{B.36})$$

$$= \check{g}_s(\rho, \frac{\pi}{2} - \theta) \quad (\text{B.37})$$

where p is an integer. Eq. B.36 implies that the Radon transform is only needed in the interval $[0; \pi/2]$. Eq. B.37 further sharpens this demand to the interval $[0; \pi/4[$. Using the general feature $\check{g}(\rho, \theta) = \check{g}(-\rho, \theta + \pi)$ implies together with Eq. B.36 that only $\rho \geq 0$ must be investigated. It is also easy to find that

$$\check{g}_s(\rho, \theta) = 0 \text{ if } \rho > \sqrt{2} \quad (\text{B.38})$$

The remaining case $\rho < \sqrt{2}$ is fairly easy due to the simple geometry. First are the intersections between the line $\rho = x \cos \theta + y \sin \theta$ and the boundaries found

$$y = -1 \Rightarrow x = x_{-1} = \frac{\rho}{\cos \theta} + \tan \theta \quad (\text{B.39})$$

$$y = 1 \Rightarrow x = x_1 = \frac{\rho}{\cos \theta} - \tan \theta \quad (\text{B.40})$$

$$x = 1 \Rightarrow y = y_1 = \frac{\rho}{\sin \theta} - \cot \theta \quad (\text{B.41})$$

Three cases exist

$$x_1 > 0 \Rightarrow \check{g}(\rho, \theta) = 0 \quad (\text{B.42})$$

$$x_1 < 1 \text{ and } x_{-1} < 1 \Rightarrow \quad (\text{B.43})$$

$$\check{g}(\rho, \theta) = \sqrt{2^2 + (x_1 - x_{-1})^2} \quad (\text{B.44})$$

$$= \frac{2}{\cos \theta} \quad (\text{B.45})$$

$$x_1 < 1 \text{ and } x_{-1} > 1 \Rightarrow \quad (\text{B.46})$$

$$\check{g}(\rho, \theta) = \sqrt{(1 - x_1)^2 + (1 - x_{-1})^2} \quad (\text{B.47})$$

These formulas are computed in their respective region giving the total Radon transform of the square. In Fig. B.8 the square is shown and Fig. B.9 shows the corresponding Radon transform.

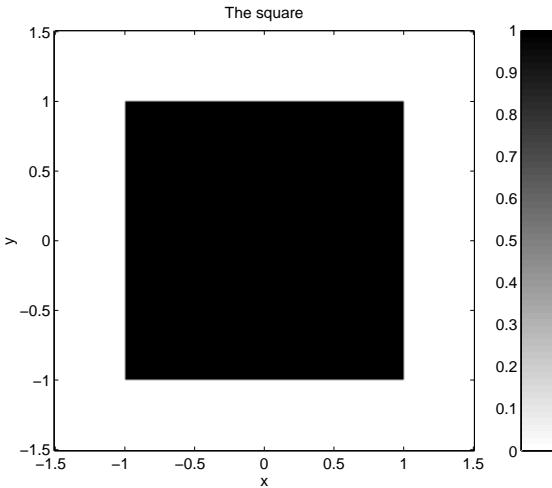


Figure B.8 The square.

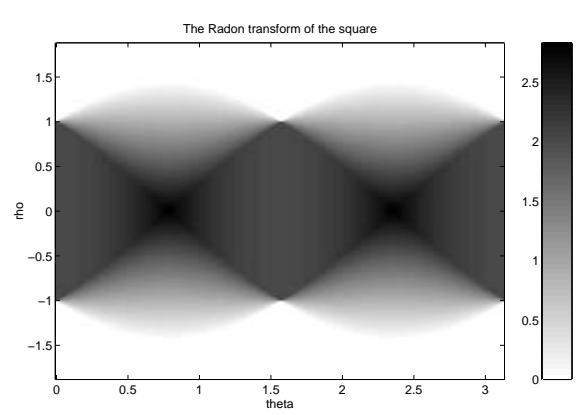


Figure B.9 The Radon transform of the square.

B.3.3 The Triangle

The following primitive is the triangle shown in Fig. B.10. This Radon transform of this primitive can also be calculated on an analytical form. The first thing is that the rotational symmetry implies that

$$\check{g}(\rho, \theta) = \check{g}(\rho, -\theta) = \check{g}\left(\rho, \theta + p\frac{2\pi}{3}\right) \quad (\text{B.48})$$

where p is an integer. These relations together with $\check{g}(\rho, \theta) = \check{g}(-\rho, \theta + \pi)$ implies that it is only relevant to investigate $\rho \geq 0$ and $0 \leq \theta < \frac{\pi}{3}$.

Three lines are relevant

- $l_- : y = -\frac{1}{\sqrt{3}}(x - 1)$, i.e., the upper slanted line of the triangle.
- $l_+ : y = \frac{1}{\sqrt{3}}(x - 1)$, i.e., the lower slanted line of the triangle.
- $l : \rho = x \cos \theta + y \sin \theta$, i.e., the transformation line.

First is the intersection between l and l_+ is found, i.e.,

$$x_+ = \frac{\sin \theta + \sqrt{3}\rho}{\sin \theta + \sqrt{3} \cos \theta} \quad (\text{B.49})$$

$$y_+ = \frac{1}{\sqrt{3}}(x_+ - 1) \quad (\text{B.50})$$

and the intersection between l and l_-

$$x_- = \frac{\sin \theta - \sqrt{3}\rho}{\sin \theta - \sqrt{3} \cos \theta} \quad (\text{B.51})$$

$$y_- = -\frac{1}{\sqrt{3}}(x_- - 1) \quad (\text{B.52})$$

Another interesting intersection is between the vertical line $x = -\frac{1}{2}$ and the integration line. The corresponding $y = y_l$ is given by

$$y_l = \frac{2\rho + \cos \theta}{2 \sin \theta} \quad (\text{B.53})$$

It is easy to realize that $x_+ > 0$ implies that the integration line does not cross the triangle, i.e., $\check{g}(\rho, \theta) = 0$.

The remaining cases all imply that the integration line intersects l_+

$$y_l < \frac{\sqrt{3}}{2} \Leftrightarrow \check{g}(\rho, \theta) = \sqrt{\left(x_+ + \frac{1}{2}\right)^2 + (y_l - y_+)^2} \quad (\text{B.54})$$

$$y_l > \frac{\sqrt{3}}{2} \Leftrightarrow \check{g}(\rho, \theta) = \sqrt{(x_+ + x_-)^2 + (y_+ - y_-)^2} \quad (\text{B.55})$$

These formulas are computed in their respective region giving the total Radon transform of the triangle. In Fig. B.10 the square is shown and Fig. B.11 shows the corresponding Radon transform.

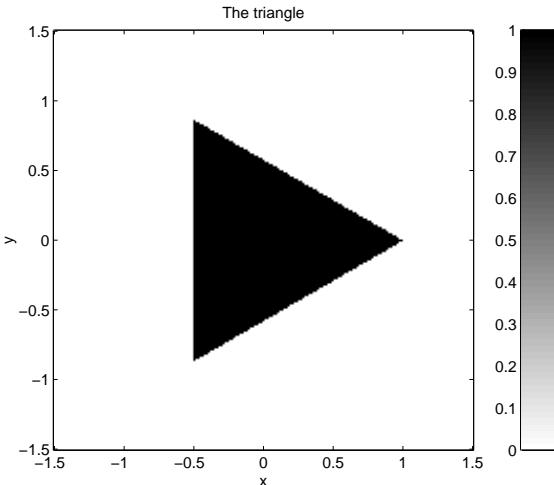


Figure B.10 The triangle.

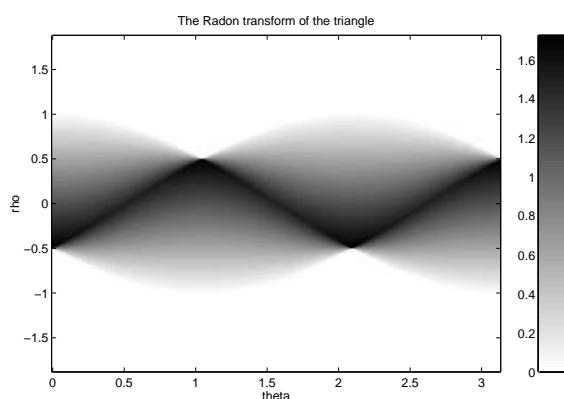
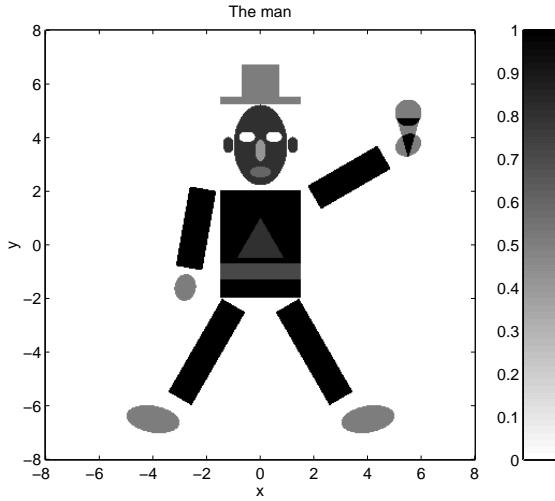
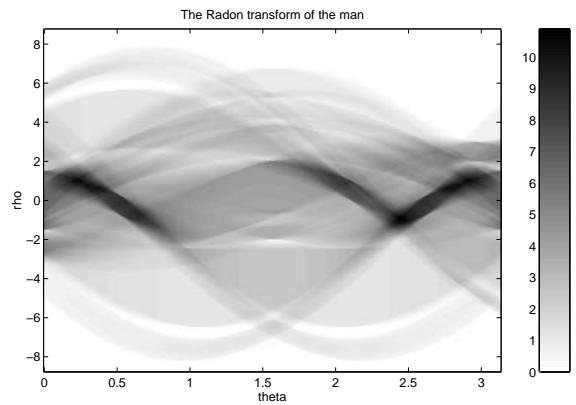


Figure B.11 The Radon transform of the triangle.

Using the presented set of primitives with proper scaling, rotation and translation as shown in the first part of this appendix, e.g., a “man” can be created as shown in Fig. B.12. The corresponding Radon transform is shown in Fig. B.13.

**Figure B.12** The man.**Figure B.13** The Radon transform of the Man.

B.3.4 The Gaussian Bell

The previous cases all concern finite objects with sharp edges. The next case is the infinite and soft varying Gaussian bell

$$g(x, y) = \exp(-x^2 - y^2) \quad (\text{B.56})$$

Due to the rotational symmetry the Radon transform can be calculated at any angle, e.g., $\theta = 0$, where the integration reduces to

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} g(x = \rho, y) dy \quad (\text{B.57})$$

$$= \exp(-\rho^2) \int_{-\infty}^{\infty} \exp(-y^2) dy \quad (\text{B.58})$$

$$= \sqrt{\pi} \exp(-\rho^2) \quad (\text{B.59})$$

The Gaussian bell is shown in Fig. B.14 and the corresponding Radon transform given in Eq. B.59 is shown in Fig. B.15. Note that the bell is infinite, thus it cannot be bounded in the image- or the Radon-domain.

B.3.5 The Pyramid

The square can be used to simulate digital images using pixels with constant excitation. This is in fact a nearest neighbour approach. In some application, e.g., Radon transformation, the nearest neighbour approach introduces too much noise. In this case a linear interpolation strategy is very common. The discrete image is convoluted with a triangle in both directions, resulting in a pyramid-like structure. If normalizing the sampling distance to 1, the convolution kernel can be written as

$$g(x, y) = \begin{cases} (1 - |x|)(1 - |y|) & \text{for } |x| < 1, |y| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (\text{B.60})$$

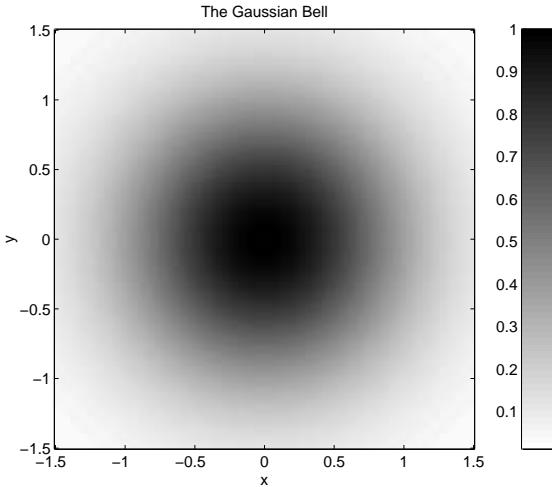


Figure B.14 The Gaussian Bell.

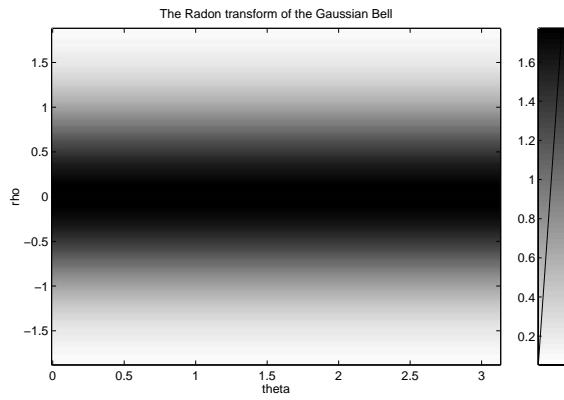


Figure B.15 The Radon transform of the Gaussian Bell.

The interpolated image can be written as the integral

$$s_i(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sum_m \sum_n s(m, n) \delta(x' - x_m) \delta(y' - y_n) g(x - x', y - y') dx' dy' \quad (\text{B.61})$$

$$= \sum_m \sum_n s(m, n) g(x - x_m, y - y_n) \quad (\text{B.62})$$

where $s(m, n)$ is the discrete image. Due to the linearity and translation property the Radon transform of the continuous function $s_i(x, y)$ can be found if the Radon transform of the kernel $g(x, y)$ can be found. This will be done.

Due to the symmetry in the pyramid-like kernel it is easy to find that

$$\check{g}(\rho, \theta) = \check{g}(-\rho, \theta) = \check{g}(\rho, \theta + p\pi/2) = \check{g}(\rho, \pi/2 - \theta) \quad (\text{B.63})$$

It is also apparent that the Radon transform is zero if $\rho > \sqrt{2}$, i.e., the Radon transform is only needed in the interval $0 < \rho < \sqrt{2}$ and $0 \leq \theta < \pi/4$.

$$\check{g}(\rho, \theta) = \frac{1}{|\cos \theta|} \int_{-\infty}^{\infty} g\left(\frac{\rho}{\cos \theta} - y \tan \theta, y\right) dy \quad (\text{B.64})$$

$$= \frac{1}{|\cos \theta|} \int_{y^*}^1 (\cos \theta - |\rho - y \sin \theta|)(1 - |y|) dy \quad (\text{B.65})$$

where

$$y^* = \max\{-1, y_1\} \quad (\text{B.66})$$

and

$$y_1 = \frac{\rho - \cos \theta}{\sin \theta} \quad (\text{B.67})$$

For test purposes it can be used that $y_1 > 1$ implies that $\check{g}(\rho, \theta) = 0$. The meaning of y_1 is indicated in Fig. B.16.

Yet another definition. It turns out that a vital parameter is

$$y_0 = \frac{\rho}{\sin \theta} \quad (\text{B.68})$$

Solving the rather simple integrals in Eq. B.65 (it only requires good bookkeeping) the Radon transform can be derived. The result can be written as

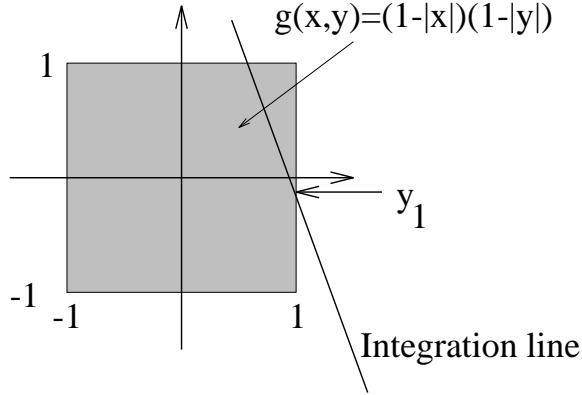


Figure B.16 The Pyramid.

$$y_1 < -1 \text{ and } y_0 > 1$$

$$\check{g}(\rho, \theta) = \frac{3 \cos \theta - \rho}{\cos^2 \theta} \quad (\text{B.69})$$

$$y_1 < -1 \text{ and } y_0 < 1$$

$$\check{g}(\rho, \theta) = \frac{9 \cos \theta - \sin \theta - 3\rho y_0 + \rho y_0^2}{3 \cos^2 \theta} \quad (\text{B.70})$$

$$-1 < y_1 < 0 \text{ and } y_0 > 1$$

$$\check{g}(\rho, \theta) = \frac{\cos \theta(9 - 3y_1 + 5y_1^2) - 3\rho + 3\rho y_1 + \rho y_1^2 + \sin \theta}{6 \cos^2 \theta} \quad (\text{B.71})$$

$$-1 < y_1 < 0 \text{ and } y_0 < 1$$

$$\check{g}(\rho, \theta) = \frac{\cos \theta(9 - 3y_1 + 5y_1^2) + \rho(3 - 3y_0 + 2y_0^2 + 3y_1 + y_1^2) - \sin \theta}{6 \cos^2 \theta} \quad (\text{B.72})$$

$$y_1 > 0 \text{ and } y_0 > 1$$

$$\check{g}(\rho, \theta) = \frac{\cos \theta(9 - 3y_1 - 5y_1^2) + \rho(3y_1 - 3 - y_0 - y_1^2) + \sin \theta}{6 \cos^2 \theta} \quad (\text{B.73})$$

$$y_1 > 0 \text{ and } y_0 < 1$$

$$\check{g}(\rho, \theta) = \frac{\cos \theta(9 - 3y_1 - 5y_1^2) + \rho(3 - 6y_0 + 3y_1 + 2y_0^2 - y_1^2) - \sin \theta}{6 \cos^2 \theta} \quad (\text{B.74})$$

In Fig. B.17 the basic ‘‘pyramid’’-function $g(x, y)$ is shown and finally Fig. B.18 shows the corresponding Radon transform.

Note that the Radon transform could also be calculated from the square using that the auto correlation function can be Radon transformed.

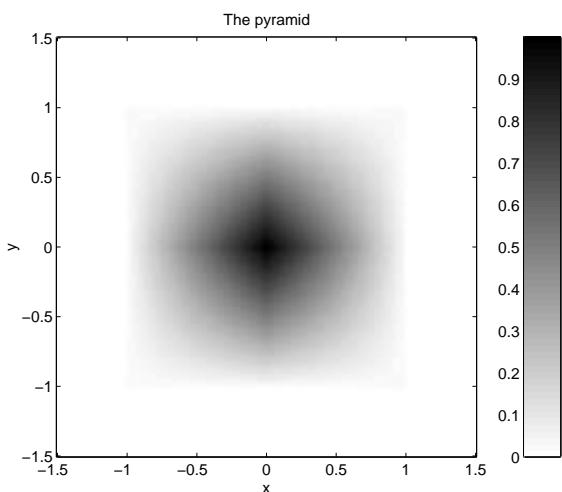


Figure B.17 The pyramid.

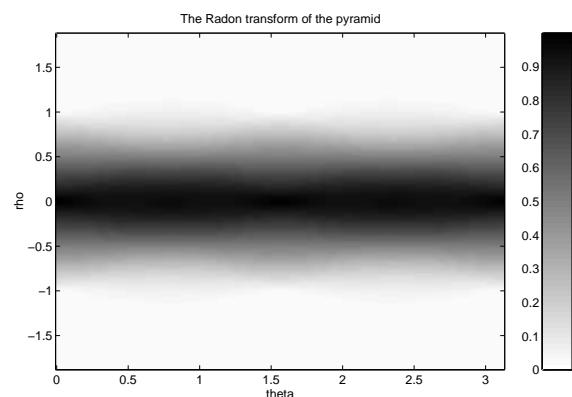


Figure B.18 The Radon transform of the pyramid.

Appendix C

Usage of the 2D Program Packages

Several software packages has been made for reconstruction of sinograms. The packages are provided for free, but protected by the GNU General Public License. All programs described in this appendix has been developed on a Linux system and implemented in C. The program should compile directly on any GNU-C Unix system. The programs are available from [10].

In Section C.1 a program called “RadonAna” is described. The program is based on the derivation of the Radon transform of several well known and some new primitives, shown in Section B.3. The program can be used to generate pairs of analytically images and their Radon transform. The program can, e.g., be used to benchmark reconstruction packages.

In Section C.2 the usage of the program “iradon” is described. In the program several of the classical reconstruction techniques have been implemented.

In Section C.3 the iterative reconstruction program “it” is described. Here slow and fast versions of ART, EM and LSCG reconstruction have been implemented.

C.1 The Analytical Sinogram Program “RadonAna”

The program `RadonAna` can be used to generate Radon transform pairs. The basic idea is to specify a set of scaled, rotated, and shifted primitives from which the Radon transform is calculated and sampled in both the image domain and the Radon domain.

The program will produce two files `OutFileNamea.fif` and `OutFileNameR.fif`, where the file `OutFileName` is specified by the user, and the a-file is the image and the r-file is the radon space.

The program uses the following parameters

`OutFileName` [String] With this parameter the base file name is determined. If `OutFileName` is `man.fif` then the image is written in `mana.fif` and the sinogram in the file `manr.fif`.

`XSamples` [Integer] Specifies the number of image samples on the first axis. Should be an odd number.

`YSamples` [Integer] Specifies the number of image samples on the second axis. Should be an odd number. If not given the program uses `XSamples=YSamples`.

`DeltaX` [Float] Specifies the sampling distance of both axes in the image.

`Xmin` [Float] Specifies the minimum sample position in the image on the first axis. If not given the image is centered around the middle.

`Ymin` [Float] Specifies the minimum sample position in the image on the second axis. If not given the image is centered around the middle.

RhoSamples [Integer] Specifies the number of samples in the distance parameter ρ in the sinogram. Should be an odd number.

ThetaSamples [Integer] Specifies the number of samples in the angular parameter θ in the sinogram. The sinogram is sampled linearly from 0 to (approximately) π radians. The sampling distance in θ is $\pi/\text{ThetaSamples}$.

DeltaRho [Float] Specifies the sampling distance in ρ . The program will center the sampling points around 0.

OverSamp [Integer] This parameter specifies whether an oversampled image and sinogram should be generated or not, i.e., if **OverSamp**=0. If, e.g., **OverSamp**=2 an image is calculated with 5 times ($2*\text{OverSamp}+1$) the resolution in both parameters. This image is then averaged so the output value make a better approximation to the average value under the sample. This technique is also applied the the sinogram. It will reduce aliasing problems.

GenerateImage [Integer] If set to 1 the image is generated.

GenerateRadon [Integer] If set to 1 the sinogram is generated.

NumberOfShapes [Integer] Here the number of primitives are specified.

DebugNiveau [String] Log files is generated if this parameter is set to **_Debug**. No log-file and no screen output can be used with **DebugNiveau=_HardCore**.

The ordering of the parameters shown above are arbitrary. After these parameters a number of lines must follow specifying a scaling, rotation, and shifting parameters of each primitive. The line uses the following parameters in this order

Shape*i* [String] where *i* is 0 to **NumberOfShapes**-1.

Type [Integer] The type of primitive, where

- 1 A circle centered around (0,0) with radius 1 and uniform signal 1 on the disk.
- 2 A square centered around (0,0). $-1 < x < 1$ and $-1 < y < 1$. The signal is 0 on the disk.
- 3 A Gaussian bell $\exp(-x^2 - y^2)$.
- 4 An even sided triangle centered around (0,0). The triangle has corners $(1, 0)$, $(-\frac{1}{2}, \frac{\sqrt{3}}{2})$ and $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$.
- 5 The image and the Radon domain is con-terminated with white Gaussian noise. Note that in this case the noise term in the Radon domain is NOT the Radon transform of the noise in the image.
- 6 A "pyramid" $f(x, y) = (1 - |x|)(1 - |y|)$ for $-1 < x < 1$ and $-1 < y < 1$.

a [Float] Scaling of the primitive in the direction of the first axis.

b [Float] Scaling of the primitive in the direction of the second axis.

x0 [Float] Shift of the primitive in direction of the first axis.

y0 [Float] Shift of the primitive in direction of the second axis.

phi [Float] Rotation of the primitive measured in degrees.

power [Float] Scaling of the intensity of the primitive.

Description [String] Optional description of the primitive.

The program is executed with the initialization file or .INI-file as sole argument. A typical .INI-file could be the one used to generate the man. Note that comments can be used, such as the line above Shape0.

```
OutFileName=man.fif
XSamples=401           Number of X samples
DeltaX=0.04            Sampling Distance in X
RhoSamples=351          Number of rho samples
OverSamp=0              Half oversampling factor.
GenerateImage=1         Generates image if 1. Optional -> Generates if omitted
GenerateRadon=1         Generates sinogram if 1. Optional -> Generates if omitted
DeltaRho=0.05            Sampling distance in rho
ThetaSamples=300          Number of theta samples. Distributed from 0 to pi.
NumberOfShapes=22          Number of simple elements.
DebugNiveau=_DDebug
```

	type	a	b	x0	y0	phi	power	description
Shape0=	1	1	1.5	0.0	3.7	0	0.8	head
Shape1=	1	0.4	0.2	0.0	2.7	0	-0.2	mouth
Shape2=	2	1.5	2.0	0.0	0.0	0	1.0	stomach
Shape3=	2	1.5	0.3	0.0	-1.0	0	-0.3	belt
Shape4=	2	1.5	0.5	3.3	2.5	30	1.0	right arm
Shape5=	2	1.5	0.5	-2.4	0.6	-100	1.0	left arm
Shape6=	4	1	1.0	0.0	0.0	90	-0.2	triangle on stomach
Shape7=	1	0.2	0.3	-1.2	3.7	0	0.8	left ear
Shape8=	1	0.2	0.3	1.2	3.7	0	0.8	right ear
Shape9=	2	1.5	0.15	0.0	5.35	0	0.5	lower part of hat
Shape10=	2	0.7	0.6	0.0	6.1	0	0.5	upper part of hat
Shape11=	1	0.3	0.2	-0.5	4.0	0	-0.8	left eye
Shape12=	1	0.3	0.2	0.5	4.0	0	-0.8	right eye
Shape13=	1	0.2	0.4	0.0	3.5	0	-0.4	nose
Shape14=	2	2.0	0.5	2.0	-4.0	-60	1.0	right leg
Shape15=	1	0.5	0.4	5.5	3.7	30	0.5	right hand
Shape16=	1	1	0.5	4.0	-6.5	10	0.5	left foot
Shape17=	2	2.0	0.5	-2.0	-4.0	60	1.0	left leg
Shape18=	1	0.5	0.4	-2.8	-1.6	-100	0.5	left hand
Shape19=	1	1	0.5	-4.0	-6.5	-10	0.5	right foot
Shape20=	4	1	0.5	5.5	4.2	-90	0.5	ice cream cone in right hand
Shape21=	1	0.5	0.5	5.5	4.9	0	0.5	only one ice cream cube!

C.2 The Direct Reconstruction Program “iradon”

In [59] the “iradon” package is described. Later additional features have been added, and here the basic possibilities are given.

The ordering of the following items which can be specified in the parameter file (.ini-file) are not important, but each require one line.

OutFile [String] Name of the file to write the reconstructed image (for the reconstruction functions). The program recognizes the following file extensions:

- fif** Float image format which includes sampling parameters.
- gif** The very popular graphics format.
- dat** raw and outdated image format.
- mat** Matlab files with one matrix.
- analyze** GE specific file format.

InFile [String] Name of the file to read the sinogram image (for the reconstruction functions). The sinogram should be a fif-file and contain the sampling parameters.

OrgFile [String] (Optional) If the name of a fif-file is provided, the sampling parameters are used to create the output-file, and measures of misfit can be made between the OrgFile and the OutFile. This can be used to input the true solution and get error levels.

Function [String] The function the program should do. Currently supported functions are

- FB** Filtered Backprojection.
- BF** Filtering After Backprojection.
- CNF** Central Slice. FFT based with Nearest Neighbour approximation.
- CBF** Central Slice. FFT based with Bilinear Interpolation.
- CNC** Central Slice. Chirp-z based with Nearest Neighbour approximation.
- CBC** Central Slice. Chirp-z based with Bilinear Interpolation.
- Forward** The InFile is forward Radon transformed into the OutFile.
- Convert** Very handy function Convert Infile directly to OutFile (in another file-format).
- Trace** Pick a trace of the InFile.
- Info** Get statistics about the Infile.

DebugLevel [String] This parameter controls the level of output. The parameter is mixed and overrules with the one used in the Print-statements (extended printf defined in the program).

- Normal** Standard level of output to screen and log file.
- Debug** Almost all output is logged to screen and log file.
- NoScreen** Screen output is disabled and log file level is ordinary.
- NoLog** Log file output is disabled and screen level is ordinary.
- HardCore** No information at all.

Palette [String] (Optional) Name of external colormap file to use when writing gif-images.

InterPol [Integer] (Optional) Interpolation level. Used by Filtered Backprojection to interpolate the sinogram. Default value is 1.

FilterType [String] (Optional) Filter choice in Filtered Backprojection. For all three choices see also **FilterCutoff**.

Ramp Ordinary Ramp filter.

Hanning The ramp filter is apodized with Hanning window.

Hamming The ramp filter is apodized with Hamming window.

FilterCutoff [Float] (Optional) For Filtered Backprojection the frequency where the filter is set to zero. The number is relative to half of the sampling frequency, i.e., should be set between 0 and 1.

SliceNumber [Integer] (Optional) If the InFile is an analyze-file with a lot of slices, then this parameter will select the slice number.

Xmin [Float] The minimum x-position of the reconstructed image. Not needed if a fif-file is provided as OrgFile.

Ymin [Float] The minimum y-position of the reconstructed image. Not needed if a fif-file is provided as OrgFile.

DeltaX [Float] Sampling distance on the x-axis. Not needed if a fif-file is provided as OrgFile.

DeltaY [Float] Sampling distance on the y-axis. Not needed if a fif-file is provided as OrgFile.

XSamples [Integer] Number of samples on the x-axis. Not needed if a fif-file is provided as OrgFile.

YSamples [Integer] Number of samples on the y-axis. Not needed if a fif-file is provided as OrgFile.

Assume that the sinogram **mysino.fif** should be reconstructed into **rec.fif** using Filtered Backprojection with the same parameters as found in **myorg.fif**, then a possible parameter file **myrec.ini** could look like:

```
InFile=mysino.fif
OutFile=rec.fif
Function=FB
DebugLevel=Normal
InterPol=1
OrgFile=myorg.fif
```

C.3 The Fast Iterative Reconstruction Program “it”

The “it” program implements three of the major iterative reconstruction techniques: ART, EM, and LSCG, but with the basic framework provided virtually all iterative reconstruction algorithms based on matrix operations can easily be implemented. The program uses many of the same elements as “iradon” and provides several new for matrix operations relevant for iterative reconstruction.

OutFileName [String] Name of the file to write the reconstructed image. File formats are as described under “iradon”.

InFileName [String] Name of the file where the sinogram image can be read. File formats are as described under “iradon”.

RefFileName [String] If provided the reconstructed image will use sampling parameters from this image. Error measures can be made between the RefFile and the OutFile.

StartFileName [String] If provided the initial guess on the reconstructed image is taken from this image, else a constant solution will be assumed from start.

Algorithm [String] The iterative reconstruction method used: Currently three are available: ART, EM and CG (LSCG).

UseFast [Integer] If 1 then fast reconstruction is used, where the system matrix is stored using sparse techniques, else slower but memory efficient reconstruction is used.

RadonKernel [String] The type of kernel used to model the system matrix. Currently available are the following methods

NN Two-level Nearest Neighbour approximation. (Memory consuming).

RNN Ray driven Nearest Neighbour discrete Radon transform based (Very fast with small system matrix).

RL Ray driven Linear Interpolation discrete Radon transform based (Fast with small system matrix).

P1 Method based on Radon transformation of square with pre-guidance (slow but good).

P2 Method based on Radon transformation of square with no pre-guidance (slower but better).

SINC Sinc interpolation methods in the image and analytically Radon of that. (Very slow).

IterationType [Integer] For ART: If set to 1 a cyclical selection of the row index is used, and else a random selection is chosen.

Iterations [Integer] For EM and CG the number of iterations before the iteration ends. For ART the number of full iterations, i.e., the number of actual ART iterations is **Iterations** times the number of rows in the system matrix.

SaveIterations [Integer] (optional) If set to 1 the current solution will be saved after each iteration.

LowestALevel [Float] (optional) If fast reconstruction is used this the matrix elements are truncated to this level relative to the sampling distance of x , $\Delta x =$. If not set the value is 0.

ConstrainMin [Float] (optional) After each iteration the solution will forced up to this limit.

ConstrainMax [Float] (optional) After each iteration the solution will forced down to this limit.

For both limit it is assumed that negative limits imply that the feature is not used.

Alpha [Float] (optional) For ART the initial update weight (if not specified then it is set to 1).

Beta [Float] (optional) For ART the multiplicative change to the weight factor, which should be less than one (if not specified then it is set to 1).

Regularization [Float] If set different from zero and using fast reconstruction, then the program will append rows to the system matrix with a simple Laplace operator. The parameter is used to control the degree of regularization.

KernelFileName [String] If using fast reconstruction the system matrix will be saved and restored with the extension <sif>. If a system matrix on the disk is incompatible with the sampling parameters, a new will be generated.

SaveMatlab [Integer] If the parameter is set to 1, then a sparse matrix is saved compatible with Matlab. The filename used will be KernelFileName.sia, where KernelFileName also should be specified.

ThetaSamples [Integer] Number of angular samples T in the sinogram, only needed if the sinogram file does not contain sampling information.

ThetaMin [Float] Start of the angular sampling (should be set to zero), only needed if the sinogram file does not contain sampling information.

DeltaTheta [Float] Angular sampling distance (should be set to π/\ThetaSamples), only needed if the sinogram file does not contain sampling information.

RhoSamples [Integer] Number of samples in the sinogram R in the ρ -direction, only needed if the sinogram file does not contain sampling information.

DeltaRho [Integer] Sampling distance in ρ , i.e., $\Delta\rho$, only needed if the sinogram file does not contain sampling information.

RhoMin [Float] Start of sampling positions in ρ (should be set to $-\Delta\rho\frac{R-1}{2}$).

Xmin [Float] The minimum x-position of the reconstructed image. Not needed if a fif-file is provided as OrgFile.

Ymin [Float] The minimum y-position of the reconstructed image. Not needed if a fif-file is provided as OrgFile.

DeltaX [Float] Sampling distance on the x-axis. Not needed if a fif-file is provided as OrgFile.

DeltaY [Float] Sampling distance on the y-axis. Not needed if a fif-file is provided as OrgFile.

XSamples [Integer] Number of samples on the x-axis. Not needed if a fif-file is provided as OrgFile.

YSamples [Integer] Number of samples on the y-axis. Not needed if a fif-file is provided as OrgFile.

DebugLevel [String] This parameter controls the level of output. The parameter is mixed and overrules with the one used in the Print-statements (an expanded printf defined in the program).

Normal Standard level of output to screen and log file.

Debug Almost all output is logged to screen and log file.

NoScreen Screen output is disabled and log file level is ordinary.

NoLog Log file output is disabled and screen level is ordinary.

HardCore No information at all.

An example of a valid ini-file for “it” is shown below. The ini-file is used to reconstruct a sinogram **smallmanr.fif** into **smallman.EM.fif** with the same sampling parameters as contained in the file **smallmana.fif**. Here fast ART is used with a discrete Radon transform (linear interpolation) system matrix saved in **syssmall.sif**. Constraints are also used.

```
InFileName=smallmanr.fif
OutFileName=smallman.EM.fif
RefFileName=smallmana.fif
KernelFileName=syssmall.sif
Algorithm=ART
UseFast=1
RadonKernel=RL
Iterations=2
ConstrainMin=0
ConstrainMax=10
DebugLevel=Normal
```

Appendix D

The Three Dimensional Radon Transformation

The general form of the Radon transformation in N dimensions [14, 82] is given by

$$\check{g}(\rho, \boldsymbol{\xi}) = \int_{-\infty}^{\infty} g(\mathbf{r}) \delta(\rho - \boldsymbol{\xi} \cdot \mathbf{r}) d\mathbf{r} \quad (\text{D.1})$$

where \mathbf{r} and $\boldsymbol{\xi} \in \mathbb{R}^N$ but $\boldsymbol{\xi}$ has only N-1 degrees of freedom, e.g., done by normalizing $|\boldsymbol{\xi}| = 1$. A common approach is to express the vector $\boldsymbol{\xi}$ in hyper-spherical coordinates [14]. In two dimensions $\boldsymbol{\xi} = (\cos \theta, \sin \theta)^T$ and in three dimensions the vector can be chosen to

$$\boldsymbol{\xi} = \begin{pmatrix} \cos \phi \sin \theta \\ \sin \phi \sin \theta \\ \cos \theta \end{pmatrix} \quad (\text{D.2})$$

The fundamental body of the Radon transform is in two dimensions a line, cf. Eq. 2.4, and in three dimensions a plane described by the kernel of the Radon transform

$$\boldsymbol{\xi} \cdot \mathbf{r} = x \cos \phi \sin \theta + y \sin \phi \sin \theta + z \cos \theta \Rightarrow \quad (\text{D.3})$$

$$\check{g}(\rho, \theta, \phi) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(\mathbf{r}) \delta(\rho - x \cos \phi \sin \theta - y \sin \phi \sin \theta - z \cos \theta) dx dy dz \quad (\text{D.4})$$

Here it can be noted that the three dimensional Radon transform will transform a three dimensional signal $g(\mathbf{r})$ into a three dimensional parameter domain $\check{g}(\rho, \theta, \phi)$.

The three dimensional Radon transform can be viewed as a convolution with a filter

$$\delta(\boldsymbol{\xi}) \quad (\text{D.5})$$

where the $\boldsymbol{\xi}$ depends on the angles. Eq. D.5 shows that the Radon transform picks out a plane perpendicular to $\boldsymbol{\xi}$.

Like it was done for the two dimensional Radon transform the 3D Radon transform can also be written without the delta function, and here the integral is two dimensional. For sake of simplicity two other vectors orthogonal vectors are introduced, where $\boldsymbol{\xi}$, $\boldsymbol{\kappa}$, and $\boldsymbol{\gamma}$ are form a orthogonal basis.

$$\boldsymbol{\kappa} = \begin{pmatrix} -\sin \phi \\ \cos \phi \\ 0 \end{pmatrix} \quad \boldsymbol{\gamma} = \begin{pmatrix} -\cos \phi \cos \theta \\ -\sin \phi \cos \theta \\ \sin \theta \end{pmatrix} \quad (\text{D.6})$$

$$\check{g}(\rho, \theta, \phi) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(r\boldsymbol{\kappa} + p\boldsymbol{\gamma} + \rho\boldsymbol{\xi}) dr dp \quad (\text{D.7})$$

In the following sections methods for inversion of the 3D Radon transform are derived.

D.1 The Three Dimensional Fourier Slice Theorem

The Fourier Slice Theorem can also be generalized to 3D. The inversion formula is based on Fourier transformation of Eq. D.1.

$$\check{G}(\nu, \boldsymbol{\xi}) = \int_{-\infty}^{\infty} \check{g}(\rho, \boldsymbol{\xi}) e^{-j2\pi\rho\nu} d\rho \quad (\text{D.8})$$

$$= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} g(\mathbf{r}) \delta(\rho - \boldsymbol{\xi} \cdot \mathbf{r}) d\mathbf{r} \right) e^{-j2\pi\rho\nu} d\rho \quad (\text{D.9})$$

$$= \int_{-\infty}^{\infty} g(\mathbf{r}) e^{-j2\pi\nu\boldsymbol{\xi} \cdot \mathbf{r}} d\mathbf{r} \quad (\text{D.10})$$

Defining the frequency parameters

$$\nu\boldsymbol{\xi} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \quad (\text{D.11})$$

then the one dimensional Fourier transform of \check{g} can be recognized to the three dimensional Fourier transform of g .

$$\check{G}(\nu, \boldsymbol{\xi}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y, z) e^{-j2\pi(xu+yv+zw)} dx dy dz \Leftrightarrow \quad (\text{D.12})$$

$$g(\mathbf{r}) = g(x, y, z) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \check{G}(\nu, \boldsymbol{\xi}) e^{j2\pi(xu+yv+zw)} du dv dw \quad (\text{D.13})$$

Note that the result is very similar to the two dimensional result and here the transformation is from a three dimensional domain into another three dimensional domain.

D.2 Filtered Backprojection in 3D

Filtered Backprojection can also be generalized into a 3D version. The inversion scheme is derived by expressing Eq. D.13 in spherical coordinates.

$$g(x, y, z) = g(\mathbf{r}) = \int_{\nu=0}^{\infty} \int_{\theta=0}^{\pi} \int_{\phi=0}^{2\pi} \nu^2 \sin \theta \check{G}(\nu, \boldsymbol{\xi}) e^{j2\pi\nu\boldsymbol{\xi} \cdot \mathbf{r}} d\phi d\theta d\nu \quad (\text{D.14})$$

Now an inherent symmetry of spherical coordinates are used

$$\check{G}(\nu, \boldsymbol{\xi}) = \check{G}(\nu, \theta, \phi) = \check{G}(-\nu, \pi - \theta, \phi - \pi) \quad (\text{D.15})$$

which is inserted in Eq. D.14, and after a simple splitting of the ϕ integral, it is found that

$$g(\mathbf{r}) = \int_{\nu=-\infty}^{\infty} \int_{\theta=0}^{\pi} \int_{\phi=0}^{\pi} \nu^2 \sin \theta \check{G}(\nu, \boldsymbol{\xi}) e^{j2\pi\nu\boldsymbol{\xi} \cdot \mathbf{r}} d\phi d\theta d\nu \quad (\text{D.16})$$

Now Eq. D.8 is inserted into Eq. D.16.

$$g(\mathbf{r}) = \int_{\nu=-\infty}^{\infty} \int_{\theta=0}^{\pi} \int_{\phi=0}^{\pi} \nu^2 \sin \theta \left(\int_{\rho=-\infty}^{\infty} \check{g}(\rho, \theta, \phi) e^{-j2\pi\nu\rho} d\rho \right) e^{j2\pi\nu\boldsymbol{\xi} \cdot \mathbf{r}} d\phi d\theta d\nu \quad (\text{D.17})$$

This equation can for convenience be written in two parts

$$\bar{g}(\rho, \theta, \phi) = \int_{\nu=-\infty}^{\infty} \nu^2 \left(\int_{\rho=-\infty}^{\infty} \bar{g}(\tilde{\rho}, \theta, \phi) e^{-j2\pi\nu\tilde{\rho}} d\tilde{\rho} \right) e^{j2\pi\nu\rho} d\nu \quad (\text{D.18})$$

$$g(\mathbf{r}) = \int_{\theta=0}^{\pi} \int_{\phi=0}^{\pi} \sin \theta \bar{g}(\boldsymbol{\xi} \cdot \mathbf{r}, \theta, \phi) d\phi d\theta \quad (\text{D.19})$$

$$= \int_{\rho=-\infty}^{\infty} \int_{\theta=0}^{\pi} \int_{\phi=0}^{\pi} \sin \theta g^{\dagger}(\rho, \theta, \phi) \delta(\rho - \boldsymbol{\xi} \cdot \mathbf{r}) d\phi d\theta d\rho \quad (\text{D.20})$$

Note the similarity between the 2D Filtered Backprojection inversion scheme and the 3D version as shown in Eqs. D.18 and D.20. Naturally Eq. D.20 is named the backprojection operator. Note also that the filtering part of 3D Filtered Backprojection requires a quadratic filter ν^2 .

These inversion schemes are generalizations of the ones shown for the two dimensional Radon transform, and implementation can be done by generalizing the techniques described in Chapter 8. It should be mentioned that [62] has a thorough description of the linogram method extended to three dimensions.

D.3 Connection between the 3D plane integrals and 3D line integrals

In this section the connection between the 3D line integrals (see Chapter 10) and the 3D plane integrals is established. This can be used to exploit the inverse 3D plane Radon transform, e.g., Filtered Backprojection shown in Eq. D.20 and D.18 to reconstruct a volume from line integrals and supplement the methods shown in Chapter 10.

Assume a plane with parameters ρ_p, θ_p, ϕ_p (index p for plane). From θ_p and ϕ_p an orthonormal basis can be found

$$\boldsymbol{\xi} = \begin{pmatrix} \sin \theta_p \cos \phi_p \\ \sin \theta_p \sin \phi_p \\ \cos \theta_p \end{pmatrix} \quad \boldsymbol{\zeta} = \begin{pmatrix} -\sin \phi_p \\ \cos \phi_p \\ 0 \end{pmatrix} \quad \boldsymbol{\eta} = \begin{pmatrix} \cos \theta_p \cos \phi_p \\ \cos \theta_p \sin \phi_p \\ -\sin \theta_p \end{pmatrix} \quad (\text{D.21})$$

Now all lines in the plane (normal to $\boldsymbol{\xi}$) is be found. The direction vector $\boldsymbol{\tau}$ of the lines can in general be written as

$$\boldsymbol{\tau} = \boldsymbol{\eta} \cos \omega + \boldsymbol{\zeta} \sin \omega \quad (\text{D.22})$$

The vectors are illustrated in Fig. D.1.

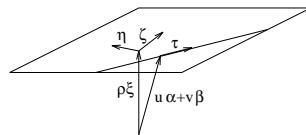


Figure D.1 The plane with normal vector $\boldsymbol{\xi}$. The line $\mathbf{r} = s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta}$ lies in the plane.

Another useful vector is $\boldsymbol{\chi}$, defined as the vector product between $\boldsymbol{\tau}$ and $\boldsymbol{\xi}$. Using that $\boldsymbol{\xi}, \boldsymbol{\eta}$ and $\boldsymbol{\zeta}$ are orthonormal vectors imply that

$$\boldsymbol{\chi} = \boldsymbol{\eta} \sin \omega - \boldsymbol{\zeta} \cos \omega \quad (\text{D.23})$$

hence the lines can be described by

$$\mathbf{r} = s\boldsymbol{\tau} + \mathbf{r}_0 \quad (\text{D.24})$$

where s is the free parameter and \mathbf{r}_0 is the base point of the lines, resolved as shown in Section 10.1. Using the symbols defined there gives

$$\mathbf{r}_0 = u\boldsymbol{\alpha} + v\boldsymbol{\beta} \quad (\text{D.25})$$

Using the basis vectors shown in Fig. D.1 the plane integral $\check{g}(\rho, \boldsymbol{\xi})$ can be expressed as

$$\check{g}_\omega(\rho, \boldsymbol{\xi}) = \int_{r=-\infty}^{\infty} \int_{t=-\infty}^{\infty} g(\rho \boldsymbol{\xi} + r \boldsymbol{\chi} + t \boldsymbol{\tau}) dt dr \quad (\text{D.26})$$

$$= \int_{r=-\infty}^{\infty} \check{g}(\theta_l, \phi_l, u, v) dr \quad (\text{D.27})$$

where θ_l and ϕ_l (The index l stands for line) are the angles corresponding to $\boldsymbol{\tau}$ found from Eq. 10.2. The parameters u and v are found from

$$\rho \boldsymbol{\xi} + r \boldsymbol{\chi} = u \boldsymbol{\alpha} + v \boldsymbol{\beta} \quad (\text{D.28})$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are found from the angles θ_l and ϕ_l .

The result in Eq. D.27 depends on the angle ω . The final estimate of the plane integral may be set to a weighted integral

$$\check{g}(\rho, \boldsymbol{\xi}) = \frac{1}{\pi} \int_{\omega=0}^{\pi} q(\omega) \check{g}_\omega(\rho, \boldsymbol{\xi}) d\omega \quad (\text{D.29})$$

$$= \frac{1}{\pi} \int_{\omega=0}^{\pi} q(\omega) \int_{r=-\infty}^{\infty} \check{g}(\theta_l, \phi_l, \boldsymbol{\alpha} \cdot (\rho \boldsymbol{\xi} + r \boldsymbol{\chi}), \boldsymbol{\beta} \cdot (\rho \boldsymbol{\xi} + r \boldsymbol{\chi})) dr d\omega \quad (\text{D.30})$$

where $q(\omega)$ is a weight designed after the available data. In general it must fulfill

$$\int_{\omega=0}^{\pi} q(\omega) d\omega = \pi \quad (\text{D.31})$$

Appendix E

Properties of the 3D line Radon Transform

From the 3D line Radon transform shown in Eq. E.1 several properties are derived.

E.1 Basic Properties of the 3D line Radon Transform

For sake of clarify the 3D line Radon transform given in Eq. 10.5, is repeated here.

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta}) ds \quad (\text{E.1})$$

E.1.1 Linearity

The first crucial property is linearity of the transform.

$$g(\mathbf{r}) = \sum_i \kappa_i g_i(\mathbf{r}) \Rightarrow \check{g}(\theta, \phi, u, v) = \sum_i \kappa_i \check{g}_i(\theta, \phi, u, v) \quad (\text{E.2})$$

where the set of κ_i are arbitrary constants.

E.1.2 Translation

It is assumed that

$$h(\mathbf{r}) = g(\mathbf{r} - \mathbf{r}_0) \Rightarrow \quad (\text{E.3})$$

$$\check{h}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta} - \mathbf{r}_0) ds \quad (\text{E.4})$$

Here the translation \mathbf{r}_0 are resolved after the basis vectors, i.e.,

$$\mathbf{r}_0 = s_0\boldsymbol{\tau} + u_0\boldsymbol{\alpha} + v_0\boldsymbol{\beta} \quad (\text{E.5})$$

This can be done in an unambiguous way

$$s_0 = \mathbf{r}_0 \cdot \boldsymbol{\tau} \quad u_0 = \mathbf{r}_0 \cdot \boldsymbol{\alpha} \quad v_0 = \mathbf{r}_0 \cdot \boldsymbol{\beta} \quad (\text{E.6})$$

This is inserted in Eq. E.4.

$$\check{h}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g((s - s_0)\boldsymbol{\tau} + (u - u_0)\boldsymbol{\alpha} + (v - v_0)\boldsymbol{\beta}) \, ds \quad (\text{E.7})$$

$$= \int_{-\infty}^{\infty} g(\tilde{s}\boldsymbol{\tau} + (u - u_0)\boldsymbol{\alpha} + (v - v_0)\boldsymbol{\beta}) \, d\tilde{s} \Rightarrow \quad (\text{E.8})$$

$$\check{h}(\theta, \phi, u, v) = \check{g}(\theta, \phi, u - \mathbf{r}_0 \cdot \boldsymbol{\alpha}, v - \mathbf{r}_0 \cdot \boldsymbol{\beta}) \quad (\text{E.9})$$

Note that the angular parameters θ and ϕ are not changed.

E.1.3 Rotation and Scaling

Rotation and scaling can be controlled by means of a general transformation matrix \mathbf{A}

$$h(\mathbf{r}) = g(\mathbf{A}\mathbf{r}) \quad (\text{E.10})$$

The diagonal elements of the $3 * 3$ matrix control the individual scaling and the off-diagonal elements are controlling the rotation. The corresponding Radon transform is given by

$$\check{h}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(\mathbf{A}(s\boldsymbol{\tau} + u\boldsymbol{\alpha} + v\boldsymbol{\beta})) \, ds \quad (\text{E.11})$$

$$= \int_{-\infty}^{\infty} g(s\mathbf{A}\boldsymbol{\tau} + u\mathbf{A}\boldsymbol{\alpha} + v\mathbf{A}\boldsymbol{\beta}) \, ds \quad (\text{E.12})$$

This will in general alter the directional vector $\boldsymbol{\tau}$ of the line, i.e., a new direction vector is needed

$$\mathbf{A}\boldsymbol{\tau} = \gamma\tilde{\boldsymbol{\tau}} \quad |\tilde{\boldsymbol{\tau}}| = 1 \quad \gamma \geq 0 \quad (\text{E.13})$$

The vector $\tilde{\boldsymbol{\tau}}$ defines cf. Eq. 10.2 two corresponding angles $\tilde{\theta}$ and $\tilde{\phi}$, which again defines the two remaining vectors $\tilde{\boldsymbol{\alpha}}$ and $\tilde{\boldsymbol{\beta}}$, cf. Eq. 10.4. Now Eq. E.13 is inserted into Eq. E.12.

$$\check{h}(\theta, \phi, u, v) = \frac{1}{\gamma} \int_{-\infty}^{\infty} g(s\tilde{\boldsymbol{\tau}} + \mathbf{A}(u\boldsymbol{\alpha} + v\boldsymbol{\beta})) \, ds \quad (\text{E.14})$$

The next step is to resolve the line offset parameters after the three new tilde vectors, which again can be done unambiguously.

$$\mathbf{A}(u\boldsymbol{\alpha} + v\boldsymbol{\beta}) = s_0\tilde{\boldsymbol{\tau}} + u_0\tilde{\boldsymbol{\alpha}} + v_0\tilde{\boldsymbol{\beta}} \quad (\text{E.15})$$

This implies that the 3D line Radon transform is given by

$$\check{h}(\theta, \phi, u, v) = \frac{1}{\gamma} \int_{-\infty}^{\infty} g((s + s_0)\tilde{\boldsymbol{\tau}} + u_0\tilde{\boldsymbol{\alpha}} + v_0\tilde{\boldsymbol{\beta}}) \, ds \quad (\text{E.16})$$

$$= \frac{1}{\gamma} \int_{-\infty}^{\infty} g(\tilde{s}\tilde{\boldsymbol{\tau}} + u_0\tilde{\boldsymbol{\alpha}} + v_0\tilde{\boldsymbol{\beta}}) \, d\tilde{s} \quad (\text{E.17})$$

$$= \frac{1}{\gamma} \check{g}(\tilde{\theta}, \tilde{\phi}, u_0, v_0) \quad (\text{E.18})$$

where the four tilde parameters are given by

$$\tilde{\boldsymbol{\tau}} = \begin{pmatrix} \cos \tilde{\phi} \cos \tilde{\theta} \\ \cos \tilde{\phi} \sin \tilde{\theta} \\ \sin \tilde{\phi} \end{pmatrix} = \frac{1}{|\mathbf{A}\boldsymbol{\tau}|} \mathbf{A}\boldsymbol{\tau}, \quad \tilde{\boldsymbol{\alpha}} = \begin{pmatrix} -\sin \tilde{\theta} \\ \cos \tilde{\theta} \\ 0 \end{pmatrix} \text{ and } \tilde{\boldsymbol{\beta}} = \begin{pmatrix} -\sin \tilde{\phi} \cos \tilde{\theta} \\ -\sin \tilde{\phi} \sin \tilde{\theta} \\ \cos \tilde{\phi} \end{pmatrix} \quad (\text{E.19})$$

$$\gamma = |\mathbf{A}\boldsymbol{\tau}|, \quad u_0 = \tilde{\boldsymbol{\alpha}} \cdot \mathbf{A} \cdot (\boldsymbol{\alpha}u + \boldsymbol{\beta}v), \quad v_0 = \tilde{\boldsymbol{\beta}} \cdot \mathbf{A} \cdot (\boldsymbol{\alpha}u + \boldsymbol{\beta}v) \quad (\text{E.20})$$

The matrix \mathbf{A} can be partitioned in three rotational matrices and one scaling matrix

$$\mathbf{A}_z \mathbf{A}_y \mathbf{A}_x \mathbf{S} \quad (\text{E.21})$$

where \mathbf{A}_x rotates in the $y - z$ coordinates, e.g.

$$\mathbf{A}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \psi_x & \sin \psi_x \\ 0 & -\sin \psi_x & \cos \psi_x \end{pmatrix} \quad (\text{E.22})$$

and \mathbf{A}_y rotates in the $x - z$ coordinates, e.g.

$$\mathbf{A}_y = \begin{pmatrix} \cos \psi_y & 0 & \sin \psi_y \\ 0 & 1 & 0 \\ -\sin \psi_y & 0 & \cos \psi_y \end{pmatrix} \quad (\text{E.23})$$

and finally \mathbf{A}_z rotates in the $x - y$ coordinates, e.g.

$$\mathbf{A}_z = \begin{pmatrix} \cos \psi_z & \sin \psi_z & 0 \\ -\sin \psi_z & \cos \psi_z & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (\text{E.24})$$

The scaling matrix \mathbf{S} is here chosen to be the last in Eq. E.21 because the interpretation of the scaling is by far easier when applied directly on the base function $g(\mathbf{r})$. The matrix can be chosen to

$$\mathbf{S} = \begin{pmatrix} \frac{1}{S_x} & 0 & 0 \\ 0 & \frac{1}{S_y} & 0 \\ 0 & 0 & \frac{1}{S_z} \end{pmatrix} \quad (\text{E.25})$$

The scaling parameters in the diagonal is chosen so they directly relate to length in the new function $h(\mathbf{r})$. The total rotation and scaling matrix defined in Eq. E.21 multiplied together becomes

$$\mathbf{A} = \begin{pmatrix} \frac{\cos \psi_x \cos \psi_y}{S_x} & \frac{\cos \psi_y \sin \psi_x}{S_y} & \frac{\sin \psi_y}{S_z} \\ -\frac{\cos \psi_z \sin \psi_x + \cos \psi_x \sin \psi_y \sin \psi_z}{S_x} & \frac{\cos \psi_x \cos \psi_z - \sin \psi_x \sin \psi_y \sin \psi_z}{S_y} & \frac{\cos \psi_y \sin \psi_z}{S_z} \\ \frac{\sin \psi_x \sin \psi_z - \cos \psi_x \cos \psi_z \sin \psi_y}{S_x} & -\frac{\cos \psi_z \sin \psi_x \sin \psi_y + \cos \psi_x \sin \psi_z}{S_y} & \frac{\cos \psi_y \cos \psi_z}{S_z} \end{pmatrix} \quad (\text{E.26})$$

E.2 Analytical Radon Transformation of Primitives

Along with the basic rules of the 3D line Radon transform it is very useful to have a set of base 3D functions and their corresponding Radon transform. In the following the Radon transform of the unit ball and a Gaussian bell are derived. This can be used to produce more complex functions written as a weighted sum of shifted, rotated and scaled primitives.

E.2.1 The ball

The first primitive is the unit ball, i.e.,

$$g_{\text{ball}}(x, y, z) = \begin{cases} 1 & \text{for } x^2 + y^2 + z^2 < 1 \\ 0 & \text{for } x^2 + y^2 + z^2 \geq 1 \end{cases} \quad (\text{E.27})$$

Due to the rotational symmetry of the primitive the corresponding Radon transform cannot depend on the two angular parameters θ and ϕ , thus the Radon transform is derived in a rotated coordinate system with τ lying along the x-axis and the vector $r_0 = u\alpha + v\beta$ along the y-axis. As illustrated in Fig. E.1 is the Radon transform merely the distance between the two intersection points between the line and the ball.

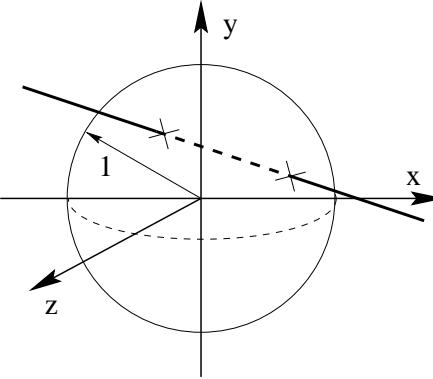


Figure E.1 The unit ball with radius 1 and the Radon transform is the length between the two intersection points between the line and the surface of the ball.

The length of r_0 is $\sqrt{u^2 + v^2}$, which implies that the Radon transform can be calculated as

$$\check{g}_{\text{ball}}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g_{\text{ball}}(s, \sqrt{u^2 + v^2}, 0) \, ds \Rightarrow \quad (\text{E.28})$$

$$\check{g}_{\text{ball}}(\theta, \phi, u, v) = \begin{cases} 2\sqrt{1 - u^2 - v^2} & \text{for } u^2 + v^2 < 1 \\ 0 & \text{for } u^2 + v^2 \geq 1 \end{cases} \quad (\text{E.29})$$

E.2.2 The Gaussian bell

The Gaussian bell is another primitive, where the Radon transform can be found analytically.

$$g_{\text{gauss}}(x, y, z) = \exp(-x^2 - y^2 - z^2) \quad (\text{E.30})$$

Again the function has rotational symmetry, hence the Radon transform does not depend on the angular parameters θ and ϕ , and the Radon transform can, e.g., be derived as

$$\check{g}_{\text{gauss}}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} e^{-s^2 - u^2 - v^2} \, ds \Rightarrow \check{g}_{\text{gauss}}(\theta, \phi, u, v) = \sqrt{\pi} e^{-u^2 - v^2} \quad (\text{E.31})$$

Appendix F

Usage of the 3D Reconstruction Tools

A software packages has been made for 3D reconstruction of sinograms (line integrals), as shown in Chapter 10. The package are provided for free, but protected by the GNU General Public License. The programs described in this appendix has been developed on a a Linux system and implemented in C. The program should compile directly on any GNU-C Unix system. Furthermore will the program compile on a SGI Onyx, and parallel options, `#pragma XXX`, have been added to take advantage of the parallel possibilities of Iris Power C. The package is available from [10].

In Section F.1 is the usage of the reconstruction program “Recon3D” described. As mentioned in Section 10.7 the program is intended for testing of different 3D reconstruction algorithms, and no Kinahan & Rogers reprojection step is currently included.

In Section F.2 is the program “3D_RadonAna” described, which can generate a volume and the corresponding four-dimensional sinogram from a set of scaled, translated, and rotated primitives.

F.1 The 3D Reconstruction Program “Recon3D”

The program “Recon3D” uses the following parameters

DebugLevel [String] This parameter controls the level of output. The parameter is mixed and overrules with the one used in the Print-statements (extended printf defined in the program).

Normal Standard level of output to screen and log file.

Debug Almost all output is logged to screen and log file.

NoScreen Screen output is disabled and log file level is ordinary.

NoLog Log file output is disabled and screen level is ordinary.

HardCore No information at all.

WorkDir [String] Directory where the data-files are stored. The style `../signals` is valid.

Sinogram [String] Name of the file containing the 4D sinogram. The file must have extension `.f4f` (binary format) or `.a4f` (ASCII format).

OrgFile [String] (optional) Name of a volume specifying the sampling parameters of the reconstructed volume. If given, then error measures will be computed with respect to this volume.

Volume [String] Name of the volume to be reconstructed. The file must have extension `.f3f` (binary format) or `.a3f` (ASCII format).

Function [String] Parameter specifying the action to be used.

- FB** Direct reconstruction using Filtered Backprojection. Result in the **Volume**.
- FAB** Direct reconstruction using Filtering after Backprojection. Result in the **Volume**.
- CS** Direct reconstruction using Fourier Slice Theorem (Experimental). Result in the **Volume**.
- Forward** Radon transform of **Volume** into the **Sinogram**.
- MART** Iterative reconstruction using MART. Result in the **Volume**.
- ART** Iterative reconstruction using ART. Result in the **Volume**.
- EM** Iterative reconstruction using EM. Result in the **Volume**.

Iterations [Integer] For EM the number of iterations before the iteration ends. For ART and MART the number of full iterations, i.e., the number of actual ART iterations is **Iterations** times the number of rows in the system matrix.

Iterative_Par_1 [Float] In ART and MART the initial weight parameter λ .

Iterative_Par_2 [Float] In ART and MART the final weight parameter λ .

Select_Par_1 [Integer] In Fourier Slice Reconstruction a value of 0 will choose a nearest neighbour interpolation in the spectrum, and a value of 1 implies that tri-linear interpolation is used.

Xmin [Float] The minimum value of x in the volume. Only needed if **OrgFile** is not specified.

Ymin [Float] The minimum value of y in the volume. Only needed if **OrgFile** is not specified.

Zmin [Float] The minimum value of z in the volume. Only needed if **OrgFile** is not specified.

DeltaX [Float] The sampling interval of x in the volume. Only needed if **OrgFile** is not specified.

DeltaY [Float] The sampling interval of y in the volume. Only needed if **OrgFile** is not specified.

DeltaZ [Float] The sampling interval of z in the volume. Only needed if **OrgFile** is not specified.

XSamples [Integer] The number of samples of x in the volume. Only needed if **OrgFile** is not specified.

YSamples [Integer] The number of samples of y in the volume. Only needed if **OrgFile** is not specified.

ZSamples [Integer] The number of samples of z in the volume. Only needed if **OrgFile** is not specified.

An example of a valid ini-file for “Recon3D” is shown below. The ini-file is used to reconstruct a sinogram `../signals/Mickey_Sino.f4f` into `Mickey_Out.f3f` with the same sampling parameters as contained in the file `Mickey_Vol.f3f`. Here Filtered Backprojection is used.

```
WorkDir=../signals
Sinogram=Mickey_Sino.f4f
Volume=Mickey_Out.f3f
OrgFile=Mickey_Vol.f3f
Function=FB
DebugLevel=Normal
Iterations=30
```

F.2 The Analytical Sinogram Program “3D_RadonAna”

The program “3D_RadonAna” uses the following parameters

DebugLevel [String] This parameter controls the level of output. The parameter is mixed and overrules with the one used in the Print-statements (extended printf defined in the program).

Normal Standard level of output to screen and log file.

Debug Almost all output is logged to screen and log file.

NoScreen Screen output is disabled and log file level is ordinary.

NoLog Log file output is disabled and screen level is ordinary.

HardCore No information at all.

OutFileName [String] First part of the name to output. Two files can be written. The volume in **OutFileName_Vol.f3f** and the sinogram in **OutFileName_Sino.f4f**. Here the style `./signals/MickeyBig` can be used.

Xmin [Float] The minimum value of x in the volume.

Ymin [Float] (optional) The minimum value of y in the volume. If not specified **Xmin** will be used.

Zmin [Float] (optional) The minimum value of z in the volume. If not specified **Xmin** will be used.

DeltaX [Float] The sampling interval of x in the volume.

DeltaY [Float] (optional) The sampling interval of x in the volume. If not specified the valued of **DeltaX** will be.

DeltaZ [Float] (optional) The sampling interval of z in the volume. If not specified the valued of **DeltaX** will be.

XSamples [Integer] The number of samples of x in the volume.

YSamples [Integer] (optional) The number of samples of y in the volume. If not specified **XSamples** will be used.

ZSamples [Integer] (optional) The number of samples of z in the volume. If not specified **XSamples** will be used.

USamples [Integer] The number of samples of u in the sinogram.

VSamples [Integer] The number of samples of v in the sinogram. If not specified **USamples** will be.

DeltaU [Float] The sampling interval of u in the sinogram.

DeltaV [Float] The sampling interval of v in the sinogram. If not specified **DeltaU** will be.

PhiSamples [Integer] The number of samples of ϕ in the sinogram. This number will be distributed from $\phi_{min} = -\psi = -\text{PhiLimit}$ to $-\phi_{min} = \psi = \text{PhiLimit}$.

PhiLimit [Float] The axial acceptance angle ψ (measured in degrees).

GenerateVolume If set to 0 then the volume will not be generated, and 1 then it will be generated.

GenerateSinogram If set to 0 then the sinogram will not be generated, and 1 then it will be generated.

NumberOfShapes Number of primitives used.

The ordering of the parameters shown above are arbitrary. After these parameters a number of lines must follow specifying a scaling, rotation, and shifting parameters of each primitive. The line uses the following parameters in this order

Shape*i* [String] where *i* is 0 to **NumberOfShapes**-1.

Type [Integer] The type of primitive, where

- 1 A ball centered around (0,0) with radius 1 and uniform signal 1 in the ball.
- 2 A 3D Gaussian bell $\exp(-x^2 - y^2 - z^2)$.

offx [Float] Shift of of the primitive in the direction of the *x*-axis.

offy [Float] Shift of of the primitive in the direction of the *y*-axis.

offz [Float] Shift of of the primitive in the direction of the *z*-axis.

rotx [Float] Rotation angle of the primitive around the *x*-axis.

roty [Float] Rotation angle of the primitive around the *y*-axis.

rotz [Float] Rotation angle of the primitive around the *z*-axis.

scalx [Float] Scaling of the primitive in direction of the *x*-axis.

scaly [Float] Scaling of the primitive in direction of the *y*-axis.

scalz [Float] Scaling of the primitive in direction of the *z*-axis.

power [Float] Scaling of the value of the primitive.

The program is executed with the initialization file or .INI-file as sole argument. A typical .INI-file could be the one used to generate the Big Mickey. Note that comments can be used, such as the line above Shape0.

```
OutFileName=../signals/MickeyBig
XSamples=151
DeltaX=0.7
USamples=71
ThetaSamples=2
PhiSamples=2
PhiLimit=90
DeltaU=0.5
NumberOfShapes=17
GenerateVolume=1
GenerateSinogram=1
DebugLevel=Normal
```

	Type	offx	offy	offz	rotx	roty	rotz	scalx	scaly	scalz	power
Shape0=	1	0.0	30.0	0.0	0	0	0	10	10	10	1
Shape1=	1	-10.7	40.5	0.0	0	0	0	5	5	5	1
Shape2=	1	10.7	40.5	0.0	0	0	0	5	5	5	1
Shape3=	1	0.0	30.0	13.0	0	0	0	3	3	3	1
Shape4=	1	0.0	24.0	9.5	0	0	0	6	2	1	1
Shape5=	1	4.5	34.5	10.0	0	0	0	2	2	2	1
Shape6=	1	-4.5	34.5	10.0	0	0	0	2	2	2	1
Shape7=	1	0.0	0.00	0.0	0	0	0	10	20	10	1
Shape8=	1	-19	20.0	0.0	0	0	30	3	15	3	1
Shape9=	1	19	20.0	0.0	0	0	-30	3	15	3	1
Shape10=	1	-18	-25.0	0.0	0	0	-30	3	15	3	1
Shape11=	1	18	-25.0	0.0	0	0	30	3	15	3	1
Shape12=	1	0.0	0.0	0.0	0	0	0	8	18	8	-1
Shape13=	1	1.0	3.0	-2.0	0	0	0	2	2	2	0.5
Shape14=	1	-2.0	2.0	2.0	0	0	0	2	2	2	0.2
Shape15=	1	0.0	30.0	0.0	0	0	0	9	9	9	-1
Shape16=	1	0.0	30.0	2.0	0	0	0	1	2	1	0.7

Part IV

Papers

Appendix G

Fast Radon Transform for Detection of Seismic Reflections

This appendix contains the paper *Fast Radon Transform for Detection of Seismic Reflections*, by Peter A. Toft and Kim V. Hansen [2]. The work has been carried out as a joint venture project between Ødegaard & Danneskiold-Samsøe and Department of Mathematical Modelling (before january 1. 96 Electronics Institute).

The results have been presented at the *Interdisciplinary Inversion Summer School 94* which took place in Mønsted, Denmark, and later this paper was presented at the *EUSIPCO 94* in Edinburgh, Scotland.

Fast Radon Transform for Detection of Seismic Reflections

Peter A. TOFT and Kim V. HANSEN[†]

Electronics Institute, Technical University of Denmark, DK-2800 Lyngby, Denmark,

Tel/Fax +45 45 93 42 07/42 88 01 17, E-Mail: ptoft@ei.dtu.dk

†Ødegaard & Danneskiold-Samsøe ApS, Kroghsgade 1, DK-2100 København Ø, Denmark,

Tel/Fax +45 35 26 60 11/35 26 50 18, E-Mail: kvh@oedan.dk

Abstract. A new method for fast detection of seismic reflections is proposed. Generally the method can be viewed as a curve matching method for images with a specific structure. In this paper the method is applied to seismic signals assembled to constitute an image in which the investigated reflections produce hyperbolic curves. The idea of the proposed method is to estimate the reflection curves by combining the multipulse technique and the FCE-algorithm [1]. The FCE-algorithm is an algorithm used for fast curve estimation within binary images. The foundation of the FCE-algorithm is the composition of a pre-condition map which significantly reduces the computational cost compared to the traditional generalized Radon transform. The proposed method is successfully applied for detection of hyperbolic reflection curves within CMP-gathers.

1 Introduction

The seismic industry has developed a method of sampling seismic signals that gives multiple coverage. The recording geometry of seismic acquisition results in signals having the same mid point between source and receiver [2]. The set of signals corresponding to the same mid point is called a CMP-gather (common mid point gather).

Without any assumptions concerning the seismic signal processing, two facts of the CMP-gather are noticed. Almost the same short signal sequence (the wavelet) is observed several times within each trace and the same short signal sequence is seen along curves within several traces. These two facts form the basis of the reflection detection. The reflections represent information concerning the geological structure of the subsurface. If the subsurface is assumed to be horizontally layered it can be shown [2] that the shape of the reflection curves are hyperbolas, described by the normal moveout equation. Currently, reflections are identified from CMP-gathers by using velocity analysis succeeded by a manual picking of primary reflections. This procedure is inconvenient due to the computational burden and the required interpretation performed by an expert. These facts motivate the development of an automatic reflection identification procedure.

The key idea of the proposed method for detection of reflection curves is to combine the multipulse technique with the FCE-algorithm [1]. The multipulse technique is used to identify the wavelet in each trace, and the FCE-algorithm is used to coordinate the es-

timation of the reflection sequence.

The proposed method works on CMP-gathers, shot-gathers, and other types of gathers.

2 The Multipulse Technique

Recently, the multipulse excited speech coding technique, which was originally proposed by Atal *et al.* [3], has been suggested for seismic deconvolution by Cookey *et al.* [4]. Basically, the multipulse model for speech assumes the speech signal to be a result of several impulses modulated by the shape of the vocal tract filter. Within seismic deconvolution models, the input signal is often modeled as an all-pole impulse and the layered earth reflectivity model is considered as an impulse train.

Fundamentally, the reflections within each trace are estimated by minimizing the mean square error between the recorded and the modeled signal. The method is characterized by successive estimation of a pre-defined number of reflections. Thus, the multipulse technique can be used to map seismic signals in a CMP-gather into a binary image, where the image value is set to one in case of a reflection and otherwise is set to zero.

3 Fast Hyperbola Estimation

A recurring problem in computer image processing is the identification of curves with specific shapes. In digital image processing a special variant of the generalized Radon transform [5], called the Hough transform

[6, 7], is used for detection of curves in images. The Hough transform detects specific values of parameters, which characterize these curves. Spatially extended curves are transformed to produce spatially compact features in a space of possible parameter values. Thus, the generalized Radon transform converts a difficult global detection problem in the image space into a more easily solved local peak detection problem in a parameter space.

The foundation of the FCE-algorithm is the composition of a pre-condition map which reduces the computational costs of the traditional generalized Radon transform. The pre-condition map is composed of irregular regions in the parameter domain which represent the interesting areas of the parameter domain, i.e., the regions of the parameter domain which contain peaks representing curves. Initially, the FCE-algorithm estimates the pre-condition map and subsequently applies a traditional generalized Radon transform within the regions specified by the pre-condition map. To generate the pre-condition map a fast mapping procedure named *image point mapping* (*IPM*) is presented. As *IPM* maps image points into the corresponding parameter values in the parameter domain, it can profit by image points with value zero.

Assuming a horizontally layered subsurface the reflection curves within a CMP-gather are described by the normal moveout equation [2]

$$t_n = \phi(\kappa_x; t_{0,z}, \sigma_s) = \sqrt{t_{0,z}^2 + (\sigma_s \kappa_x)^2} \quad (1)$$

where t_n denotes the two way travel time, $t_{0,z}$ the zero offset two way travel time, κ_x the offset, and σ_s the slowness. The subscripts denote corresponding discrete variables.

In this paper a linear sampling of the variables is chosen according to

$$\xi_i = \xi_{i,j} = \xi_{i,low} + j_i \Delta \xi_i, \quad j_i = 0, 1, \dots, J_i - 1 \quad (2)$$

where j_i is a discrete variable, $\xi_{i,low}$ denotes the lower limit, $\Delta \xi_i$ the sampling interval and J_i is the number of samples of ξ_i .

Substitution of the linear sampling into eq. (1) gives

$$\begin{aligned} n &= \phi_d(x; z, s) \\ &= \text{round}((\phi(\kappa_x; t_{0,z}, \sigma_s) - t_{low}) / \Delta t) \end{aligned} \quad (3)$$

where $\text{round}(\cdot)$ rounds to the closest integer.

Let $d(x, n)$ denote a discrete image and let $\mathcal{D}(z, s)$ denote the discrete generalized Radon transform of $d(x, n)$ defined by

$$\mathcal{D}(z, s) = \sum_{x=0}^{X-1} d(x, \phi_d(x; z, s)) \quad (4)$$

Normally, the use of rounding instead of e.g., linear interpolation will not cause problems. However, interpolation can easily be incorporated in eq. (3) and (4).

The discrete generalized Radon transform can be rewritten as

$$\mathcal{D}(z, s) = \sum_{x=0}^{X-1} \sum_{n=0}^{N-1} d(x, n) \delta(n - \phi_d(x; z, s)) \quad (5)$$

where $\delta(\cdot)$ denotes the Kronecker delta function.

Eq. (5) shows that each image point (x, n) is transformed into a parameter curve where $\phi_d(x; z, s) = n$. Thus mapping of image points with zero value is needless, as they will not contribute to $\mathcal{D}(z, s)$.

Inversion of eq. (1) and insertion of the linear sampling relations of the variables leads to

$$\begin{aligned} z &= \Omega_d(x, n; s) \\ &= \text{round}((\sqrt{t_n^2 - (\sigma_s \kappa_x)^2} - t_{0,low}) / \Delta t_0) \end{aligned} \quad (6)$$

Based on eq. (5) and (6) the estimation scheme *IPM* is defined. This can be summarized as

1. Initialize $\mathcal{D}(z, s) = 0$ for all (z, s)
2. For all image points $d(x, n)$ with a value different from zero do
 - For all possible s do

$$\mathcal{D}(\Omega_d(x, n; s), s) := \mathcal{D}(\Omega_d(x, n; s), s) + d(x, n)$$

In the hyperbolical case eq. (6) will describe an ellipse. This causes a problem for $|\Omega_d(x, n; s) - \Omega_d(x, n; s - 1)| > 1$, which results in a perforated ellipse. This perforation problem can be eliminated simply by adding $d(n, x)$ to $\mathcal{D}(z, s)$ for the parameter values skipped between two subsequent parameter vectors. Using *IPM* the ellipses from points lying on a hyperbola might not intersect in one parameter combination, but will spread out. This problem can be eliminated using a sparse sampled parameter domain. However, *GRT* requires a dense sampling of the parameter domain to ensure that all hyperbolas are detectable.

3.1 The FCE-algorithm

From the characteristics of *GRT* and *IPM* it is possible to construct a fast curve parameter estimation algorithm, the *FCE-algorithm* [1], that simultaneously estimates all curve parameters within a sparse binary image.

The FCE-algorithm can be summarized as follows:

1. Design the discrete parameter domain, i.e., choose $\xi_{i,low}$, J_i and $\Delta \xi_i$, $\xi_i \in \{t_{0,z}, \sigma_s\}$.
2. Design a reduced parameter domain for *IPM* by choosing

$$J'_i = \text{ceil} \left(\frac{J_i}{2v_{\xi_i} + 1} \right) \quad (7)$$

$$\xi'_{i,low} = \xi_{i,low} + v_{\xi_i} \Delta \xi_i \quad (8)$$

$$\Delta \xi'_i = (2v_{\xi_i} + 1) \Delta \xi_i \quad (9)$$

where $\text{ceil}(\cdot)$ rounds to the nearest upper integer. Then use IPM to estimate $\mathcal{D}_m(z', s')$ as

$$\mathcal{D}_m(z', s') = IPM\{d(x, n)\} \quad (10)$$

3. Use a threshold to remove all insignificant parameter combinations. A suitable choice could be

$$\mathcal{D}_t(z', s') = \mu(\mathcal{D}_m(z', s') - \lambda_1 X) \quad (11)$$

where X denotes the number of image lines, λ_1 a fraction defining the significance level, and $\mu(\cdot)$ the Hamilton step function.

4. Resample the parameter domain to obtain $\mathcal{D}_r(z, s)$. The resampling process can be written as

$$\mathcal{D}_r(z, s) = \mathcal{D}_t(z', s') \quad (12)$$

$$j'_i = \text{round}\left(\frac{j_i - v_{\xi_i}}{2v_{\xi_i} + 1}\right) \quad (13)$$

This gives a full size parameter domain where the interesting regions have value one while all other regions have value zero.

5. Use GRT to estimate $\mathcal{D}(z, s)$ in the regions of the parameter domain where $\mathcal{D}_r(z, s)$ is not equal to zero.

$$\mathcal{D}(z, s) = GRT\{d(x, n) | \mathcal{D}_r(z, s) \neq 0\} \quad (14)$$

6. As in step three use the threshold function with significance level λ_2 . After the thresholding all points in the parameter domain having a value different from zero represent curve parameters.

The significance levels λ_1 and λ_2 must be chosen in accordance with the properties of the image. In general λ_1 and λ_2 are not critical. A suitable choice of λ_1 and λ_2 could be $0.1 - 0.7$ and $0.4 - 0.8$, respectively. There are two reasons for a low acceptance level in step three. Firstly, IPM has a tendency to blur in the parameter domain and secondly, only regions of the parameter domain ensured not to contain image curves must be left out. A higher significance level in step six (λ_2) is justified by the required certainty for correct curve identification.

4 Numerical Example

To illustrate the performance of the presented method for determination of reflection curves within CMP-gathers a numerical example is given.

Consider a CMP-gather where the image values are represented by continuous values. The CMP-gather can be mapped into a binary image $d(x, n)$ by deconvolution based on the multipulse technique [4, 8]. Figure

1.left shows a synthetic CMP-gather and Figure 1.right the thresholded image obtained by multipulse technique deconvolution. The CMP-gather is composed of seven reflection curves. Concerning the deconvolution fifteen reflections are chosen to be estimated within each trace, which implies that eight erroneous reflections within each trace have been found. These eight reflection will act as uncorrelated noise.

Next, Figure 2 shows the parameter domain obtained by use of IPM , where $v_{t_0} = 2$ and $v_\sigma = 4$. Figure 3 shows the parameter domain achieved by applying GRT in the parameter domain region specified by the thresholded and resampled parameter domain obtained by IPM . Within the thresholding process a significance level of $\lambda_1 = 0.5$ has been used. Finally, Figure 4 shows the identified reflection hyperbolae, where a significance level of $\lambda_2 = 0.8$ has been used in the thresholding. As seen all reflection curves have been found.

5 Conclusion

A new method for fast detection of seismic reflections is presented. Generally, the method can be viewed as a curve matching method for images with a specific structure.

The foundation of the proposed method is to estimate the reflection curves by combining the multipulse technique [3, 4] and the FCE-algorithm [1]. The multipulse technique is used for wavelet identification within each trace, and the FCE-algorithm is used to coordinate the wavelet identification between the individual traces.

The proposed method is successfully applied for detection of hyperbolic reflection curves within CMP-gathers.

References

- [1] K. V. Hansen and P. A. Toft, "Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform," Submitted for publication, August 1993.
- [2] W. A. Schneider, "The Common Depth Point Stack," *Proc. IEEE*, vol. 72, no. 10, pp. 1238-1254, Oct. 1984.
- [3] B. S. Atal and J. R. Remde, "A New Model of LPC Excitation for Producing Natural-Sounding Speech at Low Bit Rates," *IEEE ICASSP*, vol. 1, 1982, pp. 614-617.
- [4] M. Cooley, H. J. Trussell and I. J. Won, "Seismic Deconvolution by Multipulse Methods," *IEEE Trans. ASSP*, no. 1, vol. 38, pp. 156-160, Jan. 1990.

-
- [5] J. Radon, "Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten," *Ber. Ver. Sächs. Akad. Wiss. Leipzig, Math-Phys. Kl.*, vol. 69, pp. 262–277, April 1917.
- [6] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Com. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972.
- [7] J. Illingworth and J. Kittler, "A Survey of the Hough Transform," *Computer Vision, Graphics, and Image Processing*, vol. 44, pp. 87–116, 1988.
- [8] K. V. Hansen and J. Larsen, "An Algorithm for Successive Identification of Reflections," *IEEE Trans. IP*, vol. 3, no. 3, May 1994.

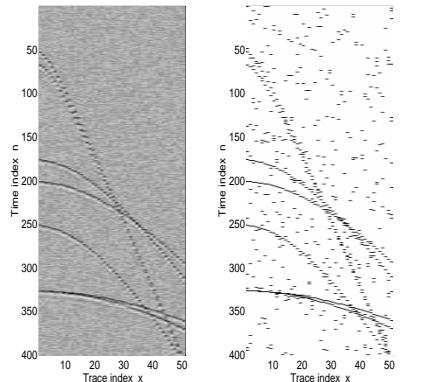


Figure 1: Left: CMP-gather. Right: Thresholded CMP-gather.

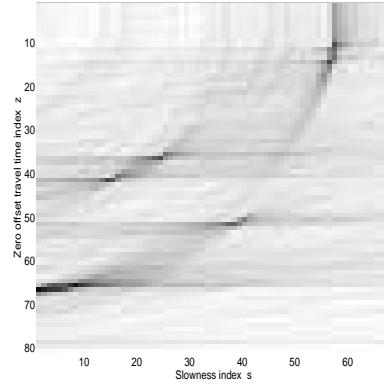


Figure 2: IPM-domain.

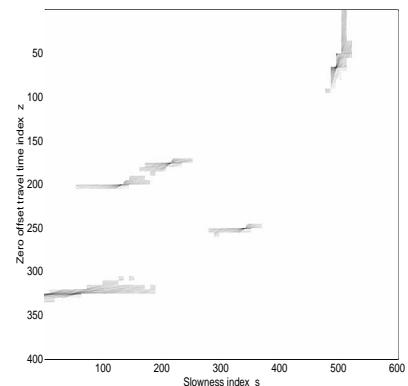


Figure 3: GRT-domain.

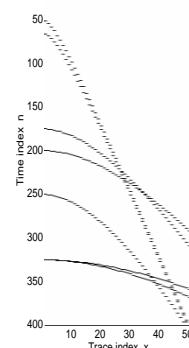


Figure 4: Identified reflection curves.

Appendix H

Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform

This appendix contains a pre-print of the paper *Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform*, by Kim V. Hansen and Peter A. Toft [1]. The work has been carried out as a joint venture project between Ødegaard & Danneskiold-Samsøe and Department of Mathematical Modelling (before january 1. 96 Electronics Institute).

The paper has been accepted for publication in the IEEE Transactions on Image Processing and should appear in the December 96 issue.

Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform

Kim V. Hansen, Oticon, Strandvejen 58, 2900 Hellerup, Phone +45 39177120,
e-mail kvh@Oticon.Oticon.dk

Peter A. Toft, Department of Mathematical Modelling, Building 349,
Technical University of Denmark, 2800 Lyngby, Denmark, Phone +45 45934207,
e-mail ptoft@ei.dtu.dk

I INTRODUCTION

A recurring problem in computer image processing is the identification of curves with specific shapes. The transform developed by Hough [1] for detection of complex patterns in binary $\{0, 1\}$ digital images has been discussed by several authors, e.g., [2], [3] and generalized for multi-dimensional pattern detection in [4]. The Hough transform can be deduced [5] as a special case of a more general transform, known as the Radon transform [6]. The Radon transform is a mapping from an image domain to a parameter domain, where the parameters characterize the curves to be identified. The Radon transform can be generalized to handle arbitrary curves [7]. The Hough version of the generalized Radon transform detects specific values of parameters, as spatially extended curves are transformed to produce spatially compact features in the parameter domain. In this way, the generalized Radon transform converts a difficult global detection problem in the image domain into a more easily solved local peak detection problem in the parameter domain.

In recent years progress has been made to understand and increase the speed of the generalized Radon transform [8, 9, 4]. Investigations concerning both the traditional and more recent generalized Radon transform for curve identification schemes point out two problems with respect to computational cost. First, the estimation schemes are not capable of exploiting image points with zero value (zero image points), and second, estimation is computationally expensive, as it estimates unneeded information.

Abstract

A new algorithm for fast curve parameter estimation based on the generalized Radon transform is proposed. The algorithm works on binary images, obtained, e.g., by edge filtering or deconvolution. The fundamental idea of the suggested algorithm is the use of a pre-condition map to reduce the computational cost of the generalized Radon transform. The pre-condition map is composed of irregular regions in the parameter domain, which contain peaks that represent curves in the image. To generate the pre-condition map, a fast mapping procedure named *image point mapping* is developed. As the image point mapping scheme maps image points into the corresponding parameter values in the parameter domain, it is possible to improve computational efficiency by recognizing image points with value zero. Initially, the suggested algorithm estimates the pre-condition map and subsequently applies the generalized Radon transform within the regions specified by the pre-condition map. The required parameter domain sampling and the resulting blurring are also investigated.

The suggested algorithm is successfully applied to the identification of hyperbolas in seismic images, and two numerical examples are given.

This is a consequence of parameter estimation, in areas of the parameter domain where curve parameters are very unlikely.

The basis for curve parameter estimation as described in this paper is a binary image. The binary image can be produced in several ways, e.g., by edge filtering or deconvolution.

In this paper, a new algorithm for fast curve estimation, called the FCE-algorithm, is presented. The key idea of the FCE-algorithm is to pre-condition the parameter domain, creating irregular regions corresponding to the parameter regions of interest. Furthermore, the FCE-algorithm takes advantage of zero image values in generating the pre-condition map. Initially, the FCE-algorithm identifies regions of the parameter domain which contain peaks representing curves in the image. Subsequently, it estimates a traditional generalized Radon transform within these regions.

The identification of hyperbolas is of particular interest within seismic signal processing. Over the years the seismic industry has developed a method of recording seismic signals resulting in images in which the investigated reflections produce curves. The recording geometry of seismic data acquisition results in signals having the same geometrical mid point between source and receiver, see, e.g., [10]. The set of signals corresponding to the same mid point is called a common mid point gather (CMP-gather). A common technique in seismic data analysis for estimation of hyperbola parameters is velocity analysis, e.g., [11]. Within the presented work some features from velocity analysis are incorporated, however, the presented method has less computational cost.

Recently, the multipulse excited speech coding technique, which was originally proposed by Atal *et al* [12], has been suggested for seismic deconvolution by Cooley *et al.* [13]. Basically, the multipulse model for speech assumes the speech signal to be a result of several impulses transformed by the shape of the vocal tract filter. In seismic deconvolution, the input signal is often modeled as an all-pole impulse and the layered earth reflectivity model is

considered as an impulse train, see, e.g., [14]. Thus, the multipulse technique can be used to map seismic signals in a CMP-gather into a binary image, where the image value is set to one in case of a reflection and zero otherwise. The reflections represent information concerning the geological structure of the subsurface.

The traditional generalized Radon transform is described in section II. Creation of the preconditioning map is described in section III, where the *image point mapping* procedure is developed. In section IV, parameter domain sampling is discussed and section V describes parameter domain blurring, which is a result of discretization. The FCE-algorithm is presented in section VI, and is applied to two numerical examples for detection of hyperbolas in section VII. Regarding notation, scalars are denoted by letters, and vectors by bold-faced letters.

II THE GENERALIZED RADON TRANSFORM

Let $f(\kappa, t)$ be a continuous signal of the continuous variables κ and t and let ξ denote an η -dimensional parameter vector defined as

$$\xi = (\xi_1, \dots, \xi_i, \dots, \xi_\eta) \quad (1)$$

where (κ, t) is called the image domain, ξ the parameter domain, and $f(\kappa, t)$ the image.

A. The Continuous Generalized Radon Transform

Let $F(\xi)$ denote the continuous generalized Radon transform of $f(\kappa, t)$, defined as

$$F(\xi) = \int_{-\infty}^{\infty} f(\kappa, \phi(\kappa; \xi)) d\kappa \quad (2)$$

where $t = \phi(\kappa; \xi)$ denotes the transformation curve. The generalized Radon transform is the integration of $f(\kappa, t)$ along the transformation curve.

Other definitions of the generalized Radon transform can be found in the literature [5], [6]. Equation 2 can be interpreted as a generalization of the Slant Stacking technique used in

geophysics, e.g., [7], but the new ideas presented in this paper can be modified to confirm with this literature.

B. The Discrete Generalized Radon Transform

Let \mathbf{j} denote the η -dimensional discrete index parameter vector defined as

$$\mathbf{j} = (j_1, \dots, j_i, \dots, j_\eta) \quad (3)$$

The correspondence between the index vector \mathbf{j} and the sampled version of the parameter vector $\boldsymbol{\xi}$ can be written as

$$\boldsymbol{\xi} = \boldsymbol{\xi}_{\mathbf{j}} = \boldsymbol{\theta}(\mathbf{j}), \text{ where } \xi_i = \theta_i(j_i) \quad (4)$$

where $\theta_i(j_i)$ is a parameter sampling function.

If a uniform sampling of the parameter domain is chosen, the function θ_i can be written

$$\xi_i = \theta_i(j_i) = \xi_{i,\min} + j_i \Delta \xi_i, \quad j_i = 0, \dots, J_i - 1 \quad (5)$$

where $\xi_{i,\min}$ denotes the lower limit and $\Delta \xi_i$ the sampling interval of ξ_i .

The parameter domain and image domain sampling are assumed to be uniform. The correspondence between the discrete image domain indices (m, n) and the continuous variables (κ, t) can be written as

$$\kappa = \kappa_m = \kappa_{\min} + m \Delta \kappa, \quad m = 0, 1, \dots, M \quad (6)$$

$$t = t_n = t_{\min} + n \Delta t, \quad n = 0, 1, \dots, N - 1 \quad (7)$$

where κ_{\min} and t_{\min} denote the lower limits and $\Delta \kappa$ and Δt the sampling interval of κ and t , respectively.

Substitution of Equation 4, 6 and 7 into the transformation curve gives

$$t = t_{\min} + n \Delta t = \phi(\kappa_{\min} + m \Delta \kappa; \boldsymbol{\theta}(\mathbf{j})) \quad (8)$$

From to Equation 8, the discrete index transformation curve $\phi_d(m; \mathbf{j})$ can be expressed as

$$\begin{aligned} \phi_d(m; \mathbf{j}) &= n \\ &= \text{round}\left(\frac{\phi(\kappa_{\min} + m \Delta \kappa; \boldsymbol{\theta}(\mathbf{j})) - t_{\min}}{\Delta t}\right) \end{aligned} \quad (9)$$

where $\text{round}(\cdot)$ rounds to the closest integer.

Let $d(m, n)$ denote the discretized version of the signal $f(\kappa, t)$, i.e., $d(m, n) = f(\kappa_m, t_n)$ and let $D(\mathbf{j})$ denote the discrete generalized Radon transform (GRT) of $d(m, n)$ defined as

$$D(\mathbf{j}) = \sum_{m=0}^{M-1} d(m, \phi_d(m; \mathbf{j})) \quad (10)$$

Notice, that the term $d\kappa$ has been omitted in the discrete generalized Radon transform as it is a constant due to the uniform sampling of κ . The use of rounding instead of, e.g., linear interpolation will normally not lead to problems. If necessary, interpolation can easily be incorporated in Equations 9 and 10.

III IMAGE POINT MAPPING

From Equation 2, the continuous generalized Radon transform $F(\boldsymbol{\xi})$ can be written as

$$F(\boldsymbol{\xi}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\kappa, t) \delta(t - \phi(\kappa; \boldsymbol{\xi})) dt d\kappa \quad (11)$$

where $\delta(\cdot)$ is the Dirac delta function.

Analogously, the discrete generalized Radon transform, from Equation 10, can be written as

$$D(\mathbf{j}) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} d(m, n) \delta(n - \phi_d(m; \mathbf{j})) \quad (12)$$

where $\delta(\cdot)$ denotes the Kronecker delta function.

Equation 12 shows that each image point (m, n) is transformed into a parameter curve where $\phi_d(m; \mathbf{j}) = n$. Thus mapping of image points with zero value is needless as they will not contribute to $D(\mathbf{j})$. The proposed estimation scheme for the Radon transform accounts for the fact that zero image values do not contribute, and does not include them in the summation in Equation 12. This has the potential to reduce the computational cost considerably.

Now, assume that ϕ is invertible in one of the parameters ξ_i , e.g., ξ_η . The inverse function $\phi_{\xi_\eta}^{-1}$ with respect to ξ_η can be written as

$$\xi_\eta = \phi_{\xi_\eta}^{-1}(\kappa, t; \boldsymbol{\xi}_r), \quad \boldsymbol{\xi}_r = (\xi_1, \dots, \xi_{\eta-1}) \quad (13)$$

Furthermore, assume that the sampling function θ_η is invertible. Then

$$\begin{aligned} j_\eta &= \text{round}(\theta_\eta^{-1}(\xi_\eta)) \\ &= \text{round}(\theta_\eta^{-1}(\phi_{\xi_\eta}^{-1}(\kappa, t; \boldsymbol{\xi}_r))) \\ &\equiv \Omega_d(m, n; \mathbf{j}_r), \quad \mathbf{j}_r = (j_1, \dots, j_{\eta-1}) \end{aligned} \quad (14)$$

Equations 12 and 14, are the basis for an estimation scheme for the generalized Radon transform which is referred to here as *Image Point Mapping* (IPM). The key steps of IPM can be summarized as

1. Initialize $D(\mathbf{j}) = 0$ for all \mathbf{j}
2. For all image points $d(m, n)$ with a value different from zero do
 - For all possible \mathbf{j}_r do

$$\mathbf{j} = (\mathbf{j}_r, \Omega_d(m, n; \mathbf{j}_r))$$

$$D(\mathbf{j}) := D(\mathbf{j}) + d(m, n)$$

Notice, that IPM is a generalization of the Hough transform [1] extended to handle arbitrary transformation curves. Like the Hough transform, the objective of IPM is parameter identification.

The fundamental difference between GRT and IPM is, that GRT requires rounding in the image domain, whereas IPM rounds in the parameter domain, as schematically shown in Figure 1.

The computational complexities of GRT and IPM, respectively, are

$$\begin{aligned} W_{GRT} &= O\left(M \prod_{i=1}^{\eta} J_i\right) \\ W_{IPM} &= O\left((MN)_r \prod_{i=1}^{\eta-1} J_i\right) \end{aligned} \quad (15)$$

where $(MN)_r$ indicates that only a reduced number of image points (nonzero values) are to be transformed. In Equation 15, the cost to test for nonzero values is assumed to be negligible.

The major advantage of IPM over GRT is the ability of IPM to ignore image points with a value of zero, making IPM especially well suited for sparse binary images.

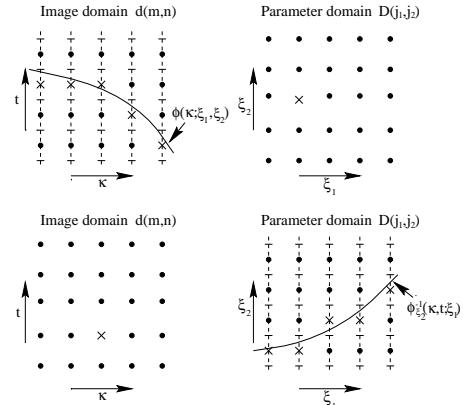


Figure 1: Top: Rounding by GRT in the image domain corresponding to a point in the parameter domain. Bottom: Rounding by IPM in the parameter domain corresponding to a point in the image domain.

If the absolute value between two successive j_η values is greater than one, i.e.,

$$|\Omega_d(m, n; \mathbf{j}_r + \boldsymbol{\varepsilon}) - \Omega_d(m, n; \mathbf{j}_r)| > 1 \quad (16)$$

where vector $\boldsymbol{\varepsilon}$ contains zeros at all $\eta-1$ entries and holds unit value at entry i , i.e., $\varepsilon_i = 1$, it is seen that IPM will map the image point (m, n) into a perforated hypercurve, i.e., a hypercurve containing missing sections. Identification of curve parameters can be difficult due to the presence of perforation holes depending on the shape and depth of the holes.

For $\eta = 2$ the perforation problem can be eliminated simply by adding $d(m, n)$ to $D(\mathbf{j})$ for the parameter values skipped between two successive parameter vectors. Consider, for example, the hyperbolic transformation curve as illustrated in Figure 2, where the parameter domain is described by $(j_1, j_2) = (s, z)$.

Figure 2 shows the mapping of an image point (m^*, n^*) to the parameter domain, that is, discretized parameter point accumulators to which the value $d(m^*, n^*)$ of the image point must be added. To investigate the perforation problem, two parameter points $(s^* - 1, z^* + 3)$ and (s^*, z^*) on the inverse transformation curve are

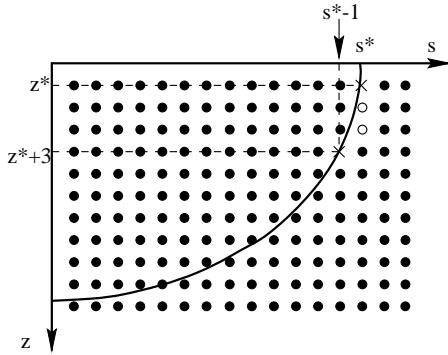


Figure 2: Illustration of the perforation problem.

considered. Following the inverse transformation curve from $(s^* - 1, z^* + 3)$ to (s^*, z^*) along the s axis gives only the two parameter points $(s^* - 1, z^* + 3)$ and (s^*, z^*) . However, following the inverse transformation curve along the z axis also gives the two intervening points $(s^*, z^* + 2)$ and $(s^*, z^* + 1)$. One way of handling such intervening points is to add the value of the image point to the parameter points skipped between two successive parameter points on the inverse transformation curve. Therefore, in this example, $d(m^*, n^*)$ must be added to the skipped parameter points $(s^*, z^* + 2)$ and $(s^*, z^* + 1)$ when following the s axis.

IV PARAMETER DOMAIN SAMPLING

For a given image $d(m, n)$ an exact determination of $\Delta\xi_i$, $\xi_{i,min}$ and J_i , which match the image, is, in general, not possible. However, some guidelines can be stated. First, some of the parameters will be bounded by the underlying physics, e.g., $\xi_{i,min}$ and $\xi_{i,max}$ will normally be limited. Second, the parameter domain can be limited by requiring that at least a fraction β of the image points addressed by the transformation curve $\phi_d(m; \mathbf{j})$ lie inside the image,

i.e.,

$$\sum_{m=0}^{M-1} I\{t_{min} < \phi(\kappa_m; \boldsymbol{\xi}) < t_{max}\} > \beta M, \forall \boldsymbol{\xi} \quad (17)$$

where $0 < \beta < 1$, and $I\{\cdot\}$ equals 1 when the logical expression is true and 0 otherwise.

Concerning use of the GRT, the sampling intervals $\Delta\xi_i$ can be chosen by requiring that

$$\max\{|\phi(\kappa; \boldsymbol{\xi} + \boldsymbol{\gamma}_i) - \phi(\kappa; \boldsymbol{\xi})|\} = \Delta t \quad \forall \kappa, \boldsymbol{\xi}, \boldsymbol{\gamma}_i \quad (18)$$

where $\boldsymbol{\gamma}_i$ is a η -dimensional vector containing zeros at all entries except entry i which is $\Delta\xi_i$.

Equation 18 states that two adjacent $\boldsymbol{\xi}$ -vectors give $t = \phi(\kappa; \boldsymbol{\xi})$ values which cannot be separated by more than one sample in the t -direction. This design criterion is not optimal as it leads to an unnecessarily dense sampling of certain parts of the parameter domain, however, it can be used as an upper sampling rate limit. Increasing the sampling rate of the parameter domain above a certain limit will not improve the resolution obtained by GRT as adjacent \mathbf{j} vectors will result in the same curve $n = \phi_d(m; \mathbf{j})$. For GRT, a coarse sampling cannot guarantee a given curve will be detected, due to the fact that no transformation curve $\phi_d(m; \mathbf{j})$ can be guaranteed to follow the image curve perfectly.

Use of IPM with a dense parameter domain sampling according to Equation 18 produces curves $j_\eta = \Omega_d(m, n; \mathbf{j}_r)$ for (m, n) values corresponding to an image curve which do not intersect in a parameter point \mathbf{j} , but will spread out over several parameter points. The reason for this spread is the individual treatment of image points as regions of zero size and not regions of, e.g., rectangular shape. This spread can be compensated by the use of a coarse sampling of the parameter domain. In addition, a coarse sampling will lead to a lower computational cost.

Summarizing, GRT requires a dense parameter domain sampling, while IPM gives rise to blurring in the parameter domain in the case of dense parameter domain sampling, but works well with a coarse sampling of the parameter

domain. This observation is the basis for the curve parameter estimation algorithm, presented in section VI.

V PARAMETER DOMAIN BLURRING

Parameter domain blurring achieved by use of GRT can be interpreted in a way that makes it usable for parameter clustering. According to Equation 2 the continuous transformation curve can be written as

$$t = \phi(\kappa; \xi), \quad \kappa \in [\kappa_{min}, \kappa_{max}] \quad (19)$$

Let Δt denote the sampling interval of t and let $\alpha\Delta t$ be a given uncertainty of t , e.g., $\alpha = 0.5$. For a parameter vector $\xi = \theta(j)$ to lie inside a band of width $2\alpha\Delta t$, symmetrically positioned around the image curve $t = \phi(\kappa; \xi)$, it must satisfy the following inequality

$$t - \alpha \Delta t < \phi(\kappa; \xi) < t + \alpha \Delta t \quad (20)$$

$$t = \phi(\kappa; \xi'), \quad \forall \kappa \in [\kappa_{min}, \kappa_{max}] \quad (21)$$

Several parameter vectors ξ will correspond to image points within the band specified in Equation 21. Therefore, in general, a band in the image around a given curve described by the parameter vector ξ' will give rise to a region in the parameter space which consists of parameter vectors that correspond to curves inside the image band. These regions are named *clusters* and the corresponding image curve with parameter ξ' is called a *center curve*. An example of a curve band around a center curve and the corresponding parameter cluster is illustrated in Figure 3.

Unfortunately, the center curves are normally unknown, and it is impossible to determine the clusters. However, it is possible to determine whether two parameter sets ξ_1 and ξ_2 belong to the same cluster. If two parameter vectors belong to the same cluster they must satisfy

$$|\phi(\kappa; \xi_1) - \phi(\kappa; \xi_2)| < 2 \alpha \Delta t, \quad \kappa \in [\kappa_{min}, \kappa_{max}] \quad (22)$$

Parameter domain points may be gathered into regions or clusters, with the guarantee that

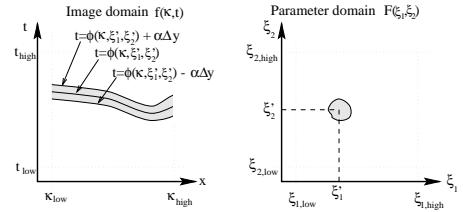


Figure 3: Left: Center curve with an uncertainty of $2\alpha\Delta t$ indicated. Right: Cluster corresponding to center curve.

all possible image curves will be represented by only one cluster in the parameter domain. This partitions the parameter domain into irregular regions, which reflect the information level of the image. Clustering can also be used to estimate parameter uncertainties, e.g., the maximum and minimum parameter value within the cluster can be used to give an estimate of the parameter uncertainty as, e.g., $\frac{1}{2}(\xi_{i,max} - \xi_{i,min} + \Delta\xi_i)$. Parameter domain clustering can be performed either before or after the transform, depending on the purpose.

VI THE FAST CURVE ESTIMATION ALGORITHM

Consider an image where the image values are represented by continuous values. This image can be mapped into a binary image $d(m, n)$, e.g., by edge filtering [15], [16] or by deconvolution [17], [13]. The proposed algorithm estimates the parameters of curves in the binary image $d(m, n)$, e.g., lines or hyperbolas.

It is well-known that direct use of the GRT is computationally expensive, and gives a lot of unnecessary information concerning estimation of curve parameters. In the light of the characteristics of GRT and IPM, a fast curve parameter estimation algorithm, the *FCE-algorithm*, is proposed which simultaneously estimates all parameters of curves having a specific shape, e.g., lines or hyperbolas. The FCE-algorithm uses IPM as a pre-conditioning procedure for GRT by selecting the regions of interest in the

parameter domain. IPM is suitable for a rapid determination of regions of interest, as it works well on a coarsely sampled parameter domain and is capable of ignoring image points with a value of zero. Another important circumstance is that GRT relates the uncertainty on the estimated curve parameters to the image domain sampling, while IPM relates it to the parameter domain sampling. The FCE-algorithm is shown in Figure 4, and is summarized as follows:

1. Design the discrete parameter domain, i.e., choose $\xi_{i,\min}$, J_i and $\Delta\xi_i$.
2. Design a reduced parameter domain for IPM by choosing

$$J'_i = \text{ceil} \left(\frac{J_i}{2v_{\xi_i} + 1} \right) \quad (23)$$

$$\xi'_{i,\min} = \xi_{\min} + v_{\xi_i} \Delta\xi_i \quad (24)$$

$$\Delta\xi'_i = (2v_{\xi_i} + 1) \Delta\xi_i \quad (25)$$

where $\text{ceil}(\cdot)$ rounds to the nearest upper integer, and v_{ξ_i} is an integer related to the resampling (see below). Then use IPM to estimate $D_{ipm}(\mathbf{j}')$ as

$$D_{ipm}(\mathbf{j}') = \text{IPM}\{d(m, n)\} \quad (26)$$

3. Use the threshold function T to give

$$D_t(\mathbf{j}') = T(D_{ipm}(\mathbf{j}')) \quad (27)$$

The threshold function T is designed to remove all insignificant parameter combinations and a suitable choice might be

$$T(D_{ipm}(\mathbf{j}')) = \mu(D_{ipm}(\mathbf{j}') - \lambda_1 M) \quad (28)$$

where M denotes the number of image lines, λ_1 a fraction defining the significance level, and $\mu(\cdot)$ the Hamilton step function. Having an image with one curve, IPM gives a corresponding parameter domain value of M . Choosing a threshold level of $\lambda_1 M$ allows some deviation for the image curve from the integration curve, e.g., to handle holes.

4. Resample the parameter domain to obtain $D_r(\mathbf{j})$. The resampling process can be written as

$$D_r(\mathbf{j}) = D_t(\mathbf{j}') \quad (29)$$

$$j'_i = \text{round} \left(\frac{j_i - v_{\xi_i}}{2v_{\xi_i} + 1} \right) \quad (30)$$

This quickly fills the entire parameter domain where the interesting regions have value one while other regions have value zero.

5. Use GRT to estimate $D(\mathbf{j})$ in the regions of the parameter domain where $D_r(\mathbf{j})$ is not equal to zero.

$$D(\mathbf{j}) = \text{GRT}\{d(m, n) | D_r(\mathbf{j}) \neq 0\} \quad (31)$$

6. As in step three, use the threshold function with a new significance level λ_2 to account for use of the GRT. After the thresholding, all points in the parameter domain having a value different from zero represent curve parameters.

7. Finally, the identified parameters are clustered into common image curves.

The resampling process takes each parameter point corresponding to the sampling interval $\Delta\xi'$ and extends it to several parameter points corresponding to the sampling interval $\Delta\xi$, $\Delta\xi_i < \Delta\xi'_i$. Thus, the FCE-algorithm operates initially in a coarsely sampled parameter domain, using IPM for determination of regions of interest. Subsequently, the sampling is refined to the required level and GRT is applied within the regions of interest.

The significance levels λ_1 and λ_2 must be chosen to reflect the parameter domain values, that can be accepted as peaks corresponding to curves in the image. In general, the precise values of λ_1 and λ_2 are not critical. For the examples, suitable ranges of values for λ_1 and λ_2 are 0.1 – 0.7 and 0.4 – 0.8, respectively. There are two reasons for the use of a lower acceptance level in step three. First, IPM has a tendency to blur in the parameter domain, and second,

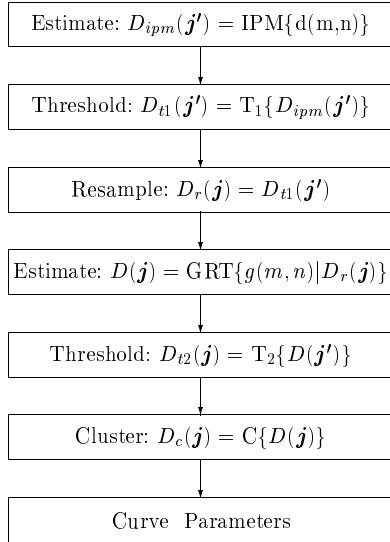


Figure 4: Flow diagram of the FCE-algorithm.

only regions of the parameter domain ensured not to contain image curves must be removed. A higher significance level in step six (λ_2) is justified by the required certainty for correct curve identification.

VII THE HYPERBOLIC TRANSFORMATION CURVE

Consider a seismic CMP-gather. The multipulse technique can be used to map the seismic signals in the CMP-gather into a binary image $d(m, n)$, where $d(m, n)$ is set to one if position n of trace number m contains a reflection, and to zero otherwise. If the subsurface is assumed to be horizontally layered, it can be shown [10] that the shape of the reflection curves are hyperbolas. Traditionally, the reflection curves within the CMP-gather are described by the *normal moveout equation* [10],

which can be written as

$$\phi(\kappa; \sigma, t_0) = t = \sqrt{t_0^2 + (\sigma \kappa)^2}, \quad \kappa \in [\kappa_{min}, \kappa_{max}] \quad (32)$$

where t denotes the two way travel time with distance κ between source and receiver, t_0 the zero offset two way travel time, and finally σ denotes the slowness. The variable κ is also known as the offset.

According to Equation 32, $\phi(\kappa; \sigma, t_0)$ can easily be inverted with respect to t_0 to yield

$$t_0 = \phi_{t_0}^{-1}(\kappa, t; \sigma) = \sqrt{t^2 - (\sigma \kappa)^2} \quad (33)$$

Both t_0 and σ are uniformly sampled and the parameter domain sampling is chosen in agreement with Equation 18. The discretized parameter domain is denoted z and s corresponding to t_0 and σ , respectively,

$$t_0 = t_{0,low} + z \Delta t_0, \quad z = 0, \dots, Z-1 \quad (34)$$

$$\sigma = \sigma_{low} + s \Delta \sigma, \quad s = 0, \dots, S-1 \quad (35)$$

The curve perforation described in section III occurs in the present case of a hyperbolic transformation curve. It is, however, eliminated by adding $d(m, n)$ to $D(s, z)$ for the parameter values skipped between two successive parameter vectors.

Assume the parameter set $(\sigma_0, t_{0,0})$ corresponds to a center curve. The parameter sets (σ, t_0) corresponding to image curves within the band of width $2\alpha\Delta t$, symmetrically positioned around the image curve $t = \phi(\kappa; \sigma_0, t_{0,0})$, can according to Equation 21 be written as

$$t - \alpha \Delta t < \sqrt{t_0^2 + (\sigma \kappa)^2} < t + \alpha \Delta t \quad (36)$$

$$t = \sqrt{t_{0,0}^2 + (\sigma_0 \kappa)^2} \quad \forall \kappa \in [\kappa_{min}, \kappa_{max}] \quad (37)$$

Figure 5 illustrates four clusters in the parameter domain. The four clusters correspond to four center curves with an uncertainty of $10\Delta t$, i.e., $\alpha = 5$. Formulas for the cluster shape is shown in appendix A.

To illustrate the performance of the FCE-algorithm two numerical examples will be given.

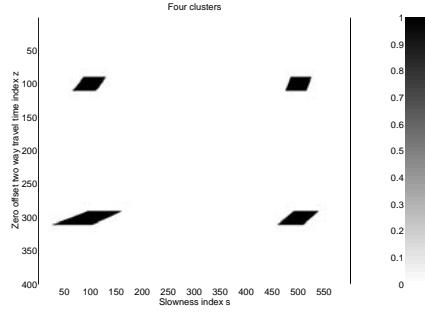


Figure 5: Four clusters in the parameter domain corresponding to an uncertainty of $10\Delta t$, i.e., $\alpha = 5$.

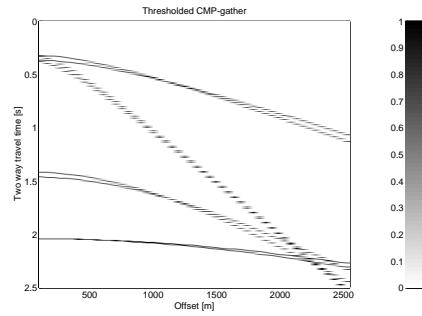


Figure 6: Synthetic binary CMP-gather composed of eight hyperbolae.

A. Example 1

In this example, the image is composed of eight hyperbolae, assembled into four groups of two hyperbolae each, as shown in Figure 6. Using GRT according to the settings in Table 1 leads to eight peaks in the parameter domain as shown in Figure 7. Thus, although the eight curves intersect and are very close, GRT is able to separate the curves in the parameter domain into eight separate peaks.

Image domain		Parameter domain	
Par.	Value	Par.	Value
M	50	S	300
N	400	Z	400
$\Delta\kappa$	50 m	$\Delta\sigma$	$1.569 \cdot 10^{-6} \text{ s/m}$
Δt	$4 \cdot 10^{-3} \text{ s}$	Δt_0	$4 \cdot 10^{-3} \text{ s}$
κ_{min}	100 m	σ_{min}	$2.3 \cdot 10^{-4} \text{ s/m}$
t_{min}	0 s	$t_{0,min}$	0 s
α	1.5	λ_1	0.5
		λ_2	0.75
		v_σ	4
		v_{t_0}	2

Table 1: Image and parameter domain settings.

The FCE-algorithm is used for fast detection of the interesting regions in parameter domain resulting from GRT. The following five figures show each step of the FCE-algorithm. Figure 8 shows the coarsely sampled parameter domain

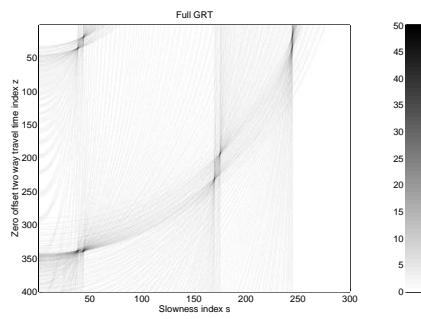


Figure 7: Full GRT estimated parameter domain.

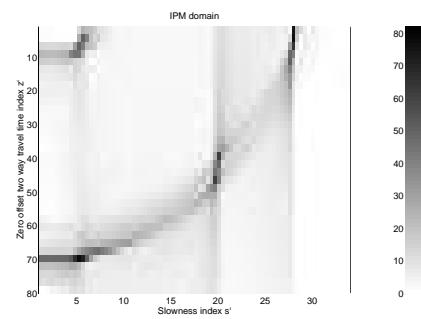


Figure 8: Parameter domain estimated by use of IPM.

obtained using IPM with $v_\sigma = 4$ and $v_{t_0} = 2$.

The parameter domain is thresholded using a significance level $\lambda_1 = 0.5$, and the resulting binary parameter domain is shown in Figure 9.

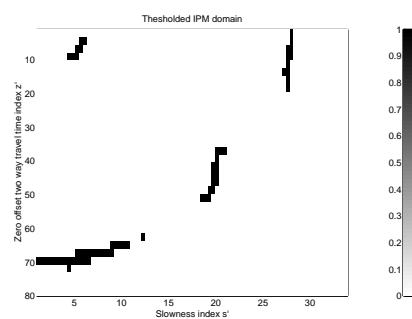


Figure 9: Thresholded IPM estimated parameter domain, where $\lambda_1 = 0.5$.

The binary parameter domain is then resampled to a full size parameter domain as shown in Figure 10.

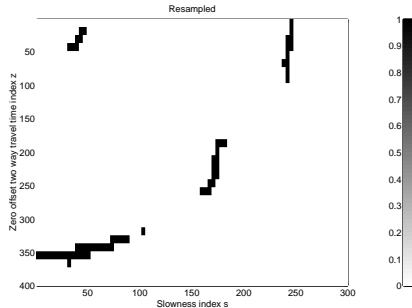


Figure 10: Resampled parameter domain.

Comparing this pre-condition map with Figure 7, indicates that all regions containing curves are contained within the pre-condition map. Next, Figure 11 shows the parameter domain obtained by using GRT within the regions specified by the pre-condition map.

Finally, the pre-conditioned GRT parameter domain is thresholded using a threshold level $\lambda_2 = 0.75$, and the result is shown in Figure 12.

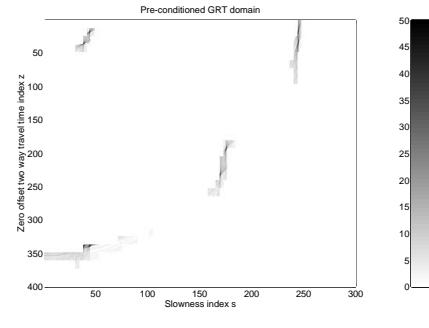


Figure 11: Parameter domain obtained using GRT within the regions specified by the resampled parameter domain shown in Figure 10.

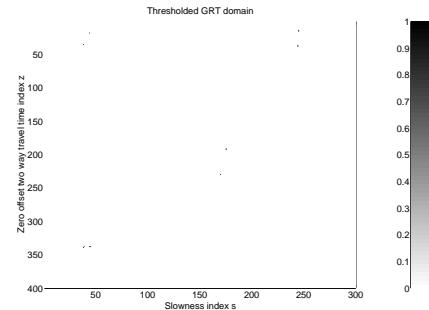


Figure 12: Thresholded GRT estimated parameter domain using $\lambda_2 = 0.75$.

The estimated parameter vectors are given in Table 2, where the clustering process has used a value of $\alpha = 1.5$. As seen from Table 2, eight groups of curve parameters are found, and the estimated curve parameters are rather close to the true parameters.

Curve Number	Parameter
1 True	(0.202 s, $2.7027 \cdot 10^{-4}$ s/m)
1 Estimated	(0.200 s, $2.705 \cdot 10^{-4}$ s/m)
2 True	(0.202 s, $6.2500 \cdot 10^{-4}$ s/m)
2 Estimated	(0.200 s, $6.261 \cdot 10^{-4}$ s/m)
2 Estimated	(0.204 s, $6.245 \cdot 10^{-4}$ s/m)
3 True	(0.232 s, $2.5000 \cdot 10^{-4}$ s/m)
3 Estimated	(0.232 s, $2.503 \cdot 10^{-4}$ s/m)
4 True	(0.232 s, $6.2500 \cdot 10^{-4}$ s/m)
4 Estimated	(0.232 s, $6.245 \cdot 10^{-4}$ s/m)
4 Estimated	(0.236 s, $6.245 \cdot 10^{-4}$ s/m)
5 True	(0.902 s, $5.0000 \cdot 10^{-4}$ s/m)
5 Estimated	(0.904 s, $4.982 \cdot 10^{-4}$ s/m)
6 True	(0.932 s, $4.5455 \cdot 10^{-4}$ s/m)
6 Estimated	(0.932 s, $4.530 \cdot 10^{-4}$ s/m)
6 Estimated	(0.932 s, $4.546 \cdot 10^{-4}$ s/m)
7 True	(1.302 s, $2.7027 \cdot 10^{-4}$ s/m)
7 Estimated	(1.300 s, $2.721 \cdot 10^{-4}$ s/m)
7 Estimated	(1.300 s, $2.737 \cdot 10^{-4}$ s/m)
8 True	(1.302 s, $2.5000 \cdot 10^{-4}$ s/m)
8 Estimated	(1.304 s, $2.456 \cdot 10^{-4}$ s/m)
8 Estimated	(1.304 s, $2.472 \cdot 10^{-4}$ s/m)

Table 2: The results of the FCE-algorithm along with the true curve parameters. The width of the cluster band is set to 3, i.e. $\alpha = 1.5$.

13, where each vertical line represents a trace and the signal values are illustrated using grey scale. The parameters for both image and parameter domains are given in Table 3.

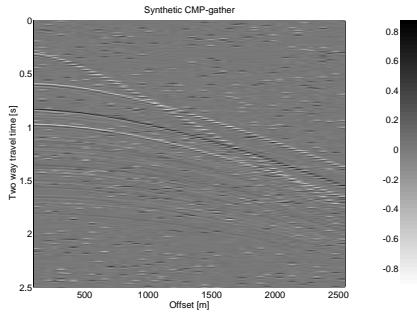


Figure 13: Synthetic CMP-gather composed of 29 reflection curves, added with uncorrelated noise.

Image domain		Parameter domain	
Par.	Value	Par.	Value
M	50	S	200
N	625	Z	625
$\Delta\kappa$	50 m	$\Delta\sigma$	$1.57 \cdot 10^{-6}$ s/m
Δt	$4 \cdot 10^{-3}$ s	Δt_0	$4 \cdot 10^{-3}$ s
κ_{min}	100 m	σ_{min}	$4.0 \cdot 10^{-4}$ s/m
t_{min}	0 s	$t_{0,min}$	0 s
α	1.5	λ_1	0.5
		λ_2	0.5
		v_σ	4
		v_{t_0}	2

Table 3: Image and parameter domain settings.

B. Example 2

To demonstrate the performance of the FCE-algorithm in the presence of noise, an example based on a noise corrupted synthetic CMP-gather is given. The subsurface model is a pure acoustic (compressional waves only) horizontally layered subsurface consisting of four finite layers and infinite top and bottom layers. The synthetic CMP-gather is produced by use of ray tracing [10]. The resulting CMP-gather is composed of 29 reflection curves, shown in Figure

In Figure 14 the corresponding true reflection curve parameters are shown, that is the values of t_0 and σ corresponding to the 29 reflection curves.

Next, the multipulse technique [12] is used to map the seismic signals in the CMP-gather into a binary image, shown in Figure 15. As seen, the mapping process results in erroneous reflections, i.e., reflections without correlation over traces. Furthermore, the reflection curves have gaps, intersect and in cases are very close.

These problems are caused by the fact that the multipulse technique is a one dimensional process.

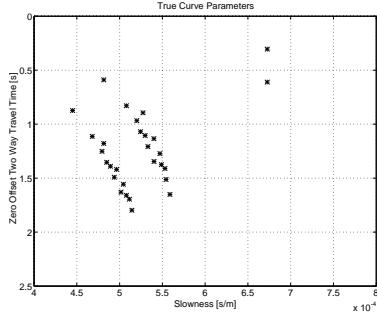


Figure 14: True reflection curve parameters.

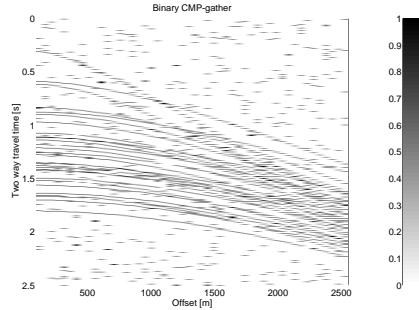


Figure 15: Binary CMP-gather obtained using multipulse technique at Figure 13.

Applying the FCE-algorithm to the binary image gives the IPM estimated parameter domain shown in Figure 16, using $v_{t_0} = 2$ and $v_\sigma = 4$.

Applying GRT on the parameter domain region specified by the resampled and thresholded IPM estimated parameter domain, with $\lambda_1 = 0.5$, results in the GRT parameter domain shown in Figure 17.

Finally, Figure 18 shows the estimated curve parameters, where a significance level of $\lambda_2 = 0.5$ in the final threshold process and $\alpha = 1.5$ in the clustering process have been used. Com-

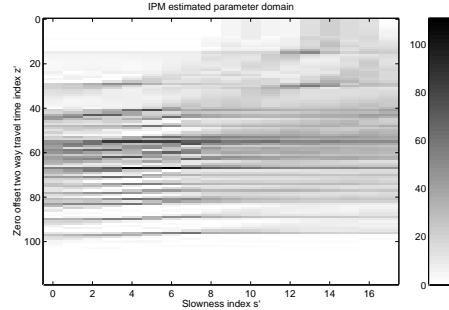


Figure 16: IPM estimated parameter domain.

paring Figure 18 with Figure 14, which shows the true reflection curves, all reflection curves are seen to have been found, and the accuracy of the estimated reflection curves is good. The deviations of the curve parameters are less than $\pm 2 \cdot 10^{-3}$ s for t_0 and $\pm 1.5 \cdot 10^{-6}$ s/m for σ .

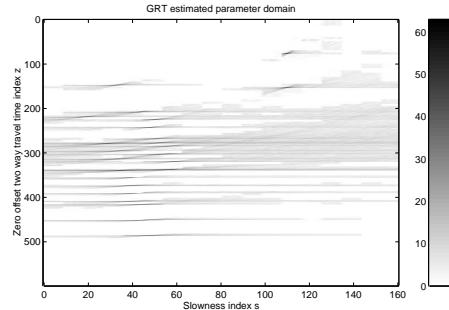


Figure 17: GRT estimated parameter domain using a pre-condition map, estimated by use of IPM.

VIII CONCLUSION

A new algorithm for fast curve parameter estimation, named the FCE-algorithm, has been presented. The algorithm identifies curve parameters by operating on a binary image, obtained, e.g., by edge filtering or deconvolution.

The fundamental idea of the algorithm is the use of pre-conditioning to reduce the computational cost of the traditional generalized Radon transform. The pre-conditioning map deter-

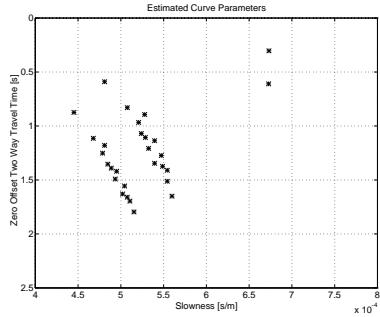


Figure 18: Estimated curve parameters using the FCE-algorithm.

ines regions of the parameter domain which contain peaks, and the generalized Radon transform is applied only in these regions. As the size of the regions is less than the full parameter domain, the pre-conditioning map reduces the computational costs when applying the generalized Radon transform. For fast generation of the pre-conditioning map, a generalization of the Hough transform named *image point mapping* has been developed. Image point mapping is computationally efficient by taking account of image points with value zero. The required parameter domain sampling and the resulting parameter domain blurring have been investigated.

The FCE algorithm was successfully applied to the identification of hyperbolas in seismic images and two numerical examples have been presented. One example demonstrates the potential of the algorithm for fast and accurate parameter estimation, and the other example illustrates the robustness of the algorithm with noise.

IX ACKNOWLEDGMENTS

We wish to thank Peter Koefoed Møller, Katja Blankensteiner, Peter Skjellerup, and Jan Larsen for useful comments on this paper.

This work was partly supported by the Danish Academy of Technical Sciences and ØD-S Holding A/S.

A CLUSTERING USING HYPERBOLIC TRANSFORMATION CURVES

In the case of hyperbolic transformation curves the exact shape of the parameter domain blurring can be calculated.

Let q denote a binary variable, $q \in \{+1, -1\}$. Thus, the lowest and highest curve within the image band specified by Equation 37 can be written

$$\sqrt{t_{0,0}^2 + (\sigma_0 \kappa)^2} + q \alpha \Delta t = \sqrt{t_0^2 + (\sigma \kappa)^2} \Leftrightarrow \\ t_0^2 = \left(\sqrt{t_{0,0}^2 + (\sigma_0 \kappa)^2} + q \alpha \Delta t \right)^2 - (\sigma \kappa)^2$$

It can be shown that κ_{low} and κ_{high} give the cluster limiting elliptical curves. The parameter cluster corresponding to the center curve $(t_{0,0}, \sigma_0)$ can be written as

$$\sqrt{\left(\sqrt{t_{0,0}^2 + (\sigma_0 \kappa_{low})^2} - \alpha \Delta t \right)^2 - (\sigma \kappa_{low})^2} < t_0 \\ < \sqrt{\left(\sqrt{t_{0,0}^2 + (\sigma_0 \kappa_{low})^2} + \alpha \Delta t \right)^2 - (\sigma \kappa_{low})^2}$$

and

$$\sqrt{\left(\sqrt{t_{0,0}^2 + (\sigma_0 \kappa_{high})^2} - \alpha \Delta t \right)^2 - (\sigma \kappa_{high})^2} < t_0 \\ < \sqrt{\left(\sqrt{t_{0,0}^2 + (\sigma_0 \kappa_{high})^2} + \alpha \Delta t \right)^2 - (\sigma \kappa_{high})^2}$$

Figure 19 (left) shows a hyperbolic center curve $(t_{0,0}, \sigma_0)$ with an uncertainty of $2\alpha\Delta t$. Figure 19 (right) shows the shape of the corresponding cluster, which is bounded by four elliptical curves. It should be noted that the cluster shape is highly dependent on the center curve $(t_{0,0}, \sigma_0)$.

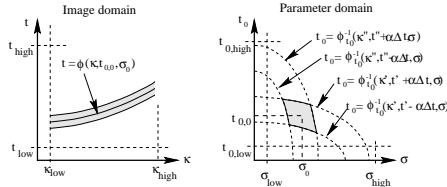


Figure 19: Left: Center curve with an uncertainty of $2\alpha\Delta t$ indicated. Right: Parameter cluster corresponding to curve band, where $\kappa' = \kappa_{low}$, $\kappa'' = \kappa_{high}$, $(t')^2 = t_{0,0}^2 + (\sigma_0 \kappa_{low})^2$, and $(t'')^2 = t_{0,0}^2 + (\sigma_0 \kappa_{high})^2$.

REFERENCES

- [1] P. V. C. Hough. A Method and Means for Recognizing Complex Patterns. US Patent: 3,069,654, Dec. 1962.
- [2] R. O. Duda and P. E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communication of the Association for Computing Machinery*, 15(1):11–15, 1972.
- [3] J. Illingworth and J. Kittler. A Survey of the Hough Transform. *Computer Vision, Graphics, and Image Processing*, 44:87–116, 1988.
- [4] H. Li, M. A. Lavin and R. J. Le Master. Fast Hough Transform: A Hierarchical Approach. *Computer Vision, Graphics, and Image Processing*, 36:139–161, 1986.
- [5] S. R. Deans. Hough Transform From the Radon Transform. *IEEE PAMI*, 3(2):185–188, 1981.
- [6] J. Radon. Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten. *Ber. Ver. Sächs. Akad. Wiss. Leipzig, Math-Phys. Kl.*, 69:262–277, April 1917.
- [7] S. R. Deans. *The Radon Transform and Some of Its Applications*. Krieger Publishing Company, Malabar, Florida, 2 edition, 1993. Previously John Wiley & Sons Inc., 1983.
- [8] G. Beylkin. Discrete Radon Transform. *IEEE tr. ASSP*, 35(2):162–172, 1987.
- [9] C. J. Haneveld and G. C. Herman. A Fast Algorithm for Computation of Radon Transforms. *Geophysical Prospecting*, 38:853–860, 1990.
- [10] W. A. Schneider. The Common Depth Point Stack. *Proceedings of the IEEE*, 72(10):1238–1254, 1984.
- [11] Özdogan Yilmaz. *Seismic Data Processing*. Society of Exploration Geophysicists, Tulsa, Oklahoma, 1987.
- [12] B. S. Atal and J. R. Remde. A New Model of LPC Excitation for Producing Natural-Sounding Speech at Low Bit Rates. In *Proceedings of ICASSP*, volume 1 of *ICASSP*, pages 614–617. IEEE, 1982.
- [13] M. Cooley, H. J. Trussel and I. J. Won. Seismic Deconvolution by Multipulse Methods. *IEEE tr. ASSP*, 38(1):156–160, 1990.
- [14] E. A. Robinson, T. S. Durrani and L. G. Peardon. *Geophysical Signal Processing*. Englewood Cliffs, New Jersey, Prentice-Hall Inc., 1980.
- [15] J. Canny. A Computational Approach to Edge Detection. *IEEE tr. PAMI*, 8(6):679–698, 1986.
- [16] K. Y. Huang, K. S. Fu, T. H. Sheen and S. W. Cheng. Image Processing of Seismograms: (A) Hough Transformation for the Detection of Seismic Patterns; (B) Thinning Processing in the Seismogram. *Pattern Recognition*, 18(6):429–440, 1985.
- [17] C. Y. Chi, J. M. Mendel and D. Hampson. A Computationally Fast Approach to Maximum-Likelihood Deconvolution. *Geophysics*, 49(5):550–565, 1984.

Appendix I

Using the Generalized Radon Transform for Detection of Curves in Noisy Images

This appendix contains the paper *Using the Generalized Radon Transform for Detection of Curves in Noisy Images*, by Peter Toft.

The work has been presented at the *IEEE ICASSP 96* and this paper appears in the proceedings [3].

USING THE GENERALIZED RADON TRANSFORM FOR DETECTION OF CURVES IN NOISY IMAGES

Peter A. Toft

Electronics Institute, Technical University of Denmark, DK-2800 Lyngby.
Email ptoft@ei.dtu.dk. Phone +45 45934207

ABSTRACT

In this paper the discrete generalized Radon transform will be investigated as a tool for detection of curves in noisy digital images. The discrete generalized Radon transform maps an image into a parameter domain, where curves following a specific parameterized curve form will correspond to a peak in the parameter domain. A major advantage of the generalized Radon transform is that the curves are allowed to intersect. This enables a thresholding algorithm in the parameter domain for simultaneous detection of curve parameters. A threshold level based on the noise level in the image is derived. A numerical example is presented to illustrate the theory.

1. INTRODUCTION

In recent years the Hough transform [1] and the related Radon transform [2] have received much attention. These two transforms are able to transform two dimensional images with lines into a domain of possible line parameters, where each line in the image will give a peak positioned at the corresponding line parameters. This has led to many line detection applications within image processing, computer vision, and seismics.

A natural expansion of the Radon transform is the (discrete) generalized Radon transform (GRT) [3, 4, 5]. Analogous to the linear Radon transform, the GRT transforms curves in the image into a discrete multi dimensional parameter domain producing peaks positioned at the corresponding curve parameters. In this way the GRT converts a difficult global detection problem into a more easily solved local peak detection problem. A major advantage of the GRT is that curves are allowed to intersect. Another major advantage that will be demonstrated in this paper, is that the GRT is very robust to noise.

In this paper a probabilistic approach is used to show that the GRT can be used for curve detection if the noise in the image is below a certain level compared to the signal values on the curves. If noise is added to

an image containing curves, the problem is that peaks in the parameter domain may or may not correspond to actual curve parameters. A threshold level, based on the noise level, is derived and applied for separation of noise and curve information in the parameter domain. A numerical example is provided to illustrate the presented theory.

2. THE GENERALIZED RADON TRANSFORM

The Generalized Radon transform, GRT, of a digital image can be defined in many ways. One way is

$$\check{g}(\mathbf{j}) = \sum_{l=0}^{L-1} g(\phi_m(l, \mathbf{j}), \phi_n(l, \mathbf{j})) \quad (1)$$

where \check{g} denote the GRT of the image $g(m, n)$ and \mathbf{j} is a multi dimensional vector containing the curve parameters. The two curve functions $\phi_m(l, \mathbf{j})$ and $\phi_n(l, \mathbf{j})$ define the curve type and are (in principle) arbitrary¹. A popular choice is the linear curve functions; e.g., normal parameters $\mathbf{j} = (\rho, \theta)$. Another frequent choice is the (τ, p) -parameters (known as slant stacking in seismics), where $\phi_m(l, \tau, p) = l$ and $\phi_n(l, \tau, p) = p l + \tau$.

Even though the GRT can be applied to any given image, the main feature of the GRT is that an image, which contains a discrete curve matching the curve functions at one parameter vector \mathbf{j}^* implies that the parameter domain $\check{g}(\mathbf{j})$, will show a peak at that specific parameter vector $\mathbf{j} = \mathbf{j}^*$. The linearity of the GRT implies that each curve in the image will be transformed into a peak in the parameter domain. In this paper a curve in an image is defined by large image values of the same sign on the curve and otherwise (approximately) zero.

Initially only two values of the GRT will be considered. The first, $\check{g}(\mathbf{j}^*)$, corresponds to a curve in the

¹ An interpolation scheme is assumed implicitly; e.g., by rounding the functions $\phi_m(l, \mathbf{j})$ and $\phi_n(l, \mathbf{j})$ to the nearest sample point.

image. Another, $\check{g}(\mathbf{j}^-)$, corresponds to a parameter vector, that does not match a curve in the image. It is assumed that $\check{g}(\mathbf{j}^*)$ is the sum of a mean signal value μ over all L samples, and $\check{g}(\mathbf{j}^-)$ covers (approximately) no samples of the curve(s) in the image. Both values of \check{g} are contaminated with noise due to noise in the image. Assume the noise in the image is nearly uncorrelated with zero mean (e.g., by subtracting a DC-value from the image) and variance σ^2 .

A classical curve detection algorithm is to determine the parameter vectors from the positions of peaks in the parameter space

$$\mathbf{j}^* = \arg \{ |\check{g}(\mathbf{j})| \geq L \mu^* \} \quad (2)$$

The reason for choosing the significance level in this way is that Eq. 1 consists of a summation over L samples, and μ^* is a lower positive bound on the mean signal level on the curve; e.g., found by estimation. The purpose of the following is to estimate whether curves having the signal level μ can be detected using Eq. 2, if the image is contaminated with the described noise.

Due to the linearity, \check{g} consists of a curve part and a noise part. If $L \gg 1$ the sum of the noise terms \check{g}_{noise} will approximately be Gaussian distributed with zero mean and variance $L\sigma^2$ due to the Central Limit Theorem. This implies that the two considered values of the GRT are distributed as $\check{g}(\mathbf{j}^*) = \mu L + \check{g}_{\text{noise}}^* \in \mathcal{N}(\mu L, L\sigma^2)$ and $\check{g}(\mathbf{j}^-) = \check{g}_{\text{noise}}^- \in \mathcal{N}(0, L\sigma^2)$. Since Eq. 2 selects the large values in the parameter domain, an important issue is the probability of detecting the correct parameter vector of the two considered

$$P_{\text{det } 2} \equiv P \{ |\check{g}(\mathbf{j}^*)| > |\check{g}(\mathbf{j}^-)| \} \quad (3)$$

Assuming that $\check{g}(\mathbf{j}^*)$ and $\check{g}(\mathbf{j}^-)$ are independent, inserting the joint probability distribution function, i.e., the product of the two individual Gaussian probability distribution functions, and using the trick of rotating the coordinate system 45 degrees, the integrals separate into one dimensional integrals which easily can be expressed by the error-function, $\text{erf}(\cdot)$

$$P_{\text{det } 2} = \frac{1}{2} \left[1 + \left(\text{erf} \left(\frac{\lambda}{2} \right) \right)^2 \right] \quad \text{and} \quad \lambda = \frac{\mu \sqrt{L}}{\sigma} \quad (4)$$

Note that in this case $P_{\text{det } 2}$, shown on Fig. 1, only depends on one parameter λ . Note that $|\lambda| > 4$ gives an almost certain detection. This is the case if $\sigma \rightarrow 0$ or $\mu \rightarrow \infty$.

When using the GRT to detect curves then the discrete parameter domain will not only have two, but \mathcal{J} different parameter vectors, where \mathcal{J} is the number of samples in the parameter domain. It is assumed that

all the noise sources in the parameter domain are independent and in the following, the detection of a single curve is analyzed. Selecting the position of the highest peak in the parameter domain, the probability of the selected parameter vector being correct, can be approximated by

$$P_{\text{det all}} \cong \prod_{i=2}^{\mathcal{J}} P_{\text{det } 2} \cong P_{\text{det } 2}^{\mathcal{J}-1} = \left(\frac{1}{2} \left[1 + \left(\text{erf} \left(\frac{\lambda}{2} \right) \right)^2 \right] \right)^{\mathcal{J}-1} \cong 1 - \frac{2\mathcal{J}}{\lambda \sqrt{\pi}} e^{-\lambda^2/4} \quad (5)$$

The last simple approximation is valid if the detection probability is close to 1 as seen from Fig. 2. Several characteristics can be noted: The Figure shows a narrow transition from low to high detection probability as a function of λ , and \mathcal{J} does not change the shape of $P_{\text{det all}}$ significantly. If demanding a high $P_{\text{det all}}$ then Eq. 5 and Fig. 2 demonstrate that \mathcal{J} should be held low; i.e., by reducing the number of samples in the parameter domain to a minimum. It should be noted that this will involve a compromise on the range of parameter vectors.

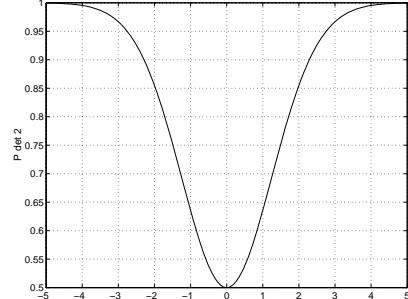


Figure 1: The probability of detecting the right curve parameters of two possible as a function of λ .

Eq. 5 can be used to set requirements on, e.g., the absolute mean signal level μ^* of the curve(s) to be detected. Demanding a detection probability $P_{\text{det all}}$ greater than P^* implies that $\mu^* = \sigma / (\lambda^* \sqrt{L})$, where λ^* can be found by from Eq. 5 with a given detection probability, summation length L , number of samples in the parameter domain \mathcal{J} , and the standard deviation σ (e.g., by found by estimation). Any $|\mu|$ less than the threshold level, μ^* , can be considered as noise. In this way it is possible to give a statistically based estimate on the thresholding level in Equation 2.

Even though the above theory is developed by analyzing one curve in the image, the theory can be used

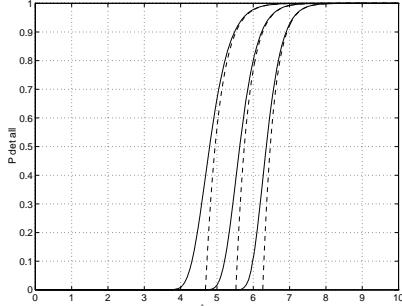


Figure 2: Solid lines show the probability of detecting the right curve parameter as a function of λ , and dashed lines show the simple approximation. From left to right $J=1000$, $J=10000$ and rightmost $J=100000$.

if the image contains few curves. Instead of having one peak in the parameter domain representing one curve in the image, each of the Z curves in the image, where $Z \ll J$, will give a peak in the parameter domain, even if the curves cross each other. With Z curves each of corresponding Z \tilde{g} -values must be larger than the rest in the parameter domain. If only a few curves are present, the rest of the parameter domain is dominated by noise, and the probability of detection for each of the Z curves can be found from Eq. 5.

The theory used to derive Eq. 5 is somewhat pessimistic in estimation of the influence of the noise. This is partly due to the assumption that all the GRT-values include summing noise over L samples. Normally some of the GRT-values will require summing up over a curve partially outside the image, where the image must be assumed equal to zero. Furthermore some correlation must be expected in the parameter domain, especially if the number of dimensions in the parameter domain is higher than two. Depending on the sampling parameters, this implies that an effective J (less than the number of samples in the parameter domain) must be used in Eq. 5.

3. AN EXAMPLE OF LINE DETECTION IN A VERY NOISY IMAGE

A noise free image containing eight lines with limited slope is created. The image, shown in Fig. 3, has 101×101 samples. The curve sampling functions are chosen to $\phi_m = l - 50$ and $\phi_n = p(l - 50) + \tau$ and L is set to 101. The offset is made in order to lower the sampling requirements in the parameter domain. The sampling distances in the parameter domain is set to $\Delta\tau = 1$ and $\Delta p = 0.01$. The line parameters are listed in Tab. 1.

No	p	τ	μ	No	p	τ	μ
1	0.10	60	1.5	5	0.00	30	1.0
2	0.25	50	-1.5	6	0.40	45	-1.0
3	-0.10	80	-1.0	7	-0.36	52	-1.0
4	-0.25	30	1.0	8	0.10	80	0.5

Table 1: Line parameters. p is the slope, τ is the offset, and μ is the curve amplitude.

To illustrate the potential of the GRT a very noisy image is generated, by adding Gaussian noise to the noise free image with zero mean and standard deviation $\sigma = 1$. It can be seen from Figure 4, that the lines are hard to identify. Choosing $P_{\text{det all}} = 0.95$, Eq. 5 gives $\lambda^* = 6.53$; i.e., only lines with $|\mu| > 0.65$ should be detectable. This implies that all but line number eight should be detectable. The absolute value of the parameter domain obtained by the use of the GRT to the noisy image is shown in Fig. 5.

Since the noisy image contains few lines with absolute curve amplitude $|\mu|$ being of the same order of magnitude as σ and has approximately zero mean, σ was estimated from the image using the ordinary central variance estimator, which gave $\hat{\sigma} = 1.04$. Setting $P_{\text{det all}}$ to 0.95, Eq. 5 results in $L\mu^* = 65.7$. This is used for thresholding of the parameter domain as shown in Figure 6. Seven of the eight line parameters are found despite the poor signal to noise ratio in the image. Note that some of the lines will be represented by a few neighbor parameter vectors. This error be can corrected by clustering neighbor parameter vectors. The error is due to the sampling of the parameter domain and the finite image size.

The theory predicted that only seven lines could be detected. The eighth line can be detected if the curve length can be increased or the noise variance can be reduced. If the theory is used with $P_{\text{det all}}$ very low, $L\mu^*$ get lower and noise peaks will appear in the parameter domain along with parameters of the eighth line. In Fig. 7 the threshold level has been reduced to, e.g., $0.7L\mu^* = 46.0$. As it can be seen, noise will now give parameter vectors which do not represent a curve. As seen from Fig. 8 a further reduction of the threshold level to, e.g., $0.5L\mu^* = 32.9$ gives a parameter domain, where all eight lines are present. Due to the noise level many false parameter vectors can also be observed.

4. CONCLUSION

A statistically based noise analysis of the generalized Radon transform has been presented, which was used to derive a threshold level in order to separate curve information and noise in the parameter domain. A numerical example was provided to illustrate the theory.

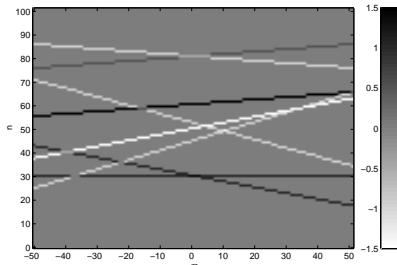


Figure 3: Noise free image with eight lines.

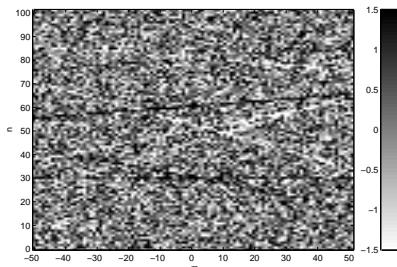


Figure 4: The same image contaminated with additive Gaussian noise.

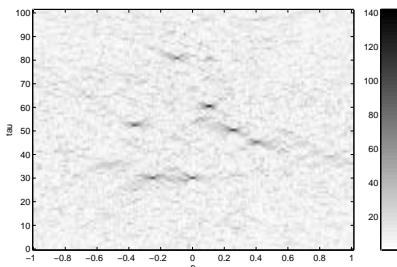


Figure 5: The absolute GRT of the noisy image. Note the peaks corresponding to the curves.

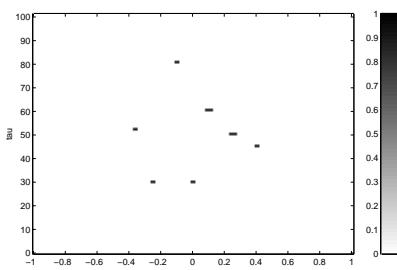


Figure 6: Threshold of the absolute GRT using the estimated threshold level.

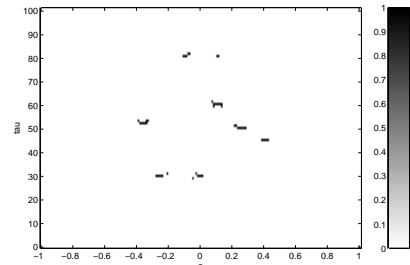


Figure 7: Threshold of the absolute GRT using 0.7 times the estimated threshold level.

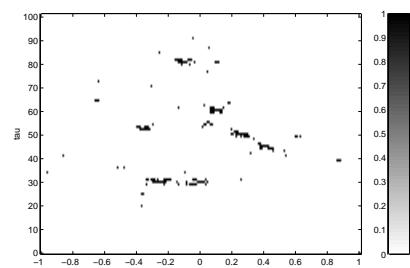


Figure 8: Threshold of the absolute GRT using 0.5 times the estimated threshold level.

5. REFERENCES

- [1] J. Illingworth. A Survey of the Hough Transform. *Computer Vision, Graphics, and Image Processing*, 44, 1988.
- [2] S. R. Deans. *The Radon Transform and Some of Its Applications*. Krieger Publishing Company, Malabar, Florida, 1993. Previously John Wiley & Sons Inc., 1983.
- [3] G. Beylkin. Discrete Radon Transform. *IEEE ASSP*, 35(2):162–172, 1987.
- [4] K. V. Hansen and P. A. Toft. Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform. Submitted for publication.
- [5] P. A. Toft and K. V. Hansen. Fast Radon Transform for Detection of Seismic Reflections. In *Signal Processing VII - Theories and Applications*, volume I, pages 229–232. EURASIP EUSIPCO94, 1994.

Appendix J

Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET

This appendix contains the paper *Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET*, by Søren Holm, Peter Toft, and Mikael Jensen. The work has been carried out at the National University Hospital in Copenhagen in 1995.

The results have been presented at the *IEEE Medical Imaging Conference 95* and this paper will appear in the conference issue [7]. The abstract can also be found in the abstract collection of the MIC 95.

A more extensive article has also been submitted, see Appendix K.

Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET

Søren Holm†, Peter Toft‡, and Mikael Jensen†

† Dept. of Nuclear Medicine, Rigshospitalet, National University Hospital,

Copenhagen, Denmark (sholm@pet.rh.dk).

‡ Electronics Institute, Technical University of Denmark (ptoft@ei.dtu.dk).

Abstract

This work determines the relative importance of noise from blank (B), transmission (T) and emission (E) scans in PET using a GE Advance scanner on a 20 cm cylinder, a brain phantom, and a torso-like ellipse (18/35 cm) with examples of human scans (brain O-15 water and F-18 FDG, heart FDG). Phantom E scans were acquired in both 2D and 3D modes as decay series with C-11 or F-18 over 3-6 decades of Noise Equivalent Counts (NEC). B and T scans were made using two pin sources (389+134 MBq) with times 16-32768 sec. In humans only a limited subset was available. In homogeneous phantoms normalized variance (var) was estimated from pixel distributions in single images. In other objects, including the human studies, calculations were performed on differences of paired images. In all cases a fit was made to a simple noise model. The cylinder data shows expected relations of T to B noise proving the adequacy of B scan times < 30 min for most purposes. For cylinder and brain phantom, contour plots are provided for var(E,T). In a typical 3D O-15 water study with 0.5M counts per central slice, a 10 min T-scan ensures that var(T) < 0.1 * var(E). Using 10 min T-scan for a static 3D FDG brain study of 10 min having 5M counts yields equal E and T variance contributions. In human body scans T noise has a relative larger importance and is often dominated by effects of line artefacts from (clusters of) zeros in the T-scan, not included in the simple model.

I. INTRODUCTION

The purpose of this work is to establish guidelines for the relative importance of the noise contributions from blank, transmission and emission scans in typical imaging situations in PET. Although much information can be deduced theoretically from existing knowledge of reconstruction algorithms [1, 2], we have mainly applied an experimental approach. The limitations in generality caused by the obvious difficulty of varying all of the many possible parameters of acquisition and reconstruction are outweighed by the advantage of the close reproduction of actual scan

setups, and direct testing in some human studies. The following paragraph describes the model used for fitting to the observed data.

II. THEORY

Due to the linearity of the reconstruction process the PET images (volumes) x_i , can be described as a weighted sum of the sinogram values s_j , with weight factors ϕ_{ij} corresponding to the actual (linear) reconstruction algorithm

$$x_i = \sum_j \phi_{ij} s_j \quad (1)$$

where the actual image pixel (voxel) is denoted by i , and j denotes the pixel position in the sinogram. In order to compensate for attenuation the sinogram used for reconstruction is calculated from an emission sinogram e_j , a transmission sinogram t_j , and a blank scan b_j . The calculation basically amounts to:

$$s_j = \frac{e_j b_j}{t_j} \quad (2)$$

although this is usually qualified by a filtering of the factor $\frac{b_j}{t_j}$.

All three types of sinograms in principle inherit their statistical properties from the Poisson statistics of the radioactive decay. A number of corrections that are performed before the stage of reconstruction described here, however, adds to the raw counts' noise. The overall effect of correction for randoms and scatter can be described as a decrease in effective counts [3], the resulting figure being widely known as Noise Equivalent Counts (NEC). A practical formulation used in this work is:

$$NEC = T \frac{(1 - SF)^2}{1 + 2f \frac{R}{T}} \quad (3)$$

where T is true counts (including scatter), R is random counts from the full field-of-view, f is the fraction of the sinogram covered by the object under investigation, and SF is the scatter fraction. In all applications of NEC in this study, the value of SF has been set to zero. Within this paper NEC per slice is used as a parameter.

A. The Noise Model

In the simplest case where the image noise is estimated from a single image set, using a Taylor expansion and disregarding noise correlation between pixels implies that the pixel noise variance $V\{x_i\}$ can be estimated by

$$\begin{aligned} V\{x_i\} &\simeq \sum_j \phi_{ij}^2 V\left\{\frac{e_j b_j}{t_j}\right\} \\ &\simeq \sum_j \phi_{ij}^2 E^2\{s_j\} \left[\frac{V\{e_j\}}{E^2\{e_j\}} + \frac{V\{b_j\}}{E^2\{b_j\}} + \frac{V\{t_j\}}{E^2\{t_j\}} \right] \end{aligned} \quad (4)$$

Due to the Poisson statistics the variance in the emission sinogram relative to the squared mean value $\frac{V\{e_j\}}{E^2\{e_j\}}$ is inversely proportional to the mean $E\{e_j\}$, i.e., inversely proportional to number of counts NEC in the emission sinogram. Similar arguments can be used for the transmission and blank scan terms, hence the total pixel variance normalized with the mean of the image can be modelled by

$$\frac{V\{x_i\}}{E^2\{x_i\}} = \frac{a}{NEC} + b + \frac{c}{T_t} + \frac{d}{T_b} \quad (5)$$

where T_t is the transmission scan time in seconds, T_b the blank scan time in seconds (for the given geometry and source strength). In the model a base term b is included to accommodate effects in the reconstruction process, not explained by the other terms, e.g., varying sensitivity of the detectors, not properly corrected for by normalization.

In this paper all measured data have been fitted to the model shown in Equation 5. In cases with heterogeneous ROI's the variance was measured for differences between equivalent and independent images [6]. In these cases the measured normalized variance has been divided by two so estimated noise parameters are directly comparable.

III. METHODS

Measurements were made on the GE Advance scanner [8, 9] with 3D acquisition and reconstruction capability. Its 9 GByte raw data disk can hold about 180 3D frames (byte mode, separate prompts and delayed), convenient for phantom decay studies. Reconstruction in our configuration takes approximately 8 minutes per frame. The scanner applies two pin sources for blank and transmission scanning. During the initial experiments the pin source activities were 389 MBq and 134 MBq, respectively, and the blank scan sinogram count rate was 0.91 Mcps. On later phantom and human studies, blank and transmission scan times have been scaled in accordance with the currently observed blank scan count rates and the values quoted in seconds (powers of 2) are therefore assumed to be directly comparable.

A. Phantom studies

Three different water-filled phantoms have been used: the 20 cm standard (NEMA) cylinder, a torso-like ellipse with

axes 18 and 35 cm, and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm.

Using initially the two homogeneous phantoms in accordance with Equation 5, emission scans were made in both 2D and 3D as decay series with F-18 starting at rates well below saturation of the scanner. In 2D a total of 22 measurements were performed starting with a half-life of F-18 and then reducing the time down to ..., 4, 2, 1 seconds, effectively reducing the number of counts by a factor of 2 for each acquisition. In 3D the dataset was limited to 10 time frames. Transmission scans were made starting from $T_t=16$ seconds (64 on the ellipse) and doubling the time for each step up to 32768 seconds. Blank scans were made from $T_b=16$ to 4096 seconds and complemented with one 100000 seconds scan.

Reconstructions were made for all relevant combinations of emission and transmission scans using the 2048 seconds blank scan. All emission scans were further reconstructed with a calculated attenuation correction (CAC) using a circular or elliptical contour respectively. For the 2D cylinder case the emission image with highest count was reconstructed with the longest transmission scan and all blank scans. The cylinder was reconstructed in a 128^2 matrix with pixel size 2.0 mm, and a Hann filter (designated '4 mm' corresponding to the approximate resulting resolution FWHM). For the ellipse, a 128^2 matrix of pixel size 4.0 mm was used with a Hann '8 mm' filter. The 3D reconstruction further contains an axial filtering (Hann) set to its minimum value of 8.5 mm. The attenuation data were preprocessed with a Gaussian filter, 8 mm for the cylinder case, and 10 mm for the ellipse. All available corrections were applied, including detector normalization, randoms subtraction, scatter correction, (slice) sensitivity and absolute calibration, and decay correction to scan start. In all slices of all reconstructed images (order of 25,000 images) the mean and variance was calculated from an ROI extending 70% of the diameter or ellipse axes. These ROI-data were imported to MATLAB for further analysis. In 2D the data were averaged over 31 (of 35) slices avoiding edge effects but ignoring the minor differences between direct and cross slices. In 3D only the central 15 slices having almost identical noise were included in the average. For each reconstructed dataset the noise was represented by the normalized variance, i.e., $\text{variance}/\text{mean}^2$, in subsequent plots and fitting. From the total rates curves acquired with the images a Noise Equivalent Count (NEC) value per slice was calculated for each scan frame. The rate dependent correction between trues and NECs due to randoms was in all these cases below 15% except in the first 3D cylinder frame where it amounted to 25%.

The brain phantom has two separate chambers ('grey' and 'white' matter, respectively). It was filled to resemble the usually quoted ratio of 4:1 between these two substances for flow or metabolism. During scanning the phantom was embedded in a Rando-Alderson whole body phantom (normally used for radiotherapy dosimetry purposes) from which a few slices of the head had been re-

moved. Also the thorax was replaced by the body ellipse phantom containing 5 times the total activity in the brain to provide more natural scatter and randoms conditions than the brain phantom alone. Two different sets of measurements have been performed. One set (repeated in 2D and 3D) was originally designed to address count rate performance [4]. The phantoms were loaded with C-11 Carbonate well above the expected saturation limit of the scanner and pairs of (60:62) seconds scans were performed every 10 minutes for about 5 hours (15 half lives). The relatively short half life of C-11 and the intention of having not only similar counts but also count rates in each of the two paired scans limited the maximum number of slice counts observed to 10^6 which would not allow a sufficiently clear distinction between the transmission scan times. One more decay series therefore was prepared in 3D with F-18 starting below maximum counts, and measuring for 12 half lives utilizing all the time for recording. Each of the half lives were split in two frames with a ratio of (0.415:0.585) yielding approximately equal counts. The C-11 series were reconstructed using an 'infinite' (32768 second) transmission scan. For the F-18 series, a set of 7 pairs of transmission scans were obtained (32,64,128,256,1024,4096,16384) seconds. Reconstructions were made to provide datasets with independent transmission and emission noise by combining the E-T frames as odd-odd, even-even. Reconstruction parameters for the brain phantom were identical to those for the cylinder except that the axial Hann filter was here replaced by a ramp filter. From both the C-11 series and the F-18 series, difference images were calculated, whole brain ROI's were placed in the original as well as the difference images for mean and standard deviation calculation respectively.

B. Human studies

For one person (case KL) included in a count rate performance and dose optimization study with O-15 [4] the usual 90 second integration time (starting 20 seconds after bolus injection) was supplemented by 2 short frames (20-22 seconds). The study comprised injections with 25-800 MBq in 2D and 50-800 MBq in 3D. Reconstruction and analysis was performed as above, although only with one 12 minutes transmission scan. The number of NECs in the short scans were about 16% of the corresponding 90 seconds frames normally used.

One person (case MN) injected with 240 MBq of FDG had a double set of emission scans (8-64 sec in 2D and 8-512 sec in 3D) of the brain starting 2 hours after injection, followed by a double set of transmission scans (32-512 sec) yielding a total of 35 data points in (E,T). Reconstruction and data analysis matched the brain phantom F-18 series.

One patient (case BB) undergoing a dynamic FDG scanning of the heart additionally had a series of transmission scans, (0.5,1,2,4,8,20) minutes. Pairs of emission scans (0.5,1,2,5 minutes) from different parts of the dynamic study were reconstructed with all the transmission scans

with subsequent data analysis in a large body circumferential ROI.

IV. RESULTS

In this section data and parameters are presented for the previously described experiments and compared to the model.

A. The Cylinder and Elliptical Body phantom

Estimated parameters for the Cylinder phantom in 2D and 3D modes, and the Elliptical Body phantom in 2D and 3D are given in table 1 and 2.

Table 1: Cylinder data 2D and 3D

	Cylinder 2D	Cylinder 3D
a [M Counts]	4.77E-2	2.70E-2
b []	1.88E-3	3.49E-4
c [sec]	6.40	1.64
d [sec]	0.96	
Min(NEC) [M-Counts]	0.43	1.48
Max(NEC) [M-Counts]	13.00	38.75
Min(T_t) [sec]	64	256
Max(T_t) [sec]	32768	32768
Min($\frac{V}{E^2}$)	5.60E-3	1.10E-3
Max($\frac{V}{E^2}$)	1.60E-1	2.46E-2
Max(Error)	1.26E-4	3.02E-6

Table 2: Elliptical body phantom data 2D and 3D

	Ellipse 2D	Ellipse 3D
a [M Counts]	2.46E-2	3.38E-2
b []	1.67E-3	1.54E-3
c [sec]	8.85	4.16
Min(NEC) [M-Counts]	0.17	2.50
Max(NEC) [M-Counts]	9.60	27.83
Min(T_t) [sec]	1024	1024
Max(T_t) [sec]	16384	32768
Min($\frac{V}{E^2}$)	4.01E-3	2.79E-3
Max($\frac{V}{E^2}$)	1.56E-1	1.94E-2
Max(Error)	4.77E-5	9.40E-6

In the following two Figures of the normalized variance are shown as a function of NEC and transmission scan length T_t . Figure 1 corresponds to the Cylinder (2D) and Figure 2 to the Cylinder (3D).

Also in the Brain phantom (3D) case and in case MN both the emission and transmission parameters have been estimated. The parameters are listed in Table 3 together with the range of the fit. Figure 3 shows the normalized variance for the Brain phantom and Figure 4 demonstrates that the model fits the measured data nicely.

To illustrate how well the emission data measured on brains agree with the phantom studies, Figure 5 shows case KL (2D/3D), case MN (2D/3D), and the Brain Phantom (2D/3D). In all cases data corresponding to only one transmission scan are shown.

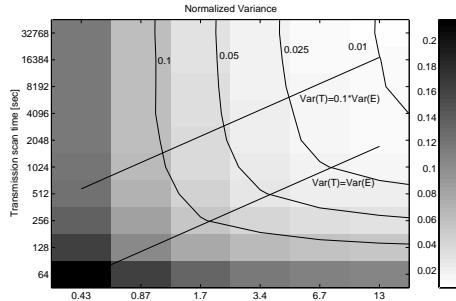


Figure 1: Normalized variance of Cylinder (2D) as a function of NEC per slice and duration of transmission scan T_t . Contours show iso-noise curves with levels indicated. Lines show where the ratio of transmission to emission noise terms is unity and 10% respectively.

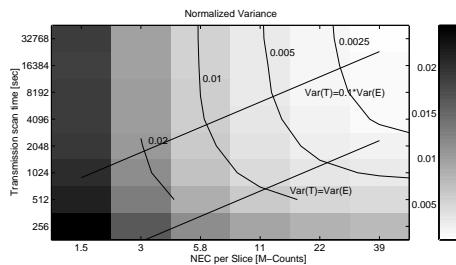


Figure 2: Normalized variance Cylinder (3D) as a function of NEC per slice and duration of transmission scan T_t .

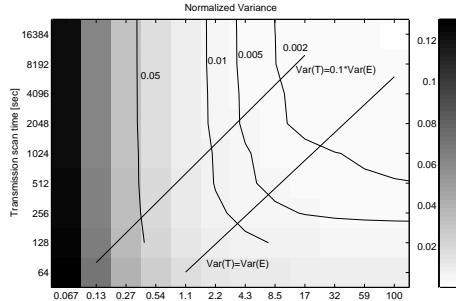


Figure 3: Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t .

Table 3: Brain phantom data 3D and Case MN 3D

	Brain Ph.	Case MN
a [M Counts]	1.67E-2	1.85E-2
c [sec]	1.01	1.59
Min(NEC) [M-Counts]	6.7E-2	6.4E-1
Max(NEC) [M-Counts]	2.2	3.6
Min(T_t) [sec]	64	64
Max(T_t) [sec]	16384	512

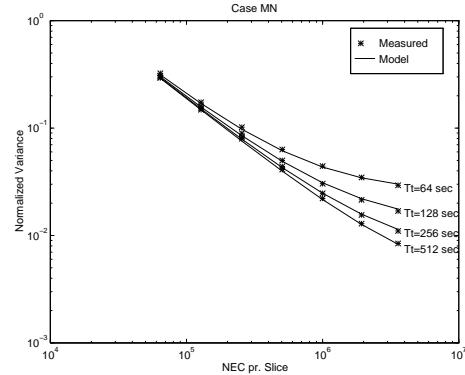


Figure 4: Normalized variance of case MN (3D) as a function of NEC per slice and duration of transmission scan T_t . Stars show the measured data and lines the fitted model.

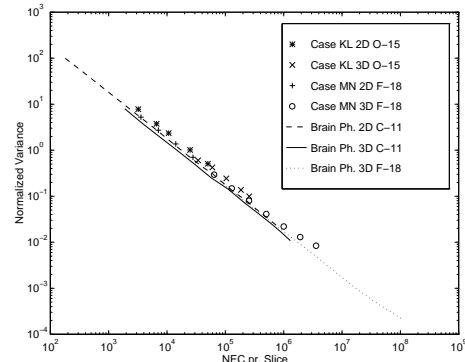


Figure 5: Normalized variance of Case KL (2D/3D), case MN (2D/3D), and Brain phantom (2D/3D) as a function of NEC per slice.

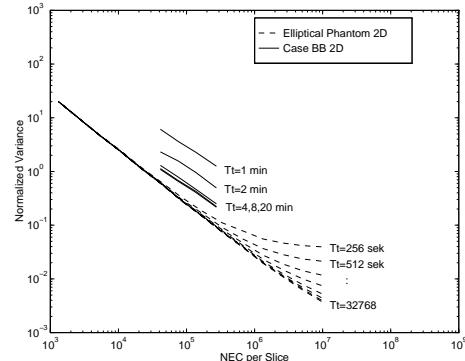


Figure 6: Normalized variance of body studies case BB (2D) and Elliptical phantom (2D) as a function of NEC per slice. Note that case BB (1,2 and 4 min) shows a different noise behavior.

Figure 6 shows body data. The 2D Elliptical Body phantom follows the previous noise model, but in case BB (large patient with arms in FOV) the noise curves for the 1 and 2 minutes transmission scans are shifted upwards. Inspection of the reconstructed images reveals large number of lines, due to (clusters of) zeros in the transmission scan.

V. DISCUSSION

The general noise properties of reconstructive tomography are well described in the literature [1, 2]. Previous descriptions of the Advance scanner also include count rate dependence in phantoms and humans [8, 4] but the noise characterization so far [5] focused on the emission noise. In this paper we have determined the relative importance of the noise contributions from blank, transmission, and emission scans for a range of imaging situations encountered.

As shown, a good fit to the empirical variance is found both in 2D and 3D studies. The numerical values of the measure chosen for the noise examination is strongly dependent on reconstruction parameters, in part because of the neighbor pixel correlation disregarded in the theory section. Most parameters, however, will affect the Emission and Transmission contributions in the same way (through ϕ_{ij} only), and therefore the model and its output is considered adequate for the purpose of identifying areas in the E-T plane where one source is dominating.

From Table 1 it is seen that the blank scan contribution to the noise as expected is only a small fraction of the transmission for same duration, the ratio being well explained by the average attenuation of the (central part of) the 20 cm Cylinder phantom. For all practical purposes, therefore, the application of a 20 minutes blank scan will ensure that the blank scan contribution is negligible since the cases where a longer transmission scan might be applied are those with a higher attenuation.

A typical 3D brain activation study with O-15 water in our setup will contain 0.5 Mcounts/central slice, Figure 5[4]. With a 10 minutes transmission scan, the transmission variance calculated from the phantom data in Table 3 is 1.67E-3 compared to the emission contribution of 3.34E-2, i.e., the transmission adds (only) 5% to the final result. For the human case MN the corresponding figures would be 2.65E-3 (transmission), 3.7E-2 (emission) and 7%, respectively. A 10 minutes, 2D FDG brain scan typically would also have 0.5-1 M counts yielding the same 5-10% ratio, while acquiring the same data in 3D would make the two contributions almost equal and therefore call for a more detailed analysis. It should be noted that in this imaging condition no attribute has been made to the additional noise from the emission correction of the transmission scan which is necessary if the transmission is performed with activity present.

Figure 6 suggests that within the emission count range

of a typical dynamic heart FDG scan, the transmission noise contribution (from a 20 minutes transmission scan) is not dominating. Data are, however, not available that can demonstrate the region in which the transmission noise curves split up. It should be emphasized that the observed shift of the short-time curves is due to line-artefacts caused by (clusters of) zeros in the transmission sinogram. This deteriorating effect is usually more important and calls for improved method of attenuation correction, e.g., by using image segmentation as described in [7].

REFERENCES

- [1] A. E. Todd-Pokropek and P. H. Jarrit. The noise characteristics of SPECT systems. In P. J. Ell and B. L. Holman, editor, *Computed Emission Tomography*, pages 361–389. Oxford University Press, Oxford, 1982.
- [2] N. M. Alpert et al. Estimation of the Local Statistical Noise in Emission Computed Tomography. *IEEE Trans. Med. Imag.*, MI-1(2):142–146, 1982.
- [3] S. C. Strother et al. Measuring PET Scanner Sensitivity: Relating Count Rates to Image Signal-to-Noise Ratios Using Noise Equivalent Counts. *IEEE Trans. Nucl. Sci.*, NS-37(2):783–788, April 1990.
- [4] S. Holm et al. 3D PET activation studies with $H_2^{15}O$ bolus injection. Count rate performance and dose optimization. In *Quantification of Brain Function using PET*. Academic Press, San Diego, 1995. In press.
- [5] S. Pajacic et al. Noise Properties of 3D PET Images. *J. Nucl. Med.*, 36:p. 105 (abstract), 1995.
- [6] S. R. Cherry et al. Improved Detection of Focal Cerebral Blood Flow Changes Using Three-Dimensional Positron Emission Tomography. *J. Cereb Blood Flow Metab.*, (4):630–638, 1993.
- [7] S. R. Meikle et al. Attenuation Correction Using Count-Limited Transmission Data in Positron Emission Tomography. *J. Nucl. Med.*, 34:143–150, 1993.
- [8] T. K. Lewellen et al. Investigations of the Count Rate Performance of General Electric Advance Positron Emission Tomograph. *IEEE Trans. Nucl. Sci.*, NS-42(4):1051–1057, August 1995.
- [9] T. R. DeGrado et al. Performance Characteristics of a Whole-Body PET Scanner. *J. Nucl. Med.*, 35(8):1398–1406, 1994.

Appendix K

Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET

This appendix contains the paper *Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET*, by Søren Holm, Peter Toft, and Mikael Jensen. The work has been carried out at the National University Hospital in Copenhagen in 1995.

The results have been presented at the *IEEE Medical Imaging Conference 95* and this paper has been submitted for publication in a conference issue of the IEEE Transactions on Nuclear Science [8].

A subset of this paper appears in [7], which is shown in Appendix J.

Estimation of the Noise Contributions from Blank, Transmission and Emission Scans in PET

Søren Holm†, Peter Toft‡, and Mikael Jensen†

† Dept. of Nuclear Medicine, Rigshospitalet, National University Hospital,
Copenhagen, Denmark (sholm@pet.rh.dk).

‡Department of Mathematical Modelling, Technical University of Denmark (ptoft@ei.dtu.dk).

Abstract

This work determines the relative importance of noise from blank (B), transmission (T) and emission (E) scans in PET using a GE Advance scanner on a 20 cm cylinder, a brain phantom, and a torso-like ellipse (18/35 cm) with examples of human scans (brain O-15 water and F-18 FDG, heart FDG). Phantom E scans were acquired in both 2D and 3D modes as decay series with C-11 or F-18 over 3-6 decades of Noise Equivalent Counts (NEC). B and T scans were made using two pin sources (≈ 500 MBq total) over 64-32768 sec. In humans only a limited subset was available. In homogeneous phantoms normalized variance (var) was estimated from pixel distributions in single images. In other objects, including the human studies, calculations were performed on differences of paired images. In all cases a fit was made to a simple noise model. The cylinder data show expected relations of T to B noise proving the adequacy of B scan times ≤ 20 min for most purposes. For the brain phantom, a contour plot is provided for var(E,T). In a typical 3D O-15 water study with 0.5M counts per central slice, a 10 min T-scan adds less than 10% to the total noise level. An example shows how to split a total scan time between E and T scans, in order to minimize the variance.

I. INTRODUCTION

PET scans require correction for attenuation in order to be quantitative. Even in brain activation studies where absolute units are often replaced by relative values (i.e., normalized to a mean value), inter-individual comparisons still assume that values across an image reflect the local tracer concentration. Corrections can be applied, e.g., by assuming uniform values of attenuation within geometrically defined boundaries or in segmented areas of the image, but a more accurate description of the attenuation requires measurement by a transmission scan. Unfortunately this also adds noise to the image and adds to the total procedure time. In some cases, mainly in body scanning, the increase in noise is immediately noticed and may even be so high that quantification must be sacrificed in order to provide a reasonable visual impression. However,

in other situations, effects that are invisible for the human observer might still be of importance when examining the subtle differences of brain activation by statistical methods. The purpose of this work is to establish guidelines for the relative importance of the noise contributions from blank, transmission, and emission scans in typical imaging situations. Although much information can be deduced theoretically from existing knowledge of reconstruction algorithms [1, 2, 3] we have here applied an experimental approach. The limitations in generality caused by the obvious difficulty of varying all of the many possible parameters of acquisition and reconstruction are outweighed by the advantage of the close reproduction of actual scan setups, including some human studies. The following Section describes the model used for fitting to the observed data. It should not be considered as a rigorous derivation; it merely lists the different structures of the datasets included in the study comprising homogeneous as well as inhomogeneous objects.

II. THEORY

Due to the linearity of the reconstruction process the PET images (volumes) x_i , can be described as a weighted sum of the sinogram values s_j , with weight factors ϕ_{ij} corresponding to the actual (linear) reconstruction algorithm

$$x_i = \sum_j \phi_{ij} s_j \quad (1)$$

where the actual image pixel (voxel) is denoted by i , and j denotes the pixel position in the sinogram. In order to compensate for attenuation the sinogram used for reconstruction is calculated from an emission sinogram e_j , a transmission sinogram t_j , and a blank scan b_j . The calculation basically amounts to:

$$s_j = \frac{e_j b_j}{t_j} \quad (2)$$

although usually qualified by a filtering of the factor $\frac{b_j}{t_j}$.

All three types of sinograms in principle inherit their statistical properties from the Poisson statistics of the radioactive decay. A number of corrections that are performed before the stage of reconstruction described here,

however, adds to the raw counts' noise. The overall effect of correction for randoms and scatter can be described as a decrease in effective counts [4, 3], the resulting figure being widely known as Noise Equivalent Counts (NEC). A practical formulation used in this work is:

$$\text{NEC} = \mathcal{T} \frac{(1 - \text{SF})^2}{1 + 2f \frac{\mathcal{R}}{\mathcal{T}}} \quad (3)$$

where \mathcal{T} is true counts (including scatter), \mathcal{R} is random counts from the full field-of-view, f is the fraction of the sinogram covered by the object under investigation, and SF is the scatter fraction. In all applications of NEC in this study, the value of SF has been set to zero.

While NEC as a global measure may be adequate for the noise description in the case of homogeneous phantoms extending the full axial field-of-view (AFOV), objects of limited extension may be better characterized by a value per slice. In the following, all NEC values quoted correspond to a single slice.

A. One Emission and One Transmission Scan

In the simplest case where the image noise is estimated from a single image set, using a Taylor expansion and assuming no noise correlation between pixels implies that the pixel noise variance $V\{x_i\}$ can be estimated by

$$\begin{aligned} V\{x_i\} &\simeq \sum_j \phi_{ij}^2 V\left\{\frac{e_j b_j}{t_j}\right\} \\ &\simeq \sum_j \phi_{ij}^2 E^2\{s_j\} \left(\frac{V\{e_j\}}{E^2\{e_j\}} + \frac{V\{b_j\}}{E^2\{b_j\}} + \frac{V\{t_j\}}{E^2\{t_j\}} \right) \end{aligned} \quad (4)$$

Due to the Poisson statistics the variance in the emission sinogram relative to the squared mean value $\frac{V\{e_j\}}{E^2\{e_j\}}$ is inversely proportional to the mean of an infinite amount of experiments $E\{e_j\}$, i.e., inversely proportional to number of counts NEC in the emission sinogram. Similar arguments can be used for the transmission and blank scan terms, hence the total pixel variance normalized with the mean of the image can be modelled by

$$\frac{V\{x_i\}}{E^2\{x_i\}} = \frac{a}{\text{NEC}} + b + \frac{c}{T_t} + \frac{d}{T_b} \quad (5)$$

where T_t is the transmission scan time in seconds, T_b the blank scan time in seconds (for the given geometry and source strength). In the model a base term b is included to accommodate effects in the reconstruction process, not explained by the other terms, e.g., varying sensitivity of the detectors, not properly corrected for by normalization. For fixed values of (either) T_t or T_b the corresponding terms may also be thought of as part of this constant term. The model applies to regions with limited structural variation, hence measuring variance in a heterogeneous object like the brain or the thorax is problematic.

B. Two Emission Scans

To overcome the problem with heterogeneous structures the difference between two PET images, corresponding to two equivalent but independent emission scans $e_j^{(1)}$ and $e_j^{(2)}$, can be calculated. Due to Equations 1 and 2 the difference image is given by

$$x_i = \sum_j \phi_{ij} \left(e_j^{(2)} - e_j^{(1)} \right) \frac{b_j}{t_j} \quad (6)$$

Using a Taylor expansion, and using that the two emission scan have the same mean values $E\{e_j\}$ and variance $V\{e_j\}$ implies that

$$x_i \simeq \sum_j \phi_{ij} \left(\left(e_j^{(2)} - E\{e_j\} \right) - \left(e_j^{(1)} - E\{e_j\} \right) \right) \frac{E\{b_j\}}{E\{t_j\}} \quad (7)$$

Thus the variance $V\{x_i\}$ is given by

$$V_{\text{two e-scans}}\{x_i\} \simeq 2 \sum_j \phi_{ij}^2 E^2\{s_j\} \frac{V\{e_j\}}{E^2\{e_j\}} \quad (8)$$

Normalized with the squared mean of the heterogeneous structure and using the same definition of constants as in Equation 5 gives that

$$\frac{1}{2} \frac{V_{\text{two e-scans}}\{x_i\}}{E^2\{x_i\}} \simeq \frac{a}{\text{NEC}} \quad (9)$$

Thus only the emission term of the noise variance can be estimated from two emission scan and one transmission scan. Since all structural information is removed by using differences, no constant term (b) can appear.

If instead two transmission scans (and one emission scan) are available, it is easy to show that two times the transmission variance can be estimated (and only that) from a normalized difference image.

C. Two Emission and Two Transmission Scans

Using the same Taylor technique with two sets of emission scans and two transmission scans, it can be shown that

$$\frac{1}{2} \frac{V_{\text{two e and t-scans}}\{x_i\}}{E^2\{x_i\}} \simeq \frac{a}{\text{NEC}} + \frac{c}{T_t} \quad (10)$$

This implies that both the emission and transmission term can be estimated in the inhomogenous case if a double set of independent and equivalent emission and transmission scans are measured and reconstructed pairwise. Note that any normalized variance shown in the following has been divided by two if measured from two scans, cf. Equations 5, 9 and 10.

D. Zeroes in the Transmission Sinogram

A common problem especially in body scanning is zeros in the transmission sinogram. One option is to replace

the zeros with a small number T_0 , so the attenuation correction can be done cf. Equation 2. Although overflow is avoided this may imply that reconstructed images contain strong lines causing a significant change in the noise structure. Let \mathcal{Z} be the set of j , where the measured t_j is zero. Assume that the variance of the blank scan is negligible and further that only the terms with zeros effectively contribute to the noise. In this case

$$\begin{aligned} V\{x_i\} &\simeq \sum_{j \in \mathcal{Z}} \phi_{ij}^2 V\left\{\frac{e_j b_j}{T_0}\right\} \\ &\simeq \sum_{j \in \mathcal{Z}} \phi_{ij}^2 V\{e_j\} \frac{E^2\{b_j\}}{T_0^2} \end{aligned} \quad (11)$$

This indicates that the number of zeros (through the sum) and the variance on the emission scan determines the noise level. Hence the variance normalized with the squared mean is approximately proportional to the number of zeros and inversely proportional to NEC. In the limit where this kind of noise is dominating, however, the reconstructed images consist of many lines and become useless anyway.

E. Limitations

Using either of subsections A, B or C to estimate the noise terms, we estimate the noise variance from a chosen ROI and normalize with the squared mean of the ROI. Due to the global features in the reconstruction process the noise tends to have only a very small correlation with the underlying structure and often it is approximately evenly distributed across the ROI's used. This implies that the normalized noise variance of the ROI depends strongly on the selected ROI, through the squared mean.

III. METHODS

Measurements were made on the GE Advance scanner [5, 6, 7] with 3D acquisition and reconstruction capability. Its 9 GByte raw data disk can hold about 180 3D frames (byte mode, separate prompts and delayed), suitable for phantom decay studies. Reconstruction in our configuration (with 10 i860 processors) takes approximately 8 minutes per frame [8]. For blank and transmission scanning, which is always performed in 2D, the scanner applies two pin sources. During the initial experiments the pin source activities were 389 MBq and 134 MBq, respectively. The blank scan sinogram count rate was 0.91 Mcps, corresponding to an average count rate per sinogram element of ≈ 0.23 cps in the center. On later phantom and human studies, blank and transmission scan times have been scaled in accordance with the currently observed blank scan count rates and the values quoted in seconds are therefore directly comparable.

A. Phantom studies

Three different water-filled phantoms have been used: the 20 cm standard (NEMA) cylinder, a torso-like ellipse with axes 18 and 35 cm, and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm.

Using initially the two homogeneous phantoms in accordance with Equation 5, emission scans were made as decay series with F-18 in 2D (total of 22 measurements) and 3D (limited to 10 time frames). Transmission scans were made starting from $T_t=64$ seconds and doubling the time for each step up to 32768 seconds. Blank scans were made from $T_b=64$ to 4096 seconds and complemented with one 100000 seconds scan.

Reconstructions were made for all relevant combinations of emission and transmission scans using the 2048 seconds blank scan. All emission scans were further reconstructed with a calculated attenuation correction (CAC) using a circular or elliptical contour respectively. For the 2D cylinder case the emission image with highest count was reconstructed with the longest transmission scan and all blank scans. The cylinder was reconstructed in a 128^2 matrix with pixel size 2.0 mm, and a 4 mm Hann filter. For the ellipse, a 128^2 matrix of pixel size 4.0 mm was used with a 8 mm Hann filter. The 3D axial filter (Hann) was also set to its minimum value of 8.5 mm. The attenuation data were preprocessed with a 8 mm Gaussian filter for the cylinder case, and 10 mm for the ellipse. All available corrections were applied, including detector normalization, randoms subtraction, and scatter correction. In all slices of all reconstructed images (order of 25,000 images) the mean and variance was calculated from an ROI extending 70% of the diameter or ellipse axes. In 2D, the data were averaged over 31 (of 35) slices avoiding edge effects but ignoring the minor differences between direct and cross slices. In 3D only the central 15 slices having almost identical noise were included in the average. For each reconstructed dataset the noise was represented by the normalized variance, i.e., variance/mean², in subsequent plots and fitting. From the total rates curves a Noise Equivalent Count (NEC) value per slice was calculated for each scan frame. The rate dependent correction between trues and NECs due to randoms did not exceed 15%.

The brain phantom has two separate chambers ('grey' and 'white' matter, respectively). It was filled to resemble the usually quoted ratio of 4:1 between these two substances for flow or metabolism. Two different sets of measurements have been performed. One set (repeated in 2D and 3D) was originally designed to address count rate performance [9, 10]. The phantoms were loaded with C-11 carbonate well above the expected saturation limit of the scanner and pairs of (60:62) seconds scans were performed every 10 minutes for about 5 hours (15 half lives). The maximum number of slice counts observed in this study was limited to 10^6 which would not allow a sufficiently clear distinction between the transmission scan

times. One more decay series was therefore prepared in 3D with F-18 measuring for 12 half lives, each split in two frames with a ratio of (0.415:0.585) yielding approximately equal NEC. The C-11 series were reconstructed using an 'infinite' (32768 sec) transmission scan. For the F-18 series, a set of 6 pairs of transmission scans, 64 - 16384 sec, were obtained. Reconstructions were made to provide datasets with independent transmission and emission noise according to Equation 10 by combining the E-T frames as odd-odd, even-even. Reconstruction parameters for the brain phantom were identical to those for the cylinder except that the axial Hann filter was replaced by a Ramp filter. From both the C-11 series and the F-18 series difference images were calculated, and ROI's placed over grey matter (central and peripheral), white matter, and whole brain in the original as well as the difference images for mean and standard deviation calculation respectively.

B. Human studies

For one person (case KL) included in a count rate performance and dose optimization study with O-15 [9], the usual 90 second integration time (starting 20 seconds after bolus injection) was supplemented by 2 short frames (20-22 sec). The study comprised of injections with 25-800 MBq in 2D and 50-800 MBq in 3D. Reconstruction and analysis was performed as above, although only with one 12 minutes transmission scan. The number of NECs in the short scans were about 16% of the corresponding 90 seconds frames normally used.

One person (case MN) injected with 240 MBq of FDG had a double set of emission scans (8-512 sec) of the brain starting 2 hours after injection, followed by a double set of transmission scans (64-512 sec) yielding a total of 28 data points in (E,T)-plane. Reconstruction and data analysis was made to match the brain phantom F-18 series.

One patient (case BB) undergoing a dynamic FDG scanning of the heart additionally had a series of 6 transmission scans (0.5 - 20 min). Pairs of emission scans (0.5,1,2,5 min) from different parts of the dynamic study were reconstructed with all the transmission scans, and subsequent data analysis made as described above (Equation 9) in a large body circumferential ROI.

C. Fitting data

From the emission data (see subsections in section II.) the parameter a was estimated by linear regression in a log-log diagram of the normalized variance using the longest transmission scan available. Depending on the setup the transmission parameter c (and the base term b) was subsequently estimated by subtracting the estimated emission part of the normalized variance, a/NEC , and again applying linear regression. This approach is very simple and was considered adequate for the purpose. Non-linear data fitting was also tried and found to give similar results.

IV. RESULTS

In this section data and parameters are presented for the previously described experiments and compared to the model. Again it should be noted that any variance shown has been divided by two if measured from paired difference scans so that the results presented are comparable and represent noise in the single images.

Estimated model parameters from the Cylinder phantom in 2D and 3D modes, the Elliptical Body phantom in 2D and 3D, The Brain Phantom, and Case MN, are given in Table 1. This table give parameters valid for any practically used value of NEC and T_t .

Table 1 Estimated model constants.

Case	Mode	a [k counts]	b []	c [sec]
Cylinder	2D	47.7	0.00188	6.40
Cylinder	3D	27.0	0.00035	1.64
Ellipse	2D	24.6	0.00167	8.85
Ellipse	3D	33.8	0.00154	4.16
Brain Ph.	3D	16.7		1.01
Case MN	3D	18.5		1.59

In Figure 1 the measured normalized variance in the Cylinder (2D) case is shown. It can be seen that the transmission scans will effectively start to add to the variance if NEC is larger than approximately 10^5 counts per slice and that they - for transmission scan lengths normally encountered - are dominating at 10^7 . Note also, that the emission noise model as judged from the linearity in the log-log plot remains valid down to the level where the noise exceeds the mean by an order of magnitude, i.e., beyond any practical application of the images as such.

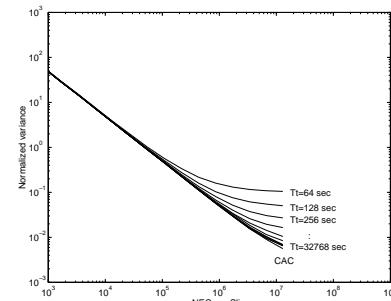


Figure 1 Normalized variance from 2D PET scan of the 20 cm Cylinder phantom as a function of NEC per slice and duration of transmission scan T_t . CAC means calculated attenuation correction.

In the Cylinder (2D) case the blank scan parameter d has also been estimated from reconstructed images corresponding to varying blank scan length T_b , and the estimated value is $d = 0.96$. By comparison with the parameter c from Table 1 the transmission scan in the cylinder case (2D) is seen to contribute approximately 6.7 times more to the variance than a blank scan with the same duration,

in accordance with the average attenuation of the central region of a 20 cm waterfilled phantom.

In Figure 2, corresponding to the Brain phantom scanned in 3D, a contour plot shows the normalized variance as a function of NEC and transmission length T_t . Shading is used to indicate the actual variance, which can be seen from the rightmost grey scale bar. Contours show approximate equidistant level of noise. Furthermore two lines are inserted to show where the transmission term equals 10% and 100% respectively of the emission term.

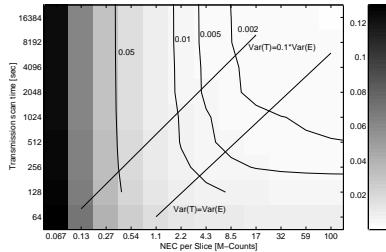


Figure 2 Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t . Contours show iso-noise curves with levels indicated. Lines show where the ratio of transmission to emission noise terms is unity and 10% respectively.

In the Brain phantom (3D) case and in case MN two emission and two transmission scans were measured, thus both emission and transmission parameters have been estimated. The parameters are listed in Table 1. Despite the approximations made for deriving the models Figures 3 and 4 demonstrate an excellent match between the measured data and the model for many decades of NEC. Note that the very low noise level, compared to Figure 1, is due to the use of difference images made from paired emission and transmission scans, which eliminates the b -term.

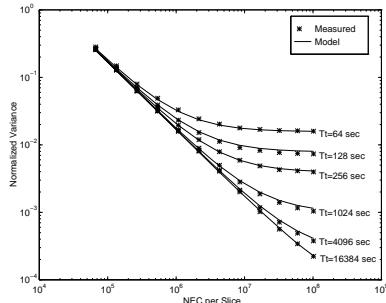


Figure 3 Normalized variance of Brain phantom (3D) as a function of NEC per slice and duration of transmission scan T_t . Stars show the measured data and lines the fitted model.

Figure 5 illustrates how well the emission data measured on human brains agree with the phantom studies. The Figure shows case KL (2D/3D), case MN (2D/3D), and the Brain Phantom (2D/3D). In all cases data corresponding to only one transmission scan are shown.

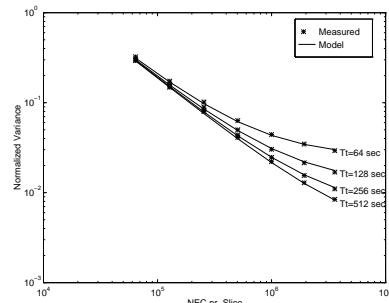


Figure 4 Normalized variance of case MN (3D) as a function of NEC per slice and duration of transmission scan T_t . Stars show the measured data and lines the fitted model.

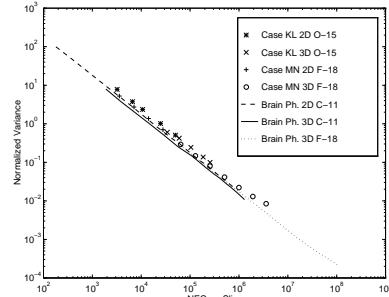


Figure 5 Normalized variance of Case KL (2D/3D), case MN (2D/3D), and Brain phantom (2D/3D) as a function of NEC per slice.

Table 2 shows noise estimates obtained from different ROIs. The emission parameter a and the transmission parameter c have been estimated for the Brain Phantom (3D) in four different ROIs. The Head ROI is an ellipse just surrounding the activity of the brain, WM is a small region within white matter, GM_p and GM_c are two small grey matter regions, peripheral and central, respectively. The averages of the measured regions are normalized to the average of WM. Note the approximately constant ratio between a and c , indicating the major influence of the average on the coefficients.

Table 2 Brain phantom data (3D) with different ROIs.

ROI	a [k counts]	c [sec]	Average
Head	17	1.0	2.07
WM	72	3.7	1.00
GM_p	4.4	0.24	3.98
GM_c	7.2	0.49	4.05

Finally Figure 6 shows data from body measurements. The 2D Elliptical Body phantom follows the previous noise model, but in case BB (large patient with arms in FOV, only a single set of transmission scans) the noise curves for the 1 and 2 minutes transmission scans are shifted

upwards. Inspection of the reconstructed images reveals large number of lines, due to (clusters of) zeros in the transmission scan (cf. Equation 11).

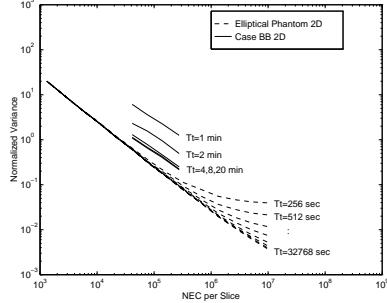


Figure 6 Normalized variance of body studies case BB (2D) and Elliptical phantom (2D) as a function of NEC per slice. Note that case BB (1,2 and 4 min) shows a different noise behavior.

V. OPTIMIZATION

One application of the estimated noise parameters is optimization of the patient time used in the scanner, [3]. Assume that the total time available for the examination of a patient is T seconds and that the transmission scan and the emission scan are distinct. The duration of the two scans are T_t and T_e respectively, and $T = T_t + T_e$. For simplicity assume that the rate of Noise Equivalent counts is constant and equals R_{NEC} . This implies that the sum of the emission and transmission term are given by

$$\frac{V}{E^2} = \frac{a}{R_{\text{NEC}}(T - T_t)} + \frac{c}{T_t} \quad (12)$$

Thus, with respect to the total noise level, the optimum duration of the transmission scan is given by

$$T_t = \frac{T}{1 + \sqrt{\frac{a}{c R_{\text{NEC}}}}} \quad (13)$$

Figure 7 shows the normalized variance of the individual noise terms and their sum as a function of transmission scan time T_t . The Figure corresponds to the parameters found in Table 1 for the Brain Phantom (3D) and the sum of the transmission scan time and emission scan time is arbitrarily set to 1800 sec. In Figure 7 it is assumed that $R_{\text{NEC}} = 1000$ counts per slice per sec, which gives an optimized normalized variance of 0.014 when $T_t = 354$ sec. Assuming $R_{\text{NEC}} = 10000$ counts per slice per sec implies that the optimum is found at $T_t = 786$ sec and the normalized variance is 0.0029.

VI. DISCUSSION

The general noise properties of reconstructive tomography are well described in the literature [1, 2]. Previous descriptions of the Advance scanner also include count rate dependence in phantoms and humans [6, 9, 10] but the

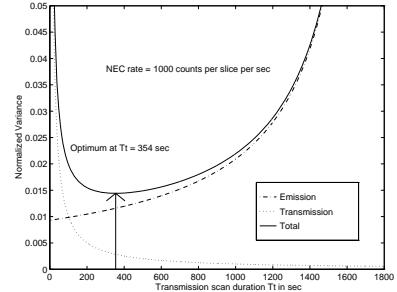


Figure 7 The transmission and emission noise terms and their sum as a function of the transmission scan duration, when the total (patient) scan time is 1800 sec. Parameters correspond to the Brain Phantom (3D). The rate of Noise Equivalent counts is 1000 counts per slice per sec.

noise characterization so far [11] focused on the emission noise. A theoretical derivation [3] using data from the GE 2048 scanner formed part of the basis for the design of the Advance scanner's 2-pin source transmission scanning system. In the present paper we have measured the relative importance of the noise contributions from blank, transmission, and emission scans for a range of imaging situations encountered. Transmission scan values are presented as scan durations in seconds for a given, specified set of sources. Optimal conditions with 2 sources of maximal strength (400 MBq) would almost double the transmission count rate. Given the current expensive Germanium-68 pin sources, the presented configuration represents a likely mean value over source life time. Results can be scaled according to the observed blank scan count rate, to accommodate for differences in number or activity of pin sources and their rotation radius.

An excellent fit to the empirical variance has been found both in 2D and 3D studies. The numerical values of the measure chosen for the noise examination is strongly dependent on reconstruction parameters, in part because of the neighbor pixel correlation disregarded in the theory section. Most parameters, however, will affect the emission and transmission contributions in the same way (through ϕ_{ij} only), and therefore the model and its output is considered adequate for the purpose of identifying areas in the E-T plane where one source is dominating. The analysis largely ignores the fact that the tomographic noise is non-stationary with higher values found towards the center as exemplified by the differently located ROIs in the brain phantom, but also in this respect is the ratio between the emission and the transmission term approximately constant.

From the estimated value of $d = 0.96$ it is seen that the blank scan contribution to the noise as expected is only a small fraction of the transmission noise for same duration, the ratio being well explained by the average attenuation of the (central part of) the 20 cm Cylinder phantom. For all practical purposes, therefore, the application of a 20 minutes blank scan will ensure that the blank scan contri-

bution is negligible since the cases where a longer transmission scan might be applied are those with a higher attenuation.

A typical 3D brain activation study with O-15 water in our setup will contain 0.5 Mcounts/central slice, (Figure 5,[9]). With a 10 minute transmission scan, the transmission variance calculated from the phantom data in Table 1 is 0.00167 compared to the emission contribution of 0.0334, i.e., the transmission adds (only) 5% to the final result. If difference images are made, the effect becomes even smaller. For the case MN the corresponding figures would be 0.00265 (transmission), 0.037 (emission) and 7%, respectively. This higher value is due to the lack of a skull for the Brain phantom.

A 10 minute, 2D FDG brain scan typically would also have 0.5-1 M counts yielding the same 5-10% ratio, while acquiring the same data in 3D would make the two contributions almost equal and therefore call for a more detailed analysis of timing. It should be noted that in this imaging condition no attribute has been made to the small additional noise from the emission correction of the transmission scan which is necessary if the transmission is performed with activity present.

Figure 6 suggests that within the emission count range of a typical dynamic heart FDG scan, the emission noise contribution (when using a 20 minute transmission scan) is dominating. Data are, however, currently not available that can demonstrate the region in which the transmission noise curves split up. It should be emphasized that the observed shifts of the curves with short transmission time are due to line-artifacts caused by (clusters of) zeros in the transmission sinogram. This deteriorating effect is usually more important and calls for an improved methods of attenuation correction, e.g., by using image segmentation as described in [12].

VII. CONCLUSION

The measured image noise variance from a GE Advance PET scanner has been modelled as a sum of terms corresponding to the noise in the emission scan, the transmission scan, and the blank scan. The weight parameters in this simple model have been determined from a large number of experiments in both 2D and 3D scan mode for the 20 cm standard (NEMA) cylinder, a torso-like ellipse with axes 18 and 35 cm, and an axially symmetrical, elliptical brain phantom (Capintec) with axes 15 and 20 cm. Furthermore, noise model parameters have been found from some human studies. For brain studies there is an excellent agreement between model and observation, and between phantoms and human studies. The estimated parameters have been used to optimize the durations of the emission and transmission scan under the constraint that their sum is constant. For studies of the heart or other body regions the model is still generally valid, but the parameters not as well documented, and the transmission noise is often dominated by line artifacts.

REFERENCES

- [1] A. E. Todd-Pokropek and P. H. Jarrit. The noise characteristics of SPECT systems. In P. J. Ell and B. L. Holman, editor, *Computed Emission Tomography*, pages 361-389. Oxford University Press, Oxford, 1982.
- [2] N. M. Alpert et al. Estimation of the Local Statistical Noise in Emission Computed Tomography. *IEEE Trans. Med. Imag.*, MI-1(2):142-146, 1982.
- [3] C. W. Stearns and D. Wack. A Noise Equivalent Counts Approach to Transmission Imaging and Source Design. *IEEE Trans. Med. Imag.*, MI-12(2):287-292, June 1993.
- [4] S. C. Strother et al. Measuring PET Scanner Sensitivity: Relating Count Rates to Image Signal-to-Noise Ratios Using Noise Equivalent Counts. *IEEE Trans. Nucl. Sci.*, NS-37(2):783-788, April 1990.
- [5] T. R. DeGrado et al. Performance Characteristics of a Whole-Body PET Scanner. *J. Nucl. Med.*, 35(8):1398-1406, 1994.
- [6] T. K. Lewellen et al. Investigations of the Count Rate Performance of General Electric Advance Positron Emission Tomograph. *IEEE Trans. Nucl. Sci.*, NS-42(4):1051-1057, August 1995.
- [7] T. Lewellen et al. Investigations of the Performance of the General Electric Advance Position Emission Tomograph in 3D mode. In *Submitted*, 1995.
- [8] P. S. Crandall and C. W. Stearns. A Scalable Multi Processor Implementation of the Reprojection Algorithm for Volumetric PET Imaging. In *Conference Record of the IEEE 1995 Nuclear Symposium and Medical Imaging Conference*, 1996. In press.
- [9] S. Holm et al. 3D PET activation studies with $H_2^{15}O$ bolus injection. Count rate performance and dose optimization. In *Quantification of Brain Function using PET*. Academic Press, San Diego, 1996. In press.
- [10] S. Holm et al. Count Rate Performance and Dose Optimization for rCBF with O-15 Water Bolus Injection in the New 3D PET. *J. Nucl. Med.*, 36:105P, 1995.
- [11] S. Pajavac et al. Noise Properties of 3D PET Images. *J. Nucl. Med.*, 36:105P, 1995.
- [12] S. R. Meikle et al. Attenuation Correction Using Count-Limited Transmission Data in Positron Emission Tomography. *J. Nucl. Med.*, 34:143-150, 1993.

Appendix L

A very fast Implementation of 2D Iterative Reconstruction Algorithms

This appendix contains the abstract and summary of the paper *A very fast Implementation of 2D Iterative Reconstruction Algorithms*, by Peter Toft and Jesper James Jensen [5]. The abstract and summary has been submitted to the *IEEE Medical Imaging Conference 96*. Note that a paper has also been submitted, see Appendix M.

A very fast Implementation of 2D Iterative Reconstruction Algorithms

Peter Toft† and Jesper James Jensen‡

† IMM, Technical University of Denmark, DK-2800 Lyngby, Denmark, Email ptoft@ei.dtu.dk.
‡ ØDS-Holding A/S, Kroghsgade 1, DK-2100 Copenhagen E, Denmark, Email jjj@oedan.dk.

abstract

One of the limitations of using iterative reconstruction methods in tomography is the slow performance compared with the direct reconstruction methods, such as filtered backprojection. In this paper we demonstrate a very efficient implementation of virtually all types of iterative reconstruction methods. The key idea of our methods is to generate the huge system matrix only once, and store it using sparse matrix techniques. From the sparse matrix we can perform the matrix vector products very fast, which implies a huge acceleration of the reconstruction. In this paper we demonstrate that iterative reconstruction algorithms can be implemented and run almost as fast as direct reconstruction algorithms. The method has been implemented in a software package which is available for free, providing ART, EM, and the Least Squares Conjugate Gradient Method.

A very fast Implementation of 2D Iterative Reconstruction Algorithms

Peter Toft† and Jesper James Jensen‡

† IMM, Technical University of Denmark, DK-2800 Lyngby, Denmark, Email ptoft@ei.dtu.dk.

‡ ØDS-Holding A/S, Kroghsgade 1, DK-2100 Copenhagen E, Denmark, Email jjj@oedan.dk.

1 Introduction

In the last 15 years the iterative reconstruction methods have gained much attention in the literature [1]. Several methods have been very prominent, such as EM (Expectation Maximization) [2, 3], ART (Algebraic Reconstruction Technique) [4], and LSCG (Least Squares Conjugate Gradient) [5]. Each of the methods formulates the reconstruction problem as a linear set of equations $\mathbf{b} = \mathbf{Ax} \Leftrightarrow b_i = \sum_{j=1}^{J-1} a_{i,j}x_j$, where \mathbf{b} is an I -dimensional vector containing the known sinogram values wrapped into a vector, and \mathbf{x} is a J -dimensional vector containing the unknown image to be reconstructed. The matrix \mathbf{A} is the system matrix, which contains the weight factors between each of the image pixels and each of the sinogram values, corresponding to line orientations.

One problem is the huge size of the system matrix, often impossible to store in memory. Assuming that memory is not available for storing the full system matrix, then one possibility is to compute the individual matrix elements in each iteration when needed. This can be done by using of the discrete Radon transform or other modelling schemes of the scanner. This solution is rather easily implemented and is viable and storage requirements are reduced to a minimum, by only requiring memory for the sinogram (\mathbf{b}) and the current solution (\mathbf{x}), and perhaps some additional temporary variables of the same size or smaller, but no system matrix is stored in memory. This implementation has a major drawback in speed, due to the many times the system matrix has to be computed during an iterative reconstruction. Each time at the same high cost.

2 Accelerated 2D Iterative Reconstruction

Here a hybrid solution is proposed for accelerating the reconstruction speed of the iterative methods and only requiring as much memory as modern workstations are currently equipped with or will be soon. The idea is to store the non-zero elements of the system matrix in the memory using sparse matrix techniques. In this way the core of the reconstruction algorithms, highly based on matrix vector multiplications, can be accelerated significantly, and thereby solve one of the major drawbacks of the iterative methods.

It is proposed that the system matrix \mathbf{A} is calculated one time only using all the modifications found for the actual scanner setup. If no specific scanner model is provided then the system matrix can be modelled and generated using the Radon transform or other simpler schemes. From the system matrix the very small values in the matrix are truncated to zero by using a very small threshold level.

The sparse structure of \mathbf{A} can be exploited by only storing the non-zero values in the fast memory. For a certain row, number i , all of the matrix elements are calculated, stored, and truncated. Hereby the number of non-zero elements in the row, denoted by Z_i , will be much smaller than the image size $J = M^2$. The values of Z_i are stored in a simple one dimensional vector. Two vectors of length Z_i indexed by an integer z can then be allocated containing both the non-zero matrix value a_z and the corresponding column index j_z . Assuming that 4 bytes are required for storing each of the vector elements then the total storage requirement is reduced to approximately $8\sum_{i=1}^I z_i \approx 8IM$ bytes, (assuming a nearest neighbour approximation with one pixel for each point along the integration lines). Assuming that a sufficient amount of memory is present most iterative algorithms can be implemented from three basic operations: Matrix vector multiplication $\mathbf{A}\tilde{\mathbf{x}}$, scalar product between the i 'th row of the system matrix and a vector $\mathbf{a}_i^T\tilde{\mathbf{x}}$, and finally multiplication with the transpose of the matrix $\mathbf{A}^T\tilde{\mathbf{b}}$.

A software package has been written in C including the proper structures for manipulating sparse matrices and vectors, along with an optimized code for computing matrix vector products, well suited for iterative reconstruction algorithms. In the package ART, EM, and LSCG have all been implemented in a fast version using sparse matrix storage of the system matrix and in a slow version where the system matrix is not stored and needed matrix entries

are computed in each part of the iterative steps. The software package is provided for free, protected by the GNU General Public License.

Several interpolation methods have been implemented for generation of the system matrix: (1): Nearest Neighbour interpolation based on the Radon transform. (2): Linear interpolation based on the Radon transform. (3): Analytical Radon transform of a square, i.e., the length through a quadratic pixel is used. (4): The Radon transform of a sinc expansion in the image domain.

3 Results

The program has been used on two types of machines. A Linux machine with a 120 MHz Pentium processor and a SGI Onyx with four 200 MHz R4400 processors, with the program running on one processor.

As an example a sinogram with 125×101 samples is reconstructed into an image with 101×101 pixels. In Table 1 the reconstruction time is shown for the fast and the slow methods on the two machines.

Times are measured for ART, EM, and the LSCG-method, where EM and LSCG were running (arbitrarily) 20 iterations, and ART 20 full iterations, i.e., 20 times the number of rows (chosen randomly), which is $20 \times 125 \times 101$ iterations. Note that all times only correspond to the actual iterations. For the fast versions of the iterative reconstruction algorithms, the time to generate the system matrix once should be added if changing the parameters determining the system matrix, e.g., the sampling parameters of the sinogram or the reconstructed image.

For this example the system matrix was modelled using discrete Radon transformation with linear interpolation, where the threshold was chosen to zero, hence the slow and the fast methods give exactly the same results. In this example the sparse system matrix required approximately 13 MBytes memory.

Note that the large difference in speedup between ART and EM/LSCG is due to a very effective implementation of the forward projection compared with the backprojection. The slow methods can be accelerated some by implementing multiplication with the transpose of the system matrix (adjoint operator) as a backprojection integral, but note that this implies that the approximation of the system matrix will be different in the forward and the backprojection part.

Machine	Fast ART	Slow ART	Fast EM	Slow EM	Fast LSCG	Slow LSCG
Pentium	26 sec	2218 sec	17 sec	5776 sec	17 sec	5250 sec
Onyx	16 sec	1306 sec	16 sec	2722 sec	16 sec	2715 sec

Table 1 Time usage for 20 iterations of EM and LSCG. For ART the time is for 20 full iterations, i.e., $20 \times 125 \times 101$ iterations.

4 Conclusion

We have demonstrated a very fast implementation of iterative reconstruction based on storing the system matrix in fast memory by using sparse techniques. The approach is mainly applicable to 2D reconstruction, due to the requirements of a sufficient amount of memory, but in principle the method can also be applied to 3D reconstruction.

The cost of the proposed idea is that a large amount of memory is required, but for ART we have demonstrated a speedup factor of approximately 70 and for EM and LSCG a speedup factor larger than 175, between the slow and the fast implementation. In the example it was showed that each iteration of ART, EM, or LSCG requires approximately one second on most modern workstations.

References

- [1] Yair Censor. Finite series-expansion reconstruction methods. In *Proc. of the IEEE*, volume 71, pages 409–419, March 1983.
- [2] L. A. Shepp and J. B. Krustal. Computerized tomography: The new medical x-ray technology. *Am. Math. Monthly*, 85:420–439, April 1978.
- [3] R. E. Carson and K. Lange. The EM parametric image reconstruction algorithm. *Journal of the American Statistical Association*, 389(80):20–25, March 1985.
- [4] Gabor T. Herman and Lorraine B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE trans. on Med. Imag.*, 12(3):600–609, Sep. 1993.
- [5] John A. Scales. Iterative methods for large, sparse, inverse calculations. Samizdat Press. Colorado School of Mines, 76 Olcott Drive, White River Junction, Vermont 05001, April 1993. Lecture notes.

Appendix M

Accelerated 2D Iterative Reconstruction

This appendix contains the paper *Accelerated 2D Iterative Reconstruction*, by Peter Toft and Jesper James Jensen, which has been submitted to IEEE Transactions on Medical Imaging [6]. A subset of this paper has been submitted to the Medical Imaging Conference 96. This paper is shown in Appendix L.

Accelerated 2D Iterative Reconstruction

Peter Toft, Department of Mathematical Modelling,
 Technical University of Denmark, DK-2800 Lyngby, Denmark, Email ptoft@ci.dtu.dk.
 Jesper James Jensen, ØDS-Holding A/S, Kroghsgade 1,
 DK-2100 Copenhagen E, Denmark, Email jjj@oedan.dk.

Abstract

One of the limitations of using iterative reconstruction methods in tomography is the slow performance compared with the direct reconstruction methods, such as filtered backprojection. In this paper we demonstrate a very fast implementation of most types of iterative reconstruction methods. The key idea of our method is to generate the huge system matrix only once, and store it using sparse matrix techniques. From the sparse matrix we can perform the matrix vector products very fast, which implies a major acceleration of the reconstruction algorithms. In this paper we demonstrate that iterative reconstruction algorithms can be implemented and run almost as fast as direct reconstruction algorithms. The method has been implemented in a software package that is available for free, providing reconstruction algorithms using ART, EM, and the Least Squares Conjugate Gradient Method.

I. INTRODUCTION

In the last 15 years the iterative reconstruction methods have gained much attention in the literature [1, 2]. Several methods have been very prominent, such as EM (Expectation Maximization) [3, 4, 5, 6, 7], ART (Algebraic Reconstruction Technique) [8, 1, 9], and LSCG (Least Squares Conjugate Gradient) [10, 11].

These methods formulate the reconstruction problem as a linear set of equations

$$\mathbf{b} = \mathbf{Ax} \Leftrightarrow b_i = \sum_{j=1}^{J-1} a_{i,j}x_j, \quad i = 1, 2, \dots, I \quad (1)$$

where \mathbf{b} is an I -dimensional vector containing the known sinogram values wrapped into a vector, and \mathbf{x} is a J -dimensional vector containing the unknown image to be reconstructed. Here \mathbf{A} is the system matrix,

which contains the weight factors between each of the image pixels and each of the values in the sinogram, corresponding to line orientations. Compared with Radon transform based direct reconstruction methods [12], the use of linear algebra has several advantages, such as easier incorporation of irregular geometries. The system matrix can model several real-world properties, such as finite, i.e., non-zero detector size and varying detector sensitivity. Furthermore regularization can easily be incorporated [13, 10] in order to affect the often ill-conditioned reconstruction problem.

One problem is the huge size of the system matrix. A 2D sinogram from, e.g., a GE Advance PET scanner contains $I = 281 * 336$ values, and reconstructed into a $J = 301 * 301$ grid, i.e., the system matrix has approximately 8.5 billion elements, requiring over 34 GBytes of memory, when using 4 bytes per matrix element. This is a large amount of memory, even looking some years into the future. Besides this aspect, it would not be wise to store all that data, due to the fact that approximately 98% of the matrix entries will be zeros. This knowledge should be incorporated into the reconstruction schemes.

Assuming that memory is not available for storing the full system matrix, one possibility is to compute the individual matrix elements in each iteration when needed. This can be done by using the Radon transform, e.g., [14] or other modelling schemes for the scanner. This approach is rather easily implemented and is viable and storage requirements are reduced to a minimum, only requiring memory for the sinogram (\mathbf{b}) and the current solution (\mathbf{x}), and perhaps some additional temporary variables of the same size or smaller, but no system matrix is stored in memory. It will be demonstrated that this implementation has a major drawback in speed, since the system matrix will be computed many times during an iterative

reconstruction. Each time at the same high computational cost.

II. ACCELERATED 2D ITERATIVE RECONSTRUCTION

Here a hybrid solution is proposed for accelerating the iterative reconstruction algorithms, but requiring as much memory as modern workstations are currently equipped with, or will be soon. The idea is to store the non-zero elements of the system matrix in the main memory using sparse matrix techniques. In this way the core of the reconstruction algorithms, highly based on matrix vector multiplications, can be accelerated significantly, and thereby solve one of the major drawbacks of the iterative methods.

It is proposed that the system matrix \mathbf{A} is calculated one time only using all the modifications found for the actual scanner setup. If no specific scanner model is provided then the system matrix can be modelled and generated using the Radon transform or other simpler schemes. From the system matrix the very small values in the matrix are truncated to zero.

$$\tilde{a}_{i,j} = \begin{cases} a_{i,j} & \text{if } a_{i,j} > \gamma \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

where the threshold γ can be chosen to a certain fraction of the maximum matrix value, e.g., $\gamma = 0.05 \max_{i,j} \{a_{i,j}\}$. If γ is chosen sufficiently low, a good compromise between resolution and the sparseness of the matrix can be reached, and normally this does not alter the behaviour of the algorithms. Current work concerns the quantification of the truncation error.

The sparse structure of \mathbf{A} can be exploited by only storing non-zero values in the fast memory. For a certain row, number i , all of the matrix elements are calculated, stored, and truncated using Eq. 2. Hereby the number of non-zero elements in the row, denoted by Z_i , will be much smaller than the image size $J = M^2$. The values of Z_i are stored in a simple, one dimensional vector. Two vectors of length Z_i , indexed by an integer z , can then be allocated and stored containing the non-zero matrix value a_z and the corresponding column index j_z . The procedure is repeated for all rows.

Assuming a nearest neighbour approximation with one pixel for each point along the integration lines and using 4 bytes for storing each of the vector elements,

the total storage requirement is then reduced to approximately $8 \sum_{i=1}^I Z_i \approx 8IM$ bytes. In the example shown above approximately 100 MBytes memory is required. Assuming this amount of memory is present most iterative algorithms can be implemented from three basic operations: Matrix vector multiplication $\mathbf{A}\tilde{\mathbf{x}}$, scalar product between the i 'th row of the system matrix and a vector $\mathbf{a}_i^T \tilde{\mathbf{x}}$, and finally multiplication with the transpose of the matrix $\mathbf{A}^T \tilde{\mathbf{b}}$.

In the following pseudo code (called Algorithms), the implementation of the matrix vector multiplication and the multiplication with the transpose of the system matrix are shown. A C++ style is used for comments and note that all indices here start at zero.

```

ALGORITHM 1 :  $\mathbf{Ax}$ 
  For i = 0 to I-1           //For all rows
    sum = 0                  //Initialize
    Set a and j to correct row //Use pointers
    For z=0 to Z(i)-1         //For row i
      sum=sum+a(z)*x(j(z))   //Increment sum
    End
    bt(i)=sum                //Store value
  End
END ALGORITHM

ALGORITHM 2 :  $\mathbf{A}^T b$ 
  For j=0 to J-1           //For all columns
    xb(j)=0                  //Initialize
  End
  For i=0 to I-1           //For all rows
    Set a and j to correct row //Using pointers
    For z=0 to Z(i)-1         //Compute sum
      xb(j(z))=xb(j(z))+a(z)*b(i) //Update sum
    End
  End
END ALGORITHM

```

III. IMPLEMENTED METHODS

A software package has been written in C including the proper structures for manipulating sparse matrices and vectors, along with an optimized code for computing matrix vector products, well suited for iterative reconstruction algorithms. In the package ART, EM, and LSCG are implemented both in a fast version using sparse matrix storage of the system matrix and in a slow version where the system matrix is not stored and needed matrix entries are computed in each step of the iterative algorithms. The software package is available for free, but protected by the GNU General Public License. Contact the authors to obtain the package.

ART For a certain row i of the matrix (i depends on the iteration number k), the general iteration step incrementing the current solution $\mathbf{x}^{(k)}$ can be found in, e.g., [1]

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \frac{b_i - \mathbf{a}_i^T \mathbf{x}^{(k)}}{\mathbf{a}_i^T \mathbf{a}_i} \mathbf{a}_i \quad (3)$$

EM The general iteration step of EM [5, 15] requires a forward projection, a backprojection, and two fast updates in each iteration.

$$\mathbf{b}^f = \mathbf{A}\mathbf{x}^{(k-1)} \quad (4)$$

$$b_i^r = \frac{b_i}{b_i^f} \quad (5)$$

$$\mathbf{x}^b = \mathbf{A}^T \mathbf{b}^r \quad (6)$$

$$x_j^{(k)} = \frac{x_j^{(k-1)} x_j^b}{s_j} \text{ where } s_j = \sum_{i=1}^I a_{i,j} \quad (7)$$

LSCG The Least Squares Conjugate Gradient method requires some initialization [11]

$$\mathbf{s}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)} \quad (8)$$

$$\mathbf{r}^{(0)} = \mathbf{p}^{(0)} = \mathbf{A}^T \mathbf{s}^{(0)} \quad (9)$$

$$\mathbf{q}^{(0)} = \mathbf{A}\mathbf{p}^{(0)} \quad (10)$$

Then for each iteration the LSCG algorithm on the normal equations becomes

$$\alpha = \frac{(\mathbf{r}^{(k-1)})^T \mathbf{r}^{(k-1)}}{(\mathbf{q}^{(k-1)})^T \mathbf{q}^{(k-1)}} \quad (11)$$

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha \mathbf{p}^{(k-1)} \quad (12)$$

$$\mathbf{r}^{(k)} = \mathbf{A}^T \mathbf{s}^{(k-1)} \quad (13)$$

$$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha \mathbf{q}^{(k-1)} \quad (14)$$

$$\beta = \frac{(\mathbf{r}^{(k)})^T \mathbf{r}^{(k)}}{(\mathbf{r}^{(k-1)})^T \mathbf{r}^{(k-1)}} \quad (15)$$

$$\mathbf{p}^{(k)} = \mathbf{r}^{(k)} + \beta \mathbf{p}^{(k-1)} \quad (16)$$

$$\mathbf{q}^{(k)} = \mathbf{A}\mathbf{p}^{(k)} \quad (17)$$

For all three methods an initial value of the solution, i.e., $\mathbf{x}^{(0)}$ is needed. In the package an image found by, e.g., a fast direct method, can be supplied and used. If not provided, all of the initial values of the vector are initialized to a properly chosen constant.

Interpolation Methods: Several interpolation methods have been implemented for computing the system matrix.

- Nearest Neighbour interpolation based on the Radon transform.
- Linear interpolation based on the Radon transform.
- Analytical Radon transform of a square, i.e., the length through a quadratic pixel is used.
- The Radon transform of a sinc expansion in the image domain.

IV. RESULTS

The program has been used on two types of machines. A Linux machine with a 120 MHz Pentium processor and an Onyx from SGI equipped with four 200 MHz R4400 processors, where the program was running on one processor.

A. Example 1

In the first example the (synthetic) sinogram has $125 * 101$ samples and the reconstructed image has $101 * 101$ samples. In Table 1 the reconstruction times on both machines are shown for the fast and the slow method as well as the ratio between the execution times (slow/fast).

Times are measured for ART, EM, and the LSCG-method, when EM and LSCG were running (arbitrarily) 20 iterations, and ART 20 full iterations, i.e., 20 times the number of rows (chosen randomly), which is $20 * 125 * 101$ iterations in Eq. 3. Note that all times only correspond to the actual iterations. For the fast versions of the iterative reconstruction algorithms, the time to generate the system matrix once should be added if changing the system matrix, e.g., when changing the sampling parameters of the reconstructed image.

For this example the system matrix was modelled using discrete Radon transformation with linear interpolation, where the threshold γ was chosen to zero, hence the slow and the fast methods give exactly the same results. Note that the large difference in speedup between ART and EM/LSCG is due to the implementation of the forward projection is more efficient than the multiplication with the transpose of the system matrix. The slow methods can be accelerated some by implementing multiplication with the transpose of the system matrix (adjoint operator) as a backprojection integral, but note that this implies

that the approximation of the system matrix will be different in the forward and the backprojection part. The sparse system matrix for this transformation geometry required approximately 13 MBytes, and each iteration requires approximately one second.

Machine	Type	ART	EM	LSCG
Pentium	Fast	26 sec	17 sec	17 sec
	Slow	2218 sec	5776 sec	5250 sec
	Ratio	85	340	309
SGI Onyx	Fast	16 sec	16 sec	16 sec
	Slow	1306 sec	2722 sec	2715 sec
	Ratio	82	170	170

Table 1 Time usage for 20 iterations of EM and LSCG. For ART the time is for 20 full iterations, i.e., $20 \times 125 \times 101$ of the iterations used in Eq. 3. The time measurements are for a sinogram with 125×101 samples reconstructed into a 101×101 samples image.

B. Example 2

In Fig. 1 a 2D sinogram with 281×336 samples is shown, which was measured on a GE Advance PET scanner. The sinogram is reconstructed into a (large) image with 301×301 samples. The system matrix has 94416×90601 elements of which 0.23% are non-zero when modelling then system matrix using the Radon transform of a square. On the Onyx it required 1466 seconds to generate the 157 MBytes sparse matrix, and each iteration of EM required 15 seconds in the fast version and 11678 sec in the slow implementation. After 10 iterations the algorithm was stopped. The reconstructed image is shown in Fig. 2. For sake of comparison, the same sinogram reconstructed image using Filtered Backprojection is shown in Fig. 3, and this reconstruction used 6 seconds on the Onyx.

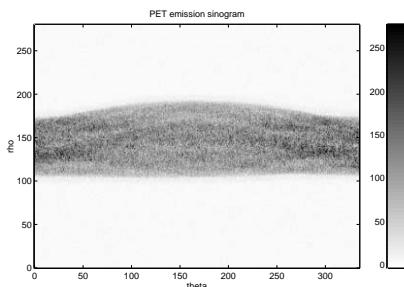


Figure 1 A PET sinogram of a human brain.

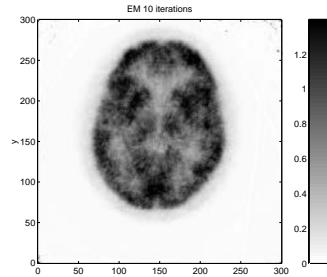


Figure 2 The reconstructed image after 10 iterations of EM.

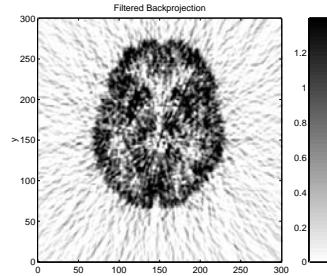


Figure 3 The reconstructed image using Filtered Backprojection with a ramp filter.

C. Example 3

Here the LSCG method is used to reconstruct a 101×101 image from the sinogram shown in Fig. 1. On the Onyx 162 seconds is required to generate the system matrix and each iteration required 3 seconds. The reconstructed image after 11 iterations is shown in Fig. 4.

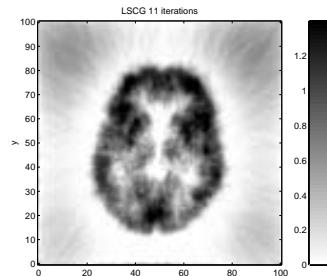


Figure 4 The reconstructed image using 11 iterations of the Least Squares Conjugate method. The artifacts around the brain can be removed by proper masking.

V. CONCLUSION

We have demonstrated a very fast implementation of iterative reconstruction comparable in speed with direct reconstruction methods. The implementation is based on storing of the system matrix in fast memory using sparse techniques. The approach is mainly applicable to 2D reconstruction, due to the requirements of a sufficient amount of memory, but in principle the method can also be applied to 3D reconstruction.

The cost is that a large amount of memory is required, but for ART we have demonstrated a speedup factor of approximately 80 and for EM and LSCG 170-340 depending on the machine, for a fixed transformation geometry and interpolation level. Once again these factors could be somewhat moderated by a faster implementation of the multiplication with the inverse system matrix.

ACKNOWLEDGMENT

We thank associate professor Lars Kai Hansen for support and useful comments.

REFERENCES

- [1] Yair Censor. Finite series-expansion reconstruction methods. In *Proc. of the IEEE*, volume 71, pages 409–419, March 1983.
- [2] Julia K. Olden and Paul C. Johns. Matrix formulation of computed tomography reconstruction. *Physics in Medicine & Biology*, 38:1051–1064, 1993.
- [3] L. A. Shepp and J. B. Krustal. Computerized tomography: The new medical x-ray technology. *Am. Math. Monthly*, 85:420–439, April 1978.
- [4] L. A. Shepp Y. Vardi and L. Kaufman. A statistical model for positron emission tomography. *Journal of the American Statistical Association*, 80(389):8–37, March 1985. The pages include comments and discussion by several authors.
- [5] R. E. Carson and K. Lange. The em parametric image reconstruction algorithm. *Journal of the American Statistical Association*, 80(389):20–25, March 1985.
- [6] Linda Kaufman. Implementing and accelerating the em algorithms for positron emission tomography. *IEEE Trans. Med. Imag.*, MI-6(1):37–51, march 1987.
- [7] John M. Ollinger. Maximum-Likelihood reconstruction of transmission images in emission computed tomography via the EM algorithm. *IEEE Trans. Med. Imag.*, 13(1):89–101, March 1994.
- [8] Gabor T. Herman and Lorraine B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE Trans. Med. Imag.*, 12(3):600–609, Sep. 1993.
- [9] Huaiqun Guan and Richard Gordon. A projection access order for speedy convergence of art (algebraic reconstruction technique): a multi-level scheme for computed tomography. *Phys. Med. Biol.*, 39:2005–2022, 1994.
- [10] Linda Kaufman and Arnold Neumaier. Image Reconstruction Through Regularization by Envelope Guided Conjugate Gradients. In *Bell Labs Report 11274-94-14*. Submitted to IEEE Trans. Medical Imaging.
- [11] John A. Scales. Iterative methods for large, sparse, inverse calculations. Samizdat Press. Colorado School of Mines, 76 Olcott Drive, White River Junction, Vermont 05001, April 1993. Lecture notes.
- [12] S. R. Deans. *The Radon Transform and Some of Its Applications*. Krieger Publishing Company, Malabar, Florida, 2 edition, 1993. Previously John Wiley & Sons Inc., 1983.
- [13] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems*. Polyteknisk Forlag, Lyngby, Denmark, 1996. Doctoral Dissertation.
- [14] Peter Toft. *The Radon Transform - Theory and Implementation*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, 1996.
- [15] G. T. Herman and D. Odhner. Performance evaluation of an iterative image reconstruction algorithm for positron emission tomography. *IEEE Trans. Med. Imag.*, 10:336–246, 1991.

Appendix N

Mean Field Reconstruction with Snaky Edge Hints

This appendix contains the paper *Mean Field Reconstruction with Snaky Edge Hints*, by Peter Alshede Philipsen, Lars Kai Hansen and Peter Toft, which has been presented at the *Interdisciplinary Inversion Conference 95* in Aarhus. The paper will appear in *INVERSE METHODS - Interdisciplinary elements of Methodology, Computation and Application*, which will be published from Springer Publications in 1996 [11].

The paper has also been presented and published in the proceedings of the *Fourth Danish Conference for Pattern Recognition and Image Analysis 95* [12].

In the paper mean field techniques have been used to improve image quality by using strong priors in the restoration of PET reconstructed images. This work is based on the results shown in [61].

Mean Field Reconstruction with Snaky Edge Hints

Peter Alshede Philipsen, Lars Kai Hansen and Peter Toft
 CONNECT, Electronics Institute, build. 349
 Technical University of Denmark,
 DK-2800 Lyngby, Denmark
 email: pap,lkhansen,ptoft@ei.dtu.dk

16/6-95

1 Introduction

Reconstruction of imagery of a non-ideal imaging system is a fundamental aim of computer vision. Geman and Geman [4] introduced Metropolis sampling from Gibbs distributions as a simulation tool for visual reconstruction and showed that a *Simulated Annealing* strategy could improve the efficiency of the sampling process. The sampling process implements a stochastic neural network with symmetric connections. Peterson and Anderson applied the *Mean Field* approximation, and observed substantial improvements in speed and performance [6].

In the next section the Bayesian approach to reconstruction is outlined, as model example we study the so-called *Weak Membrane* model. The Weak Membrane is a popular vehicle for piece-wise smooth reconstruction and involves edge units (called line processes in [4]). Edge unit control has shown to be a major challenge in applications of the Weak Membrane. However, recently there has been substantial progress in use of deformable models for contour (2D) and surface (3D) modeling, see, e.g., [2, 3]. In this contribution we suggest to combine the two approaches, in particular we show how a “Snake” contour model may be used to produce efficient edge hints to the Weak Membrane. Section three contains experiments and concluding remarks.

2 Bayesian Visual Reconstruction

The basic idea is to consider both the source (un-degraded) signal and the processes of the imaging system as stochastic processes. The Bayes formula can then be used to obtain the distribution $P(V|d)$ of the reconstructed signal V , conditioned on the observed degraded signal d :

$$P(V|d) = \frac{P(d|V)P(V)}{P(d)} \quad (1)$$

This conditional distribution is the product of the distribution of the imaging system process: $P(d|V) \equiv P(V \rightarrow d)$, and the *prior* distribution of the reconstructed signal $P(V)$. $P(V|d)$ of Equation (1) is referred to as the *posterior* distribution. A useful estimate of the reconstructed signal is given by the location of the mode of the posterior distribution, the so-called *Maximum A Posteriori* estimate.

2.1 The Weak Membrane Model

The *prior* distribution reflects our general insight on the objects being imaged, for example expressing that the image represents extended 3D structures etc. The Weak Membrane is a simple model for reconstruction of piece-wise continuous intensity surfaces (2D signals) from noisy observations [4]. In the lattice $(m, n) \in \mathbb{Z}^2$ version the reconstructed surface is described by the intensity values $V_{m,n}$. The prior information formalizes the expectation that neighbor intensity values should be close, except when they are disconnected by the rare occasion of an active edge-unit. We introduce horizontal $h_{m,n}$ and vertical edges $v_{m,n}$. The *prior* probability distribution for a membrane reads:

$$P(V, h, v) = Z_1^{-1} \exp(-E_{\text{prior}}(y, h, v)) \quad (2)$$

where $E_{\text{prior}}(y, h, v)$ is the energy or cost function:

$$\begin{aligned} E_{\text{prior}}(V, h, v) &= \frac{\nu}{2} \left[\sum_{m,n} (1 - h_{m,n})(V_{m,n} - V_{m+1,n})^2 + (1 - v_{m,n})(V_{m,n} - V_{m,n+1})^2 \right] \\ &+ \sum_{m,n} \mu_{m,n}^h h_{m,n} + \sum_{m,n} \mu_{m,n}^v v_{m,n} - \sum_{m,n} \gamma_{m,n} h_{m,n} h_{m,n+1} v_{m,n} v_{m+1,n} \end{aligned} \quad (3)$$

and Z_1^{-1} is a normalization constant. This is a *Gibbs* distribution¹. Note that last terms in the cost function act as local *chemical potentials* for control of the number of active edge-units (i.e., $h_{m,n} = +1$).

The degradation process produces the measurements $(V_{m,n} \rightarrow d_{m,n})$. In the Weak Membrane example we will for simplicity study addition of zero-mean, white Gaussian noise with variance σ^2 . Note that any other model of the degeneration process can be incorporated here:

$$P[d|V] = Z_2^{-1} \exp\left(-\frac{1}{2\sigma^2} \sum_{m,n} (V_{m,n} - d_{m,n})^2\right) \quad (4)$$

Using Bayes formula (1), we can combine Equations (2) and (4), to obtain the parameterized posterior distribution:

$$P[V, h, v|d] = Z^{-1} \exp(-E(V, h, v, d)) \quad (5)$$

where Z is a normalization constant and the energy function is given by,

$$E(V, h, v, d) = \frac{1}{2\sigma^2} \sum_{m,n} (V_{m,n} - d_{m,n})^2 + E_{\text{prior}}(V, h, v, d). \quad (6)$$

2.2 Network design

Inspecting the posterior distribution we note that the three-component process $(V_{m,n}, h_{m,n}, v_{m,n})$ realizes a *Compound Random Markov Field*. This property ensures that the neural network implementation only involves neighbor connectivity. To enhance the sampling efficiency we use a simulated annealing strategy as recommended by Geman and Geman [4]. The scheme is implemented by introducing a *temperature* T in the Gibbs distribution and design the sampler with a

¹A distribution of the form $P(x) = Z^{-1} \exp(-E(x)/T)$, where $E(x)$ is a cost-function, bounded from below, and T is a parameter.

decreasing sequence of temperatures ending up at $T = 1$. The temperature dependent distribution reads:

$$P[V, h, v|d] = Z_T^{-1} \exp\left(-\frac{E[V, h, v, d]}{T}\right) \quad (7)$$

Sampling of this distribution, as investigated by Geman and Geman, is too slow for most applications and it is therefore recommended to invoke a deterministic approximation scheme to obtain the necessary averages [1, 6]. The self-consistent Mean Field equations for the Weak Membrane model in the k 'th iteration read (see also [5] for an introduction):

$$\begin{aligned} V_{m,n}^{k+1} &= \kappa^{-1} \left[\sigma^{-2} d_{m,n} + (1 - h_{m,n}^k) V_{m+1,n}^k + (1 - h_{m-1,n}^k) V_{m-1,n}^k \right. \\ &\quad \left. + (1 - v_{m,n}^k) V_{m,n+1}^k + (1 - v_{m,n-1}^k) V_{m,n-1}^k \right] \\ h_{m,n}^{k+1} &= \tanh \left[\beta \left(\frac{1}{2} (V_{m,n}^k - V_{m+1,n}^k)^2 - \mu_h - \gamma_{m,n} \Gamma_{m,n}^h(k) \right) \right] \\ v_{m,n}^{k+1} &= \tanh \left[\beta \left(\frac{1}{2} (V_{m,n}^k - V_{m,n+1}^k)^2 - \mu_h - \gamma_{m,n} \Gamma_{m,n}^v(k) \right) \right] \end{aligned} \quad (8)$$

where $\beta = 1/T$ and

$$\Gamma_{m,n}^h(k) = h_{m,n+1}^k v_{m,n}^k v_{m+1,n}^k + h_{m,n-1}^k v_{m,n-1}^k v_{m+1,n-1}^k \quad (9)$$

$$\Gamma_{m,n}^v(k) = v_{m+1,n}^k h_{m,n}^k h_{m,n+1}^k + v_{m-1,n}^k h_{m-1,n}^k h_{m-1,n+1}^k \quad (10)$$

$$\kappa = \sigma^{-2} + 4 - h_{m,n}^k - h_{m-1,n}^k - v_{m,n}^k - v_{m,n-1}^k \quad (11)$$

We have used the same symbols for the averaged quantities as for the stochastic, but that should not lead to confusion since the two never occur in the same expression. The coupled equations are solved by straightforward iteration as shown in Equation (8), defining a recursive nearest neighbor connected *cellular neural network*.

2.3 Snake hints

There has been much recent progress in the use of deformable models for edge and surface identification, see, e.g., [2]. In this presentation we suggest to produce strong edge hints for the Weak Membrane through a Snake deformable model. The Snake is defined to be a periodic set of N points in the visual field of an image: $\mathbf{r}_j = (x_j, y_j)$. Periodicity meaning $\mathbf{r}_{j+pN} = \mathbf{r}_j, p \in \mathbb{Z}$. The Snake energy function consists in a form control part E_{form} and a match part E_{match} .

$$E = E_{\text{form}} + E_{\text{match}} \quad (12)$$

In this simple version the form control preserves total length,

$$E_{\text{form}} = \frac{a}{2} (d - d_0)^2 \quad (13)$$

$$d = \sum_j |\mathbf{r}_j - \mathbf{r}_{j+1}| \quad (14)$$

where d_0 is the initial Snake length. More complex form controls can be designed that preserve shape, corners, etc. The Snake match energy is designed to ensure that the Snake points track edge contours in the image field. This effect may be obtained by letting the Snake points seek local maxima in the gradient energy map of the image, $G(\mathbf{r})$. The total Snake match energy is then given by,

$$E_{\text{match}} = - \sum_j G(\mathbf{r}_j). \quad (15)$$

Snake dynamics is established through gradient descent,

$$\frac{1}{\eta} \frac{\partial \mathbf{r}_j}{\partial t} = -\frac{\partial E}{\partial \mathbf{r}_j} = a(d - d_0) \frac{\partial d}{\partial \mathbf{r}_j} + \frac{\partial G(\mathbf{r}_j)}{\partial \mathbf{r}_j} \quad (16)$$

3 Experiments and concluding remarks

In numerous image processing applications quite detailed prior information can be devised. In, e.g., brain scan reconstruction detailed atlases are known describing the generic brain topography under various scanning modes. To illustrate the potential in brain scans of using strong edge hints we want to reconstruct a head phantom shown in Figure 1. This is done from a noise corrupted image shown in Figure 2. Under the reconstruction process we also want to estimate edges.

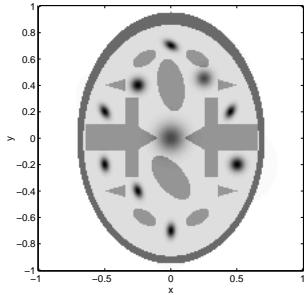


Figure 1: Original head phantom without noise.

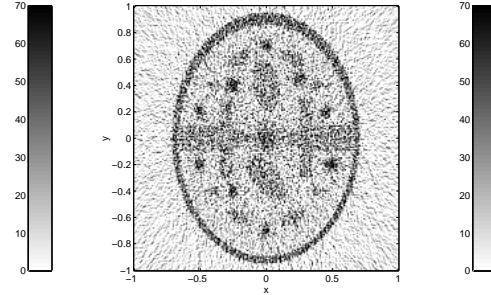


Figure 2: Noise corrupted head phantom shown in same color scale.

Snakes are initialized in seven generic positions, shown in Figure 3. For the particular instance the Snake equilibrates in about 100 iterations as illustrated in Figure 4. The approach to equilibrium is quite sensitive to the topography of the image and careful control of η is necessary. This problem may be relieved by use of a pseudo-second order search direction instead of the gradient of the edge energy map.

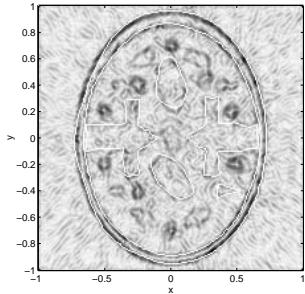


Figure 3: Edge energy map with initial snake positions shown in white.

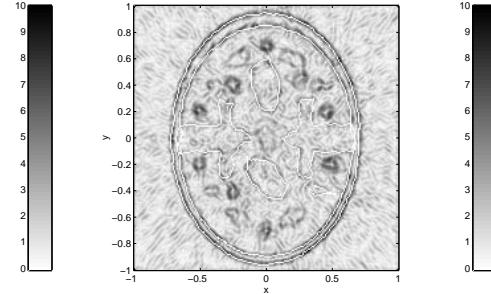


Figure 4: Edge energy map with equilibrated Snake positions shown in white.

Subsequently, hints are created by modulating the chemical potentials $\mu_{m,n}^h, \mu_{m,n}^v, \gamma_{m,n}$. This modulation imply that edge units are strongly suppressed outside of the region suggested by the Snakes. Finally we relax the cellular network as defined in Equation (8).

The result of the cellular net is presented in Figures 5 (without Snakes) and 6 (using Snakes). Note that the intensity level inside the region of the Snake has been estimated more closely using Snakes. This might be of significant importance in, e.g., brain activation studies, in which the minute differences in brain activity between activated and resting states are investigated. Using Snakes enhances the Signal to Noise ratio (SNR) with 12.5 dB. Without Snakes the improvement is 11.3 dB. The improvement is due to a better estimation of edges.

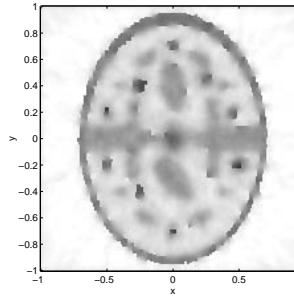


Figure 5: Restored head phantom using Mean Field Annealing.

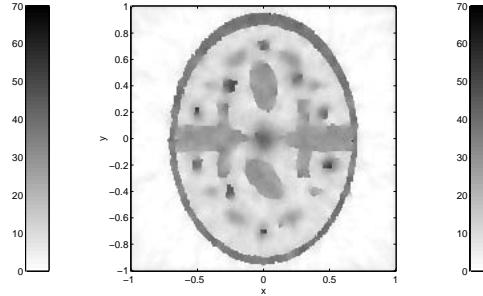


Figure 6: Restored head phantom using Snakes as edge priors to the Mean Field Annealing.

In Figures 7 and 8 are shown the active edges without using Snakes respectively with Snakes. As seen from Figure 8 it is possible to incorporate strong priors in order to obtain closed edge contours. This can be used to segment the image.

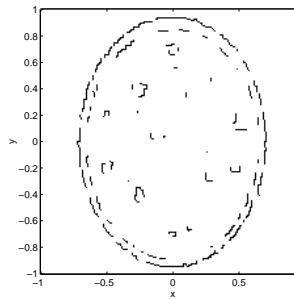


Figure 7: Estimated edges without using snakes. Active edges are marked black.

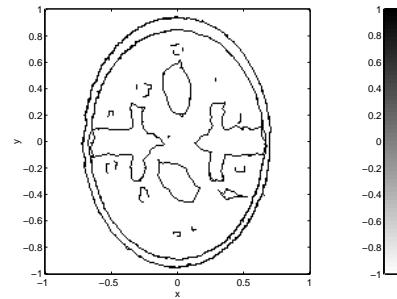


Figure 8: Estimated edges using Snakes as priors

Another way of evaluating the results is to examine the histograms of the images. Figure 9 shows the histogram of the original noise free image and Figure 10 shows, in the same interval, the histogram of the noise corrupted image.

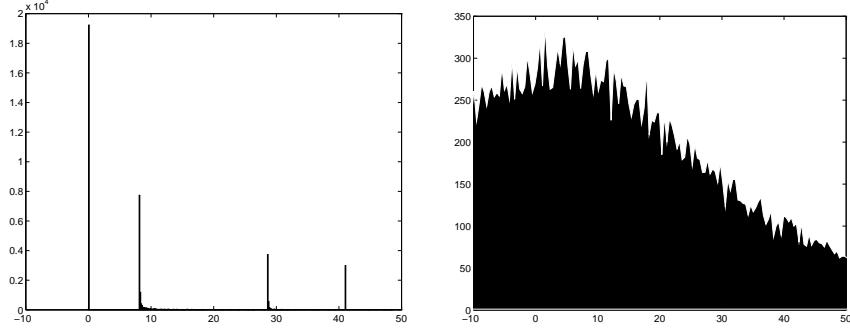


Figure 9: Histogram of original noise free head phantom.

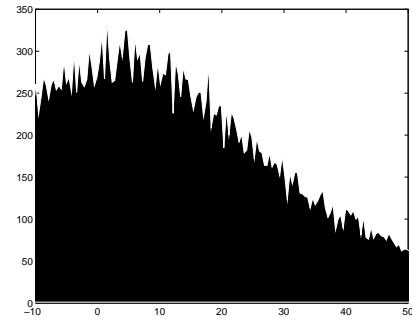


Figure 10: Histogram of the observed noise corrupted head phantom.

Without using Snakes the histogram after restoration can be seen in Figure 11. Finally Figure 12 shows the histogram corresponding the restored image using Snakes as a prior. From the Figures it can be seen that the two higher levels (29 and 41) is better resolved using Snakes and compared to the noisy histogram shown in Figure 10 the result is far better.

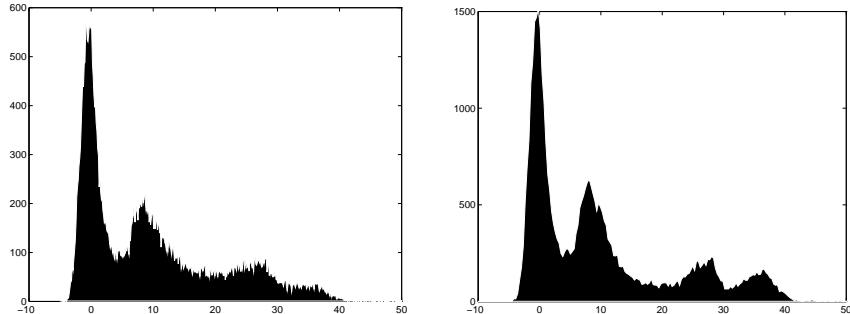


Figure 11: Histogram of the restored image without using Snakes.

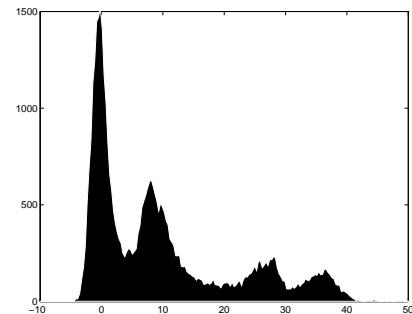


Figure 12: Histogram of the restored image using Snakes as priors.

In conclusion we have shown that strong structural priors may be introduced in the Weak Membrane model by invoking deformable models like Snakes. The present study was based on a head phantom; we are currently pursuing the viability of the approach in the context of Positron-Emission-Tomography.

Acknowledgment

This research is supported by the Danish Research Councils for the Natural and Technical Sciences through the Danish Computational Neural Network Center.

References

- [1] A. Blake: *Comparison of the Efficiency of Deterministic and Stochastic Algorithms for Visual Reconstruction*. IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-11** 2-12 (1989).
- [2] L. Cohen and I. Cohen: *Finite Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images*. IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-15** 1133-1147 (1993).
- [3] X. Ouyang, W.H. Wong, V.E. Johnson, X. Hu, C.-T. Chen: *Incorporation of Correlated Structural Images in PET Image Reconstruction*. IEEE Transactions on Medical Imaging **13** 627-640 (1994).
- [4] D. Geman and S. Geman: *Stochastic Relaxation, Gibbs distributions and the Bayesian restoration of images*. IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-6** 721-741 (1984).
- [5] J. Hertz, A. Krogh and R.G. Palmer: *Introduction to the Theory of Neural Computation*. Addison Wesley, New York (1991).
- [6] C. Peterson and J.R. Anderson: *A Mean Field Learning Algorithm for Neural Networks*. Complex Systems **1** 995-1019, (1987).
- [7] M.W. Roth: *Survey of Neural Network Technology for Automatic Target Recognition*. IEEE Transactions on Neural Networks **1**, 28-43, (1990).

Appendix O

Detection of Lines with Wiggles using the Radon Transform

This appendix contains the paper *Detection of Lines with Wiggles using the Radon Transform*, by Peter Toft [4].

The paper has been submitted to the *NORSIG'96 - 1996 IEEE Nordic Signal Processing Symposium* in Espoo, Finland.

Detection of Lines with Wiggles using the Radon Transform

Peter Toft

Department of Mathematical Modelling, Technical University of Denmark,
DK-2800 Lyngby, Denmark, Email pto@imm.dtu.dk.

Abstract

The discrete Radon transform is a useful tool in image processing for detection of lines (or in general curves) in digital images.

One of the key properties of the discrete Radon transform is that a line in an image is transformed into a peak in the parameter domain, where the position of the peak corresponds to the line parameters.

What often is needed, is to determine whether a Radon transform based curve detection algorithm will work in presence of noise. This paper regards detection of lines in images, where the lines are assumed to have wiggles. A theoretical analysis is given providing analytical expressions for this kind of noise.

I. INTRODUCTION

The Radon transform can be defined in various ways. In general image processing the lines are often parameterized using normal parameters [1], and in seismic signal processing the τ - p transform or slant stack version of the Radon transform use slope p and offset τ to parameterize the lines [2, 3]. Here the last definition will be used.

$$\check{g}(p, \tau) = \int_{-\infty}^{\infty} g(x, px + \tau) dx \quad (1)$$

One of the key properties of the (discrete) Radon transform is that a line in an image is transformed into a peak in the parameter domain [4], where the position of the peak corresponds to the line parameters. In this way, the Radon transform converts a difficult global detection problem in the image domain into a more easily solved local peak detection problem in the parameter domain.

For a discrete image $g(m, n)$, a nearest neighbour approximation of Eq. 1 will be used

$$\check{g}(\alpha, \beta) = \sum_{m=0}^{M-1} g(m, [\alpha m + \beta]) \quad (2)$$

where $[\cdot]$ denotes rounding to nearest neighbour. Lately analytical expressions for the probability of detecting a

curve in presence of additive noise has been analyzed [5], and here another kind of noise is considered, namely that the lines in the images might not be perfectly linear but include some random misalignment, here called wiggles.

II. LINES WITH WIGGLES

Assuming that a line in the image can be modelled as

$$g(m, n) = \delta(n - [\alpha^* m + \beta^* + \lambda]) \quad (3)$$

where $\delta(\cdot)$ is the Kronecker delta function, i.e., the Gaussian distributed noise term $\lambda \in \mathcal{N}(0, \sigma^2)$ determines the change in position of the line in the n -direction, and it will be assumed that the noise terms λ are uncorrelated as a function of m .

What now is considered, is to find the average value (and shape) of the peak (if any is found) in the discrete parameter domain as a function of the noise deviation σ . Eq. 3 implies that the probability of the sample $g(m, n)$ being 1 is given by

$$P\{g(m, n) = 1\} \quad (4)$$

$$= P\{n = [\alpha^* m + \beta^* + \lambda]\} \quad (5)$$

$$= P\left\{-\frac{1}{2} < n - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \quad (6)$$

and assuming that the sample point (m, n) is given from the nearest neighbour mapping in the discrete Radon transform, $n = [\alpha m + \beta]$, then the rounding function is modelled as an additive (noise) term ω

$$P\{g(m, n) = 1\} \quad (7)$$

$$= P\left\{-\frac{1}{2} < [\alpha m + \beta] - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \quad (8)$$

$$= P\left\{-\frac{1}{2} < \alpha m + \beta + \omega - \alpha^* m - \beta^* - \lambda < \frac{1}{2}\right\} \quad (9)$$

$$= P\left\{-\frac{1}{2} < \zeta + \omega - \lambda < \frac{1}{2}\right\} \quad (10)$$

$$= \Phi\left(\frac{\frac{1}{2} + \zeta + \omega}{\sigma}\right) - \Phi\left(\frac{-\frac{1}{2} + \zeta + \omega}{\sigma}\right) \quad (11)$$

$$\zeta = (\alpha - \alpha^*)m + \beta - \beta^* \quad (12)$$

where $\Phi(\cdot)$ is the Gaussian probability function, and ζ is a displacement between the true line parameters (α^*, β^*) and the parameters (α, β) at a given position m .

Due to many values of m used in Eq. 2, it is a reasonable approximation to model ω as a uniformly distributed variable between the limits $-1/2$ and $1/2$, i.e., $\omega_m \in \mathcal{U}(-1/2, 1/2)$. Eq. 11 implies that the average contribution from the pixel $g(m, n)$ to the discrete Radon transform is given by

$$E\{\check{g}(\alpha, \beta)|g(m, n)\} = \int_{\omega=-\frac{1}{2}}^{\omega=\frac{1}{2}} P_m(\omega) d\omega \quad (13)$$

$$= \int_{\omega=-\frac{1}{2}}^{\omega=\frac{1}{2}} \Phi\left(\frac{\frac{1}{2} + \zeta + \omega}{\sigma}\right) - \Phi\left(\frac{-\frac{1}{2} + \zeta + \omega}{\sigma}\right) d\omega \quad (14)$$

In order to get an analytical expression for $E\{\check{g}(\alpha, \beta)|g(m, n)\}$ the Gaussian probability function is approximated

$$\Phi(x) \approx \frac{1}{2} \left(1 + \tanh\left(\frac{x}{\gamma}\right)\right) \text{ where } \gamma = \sqrt{\frac{\pi}{2}} \quad (15)$$

i.e., the integral of $\Phi(\cdot)$ can be approximated by

$$\int \Phi(x) dx \approx \frac{\gamma}{2} \left(\frac{1+x}{\gamma} + \log \cosh\left(\frac{x}{\gamma}\right) \right) \quad (16)$$

where $\log(\cdot)$ is the natural logarithm.

If Eq. 16 is inserted into Eq. 14, followed by some rearrangements of the expressions, it is found that

$$E\{\check{g}(\alpha, \beta)|g(m, n)\} = \frac{\gamma\sigma}{2} \log \left(\frac{\cosh\left(\frac{2\zeta}{\gamma\sigma}\right) + \cosh\left(\frac{2}{\gamma\sigma}\right)}{\cosh\left(\frac{2\zeta}{\gamma\sigma}\right) + 1} \right) \quad (17)$$

In Fig. 1 the average weight to the discrete Radon transform is shown as a function of σ and ζ .

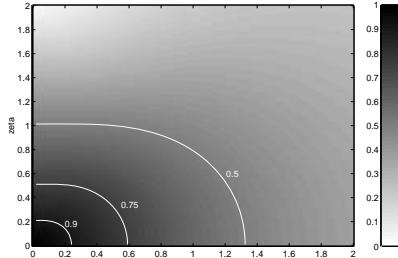


Figure 1 The average weight to the discrete Radon transform $E\{\check{g}(\alpha, \beta)|g(m, n)\}$ as a function of the noise deviation σ on the first axis, and the displacement ζ on the second axis.

A special case of interest regards $\zeta = 0$, i.e., where the coordinates of the line in the image matches exactly a sample in the discrete parameter domain.

$$\zeta = 0 \Rightarrow E\{\check{g}(\alpha, \beta)|g(m, n)\} \quad (18)$$

$$= \sigma \sqrt{\frac{\pi}{8}} \log \left(\frac{1}{2} \left(1 + \cosh\left(\frac{\sqrt{8}}{\sigma\sqrt{\pi}}\right) \right) \right) \quad (19)$$

In Fig. 2 the average weight to the discrete Radon transform is shown as a function of σ , when the displacement is assumed negligible. It can be seen that the function agrees well with a simulated result shown in Fig. 3, that is found by generating 10000 images with the correct noise amplitude σ and computing the discrete Radon transform only for the parameter set matching the true parameters. An even better agreement can be found using a numerical integration of Eq. 14, but Eq. 16 provides an simple analytical result that approximates the simulated results well, though it can also be seen that the theoretical model has a small bias, when $\sigma \approx 0.1$.

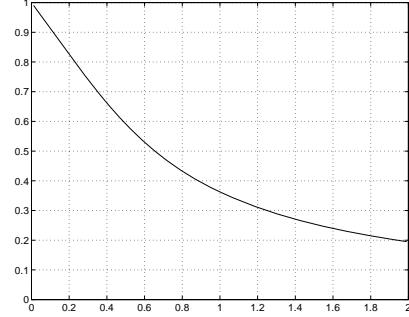


Figure 2 The theoretical result of the average weight to the discrete Radon transform, found in Eq. 19, as a function of σ when the displacement ζ is negligible.

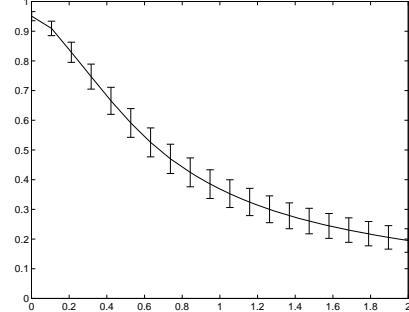


Figure 3 The simulated average value of the peak using 10000 images for each noise level. The vertical error bars show the measured deviation.

Fig. 2 or Eq. 19 can be used to predict whether line parameters might be estimated given that the lines have wiggles. If, e.g., demanding that the peak value cannot decrease more than 50% due to wiggles, then σ cannot exceed approximately 0.7. And a 25% decrease of the peak value is found if $\sigma \approx 0.25$.

III. EXAMPLES

In Fig. 4 is shown an image containing one line without noise, and Fig. 5 shows the discrete parameter domain in the area of the peak. Next, Fig. 6 shows an image where $\sigma = 0.5$, and Fig. 7 shows the corresponding discrete parameter domain. On the figure-axes, the continuous variables, indicated in Eq. 1, have been used. This merely imply a linear scaling of the discrete parameters used in Section II.. The images have been scaled individually according to the minimal and maximal values, and it can be seen that the maximal value here is around 65, and the shape is more uneven than seen from Fig. 5. This value lies within the error bars shown in Fig. 3.

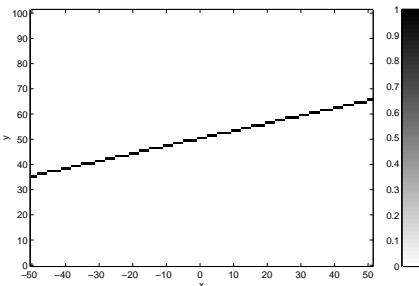


Figure 4 An image containing one line without any kind of noise.

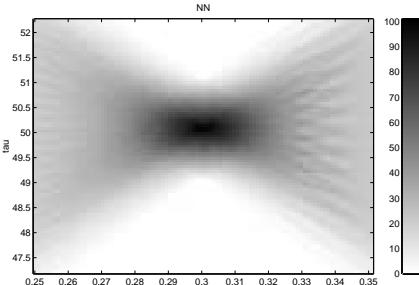


Figure 5 The discrete Radon transform of the image shown in Fig. 4 zoomed in at the peak.

IV. CONCLUSION

The conclusion is that a peak is generated in the discrete parameter domain corresponding to each of the wiggly line in the image, and the value of the peak will be lowered corresponding to the noise level. Analytical expressions have been given to determine the actual reduction of the peak value as a function of the noise level.

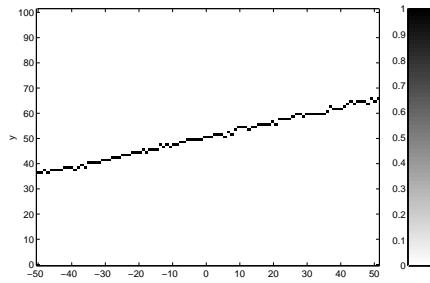


Figure 6 An image containing one wiggly line where $\sigma = 0.5$.

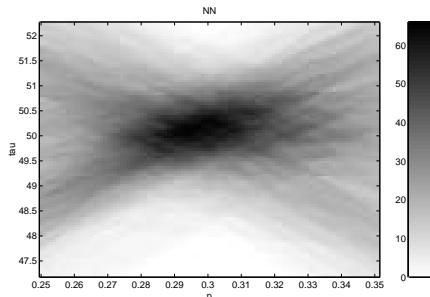


Figure 7 The discrete Radon transform of the image shown in Fig. 6 zoomed in at the peak.

REFERENCES

- [1] S. R. Deans. *The Radon Transform and Some of Its Applications*. Krieger Publishing Company, Malabar, Florida, 2 edition, 1993. Previously John Wiley & Sons Inc., 1983.
- [2] Gregory Beylkin. Discrete Radon Transform. *IEEE Tr. ASSP*, 35(2):162–172, 1987.
- [3] T. S. Durrani and D. Bisset. The Radon Transform and its Properties. *Geophysics*, 49(8):1180–1187, Aug. 1984. Correction note in vol. 50, no. 5, pp. 884–886, May 1985.
- [4] Peter Toft. *The Radon Transform - Theory and Implementation*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, 1996.
- [5] Peter Toft. Using the Generalized Radon Transform for Detection of Curves in Noisy Images. In *Proceedings*, volume 4, pages 2221–2225. IEEE ICASSP, May 1996.

Thesis Bibliography

- [1] K. V. Hansen and P. A. Toft. Fast Curve Estimation Using Pre-Conditioned Generalized Radon Transform. *IEEE Image Processing*, 1996. To appear in IEEE Image Processing. Preprint Department of Mathematical Modelling, Technical University of Denmark, DK2800 Lyngby.
- [2] P. A. Toft and K. V. Hansen. Fast Radon Transform for Detection of Seismic Reflections. In *Signal Processing VII - Theories and Applications*, volume I, pages 229–232. EURASIP EUSIPCO94, 1994.
- [3] Peter Toft. Using the Generalized Radon Transform for Detection of Curves in Noisy Images. In *Proceedings*, volume 4, pages 2221–2225. IEEE ICASSP, May 1996.
- [4] Peter Toft. Detection of Lines with Wiggles using the Radon Transform. *Submitted to NOR-SIG'96*, 1996. Preprint from Department of Mathematical Modelling, Technical University of Denmark.
- [5] Peter Toft and Jesper James Jensen. A Very Fast Implementation of 2D Iterative Reconstruction Algorithms. *Submitted IEEE Medical Imaging Conference*, 1996.
- [6] Peter Toft and Jesper James Jensen. Accelerated 2D Iterative Reconstruction. *Submitted*, 1996. Preprint from Department of Mathematical Modelling, Technical University of Denmark.
- [7] Søren Holm Peter Toft and Mikael Jensen. Estimation of the noise contributions from Blank, Transmission and Emission scans in PET. In *To appear in proceedings*. IEEE MIC, October 1995.
- [8] Søren Holm, Peter Toft, and Mikael Jensen. Estimation of the noise contributions from Blank, Transmission and Emission scans in PET. *IEEE Transactions on Nuclear Science*, 1996. Submitted.
- [9] Technical University of Denmark. Section for Digital Signal Processing Department of mathematical Modelling. Homepage of the danish part of the Human Brain Project. <http://hendrix.ei.dtu.dk>.
- [10] Peter Toft. Homepage of Peter Toft. Section for Digital Signal Processing IMM. <http://www.ei.dtu.dk/staff/ptoft/ptoft.html>.
- [11] Peter Alshede Philipsen, Lars Kai Hansen and Peter Toft. Mean Field Reconstruction with Snaky Edge Hints. In *To appear in INVERSE METHODS - Interdisciplinary elements of Methodology, Computation and Application*, 1995.

- [12] Peter Alshede Philipsen, Lars Kai Hansen and Peter Toft. Mean Field Reconstruction with Snaky Edge Hints. In Peter Johansen, editor, *Proceedings fra den Fjerde Danske Konference om Mønsterkendelse og Billedanalyse*. DIKU, August 1995.
- [13] J. Radon. Über die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten. *Ber. Ver. Sächs. Akad. Wiss. Leipzig, Math-Phys. Kl.*, 69:262–277, April 1917. In German. An english translation can be found in S. R. Deans: *The Radon Transform and Some of Its Applications*, app. A.
- [14] S. R. Deans. *The Radon Transform and Some of Its Applications*. Krieger Publishing Company, Malabar, Florida, 2 edition, 1993. Previously John Wiley & Sons Inc., 1983.
- [15] S. R. Deans. Hough Transform From the Radon Transform. *IEEE PAMI*, 3(2):185–188, 1981.
- [16] Peter Toft. Radontransformation. Lecture Note for DTU course 4235: Advanced Digital Signal Processing, Electronics Institute, Technical University of Denmark, 1994. In Danish.
- [17] J. Canny. A Computational Approach to Edge Detection. *IEEE Trans. PAMI*, 8(6):679–698, 1986.
- [18] R. Deriche. Using Canny's Criteria to Derive a Recursively Implemented Optimal Edge Detector. *International Journal of Computer Vision*, 2(1):167–187, 1987.
- [19] Özdogan Yilmaz. *Seismic Data Processing*. Society of Exploration Geophysicists, Tulsa, Oklahoma, 1987.
- [20] T. S. Durrani and D. Bisset. The Radon Transform and its Properties. *Geophysics*, 49(8):1180–1187, Aug. 1984. Correction note in vol. 50, no. 5, pp. 884–886, May 1985.
- [21] Athanasios Papoulis. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 2. edition, 1984.
- [22] Peter Toft. RadonAna: A C-based Package for Analytically Based Radon Transformation. Technical report, Electronics Institute, DTU, 1995.
- [23] P. V. C. Hough. A Method and Means for Recognizing Complex Patterns. US Patent: 3,069,654, Dec. 1962.
- [24] R. O. Duda and P. E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communication of the Association for Computing Machinery*, 15(1):11–15, 1972.
- [25] J. Illingworth and J. Kittler. A Survey of the Hough Transform. *Computer Vision, Graphics, and Image Processing*, 44:87–116, 1988.
- [26] V. F. Leavers. *Shape Detection in Computer Vision Using the Hough Transform*. Springer-Verlag, 1992.
- [27] Wilson Lam et al. An Analysis on Quantizing the Hough space. *Pattern Recognition Letters*, 15:1127–1135, Nov. 1994.
- [28] J. Princen et al. A Comparison of Hough Transform Methods. In *Third International Conference on Image Processing and its Applications*, pages 73–77. IEE, 18-20 July 1989.

- [29] J. Princen et al. A Formal Definition of the Hough Transform: properties and Relationships. *Journal of Mathematical Imaging and Vision*, 1:153–168, 1992.
- [30] Lei Xu et al. A new curve detection method: Randomized Hough Transform (RHT). *Pattern Recognition Letters*, 11(5):331–338, 1990.
- [31] Keikke Kälviäinen et al. Probabilistic and Non-probalistic Hough Transforms: Overview and Comparisons. In *Randomized Hough Transform: New Extentions*, number 35 in Research Letters. Lappeenranta University of Technology, 1994. email: Heikki.Kalvianen@lut.fi.
- [32] Katja Blankensteiner og Peter Toft. Development and Implementation of a Fast and Robust Algorithm for Estimation of the Generalized Radon Transform for Seismical Signal Processing. Master's thesis, Electronics Institute. Technical University of Denmark, 1993. In danish.
- [33] Piang Liang. A New and Efficient Transform for Curve Detection. *Journal of Robotic Systems*, 8(6):841–847, 1991.
- [34] D. H. Ballard. Generalizing the Hough Transform to detect arbitrary shapes. *Pattern Recognition Letters*, 13(2):111–122, 1981.
- [35] H. Li, M. A. Lavin and R. J. Le Master. Fast Hough Transform: A Hierachical Approach. *Computer Vision, Graphics, and Image Processing*, 36:139–161, 1986.
- [36] Gregory Beylkin. Discrete Radon Transform. *IEEE Tr. ASSP*, 35(2):162–172, 1987.
- [37] C. J. Haneveld and G. C. Herman. A Fast Algorithm for Computation of Radon Transforms. *Geophysical Prospecting*, 38:853–860, 1990.
- [38] W. A. Schneider. The Common Depth Point Stack. *Proceedings of the IEEE*, 72(10):1238–1254, 1984.
- [39] E. A. Robinson, T. S. Durrani and L. G. Peardon. *Geophysical Signal Processing*. Englewood Cliffs, New Jersey, Prentice-Hall Inc., 1980.
- [40] B. S. Atal and J. R. Remde. A New Model of LPC Excitation for Producing Natural-Sounding Speech at Low Bit Rates. In *Proceedings of ICASSP*, volume 1 of *ICASSP*, pages 614–617. IEEE, 1982.
- [41] M. Cookey, H. J. Trussel and I. J. Won. Seismic Deconvolution by Multipulse Methods. *IEEE Trans. ASSP*, 38(1):156–160, 1990.
- [42] Kim Vejlby Hansen. *Detection of Primary Reflection Curves in CMP gathers*. PhD thesis, Electronics Institute, Technical University of Denmark, Joint work with Ødegaard & Danneskiold-Samsøe., 1994.
- [43] George A. McMechan and Richard Ottolini. Direct observation of a $p\text{-}\tau$ curve in a slant stacked wave field. *Bulletin of the Seismological Society of America*, 70(3):775–789, 1980.
- [44] Robert Phinney et al. Transformation and Analysis of Record Sections. *Journal of Geophysical Research*, 86(B1):359–377, Jan. 1981.
- [45] John B. Diebold and Paul L. Stoffa. The Travelttime Equation, tau-p mapping, and inversion of common midpoint data. *Geophysics*, 46(3):238–254, March 1981.

- [46] Friedemann Wenzel et al. Seismic modelling in the domain of intercept time and ray parameter. In The Multidimensional Signal Processing Society, editor, *Selected Papers in Multidimensional Digital Signal Processing*, pages 377–393. IEEE Acoustics, Speech, and Signal Processing Society, 1986.
- [47] Paul Stoffa et al. Direct mapping of seismic data to the domain of intercept time and ray parameter - a plan-wave decomposition. *Geophysics*, 46(3):255–267, March 1981.
- [48] K. V. Hansen and J. Larsen. An Algorithm for Successive Identification of Reflections. *IEEE IP*, 3(3):281–291, May 1994.
- [49] K. Y. Huang *et al.* Image Processing of Seismograms: (A) Hough Transformation for the Detection of Seismic Patterns; (B) Thinning Processing in the Seismogram. *Pattern Recognition*, 18(6):429–440, 1985.
- [50] C. Y. Chi, J. M. Mendel and D. Hampson. A Computationally Fast Approach to Maximum-Likelihood Deconvolution. *Geophysics*, 49(5):550–565, 1984.
- [51] A. Blake and A. Zisserman. Comparison of the Efficiency of Deterministic and Stochastic Algorithms for Visual Reconstruction. *IEEE Trans. Neural Networks*, 11(1):2–12, 1989.
- [52] Michel Lavergne. *Seismic Methods*. Graham & Trotman Limited, Sterling House, London, 2 edition, 1989. Also at Krieger Inc.
- [53] Bjørn Ursin. Seismic velocity estimation. *Geophysical Prospecting*, 25:658–666, 1977.
- [54] C. Hewitt Dix. Seismic Velocities from Surface Measurements. *Geophysics*, XX(1):68–86, Jan. 1955.
- [55] Enders A. Robinson and Eduard J. Douze. Ray tracing and seismic modelling. In *Proceedings of the Fourteenth International Symposium*, volume 14 of *Acoustical Imaging*, pages 169–186, 1985.
- [56] Lesley M. Murphy. Linear feature detection and enhancement in noisy images via the Radon transform. *Pattern Recognition Letters*, 4:279–284, Sep. 1986.
- [57] Department of Molecular and UCLA Medical Pharmacology. Homepage: Let's Play PET. <http://www.nuc.ucla.edu/lpp/lpphome.html>.
- [58] J. Carlson et al. Fundamentals of medical imaging. Notes for the tutorial course with title Fundamentals of Medical Imaging held at the IEEE Medical Imaging Conference (MIC) 1995., 1995.
- [59] Jesper J. Jensen and Peter A. Philipsen. Direkte Methods til rekonstruktion af PET billede. Master's thesis, Electronics Institute. Technical University of Denmark, 1995. In danish.
- [60] Jesper J. Jensen. Iterative Methoder til rekonstruktion af PET billede. Master's thesis, Electronics Institute. Technical University of Denmark, 1995. In danish.
- [61] Peter A. Philipsen. Preprocessering og postprocessering af PET billede. Master's thesis, Electronics Institute. Technical University of Denmark, 1995. In danish.
- [62] Caroline Axelsson. *Direct Fourier Methods in 3D-Reconstruction from Cone-Beam Data*. PhD thesis, Linköping University, Dep. of Electrical Engineering, S-581 83 Linköping, Sweden, Jan. 1994. E-mail caroline@isy.liu.se.

- [63] Frank Natterer. *The Mathematics of Computerized Tomography*. John Wiley & Sons, 2 edition, 1986.
- [64] S. S. Orlov. Theory of three dimensional reconstruction. 1. Conditions for a complete set of projections. *Sov. Phys. Crystallogr.*, 20(3):312–314, 1975.
- [65] S. S. Orlov. Theory of three dimensional reconstruction. The recovery operator. *Sov. Phys. Crystallogr.*, 20(4):429–433, 1975.
- [66] James G. Colsher. Fully three-dimensional positron emission tomography. *Physics in Medicine and Biology*, 25(1):103–115, 1980.
- [67] R. Clack. Towards a complete description of three-dimensional filtered backprojection. *Physics in Medicine & Biology*, 37(3):645–660, March 1992.
- [68] R. Clack, D. Townsend and M. Defrise. An Algorithm for three-dimensional Reconstruction Incorporating Cross-Plane Rays. *IEEE Trans. Med. Imag.*, MI-8(1):32–42, 1989.
- [69] Charles Stearns. *Accelerated Image Reconstruction For A Cylindrical Positron Tomograph Using Fourier Domain Methods*. PhD thesis, Massachusetts Institute of Technology, 1990.
- [70] J. G. Rogers, R. Harrop and P. E. Kinahan. The Theory of Three-Dimensional Image Reconstruction for PET. *IEEE Trans. Med. Imag.*, MI-6(3):239–243, September 1987.
- [71] P. E. Kinahan and J. G. Rogers. Analytic 3D Image Reconstruction Using All Detected Events. *IEEE Transactions on Nuclear Science*, 36(1):964–968, Feb. 1989.
- [72] Maria Magnusson. *The Linogram Method for Image Reconstruction from Projections*. PhD thesis, Linköpings tekniska högskola, Dep. of Electrical Engineering, S-581 83 Linköping, Sweden, 1889. Revised version 1993.
- [73] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, Baltimore and London second, 2 edition, 1989.
- [74] A. Mallon and P. Grangeat. Three-dimensional PET reconstruction with time-of-flight measurements. *Physics in Medicine & Biology*, 37(2):717–29, 1992.
- [75] Peter Toft. Using the Inverse Radon transform in PET. Technical report, Technical University of Denmark, Electronics Institute, DK2800 Lyngby, 1994.
- [76] Michael Lautsch. A Spline Inversion Formula for the Radon Transform. *SIAM J. Numer. Anal.*, 26(2):456–467, 1889.
- [77] Yves Nievergelt. Elementary Inversion of Radon's Transform. *SIAM Review*, 28(1):79–84, March 1986.
- [78] R. M. Mersereau. Recovering Multidimensional Signals from their Projections. *Computer Graphics and Image Processing*, 1(4):179–195, Dec. 1973.
- [79] Dan E. Dudgeon and Russel M. Mersereau. *Multidimensional Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, N.J, 1984.
- [80] L. A. Shepp and J. B. Krustal. Computerized tomography: The new medical x-ray technology. *Am. Math. Monthly*, 85:420–439, April 1978.

- [81] A. Rosenfeld and A. C. Kak. *Digital Picture Processing*, volume 1. Academic Press, 2. ed. edition, 1982.
- [82] Sigurdur Helgason. *The Radon Transform*, volume 5 of *Progress in Mathematics*. Birkhäuser, Boston Basel Stuttgart, 1980.
- [83] P. A. M. Dirac. *The Principles of Quantum Mechanics*. Oxford University Press, Oxford, 1984.
- [84] Gregory Beylkin. The Inversion Problem and Application of the Generalized Radon Transform. *Communications on Pure and Applied Mathematics*, XXXVII:579–599, 1984.
- [85] C. H. Chapman. Generalized Radon transforms and slant stacks. *Geophys. J. R. ast. Soc.*, 66:445–453, 1981.
- [86] Lennart Råde and Bertil Westergren. *BETA: Mathematics Handbook*. Chartwell-Bratt Ltd, 1980.
- [87] A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Englewood Cliffs, New Jersey, Prentice-Hall Inc., 1989.
- [88] Simon Boel Pedersen. *Signalanalyse III*. Elektronisk Institut, DTU, 1990. In danish.
- [89] Peter Koefod Møller. *Den hurtige Fouriertransformation (FFT)*. Electronics Institute, DTU, 1991. In Danish.
- [90] William H. Press *et. al.* *Numerical Recipes in C*. Cambridge University Press, 2. edition, 1992.
- [91] Paul Kinahan. Personal Communication.
- [92] P. R. Edholm and G. T. Herman. Linograms in Image Reconstruction from Projections. *IEEE Trans. Med. Imag.*, 6(7):301–307, Dec. 1987.
- [93] Kevin J. Moraski and David C. Munson, Jr. Fast tomographic reconstruction using chirp-z interpolation. *IEEE Trans. Med. Imag.*, 1991.
- [94] L. R. Rabiner et al. The Chirp z-Transform Algorithm. *IEEE Trans. Audio and Electroacoustics*, AU-17(2):86–92, 1969.
- [95] L. I. Bluestein. A linear filtering approach to the computation of the discrete Fourier transform. *NEREM Rec.*, pages 218–219, 1968.
- [96] Yair Censor. Finite Series-Expansion Reconstruction Methods. In *Proc. of the IEEE*, volume 71, pages 409–419, March 1983.
- [97] Julia K. Older and Paul C. Johns. Matrix formulation of computed tomography reconstruction. *Physics in Medicine & Biology*, 38:1051–1064, 1993.
- [98] Robert M. Lewitt. Alternatives to voxels for image representation in iterative reconstruction algorithms. *Phys. Med. Biol.*, 37(2):706–716, 1992.
- [99] Jules S. Jaffe. Limited Angle Reconstruction Using Stabilized Algorithms. *IEEE Trans. Med. Imag.*, 9(3):338–344, Sep. 1990.

- [100] S. Kawata and O. Nalcioglu. Constrained iterative Reconstruction by the Conjugate Gradient Method. *IEEE Trans. Med. Imag.*, MI-4(2):65–71, June 1985.
- [101] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems*. Polyteknisk Forlag, Lyngby, Denmark, 1996. Doctoral Dissertation.
- [102] G. H. Golub and C. F. Van Loan. *Matrix Computations*, volume 4. The Johns Hopkins University Press, Baltimore and London second, 1991.
- [103] Per Christian Hansen. Personal Communication.
- [104] Mark F. Smith. Simultaneously constraining spect activity estimates with primary and secondary energy window projection data. *IEEE Trans. Med. Imag.*, 13(2), June 1994.
- [105] Mark F. Smith. Generalized matrix inverse reconstruction for spect using a weighted singular value spectrum. In *Abstract Book of IEEE Nuclear Science Symposium and Medical Imaging Conference*, page 161. IEEE, 1995. Also see the Conference Issue.
- [106] Gabor T. Herman and Lorraine B. Meyer. Algebraic Reconstruction Techniques Can Be Made Computationally Efficient. *IEEE Trans. Med. Imag.*, 12(3):600–609, Sep. 1993.
- [107] S. Kaczmarz. Angenährte Auslösung von Systemen linearer Gleichungen. *Bull. Int. Acad. Pol. Sci. Lett. A*, 35:355–357, 1937.
- [108] Huaiqun Guan and Richard Gordon. A projection access order for speedy convergence of ART (algebraic reconstruction technique): a multilevel scheme for computed tomography. *Phys. Med. Biol.*, 39:2005–2022, 1994.
- [109] G. T. Herman. *Image Reconstruction from Projections: The fundamentals of Computerized Tomography*. Academic Press, 1980.
- [110] Paul E. Kinahan and Joel Karp. Determining when 3D reconstruction methods are required in volume PET imaging. In *Conference Record*, volume 2, pages 904–906. IEEE Nuclear Science Symposium & Medical Imaging Conference, 1992.
- [111] Linda Kaufman. Implementing and Accelerating the EM Algorithms for Positron Emission Tomography. *IEEE Trans. Med. Imag.*, MI-6(1):37–51, march 1987.
- [112] L. A. Shepp Y. Vardi and L. Kaufman. A Statistical Model for Positron Emission Tomography. *Journal of the American Statistical Association*, 80(389):8–37, March 1985. The pages include comments and discussion by several authors.
- [113] R. E. Carson and K. Lange. The EM parametric image reconstruction algorithm. *Journal of the American Statistical Association*, 389(80):20–25, March 1985.
- [114] G. T. Herman and D. Odhner. Performance evaluation of an iterative image reconstruction algorithm for positron emission tomography. *IEEE Trans. Med. Imag.*, 10:336–246, 1991.
- [115] Linda Kaufman. Maximum Likelihood, Least Squares, and Penalized Least Squares for PET. *IEEE Trans. Med. Imag.*, 12(2):200–214, June 1993.
- [116] John M. Ollinger. Maximum-Likelihood reconstruction of transmission images in emission computed tomography via the EM algorithm. *IEEE Trans. Med. Imag.*, 13(1):89–101, March 1994.

- [117] John A. Scales. Iterative methods for large, sparse, inverse calculations. Samizdat Press. Colorado School of Mines, 76 Olcott Drive, White River Junction, Vermont 05001, April 1993. Lecture notes.
- [118] Linda Kaufman and Arnold Neumaier. Image Reconstruction Through Regularization by Envelope Guided Conjugate Gradients. Technical report, Bell Labs, 1993. Report 11274-94-14.
- [119] Jesper James Jensen. *3D Visualisering*. IMM Technical University of Denmark, <ftp://ei/dist/1995/james.3Dvis.ps.Z> and <ftp://ei/dist/Radon/polyr.tar.gz>. Manual in Danish. A software package called Polyr for converting a volume into a surface mesh has also been made.
- [120] Technical University of Denmark. Section for Digital Signal Processing. Home of the OOGI database of 3D objects. <http://hendrix.ei.dtu.dk/oogl/ooglhome.html>.
- [121] M. Daube-Witherspoon and G. Muehllehner. Treatment of axial data in threee-dimensional PET. *J. Nucl. Med.*, 28:1717–1724, 1987.
- [122] M. Defrise. A factorization method for the 3D x-ray transform. *Inverse Problems*, 11:983–994, 1995.
- [123] The Geometry Center. Homepage of The Geometry Center. Authors of Geomview. <http://www.geom.umn.edu>.
- [124] A. E. Todd-Pokropek and P. H. Jarrit. The noise characteristics of SPECT systems. In P. J. Ell and B. L. Holman, editor, *Computed Emission Tomography*, pages 361–389. Oxford University Press, Oxford, 1982.
- [125] N. M. Alpert et al. Estimation of the Local Statistical Noise in Emission Computed Tomography. *IEEE Trans. Med. Imag.*, MI-1(2):142–146, Oct. 1982.
- [126] C. W. Stearns and D. Wack. A Noise Equivalent Counts Approach to Transmission Imaging and Source Design. *IEEE Trans. Med. Imag.*, MI-12(2):287–292, June 1993.
- [127] S. C. Strother et al. Measuring PET Scanner Sensitivity: Relating Count Rates to Image Signal-to-Noise Ratios Using Noise Equivalent Counts. *IEEE Trans. Nucl. Sci.*, NS-37(2):783–788, April 1990.
- [128] T. R. DeGrado et al. Performance Characteristics of a Whole-Body PET Scanner. *J. Nucl. Med.*, 35(8):1398–1406, 1994.
- [129] T. K. Lewellen et al. Investigations of the Count Rate Performance of General Electric Advance Positron Emission Tomograph. *IEEE Trans. Nucl. Sci.*, NS-42(4):1051–1057, August 1995.
- [130] T. Lewellen et al. Investigations of the Performance of the General Electric Advance Position Emission Tomograph in 3D mode. In *Submitted*, 1995.
- [131] P. S. Crandall and C. W. Stearns. A Scalable Multi Processor Implementation of the Reprojection Algorithm for Volumetric PET Imaging. In *Conference Record of the IEEE 1995 Nuclear Symposium and Medical Imaging Conference.*, 1995. In press.

- [132] S. Holm et al. 3D PET activation studies with $H_2^{15}O$ bolus injection. Count rate performance and dose optimization. In *Quantification of Brain Function using PET*. Academic Press, San Diego, 1995. In press.
- [133] S. Holm et al. Count Rate Performance and Dose Optimization for rCBF with O-15 Water Bolus Injection in the New 3D PET. *J. Nucl. Med.*, 36:105P, 1995.
- [134] S. Pajovic et al. Noise Properties of 3D PET Images. *J. Nucl. Med.*, 36:p. 105 (abstract), 1995.
- [135] S. R. Meikle et al. Attenuation Correction Using Count-Limited Transmission Data in Positron Emission Tomography. *J. Nucl. Med.*, 34:143–150, 1993.
- [136] X. Ouyang *et al.* Incorporation of Correlated Structural Images in PET Image Reconstruction. *IEEE Trans. Med. Imag.*, 13:627–640, 1994.
- [137] Chien-Min Kao et al. Improved Bayesian Image Reconstruction in Positron Emission Tomography. In *Conference Issue of IEEE Medical Imaging Conference*, 1995. To appear.
- [138] L. Cohen and I. Cohen. Finite Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1993.
- [139] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated Volume Rendering and Tomographic Reconstruction using Texture Mapping Hardware. In *1994 Symposium on Volume Visualization*, pages 91–98. ACM SIGGRAPH, ACM Press, oct 1994.
- [140] Gel'fand et al. *Generalized Functions*, volume 5. Academic Press, Baltimore and London second, 2 edition, 1964.

Index

– Symbols –

Δx	7
Ω_ψ	162
Ω_ψ	165
$\delta(\cdot)$	195
γ	146
ϕ	161
ψ	162
ρ	24, 86
τ	3
τ - p transform	3
θ	24, 86, 161
p	3
x_{min}	7
3D line Radon transform	161
3DRP	168

– A –

adjoint	96, 97, 133
transpose matrix	134
Algebraic Reconstruction Technique	<i>see</i> ART
apodizing functions	108
apodizing window	109, 115
ART	137, 141, 149, 151
attenuation coefficient	86
attenuation correction	93, 159
axial acceptance angle	162, 174

– B –

backprojection	96
complexity	111
Backprojection of Filtered Projections	96
backprojection operator	102, 111
backprojection operator in 3D	164
base point vector	160
bin	90
blank scan	92, 180
brain phantom	183

– C –

C-11	183
center curve	55
Central Limit Theorem	77
Central Slice Theorem	<i>see</i> Fourier Slice Theorem
CG	145
Chirp-z transform	120
circular disc	201
classical curve detection algorithm	77
clusters	55
CMP-gather	49, 58
common mid point	58
common mid point gather	49
compact support	100
Conjugate Gradient algorithm	<i>see</i> CG
constraints	134, 151, 153
convolution backprojection	110
counts	91, 180
CT-scanner	86

– D –

delta function	x, 195
DFT	<i>see</i> discrete Fourier transform
directional vector	160
discrete Fourier transform	105
discrete Radon transform	8

– E –

EM	141, 149, 151, 153, 156
emission sinogram	90, 180
events	168
Expectation Maximization	<i>see</i> EM

– F –

F-18	183
Fast Fourier Transform	<i>see</i> FFT
FCE-algorithm	56
FFT	105
radix-2	106
zero padding	106

Filtered Backprojection.....88, **96**, 102, 107,
113, 143, 149, 153, 164, 180

3D166
discrete implementation113

Filtering after Backprojection.....**98**, 165
3D165
Implementation114

Finite Element**192**
Fourier Slice Theorem.....88, **95**, 101, 127

3D163, 218
complexity116
zero frequency99

Fourier transform**95**
full angular coverage.....**162**

— G —

generalized Hough transform**53**

generalized Radon transform.....**50**, 56, **76**
blurring.....55

Geomview**173**

glucose metabolism**88**

GRT*see* generalized Radon transform

— H —

Hamilton step functionx

high pass filter.....**165**

Hilbert transform**96**

— I —

ill-conditioned problem**131**, 136

image domain**6**

Image Point Mapping**52**, 56

interpolation.....8, **9**, 27, **111**

linear.....111

nearest neighbour.....111

sinc.....29

IPM.....*see* Image Point Mapping

— J —

Jacobian**164**

— K —

Kinahan & Rogers reprojeciton**168**

Kronecker delta function**69**

— L —

L2 measure of misfit**151**

Least Squares Conjugate Gradient*see* LSCG

least squares solution**145**

Likelihood**142**

line**160**

line description**160**

line integrals**161**

line parameters**3**

linear algebra**129**

Linogram method**119**

LSCG**145**, 149, 151, 153, 156

— M —

MART**141**

matrix vector formulation**130**

Maximum Likelihood Reconstruction**141**

mesh of triangles**173**

modified L2-measure of misfit**123**

— N —

NEC**180**

noise**76**

Noise Equivalent Counts*see* NEC

normal equations**134**, 145

normal form**23**

normal moveout equation**58**

Normal Radon transform

angular sampling**27**

parameters**26**

normal Radon transform**23**

analytically**201**

convolution**199**

Gaussian bell**205**

limits of the parameter domain99

linearity**197**

normal parameters**24**

properties**197**

pyramid**205**

rotation**197**

scaling**198**

shifted**197**

square**202**

symbol**23**

triangle**203**

null-space**167**

— O —

O-15**184**

offset**3**, **58**

orthogonal basis**160**

— P —

parameter domain**4**

- four-dimensional 161
 peaks in the parameter space **77**
 PET **88**, 159
 attenuation correction 92
 PET scanner 89
 photon **91**, **92**
 Poisson statistics **180**
 Polyr **173**
 Positron Emission Tomography *see* PET
 pre-conditioning **56**
 projection **87**, 131
- R –**
- Radon space 4
 Radon transform **3**, 129, 161
 3D line 161
 Discrete Radon transform 10
 Fourier Transform 95
 general form 217
 plane 217
 Random Radon transform 47
 Ram-Lak filter *see* ramp filter
 ramp filter **107**, **108**
 2D 116
 rebinned **170**
 reconstruction **88**, 91
 regularization **134**, **136**, 156
 resampled **170**
 ROI **148**, 182
 rotation matrix **161**
- S –**
- sampling **11**, **29**, 79
 sampling criterion **10**, 29
 sampling distance 7
 seismic signal processing 49
 series expansion **129**
 Shepp Logan Phantom **199**
 sinc interpolation Radon transform **10**, 29,
 133
 Single Slice Rebinning *see* SSRB
 Singular Value Decomposition *see* SVD
 singular values 131, **135**
 sinogram **90**, 130, **161**
 slant stacking **3**
 inversion formulas 100
 slanted lines **3**
 slices **89**
 slope **3**
- slowness **59**
 SPECT-scanner **92**
 SSRB **168**
 stacking **3**
 start guess **140**
 SVD **135**, 156
 system matrix **130**
- T –**
- Taylor expansion **180**
 The Hough transform **37**
 threshold **146**
 threshold level **76**
 Time-of-Flight PET scanners **90**
 trace **58**
 transmission scan **92**
 transmission sinogram 180
 zeros 182
 two way travel time **58**
- U –**
- unitary **161**
- V –**
- visualization **173**
 vote **38**
- W –**
- Whittaker-Shannon sampling theorem **10**
 wiggles **69**, 71, 74
 window **106**, 108
- Z –**
- zero offset two way travel time **59**