

# **Craniofacial Computer- Assisted Surgical Planning and Simulation**

**By**

**Emine EKİN**

**A Dissertation Submitted to the  
Graduate School in Partial Fulfillment of the  
Requirements for the Degree of**

**MASTER OF SCIENCE**

**Department: Computer Engineering  
Major: Computer Software**

**Izmir Institute of Technology  
Izmir, Turkey**

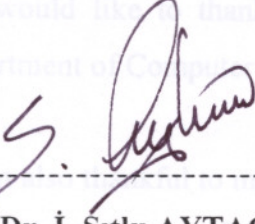
**September, 1999**

**İZMİR YÜKSEK TEKNOLOJİ ENSTİTÜSÜ  
REKTÖRLÜĞÜ  
Kütüphane ve Dokümantasyon Daire Başkanlığı**

# ACKNOWLEDGMENTS

We approve the thesis of **Emine EKİN** to Prof. Dr. Sıtkı Aytac, my adviser, who encouraged me to study the subject and gave support during this study.

I would like to thank to the staff, especially my professor Tatyana Yakimo, in Department of Computer Engineering, Dokuz Eylul University.



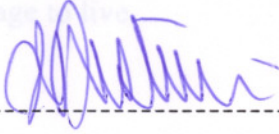
Date of Signature

28.09.1999

Prof. Dr. İ. Sıtkı AYTAC

Supervisor

Department of Computer Engineering



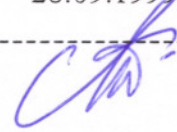
28.09.1999

Prof. Dr. Halis PÜSKÜLCÜ

Department of Computer Engineering

28.09.1999

Emine EKİN



Prof. Dr. Ali BARUTÇU

Faculty of Medicine, Dokuz Eylul University



28.09.1999

Prof. Dr. İ. Sıtkı AYTAC

Head of Department

## ACKNOWLEDGMENTS

I would like to say my gratitude to Prof. Dr. Sıtkı Aytaç, my adviser, who encouraged me to study the subject and gave support during this study.

I would like to thank to the staff, especially my professor Tatyana Yakhno, in Department of Computer Engineering, Dokuz Eylül University.

I'm also thankful to my friends for their patience. They always listened my problems about this study even if they did not know the solution.

And my family, I owe them lots of thing. If they were not with me, I could not manage to live...

Emine EKİN

## ABSTRACT

In this work, mainly, the first step of craniofacial surgical simulation and planning procedure, reconstruction of tomography images is investigated. The Direct Reconstruction techniques and algebraic methods are described in detailed mathematical formulations for both two and three-dimensional cases.

And also a brief description of medical imaging techniques and the terminology of craniofacial surgical simulation and planning is given.

In the last chapter the implementation details, the algorithms developed and some resulting images are given and the images are compared with respect to the used algorithms.

# ÖZ

## STYLE CONVERSIONS

Bu çalışmada, ana olarak, kafatasının operasyon öncesi modellenmesi ve operasyonun planlanması işleminin ilk aşaması olan tomografik resimlerin üç boyutlu yeniden yapılandırılması incelenmiştir. Bu doğrultuda, iki ve üç boyutlu ortamlar için, doğrudan yöntemlerle birlikte cebirsel yöntemler de detaylı matematiksel formüllerle ifade edilmiştir.

Ayrıca, tıpta görüntüleme yöntemleri ve kafatasının operasyon öncesi modellenmesi ve operasyonun planlanması konuları özetlenmiştir.

Çalışmanın son bölümünde ise, uygulama detayları, geliştirilen algoritmalar, bu algoritmalar uygulanarak elde edilen resimler ve karşılaştırmaları sunulmuştur.

- Vectors are always column vectors, and  $i^{\text{th}}$  rows of the matrix are denoted  $a_i$ , i.e.

$$A = \begin{pmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_n^T \end{pmatrix}$$

- Algorithms will use the Courier Font.
- The delta function is denoted by  $\delta(\ast)$ .
- The Heaviside step function is denoted by  $\rho(\ast)$ . The function is zero if the argument is negative, and one if the argument is positive.
- Fourier transform pairs are marked as  $g(t) \leftrightarrow G(f)$ .
- In convolutions convolutions are denoted by  $\ast$  for a one-dimensional,  $\ast\ast$  for a two-dimensional, and  $\ast\ast\ast$  for a three-dimensional convolution.
- In the algorithms no convolution is found,  $\ast$  is used for multiplication.
- The scalar product between two (column) vectors  $a$  and  $b$  are denoted by  $a \cdot b = a^T b$ .

## FUNCTIONS AND VARIABLES

- The input facet and the reconstructed function  $f(x,y)$
- Projection angle  $\theta$

## STYLE CONVERSIONS

- Equation has been abbreviated to Eq. Likewise Equations Eqs.
- The letters  $m, n, k, h, l$  are used to denote the integer type variables, and  $x, y, z$  denote continuous type variables.
- Vectors are denoted by bold- faced small letters or Greek letters, as  $\mathbf{b}$  or  $\xi$ .
- Transpose of a vector 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  $a_{ij}$  corresponding to row  $i$  and column  $j$ .
- Vectors are always column vectors, and  $i^{\text{th}}$  rows of the matrix are denoted  $a_i$ , i.e.

$$\mathbf{A} = \begin{pmatrix} a_1^T \\ a_2^T \\ \dots \\ a_i^T \\ \dots \\ a_i^T \end{pmatrix}$$

- Algorithms will use the Courier font.
- The delta function is denoted by  $\delta(\bullet)$ .
- The Hamilton step function is denoted by  $\mu(\bullet)$ . The function is zero if the argument is negative, and one if the argument is positive.
- Fourier transform pairs are marked as  $g(t) \leftrightarrow G(f)$ .
- In equations convolutions are denoted by  $*$  for a one dimensional,  $**$  for a two dimensional, and  $***$  for a three dimensional convolution.
- In the algorithms no convolution is found,  $*$  is used for multiplication.
- The scalar product between two (column) vectors  $\mathbf{a}$  and  $\mathbf{b}$  are denoted by  $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b}$

## FUNCTIONS AND VARIABLES

- The object function and the reconstructed function  $f(x, y)$
- Projection angle  $\theta$

- Projection axis, sinogram  $r$
- The sinogram or Radon transform  $p(r, \theta)$
- The filtered sinogram  $q(r, \theta)$

### FOURIER TRANSFORMS

- A function  $f(x, y)$
- The 1D Fourier transform of  $f(x, y)$  in the first variable  $F(X, y)$
- The 1D Fourier transform of  $f(x, y)$  in the second variable  $F(x, Y)$
- The 2D Fourier transform of  $f(x, y)$   $F(X, Y)$
- The 1D Fourier transform  $F(X) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi Xx} dx$
- The 1D inverse Fourier transform  $f(X) = \int_{-\infty}^{\infty} F(x) e^{j2\pi xX} dX$
- The 2D Fourier transform  $F(X, Y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(xX+yY)} dx dy$
- The Fourier transform operator  $\mathfrak{F}$
- The Radon transform operator  $\mathfrak{R}$
- The inverse Fourier transform operator  $\mathfrak{F}^{-1}$
- The inverse Radon transform operator  $\mathfrak{R}^{-1}$
- The Fourier transform in the  $r$ -variable  $\mathfrak{F}_r$
- The inverse Fourier transform in the  $R$ - variable  $\mathfrak{F}_R^{-1}$

### DEFINITIONS

- Radon Space:  
Projection data  $p(r, \theta)$  put in the  $(x, y)$  space where  $(x, y) = (r \sin \theta, r \cos \theta)$
- Sinogram projections:  
Parallel equidistant projection data taken at equidistant angles  $\theta$
- Linogram projections:  
Parallel projection data taken in equidistant  $\tan \theta$  steps for  $-45^\circ \leq \theta \leq 45^\circ$ , an  $\sin$  equidistant  $-\cot \theta$  steps for  $45^\circ \leq \theta \leq 135^\circ$ . The projection data are equidistant within each projection, but varies between the projections with the sample distances as follows,  $const \cdot \cos \theta$  for  $-45^\circ \leq \theta \leq 45^\circ$ , and  $const \cdot \sin \theta$  for  $45^\circ \leq \theta \leq 135^\circ$ .

# CONTENTS

<b>CONTENTS</b> .....	<b>vii</b>
<b>LIST OF FIGURES</b> .....	<b>x</b>
<b>CHAPTER 1 INTRODUCTION</b> .....	<b>1</b>
<b>CHAPTER 2 MEDICAL IMAGING</b> .....	<b>3</b>
2.1. COMPUTED TOMOGRAPHY (CT).....	4
2.2. SPECT.....	11
2.3. PET.....	14
2.4. MRI.....	16
2.5. ULTRASOUND IMAGING.....	19
<b>CHAPTER 3 COMPUTER ASSISTED CRANIOFACIAL SURGERY</b> .....	<b>22</b>
3.1. CRANIOFACIAL SURGERY.....	22
3.2. SURGICAL SIMULATION.....	22
3.3. SOME EXAMPLES OF CRANIOFACIAL DEFORMITIES .....	23
3.4. CRANIOFACIAL SURGICAL SIMULATION AND PLANNING .....	25
3.4.1. THE BASIS FOR SURGICAL SIMULATION .....	26
3.5. PREVIOUS WORK.....	30
<b>CHAPTER 4 THE BASICS OF RECONSTRUCTION METHODS</b> .....	<b>33</b>
4.1. RECONSTRUCTION PRINCIPLES.....	33
4.2. LINE INTEGRALS AND PROJECTIONS .....	36
4.3. THE FOURIER SLICE THEOREM.....	38
4.4. THE FILTERED BACKPROJECTION METHOD.....	41
4.4.1. INTUITIVE DESCRIPTION.....	41
4.4.2. FORMAL PROOF.....	44
4.5. FILTERING AFTER BACKPROJECTION .....	47
4.6. THE DIRECT FOURIER METHOD .....	49
4.7. THE LINOGRAM METHOD.....	51
4.8. RECONSTRUCTION ALGORITHMS BASED ON LINEAR ALGEBRA .....	55
4.8.1. FROM THE RADON TRANSFORM TO LINEAR ALGEBRA BASED RECONSTRUCTION .....	55
4.8.2 CALCULATION OF MATRIX ELEMENTS .....	58



4.8.2.1 Pixel Oriented Nearest Neighbour Approximation.....	59
4.8.2.2. Discrete Radon Transform .....	60
4.8.2.3. The Sinc Interpolation Strategy .....	60
4.4.3. DUALITY BETWEEN MATRIX OPERATIONS AND THE RADON TRANSFORM	61
4.4.4. ITERATIVE RECONSTRUCTION USING ALGEBRAIC RECONSTRUCTION	
TECHNIQUE (ART) .....	62
4.8.4.1 Art With Constraints .....	66
4.8.4.2 Initialization .....	66
4.4.5. THE EM ALGORITHM.....	67
<b>CHAPTER 5 THREE DIMENSIONAL RECONSTRUCTION.....</b>	<b>72</b>
5.1. LINES IN A THREE DIMENSIONAL SPACE .....	72
5.2. FOURIER SLICE RECONSTRUCTION IN 3D .....	75
5.3. THE BACKPROJECTION BASED INVERSION OF LINE INTEGRALS IN 3D .....	76
5.4. FILTERING AFTER BACKPROJECTION OF LINE INTEGRALS IN 3D .....	78
5.5. FILTERED BACKPROJECTION OF LINE INTEGRALS IN 3D .....	80
<b>CHAPTER 6 IMPLEMENTATION DETAILS .....</b>	<b>83</b>
6.1. NUMERICAL IMPLEMENTATION OF 2D DIRECT RECONSTRUCTION ALGORITHMS	
	83
6.1.1. USING THE DFT TO APPROXIMATE THE FOURIER TRANSFORMATION .....	83
6.1.1.1 The FFT Applied For Filtering .....	84
6.1.2. DISCRETE IMPLEMENTATION OF BACKPROJECTION .....	91
6.1.3. IMPLEMENTATION OF FILTERING AFTER BACKPROJECTION .....	95
6.1.4. IMPLEMENTATION OF THE FOURIER SLICE THEOREM.....	98
6.2. IMPLEMENTATION OF 3D DIRECT RECONSTRUCTION ALGORITHMS .....	102
6.2.1. IMPLEMENTATION OF THE 3D RECONSTRUCTION METHODS .....	103
6.2.1.1 Implementation Of The Backprojection Operator .....	104
6.2.1.2. Implementation Of The Radon Transform Operator .....	106
6.3. EXAMPLES USING DIRECT RECONSTRUCTION ALGORITHMS.....	108
6.3.1. RECONSTRUCTION WITH VARYING IMAGE SIZE .....	110
6.3.2. RECONSTRUCTION INTO AN OVERSAMPLED IMAGE.....	112
6.4. EXAMPLES USING ITERATIVE RECONSTRUCTION ALGORITHMS .....	114
6.5. EXAMPLES OF 3D RECONSTRUCTED VOLUMES .....	116
6.5.1. RECONSTRUCTION OF A BALL.....	116

6.5.2. RECONSTRUCTION OF THE MICKEY PHANTOM.....	118
<b>CHAPTER 7 RESULTS AND FUTURE WORK.....</b>	<b>123</b>
<b>REFERENCES.....</b>	<b>124</b>
<b>APPENDIX A .....</b>	<b>A1</b>
<b>APPENDIX B .....</b>	<b>B1</b>
1.1 The backprojector.....	7
1.2 Cross-section of the X-ray source, the patient, and the detectors in (a) a 2.1 reconstruction CT scanner, (b) a fourth generation CT scanner.....	9
2.2 (a) Fanbeam scanner (b) Conebeam scanner.....	10
3.1 SPECT geometry. (a) Cutting through the $\gamma$ -camera and the patient. (b) The camera from above.....	12
3.2 Beam artifacts in SPECT.....	13
3.3 Geometry of ectomography.....	13
3.4 The principle of the PET.....	15
4.1 Cross-sectional images of the human head, MRI.....	19
4.2 Cross-sectional images of the human head, CT.....	19
4.3.....	23
4.4.....	24
4.5.....	24
4.6.....	25
4.7.1 An object slice $f(x,y)$ and one of its projections.....	34
4.7.2 An object $f(x,y)$ and one of its projections $p(r,\theta)$ .....	36
4.8.1 Projection geometries. (a) The geometry for parallel projections. (b) The geometry for fanbeam projections.....	37
4.8.2 Collection of projections of fan object disk in a sinogram.....	38
4.8.3 A point in the object disk gives rise to a sinusoid in the sinogram.....	38
4.8.4 The projection slice theorem illustrated.....	39
4.8.5 A special case of the projection slice theorem.....	40
4.8.6 A filtered projection $q(r,\theta)$ is smeared out (backprojected) over the reconstruction plane.....	41

## LIST OF FIGURES

Figure 2.1 Recording of an X- ray image of the human head.....	5
Figure 2.2 Linear classical tomography.....	6
Figure 2.3 A special kind of classical transversal tomography. The projection recorder.	6
Figure 2.4 The backprojector.....	7
Figure 2.5 Cross- section of the X- ray source, the patient, and the detectors in (a) a third generation CT scanner , (b) a fourth generation CT scanner. ....	9
Figure 2. 6 (a) Fanbeam scanner (b) Conebeam scanner.....	10
Figure 2.7 SPECT geometry. (a) Cutting through the $\gamma$ - camera and the patient. (b) The $\gamma$ - camera from above.....	12
Figure 2.8 Source artifacts in SPECT.....	13
Figure 2.9 Geometry of ectomography.....	13
Figure 2.10 The principle of the PET. ....	15
Figure 2.11 Cross- sectional images of the human head, MRI.....	19
Figure 2.12 Cross- sectional images of the human head, CT .....	19
Figure 3.1 .....	23
Figure 3.2 .....	24
Figure 3.3 .....	24
Figure 3.4.....	25
Figure 4.1 An object slice $f(x,y)$ and one of its projections .....	34
Figure 4.2 An object $f(x,y)$ and one of its projections $p(r, \theta_j)$ . ....	36
Figure 4.3 Projection geometries. (a) The geometry for parallel projections. (b) The geometry for fanbeam projections. ....	37
Figure 4.4 Collection of projections o fan object disk in a sinogram.....	38
Figure 4.5 A point in the object disk gives rise to a sinusoid in the sinogram. ....	38
Figure 4.6 The projection slice theorem illustrated. ....	39
Figure 4.7 A special case of the projection slice theorem. ....	40
Figure 4.8 A filtered projection $q(r,\theta_i)$ is smeared out (backprojected) over the reconstruction plane.....	41

Figure 4.9 A set of projections followed by a set of backprojections causes some sort of low- pass filtering in the image. This is easy to see when the object consists of a single point.....	42
Figure 4.10 Projection followed by backprojection causes low- pass filtering in the image. The equivalent filter function is $1/\zeta$ in the spatial domain or $1/P$ in the Fourier domain. To compensate for the low- pass filtering we can multiply the image with $P$ in the Fourier domain.....	43
Figure 4.11 The band- limited version of the ramp- filter $H(R)=  R $ shown in both domains.....	43
Figure 4.12 An overview of the FBM.....	44
Figure 4.13 Backprojection takes place when, for a given $\theta$ , a filtered projection $q(r, \theta)$ is smeared .....	47
Figure 4.14 An overview of the DFM .....	51
Figure 4.15 a) The detector orientation is fixed along the x- axis for all projections $- 45^\circ \leq \theta \leq 45^\circ$ .....	52
Figure 4.16 a) Sinogram sampling of the Radon space. b) The Fourier space corresponding to sinogram sampling. c) Linogram sampling of the Radon space. d) The Fourier space corresponding to linogram sampling. ....	53
Figure 4.17 The linogram method. ....	54
Figure 4.18 The matrix element $a_{ij}$ can be considered as the weight factor between a certain sinogram value numbered by $i$ and the image pixel $j$ . ....	57
Figure 4.19 The line with index $i$ , corresponding to $(\rho_r, \theta_r)$ , crosses the square pixel centered at $(x_m, y_n)$ .....	59
Figure 4.20 The different cases of iteration in a two parameter problem using ART. Depending on the orthogonality of hyperlines (here lines) the convergence can be slow (left most figure) or fast (right most figure)(Toft 1996). ....	64
Figure 5.1 The $(x,y,z)$ coordinate system and a line lying along to the $\tau$ axis. The base point vector $r_0$ can be wirtten as a linear combination of $\alpha$ and $\beta$ . ....	73
Figure 5.2 Normalized angular part of the filter as a function of $\phi_v = \arcsin(v_z /  v )$ for $\psi = 10^\circ, 20^\circ, \dots, 80^\circ$ .....	80
Figure 6.1 Filling an array $h(n)$ into a larger array $g(n)$ , when using radix- 2 FFT.....	85

Figure 6.2 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 the zeros have been padded, and the lower right shows the corresponding absolute spectrum, which appears more smooth. ....	86
Figure 6.3 The amplitude of the Ram- Lak 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, $v_f$ . ....	89
Figure 6.4 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.....	89
Figure 6.5 Upper left shows the sinogram for a fixed value of $\theta$ and varying $\rho$ . Upper right shows the corresponding discrete spectrum, and it can be noted that there is 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. ....	90
Figure 6.6 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. ....	94
Figure 6.7 Filtered Backprojection from a sinogram with 5 angular samples ( $T=5$ ) ....	95
Figure 6.8 Filtered Backprojection from a sinogram with 10 angular samples ( $T=10$ ) .	95
Figure 6.9 Filtered Backprojection from a sinogram with 20 angular samples ( $T=20$ ) .	95
Figure 6.10 Filtered Backprojection from a sinogram with 100 angular samples ( $T=100$ )	95
Figure 6.11 A real valued image gives a spectrum with complex conjugate pairs.....	96
Figure 6.12 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. ....	99
Figure 6.13 Three ways that the polar and the quadratic spectrum can match.....	100
Figure 6.14 Strategy 1: A weighted sum of the four closest polar samples is used to estimate the spectrum on the quadratic grid. ....	102

Figure 6.15 Strategy 2: Any of the samples on the polar grid are distributed with certain weights to the quadratic grid.....	102
Figure 6.16 Sinogram of the phantom .....	108
Figure 6.17 The original head phantom.....	109
Figure 6.18 The reconstructed head phantom using filtered backprojection.....	109
Figure 6.19 The reconstructed head phantom using filtering after backprojection .....	109
Figure 6.20 The reconstructed head phantom using Fourier slice theorem.....	109
Figure 6.21 L2 error as a function of the reconstructed image size M for Filtered Backprojection, Filtering after Backprojection, and a Fourier slice implementation.	110
Figure 6.22 The absolute error between the original image and the reconstructed image using Filtered Backprojection.....	111
Figure 6.23 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. ....	112
Figure 6.24 Sinogram of a square with $R=251$ , $T=200$ , and $\Delta\rho=0.0025$ .....	112
Figure 6.25 Reconstructed image with $M=101$ and $\Delta x=0.01$ using Fourier Slice Theorem.....	113
Figure 6.26 Reconstructed image with $M=101$ and $\Delta x=0.01$ using Filtering after Backprojection.....	114
Figure 6.27 A slice of a sinogram of human brain. ....	115
Figure 6.28 The reconstructed image after 10 iterations of EM.....	115
Figure 6.29 The reconstructed image using Filtered Backprojection with a ramp filter.	116
Figure 6.30 Reconstructed ball using 3D Filtered Backprojection.....	117
Figure 6.31 Reconstructed ball using 3D Filtering after Backprojection.....	118
Figure 6.32 The Mickey phantom.....	119
Figure 6.33 Reconstructed phantom in the central (x,y) plane usign Filtered Backprojection.....	119
Figure 6.34 Reconstructed phantom in the central (y,z) plane using Filtering after Backprojection.....	120

Figure 6.35 Reconstructed phantom in the central (x,y) plane using Filtered Backprojection.....	120
Figure 6.36 Reconstructed phantom in the central (y,z) plane using Filtered Backprojection.....	121
Figure 6.37 Reconstructed phantom in the central (x,y) plane using five- iterations of EM. ....	121
Figure 6.38 Reconstructed phantom in the central (y,z) plane using five iterations of EM. ....	122
Figure A.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. ....	A5

# CHAPTER 1

## INTRODUCTION

Medical imaging technologies are altering the nature of many medical professions today. During the last decade, many radiology departments have installed powerful imaging equipment for imaging modalities like Magnetic Resonance (MR), Computed Tomography (CT), and Positron Emission Tomography (PET). These systems have spawned new specialities, and have fundamentally changed the way many diseases are diagnosed, and even the way they are treated.

Smaller CT scanners are being marketed by the imaging system manufacturers, who expect such systems to be increasingly used at clinical department instead of at centralized radiology departments. In addition, cheap Ultra Sound (US) systems are increasingly being used in the clinical departments.

Unfortunately, although the “mechanical” equipment has seen a dramatic development, the real power of these systems has not been released. The software and knowledge, needed to take full advantage of the imaging systems, has not followed the development of hardware.

It is, therefore, still not unusual that hospitals, even with powerful expensive 3D scanners, only have diagnostic procedures for handling 2D image slices. In addition, when hospitals do use the 3D reconstruction capabilities of the scanners, the available software is not sufficiently flexible and advanced, to allow real interaction with the 3D image data and provide useful support for diagnosis.

Only advanced medical imaging research laboratories, which have their own software development capability and access to the latest technology through technical partners, are today able to explore more advanced uses of the 3D images produced by the scanners. Typical for those laboratories has been a specific focus on technical research. The groups have participated in large technical advanced research projects, ie. European Union projects, and have built up the necessary technical expertise through these projects.

But from these research projects, and other research that is being performed in the field of medical imaging, new technology is becoming available. Recent years have



seen the development of many image processing, and computer graphics algorithms. Algorithms that ultimately will lead to better medical imaging software for diagnosis, treatment planning, surgery training, and surgery assistance.

The work I proposed may be named as “passive aided surgery” in a way as in (Subsol 1995). It is passive in the sense that the computer aids the surgeon by displaying useful information but does not intervene in the diagnosis or surgery process itself.

If atlas visualization and registration are added to a surgery simulation program (craniofacial surgery (Delingette et al. 1994) or stereotactic surgery (Hardy 1994)), the user can manipulate more easily the patient data thanks to automatic labelling and to contrast the patient data with the normalized atlas. A medical doctor can then plan surgical procedures.

Moreover if the registration algorithm can work in real time (at least a frequency level of hertz), it becomes possible to superimpose the visualization of the atlas on the real image of the surgery. This is the principle of “enhanced reality” developed by several teams.

## CHAPTER 2

### MEDICAL IMAGING

The study of medical imaging is concerned with the interaction of all forms of radiation with the tissue and the development of appropriate technology to extract clinically useful information from observations of this interaction. Such information is usually displayed in an image format. Medical images can be as simple as a projection or shadow image or complicated as a computer reconstructed image- as produced by Computerized Tomography (CT) using X- rays or by Magnetic Resonance Imaging (MRI) using intense magnetic fields.

Although strictly speaking, medical imaging began in 1895 with Röntgen's discoveries of X- rays and the ability of X- rays to visualize bones and other structures within the living body, contemporary medical imaging began in the 1970s with the advent of Computerized Tomography. Early or what we called classical, medical imaging utilizes images that are a direct manifestation of the interaction of some form of radiation with tissue. First example of classical imaging, the conventional X- ray procedure in which a beam of X- rays is directed through the patient onto a film. The developed film provides a shadow image of the patient which is a direct representation of the passage of X- rays through the body.

As a second example, think over a conventional nuclear medicine procedure. A radioactive material is injected into the patient and its course followed by a detector, which is moved over the patient in a specified manner. It provides a measure of physiological function from the time course of radioisotope uptake. The conventional nuclear medicine image is a direct measure of location and concentration of the radioactive isotope used.

As a final example of classical medical imaging, consider conventional medical ultrasound. Here, a pulse of ultrasonic energy is propagated into the patient and the backscattered echo signal is recorded by the same transducer. By angulating or moving the transducer positionally sequential echo signals are recorded, and a cross-sectional image of the subject is displayed directly on a video monitor. Ultrasound images are

really a mapping of echo intensities and are a direct result of the interaction of the ultrasound pulse with tissue (Cho et al. 1993).

## 2.1. Computed Tomography (CT)

Computed Tomography (CT) is an outstanding break-through in technology as well as in medical diagnostics. Its global success paved the way for a new scientific discipline called *imaging*, which encompasses such more recent developments such as Synthetic Aperture Radar (SAR) and Magnetic Resonance Imaging (MRI).

The meaning of the word *Computed Tomography* is as follows. According to Webster's dictionary tomography comes from the Greek word *tomos*, meaning slice or section, and the Greek word *graphein*, meaning to write. *Computed* comes from the demand of heavy computation.

Although CT is a general technique, its history is strongly related to the use of X-ray imaging in medicine. The history of CT may be summarised by the following events.

1895	William K. Röntgen discovers X-ray emission
1901	Röntgen receives the first Nobel prize in Physics for this discovery
1917	Radon formulates a new mathematical 2D-transform
1914- 32	Development of classical tomography
1972	Hounsfield gets patent on the first CT scanner using an algebraic reconstruction method
1974	The Filtered Backprojection Method (FBM) starts to replace algebraic reconstruction techniques in commercial CT scanners
1978	Third generation fanbeam tomography with FBM is established
1979	Hounsfield and Cormack receives the Nobel prize in medicine for CT.

Although Röntgen was unaware of the fact, X-rays are electromagnetic (EM) waves in the interval  $0.5$  to  $10^{-2} \text{Å}$ . These EM waves are well suited for attenuation measurements in the human body. Waves with longer wavelengths, such as light, are too much attenuated and shorter wavelengths, such as gamma rays, are transmitted almost completely without attenuation. In addition, to obtain a reasonable image

resolution, the wavelength must at length be less than 1 mm. The X-rays satisfy also this demand.

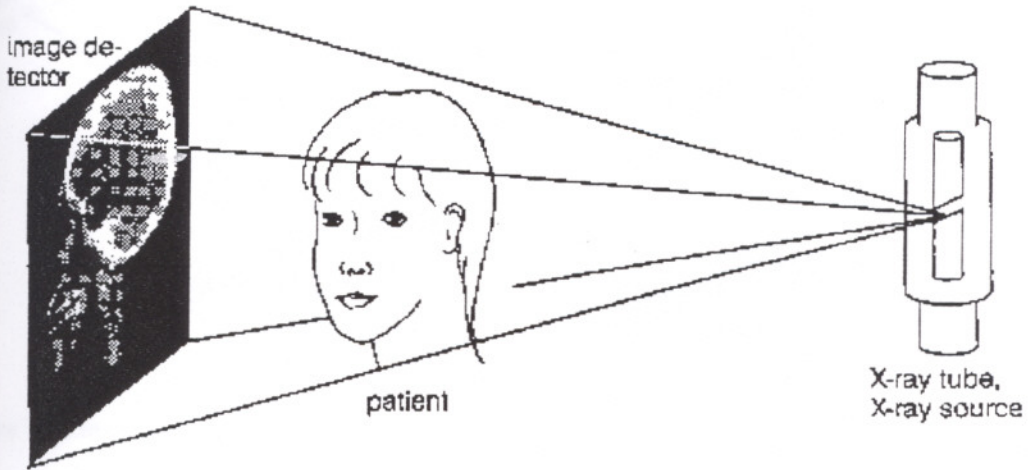


Figure 2.1 Recording of an X-ray image of the human head.

X-ray transmission images are obtained by placing the patient between an X-ray source and an image detector, Figure 2.1. The X-rays travel through the human body where they are attenuated. When they hit the detector, a picture is produced where the contrast is produced by different attenuation for different rays. By the unexperienced viewer, the X-ray image is normally perceived as a “negative”.

Tomography was invented surprisingly early. In order not to confuse this kind of tomography with the much later developed computed tomography, the first kind of tomography will be called **classical tomography**. By classical tomography it is possible to obtain a sharp picture of a specific plane of the body, while all other planes are rendered unsharp by blurring in proportion to their distance from the specific plane.

Classical tomography is obtained by moving the X-ray source and the image detector in relation to the patient in such a way that the image of the specific plane is projected on to the detector without motion. The images of all other planes are thus depicted with blur. The simplest form of classical tomography is linear tomography, shown in Figure 2.2.

The X-ray source and the 2D detector move during the exposure along linear paths synchronously and in opposite directions. They move in different planes, parallel to each other and to the specific plane. Related techniques employ circular (*circular classical tomography*) or spiral movements of X-ray source and detector.

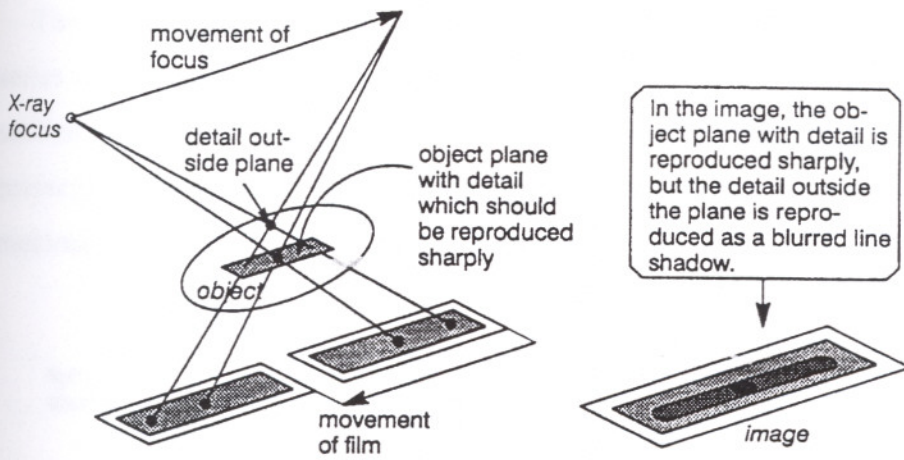


Figure 2.2 Linear classical tomography

In **classical transversal tomography** cross-sections of the human body are depicted. The source and the detector move in circles around the patient, parallel to each other and to the specific plane. In one kind of transversal tomography, the source, the specific plane, and the detector are in the same plane so that the specific plane is projected from edge to edge onto the detector. This kind of transversal tomography was never implemented commercially, but may be considered as a precursor to computed tomography. The principle of projection recording in transversal tomography is shown in Figure 2.3. Here the object or the patient moves in a circle, instead of the source and the detector, but the effect is the same.

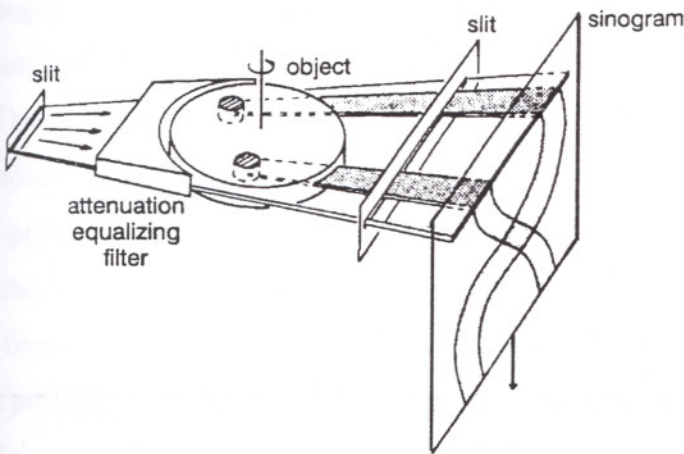


Figure 2.3 A special kind of classical transversal tomography. The projection recorder.

The image obtained after the recording of the projections is called a *sinogram*, a series of 1D projections piled on top of each other. To get a real image of the cross-section, post-processing or so called *back-projection* must be performed, where 1D projections are smeared back over the image plane. One way to do is to use an analog mechanical/ optical device shown in Figure 2.4.

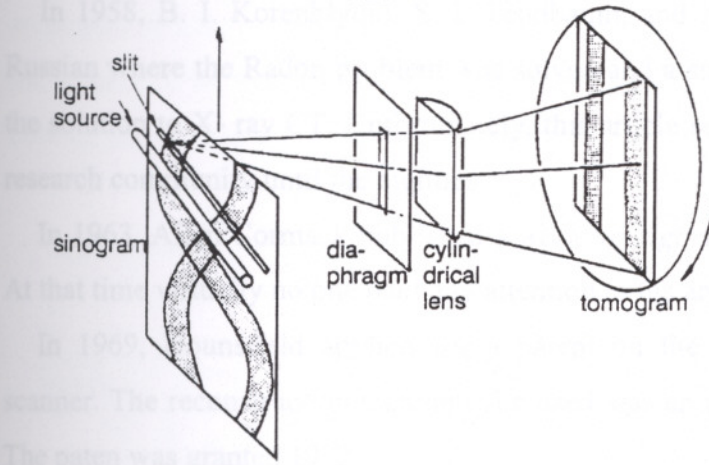


Figure 2.4 The backprojector

Many of the early inventors of **computed tomography (CT)** had very little knowledge about the long development of classical tomography. In CT, object outside the cross-section will not influence the image as in classical tomography. Neither will objects in the cross-section be smeared out as in that special type of transversal tomography mentioned above. In fact, theoretically the cross-section can be reconstructed perfectly without smearing or blurring.

The basic reconstruction problem in CT is to find the 2D function  $f(x,y)$  from the complete set of line integrals  $p(r, \theta)$  through the function. As was pointed out in 2.1 this is a problem of inversion, and it has been solved independently in different ways by a number of researchers working in very different fields. Cormack has made a search in the literature for early contributions. He found that the Dutch physicist H. A. Lorenz was probably the first to find a solution according to a reference from Bockvinkel 1906.

As mentioned above, in 1917 Johann Radon found a solution. After him the problem is often called the Radon's problem. The complete set of line integrals through the function  $f(x,y)$  is called the Radon transform, and the Radon's solution of the problem is called Radon's inversion formula. The difficulty with the Radon inversion formula is

that it does not produce an obvious solution to a working algorithm, due to a zero valued denominator for lines through the origin.

The first implementation of a solution to Radon's problem to be used in practice was made in 1956 by Ronald Bracewell in the field of radio astronomy. He reconstructed the radiation distribution of the sun disk from line integral across.

In 1958, B. I. Korenblyum, S. I. Tetelbaum, and A. A. Tyutin wrote an article in Russian where the Radon problem was solved and a suggestion was made for applying the solution to X- ray CT. Unfortunately, this article remained unknown to the western research community until the eighties.

In 1963, Allan Cormack published a *working algorithm* for calculating a CT image. At that time virtually no one paid any attention to his article.

In 1969, Hounsfield applied for a patent on the first computerized tomography scanner. The reconstruction technique he used was an algebraic reconstruction method. The patent was granted 1972.

For their discoveries in the field of computed tomography, Hounsfield and Cormack received the Nobel prize in medicine 1979.

Up to 1969 it seems that all researchers in the field worked independently and without knowledge of each other. The commercial activities grew steadily from 1970 and onwards.

In commercial CT scanners the algebraic reconstruction methods began to be replaced by the FBM around 1974. It was shown that the processing time as well as the image quality (for good projection data) was much better for the FBM than for algebraic methods.

The first and second generation CT- scanners generated parallel projection data using a slow and photon wasting procedure. The third generation, the **fanbeam tomography** scanners introduced in 1978, generated so called fanbeam projection data, where all rays in one projection originating from the same source point. The reconstruction method used for fanbeam tomography was a modified, more complicated FBM. In a third generation CT scanner (shown in Figure 2.5a), both the X- ray source and the detector array rotates synchronously around the patient. In a fourth generation CT scanner only the X- ray tube rotates around the patient inside a ring of stationary detectors (Figure 2.5b)

In the early fanbeam scanners the fanbeam projection data were interpolated to parallel data (so called rebinning) before reconstruction so that a parallel filtered backprojection algorithm could be employed. Nowadays it is more usual to apply a fanbeam filtered backprojection algorithm directly on the projection data. These algorithms are somewhat more complicated than the parallel algorithm and were first developed by Lakshminarayanan for equispaced collinear detectors and later by Herman and Naparstek for equiangular rays. The benefit of no rebinning seems to outweigh the increase in complexity.

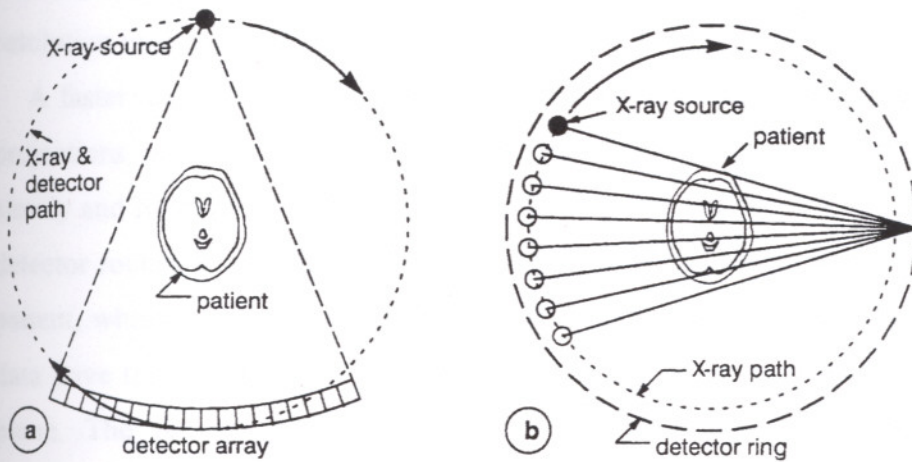


Figure 2.5 Cross- section of the X- ray source, the patient, and the detectors in (a) a third generation CT scanner , (b) a fourth generation CT scanner.

A high quality X- ray CT scanner of 1993 typically has the following technical data

- Type of X- ray beam: fanbeam
- Number of projections: approximately 1000
- Number of detector values per projection: approximately 700
- Reconstructed image format: 512 \*512 pixels
- Dynamic intensity range in reconstructed image: 12 bits
- Data acquisition per slice: 1-2 seconds
- Reconstruction time per slice: 2-5 seconds
- Cost: 0.5- 1M\$

X- ray CT has its widest use in medicine, but CT is also used in **Industrial applications** in the field of nondestructive testing (NDT) of industrial parts and materials. There are now several companies, mostly in USA, which produce special



industrial CT- scanners. Typically, the applications for NDT concerns expensive products, safety demanding applications and material development, for instance checking the quality of ammunition, inspection of turbo- blades for aeroplane engines. and testing of composite- and powder- material products.

**3D- imaging** is of interest for both medical and industrial applications. The common technique today to obtain an image volume is rather primitive. It is performed by moving the patient couch stepwise through the CT scanner ring and collects the projection data for a whole set of 2D- slices. The disadvantage with this procedure is that it is slow and that the resolution from slice to slice usually is much lower than the resolution inside 2D- slices.

A faster method is to use is so called **helical scanning** which gives a whole set of projections in a comparatively short data acquisition time. The patient is translated slowly and linearly through the CT- scanner ring. Simultaneously the X- ray source and detector rotates continuously. The net result is a helical or spiral movement around the patient, which supplies projection data for a whole set of slices. The spiral projection data have to be interpolated to a set of 2D- slice projection data aligned in the same plane. The reconstruction of each individual 2D- slice can then be done with conventional 2D techniques.

The fastest 3D data acquisition method is cone beam scanning instead of the conventional fanbeam scanning. Initially, all X- ray sources generate conebeam rays. To obtain fanbeam rays, most of the conebeam rays are screened off. A conebeam source requires a 2D detector, see Figure 2. 6.

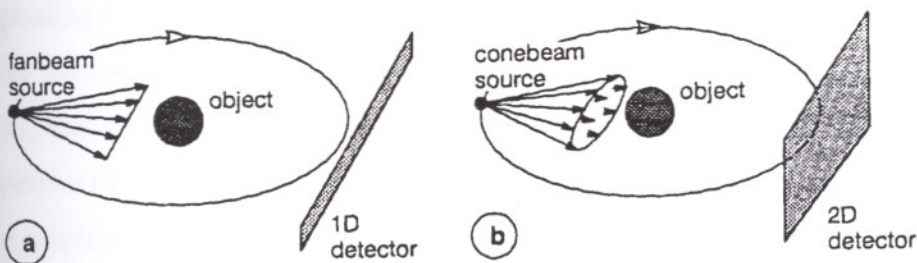


Figure 2. 6 (a) Fanbeam scanner (b) Conebeam scanner

Exact reconstruction of **3D- volumes** from the 2D-conebeam projection data is not a straightforward matter, however. The conventional FBM for parallel or fanbeam projection data consisting of filtering of projection data followed by backprojection

along the projection rays. Attempts have been made to extend this method to conebeam projection data. The result is a 3D-image volume with varying image quality from a perfect reconstruction of the central slice to heavy artifacts in the parts illuminated at large cone- angles. One reason for the bad image quality is rather fundamental. A conebeam source moving circularly around the object cannot give sufficient projection data for a perfect reconstruction. This is the problem of *missing data*. The projection data can be made complete in various ways, for instance, by adding a vertical line movement at source at right angle to the circular path. The exact mathematical solution to the reconstruction problem of conebeam projection data was given in (Kudo and Saito 1990, Grangeat 1991).

## 2.2. SPECT

Conventional CT images the attenuation of X- rays in an object slice. Emission CT, on the other hand, images the distribution of a radioactive isotope inside an object slice. One technique is called SPECT (Single Photon Emission Computed Tomography), where one single gamma photon is emitted for each nuclear event. SPECT is well described in the literature, see for example (Kok and Slaney 1987)

The data acquisition starts by depositing isotopes in the patient. The isotopes are injected or inhaled in the form of radio- pharmaceutical. After some time these substances tend to aggregate and deposit themselves in specific organs and tissues. The concentration of the isotope can then be imaged by measuring the gamma photons originating from radioactive decay of the isotope.

As the concentration of the isotope changes with time not only due to radioactive decay, but also because of flow and biochemical kinetics within the body, functional properties of the human body can also be measured. However, its important that the time constant of these human functions is large enough to measure a sufficient number of gamma photons for computation of one 3D image.

The geometry of SPECT is shown in Figure 2.7. The radioactive isotopes have migrated and accumulated in a specific area in the patient. Gamma photons are emitted from the isotopes and are then detected by the  $\gamma$ - camera. In the scintillator crystal the gamma photons are converted to light photons, which are detected and magnified by photomultipliers. In front of the  $\gamma$ - camera is a collimator consisting of many parallel

tubes of lead. The lead tubes restricts the flight paths of the measured photons so that a parallel projection of the patient is produced for each position of the camera as it moves in a circle around the patient. In principle, from these 2D projections, the reconstruction can be done in the same way as for X- ray CT where object cross- section is reconstructed from sets of line integrals, i. e. 1D projections. Since the  $\gamma$ - camera is two dimensional, as shown in Figure 2.7, it is possible to achieve a whole 3D volume of image data after reconstruction.

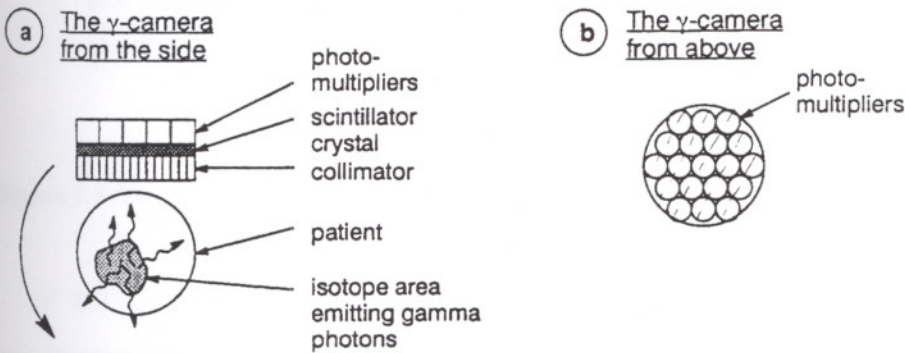


Figure 2.7 SPECT geometry. (a) Cutting through the  $\gamma$ - camera and the patient. (b) The  $\gamma$ - camera from above.

There are several difficulties with SPECT. One is the unwanted attenuation of photons as they travel from the isotope source towards the detector. The attenuation depends both upon the photon energy and the nature of the tissue. In Figure 2.8, photons originating from an isotope source at  $g_2$  are further away from the detector than  $g_1$  photons, and would therefore be more attenuated. To compensate for the attenuation, the reconstruction formula could be modified. However, this modification is neither straightforward nor simple and the compensation is not perfect.

Another source of error is also shown in Figure 2.8. Photons originating from a source between two nonparallel lines  $l_0$  and  $l_3$ , but outside the area between  $l_1$  and  $l_2$  may also be detected by the detector. Seen as probe, the collimator loses its sharpness with distance. Therefore the resolution becomes lower for points lying far away from the detector.

Typically, the images obtained with SPECT have the 64 or 128. The rather low image quality in comparison with X- ray CT images (see for example Figure 2.12) is due to a lower number of detected photons per pixel. Unfortunately, for safety reasons

the dosage of the radioactive isotope cannot be increased. A certain increase in signal-to-noise ratio (SNR) is obtained by using two or three cameras at the same time.

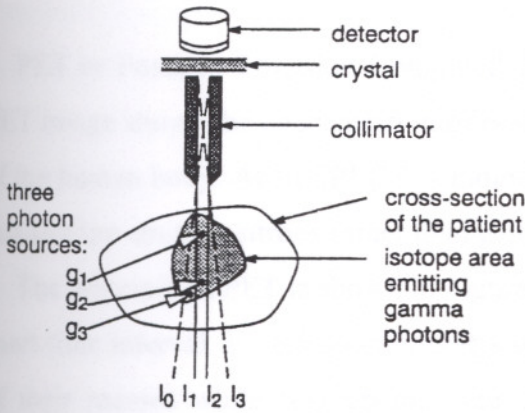


Figure 2.8 Source artifacts in SPECT

If CT is filtered version of transversal tomography, by the same token **ectomography** can be called the filtered version of circular classical tomography. Ectomography is a new technique which is on the verge of being introduced clinically. It is preferably meant for measurements in the same area as SPECT, i.e. measurement of  $\gamma$ -radiation from an isotope inside the body. Figure 2.9 shows the geometry of an ectomography scanner. Integrals of the slanted lines through the body are recorded by a rotating device. For the reconstruction similar methods as in CT are used. Exact reconstruction is not possible because of missing data. Even so, there are certain situations when this can be tolerated and where the ectomography may be more suitable than traditional SPECT.

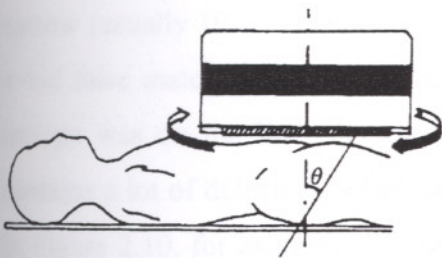


Figure 2.9 Geometry of ectomography.

### 2.3. PET

PET or Positron Emission Computed Tomography is another type emission CT. A PET image shows the concentration of positron emitting isotopes inside a cross-section of the human body. As in SPECT, a radioactive isotope is distributed to the patient, but this isotope emits positrons rather than gamma photons.

The principle of PET is shown in Figure 2.10. A positron can exist in nature for just a short time interval. It very soon interacts with an electron, which results in annihilation of their masses while two photons are created, each of them having an energy of 511keV and travelling in exact opposite directions. Note that the mass  $m$  of an electron or positron is  $9.1091 \cdot 10^{-31}$  kg and due to

$$E = mc^2 = 8.19 \cdot 10^{-14} \text{Ws} = 511 \text{keV} \quad (2.1)$$

this is equivalent to the wavelength

$$\lambda = \frac{h}{E} = 8.09 \cdot 10^{-21} \text{m} \quad (2.2)$$

An emitted positron is detected by *time coincides* of two photons arriving at two diametrically opposite detector elements. Evidently, the time window must be very narrow (usually 10-25 ns) and the number of emission events must be kept rather low to avoid false matches. When a match is confirmed, we know that the positron-emitting isotope was on the line (or strip) between the two reacting detectors. APET device contains a lot of detectors, where many coincidence lines are possible at the same time. In Figure 2.10, for example, a ring of detectors surrounding the body is indicated. The recording events constitute line integrals of the isotope concentration within the body. Hence, in principle, the reconstruction can then be done in same ways as for X-ray CT where an object cross-section is reconstructed from sets of line integrals, i. e. projections.

There are two clear advantages of PET compared to SPECT. One is that no collimators are needed. Collimation is done by nature itself and the time windowing. Consequently, the photon efficiency is higher. The other advantage is easy attenuation compensation, see Figure 2.10. From the source S (an annihilated positron- electron pair) two gamma photons  $g_1$  and  $g_2$  are emitted and travelling towards the detectors  $D_1$  and  $D_2$  respectively. The distance that  $g_1$  travel in the body is  $x_0 - x_1$  and the distance that  $g_2$  travel in the body is  $x_2 - x_0$ . The photons  $g_1$  and  $g_2$  are then attenuated with the probability functions

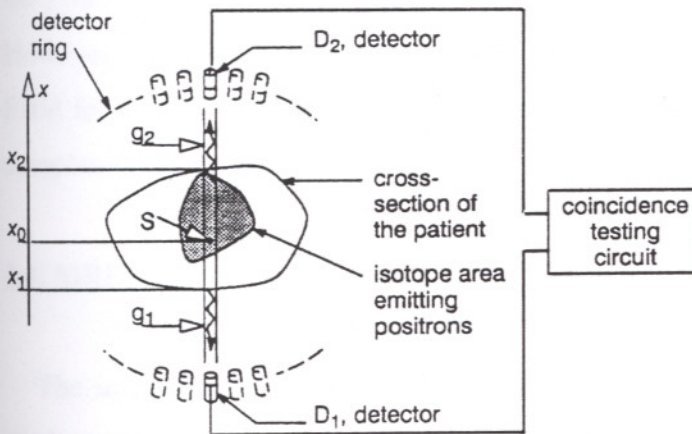


Figure 2.10 The principle of the PET.

$$\exp\left[-\int_{x_1}^{x_0} \mu(x)dx\right] \text{ and } \exp\left[-\int_{x_0}^{x_2} \mu(x)dx\right], \text{ respectively,}$$

where  $\mu(x)$  is the attenuation coefficient along the actual path in the body. The total attenuation  $\mu_{tot}$  is independent of the source position  $x_0$  along the actual path since

$$\begin{aligned} \mu_{tot} &= \exp\left[-\int_{x_1}^{x_0} \mu(x)dx\right] \cdot \exp\left[\int_{x_0}^{x_2} \mu(x)dx\right] \\ &= \exp\left[-\int_{x_1}^{x_0} \mu(x)dx - \int_{x_0}^{x_2} \mu(x)dx\right] = \exp\left[-\int_{x_1}^{x_2} \mu(x)dx\right] \end{aligned} \quad (2.3)$$

The attenuation factor for every line through the body can then be estimated and put into the reconstruction formula. An elegant and exact way to estimate the attenuation

factors is by means of a so-called transmission study, which is made with an initial step in the PET examination. During this calibration event 511 keV gamma photons are generated not by annihilated positrons but by a controllable gamma ray- source.

Typically, the resolution of the images obtained with PET are the same as for the SPECT, i.e. 64, 128, or maybe 256. A disadvantage with PET is high cost. An examination with PET requires isotopes, which usually have to be produced by a cyclotron. The total system with both PET- camera and cyclotron is considerably more expensive than a SPECT scanner.

The Linogram Method (LM) and other Direct Fourier Methods (DFM) may replace the Filtered Backprojection Method (FBM) in SPECT and PET as well as in X- ray CT. However, the LM and other DFM have their greatest computation advantage over the FBM for large image sizes, since their complexity is  $O(N^2 \log N)$  compared to the  $O(N^3)$ -complexity for FBM.

## 2.4. MRI

The images produced by MRI (Magnetic Resonance Imaging) are similar to those produced by CT in the sense that they represent cross- sections of the human body. Here we will give a simplified description of MRI in terms of classical physics as well as a simplified description of two reconstruction methods.

MRI is based on the fact that any atom with an odd number of nucleons (protons or neutrons) has a magnetic moment, which is due to the *spin* of the nucleus. When the atom is placed in a strong magnetic field, the magnetic moment of the nucleus tends to line up with the field. The magnetic moment can then be perturbed by another rotating magnetic field. This causes the magnetic field to *precess*. After the perturbation, during the return to equilibrium state a radio frequency signal is emitted. This signal can be measured and reconstruction methods can be applied to compute an image of the precessing nuclei density distribution inside the object.

The totally dominating measured element is the hydrogen nucleus, consisting of one single proton. Hydrogen is abundant in the human body, in water, fat, and all organic substances. Thus simplistically, MRI normally measures the hydrogen density in the human body.

The magnetic moment is associated with an angular momentum. The magnetic moment  $\overline{m}[\text{Am}^2]$  and the angular momentum  $\overline{L}[\text{Nms}]$  are related as

$$\overline{m} = \gamma \overline{L} \quad (2.4)$$

where  $\gamma[\text{Am/Ns}]$  is the gyro- magnetic ratio, a property of the material. When the magnetic moment is placed in a strong magnetic field  $\overline{B}_0$  it tends to line up with the field. In classical physics, it is well known that a static magnetic field  $\overline{B}_0$  acts on angular momentum with a torque. The torque influences the angular momentum (and therefore also the magnetic moment) to precess like a gyroscope around the direction of the static magnetic field.

It can be shown that the precessing frequency, called the *Larmor frequency*, is

$$\omega_0 = \gamma B_0 \quad (2.5)$$

where  $\omega_0$  [rad/s] is the angular frequency,  $\gamma$  [Am/Ns] is the gyro- magnetic ratio, and  $B_0$  [N/Am] = [Vs/m<sup>2</sup>] is the static magnetic field.

In reality the topic is more complicated than we have indicated here. Among other things, we know from quantum mechanics that the angular momentum of any system can only take certain discrete values. Here we have not taken the discreteness under consideration. Fortunately, the classical theory will give the same result as the quantum mechanic approach if small partial volume elements consisting of several nuclei are regarded instead of individual nuclei.

The hydrogen nuclei can be *excited* by a rotating magnetic field generated by coils around the object. The frequency of this rotating field must be equal to the Larmor frequency  $\omega_0$  to obtain **resonance**, otherwise the excitation will be negligible. In our classical model the excitation causes the precession angle  $\theta$  to increase according to

$$\theta = \gamma B_1 t_p \quad (2.6)$$



where  $t_p$  is the duration of the rotating magnetic field pulse. Common values of  $\overline{B}_0$  are 0.1- 2T. This gives resonance frequencies  $\omega_0$  in the radio frequency range of 3- 60 MHz. The excitation signal is often called “the radio frequency signal” because of this frequency range.

If the static strong magnetic field ( $\overline{B}_0$ ) lies in the Z- direction and the rotating magnetic field ( $\overline{B}_1$ ) lies in the xy- plane, the total vector  $\overline{B}$  is given by

$$\overline{B} = \overline{B}_0 + \overline{B}_1 = B_0 \hat{z} + B_1 (\hat{x} \cos \omega_0 t + \hat{y} \sin \omega_0 t) \quad (2.7)$$

where  $\hat{x}$ ,  $\hat{y}$ , and  $\hat{z}$  are unit vectors.

After the time  $t_p$  the excitation field  $\overline{B}_1$  is turned off. The precession of the nuclei then undergoes FID (free induction decay) as they return to their equilibrium state. During this process an electromagnetic signal at the Larmor frequency  $\omega_0 = \gamma \overline{B}_0$  is emitted. This signal can be detected by the same coils that earlier produced the rotating magnetic field. The signal is proportional to the hydrogen nuclei density of the material.

A hydrogen nucleus returns to its equilibrium state with the time constant  $T_1$  known as the longitudinal or spin- lattice relaxation time. The transverse or spin- spin relaxation time,  $T_2$  determines the time for the individual magnetic moments to come out of phase. There are also a third, more practical parameter  $T_2^*$  which includes  $T_2$  as well as local inhomogeneities of the static magnetic field. Because of physical constraints the following relationship always holds

$$T_2^* \leq T_2 \leq T_1 \quad (2.8)$$

The FID signal is attenuated both according to the  $T_1$  and  $T_2^*$  time constants. Some typical values for  $T_1$  and  $T_2^*$  are 0.5s and 50 ms, respectively. Except for imaging the nuclei density, imaging of the time constants  $T_1$  and  $T_2^*$  can provide useful images since the relation  $T_1 / T_2^*$  differs greatly between various tissues of the human body.

MRI is presently undergoing a very lively development. 2D images as well as whole 3D volumes with higher resolution and more sample points are produced. Other nuclei

than the hydrogen area of great interest for the clinicians and are beginning to be exploited. Not only the anatomy, but also the dynamic physiology of the human body can be visualised by fast 3D- imaging. Various new contrast fluids used together with MRI give images with new information.

In Figure 2.11, Figure 2.12, an MRI image data and a CT- image is shown to top to down, respectively. The CT image is a horizontal slice of the human head, which is the only practically viable orientation for CT cross- sections. The MRI technique is more flexible with regard to orientation and to produce the vertical slices in Figure 2.11, Figure 2.12 is no problem whatsoever. MRI and CT images show different things. MRI shows (mainly) hydrogen nucleus density, while CT shows matter density. Bone matter, for example, gives a large signal in CT. In Figure 2.12, the white material is the skull. In MRI, on the other hand, bone gives no signal. Consequently, the skull forms a black area between the grey skin and the grey brain in Figure 2.11.



Figure 2.11 Cross- sectional images of the human head, MRI



Figure 2.12 Cross- sectional images of the human head, CT

## **2.5. Ultrasound Imaging**

Medical ultrasound has become an important and widely accepted means for the noninvasive imaging of the human body. Studies in the wide range of clinical settings have shown it to be accurate and versatile technique yielding cross- sectional tomographic images, with little risk or discomfort and with reasonable resolution. Perhaps the most important features of this modality are, first, its ability to produce real-time images of moving structures and, second, its low cost. The first is essential for

cardiac and fetal applications; the second imperative for the efficacious practice of medicine.

All diagnostic ultrasound units now in routine clinical service are based on a simple pulse-echo technique in which the backscattered signal from an interrogating ultrasound pulse is detected as a function of time by the same transducer that served as a source. By moving transducer over a body region or by using an array of such transducers, a cross-sectional image (actually a mapping of echo intensities) can be formed. All current ultrasound-imaging systems are based on envelope detection methods and therefore only capable of displaying echo amplitude (i.e., intensity) information. Although present ultrasound systems have been clinically quite successful, they actually utilize only a small portion of the information available in the echo waveform. For example, phase information is recorded by the transducer, which is a phase-sensitive device but is not utilized in present display or measurement schemes. Other parameters such as the spectral characteristics of the echo waveform are also easily obtained but are not utilized.

In many ways, ultrasound has been the poor stepchild of medical imaging: despite technological enhancement and image improvement, today's ultrasound systems have essentially the same block diagram as those from the beginnings of the modality over 40 years. There are at least three reasons why ultrasound has remained in its "classical" imaging stage and has not yet to utilize, at least on a clinical basis, the reconstruction formalism so much a part of contemporary imaging and the unifying volume for this volume. First, and perhaps the most surprising, is the fact that data rates in ultrasound are higher than even present computer technology can handle. Since ultrasound propagates through tissue fairly slowly each image contains a large number of interactions. Retaining both the phase and amplitude information associated with the echo waveform, the database associated with a single ultrasonogram can contain several megabytes of information. To record in real time and to process in near real time, a sequence of such data sets poses significant computational demands.

A second reason ultrasound has remained at the classical imaging stage has to do with the fact that the interaction between sound and tissue is an exceedingly complex process. Major factors that describe the propagation of ultrasound in tissue include the density, elasticity, sound velocity, specific acoustical impedance, absorption, scattering, and the parameter of nonlinearity. In general these parameters are frequency and

temperature dependent and, are also a function of tissue type and pathological state. In many cases rather subtle changes in pathology can significantly alter the propagation process. By contrast the propagation of X- rays in tissue is quite simple, determined largely by the density of the material. Thus, in principle, an immense of information could be extracted from an analysis of the interaction of sound with tissue. The problem is sorting out what is in effect an overload of complex information in a realistic manner.

Finally a third reason ultrasonic imaging has not developed further si related to our lack of knowledge of fundamental interactive processes. Although our knowledge of ultrasound physics and of the interaction of ultrasound with tissue has certainly increased recently, our knowledge is still far from complete. Clearly, it is difficult to engineer a medical imaging system without all of the basic physics first in place.

## CHAPTER 3

# COMPUTER ASSISTED CRANIOFACIAL SURGERY

### 3.1. Craniofacial Surgery

Craniofacial surgery involves permanently or temporarily removing part of the craniofacial and skull bones to access deeper regions consists of operations on the soft tissues and bones in the head and face. These operations are done primarily on children born with abnormal shapes of the head and face. Sometimes such surgery is done in-patients suffering with tumors, injuries or other disorders. Reconstructive surgery is not applied to children with birth deformities, but used to remove tumors, both benign and malignant, that would otherwise not be accessible. These tumors usually begin in sinuses, nose throat, eye sockets, or ears and grow toward the brain. A team of doctors composed of head & neck surgeon, neurosurgeon, neuro- otologist, and reconstruction surgeon share their expertise to facilitate the surgery and reconstruction.

Successful execution of a surgical operation necessitates knowledge of the relevant anatomy, physiology, pathology, and wound healing, as well as the technical specifics of the intervention. Historically, this knowledge has been acquired through reading, didactic teaching, dissection room practicum, intraoperative observation, personal operative experience, and animal experimentation. Common problems with minimally altered anatomy lend themselves well to this educational process. Rare conditions with markedly aberrant anatomy are more problematic, craniofacial anomalies are such conditions.

### 3.2. Surgical Simulation

Just like flight simulators are essential today in the education of aircraft pilots, surgery simulators are expected to have an important impact on education of surgeons in the future (Satava 1990).

Currently training of surgeons is either performed using animals, cadavers, or actual human patients. But it is expected that these training methods will be restricted in the

future. Animal protection groups are pressuring the medical industry to restrict the use of animals for experiments, cadavers are generally problematic, and the use of direct training (under supervision) on human patients will probably increasingly be limited by insurance problems and patient scepticism (Nielsen 1997). In addition, the facilities necessary for training on animals are quite expensive.

Virtual reality techniques are, therefore, being developed to replace experimentation on live objects with simulated environments. In particular, the fields of craniofacial and minimally invasive surgery have seen the development of some initial surgery simulators.

### 3.3. Some Examples of Craniofacial Deformities

The most common reason for doing craniofacial surgery is to correct abnormal head and face shapes, which certain children are born with or develop after birth. This is due to craniosynostosis.

In this disorder, a child presents with a characteristic deformity of the head, or head and face. It is typically an intrauterine (inside the womb) event. This is probably due to a misdirected message from a gene. It is not due to a defect in parents. Sometimes, postnatal (after birth) synostosis can occur as well. There are rare cases where this is due to inborn errors of metabolism where the body chemicals affect bone growth abnormally.

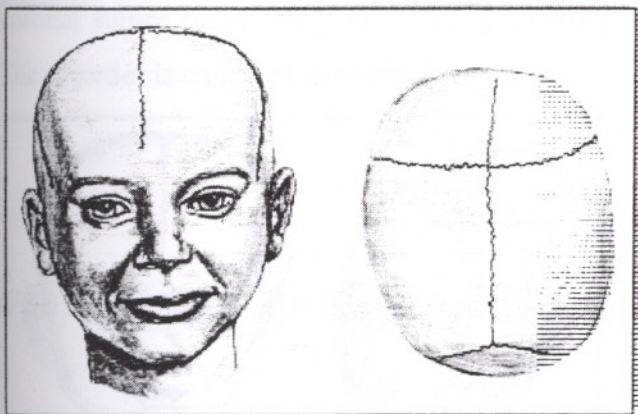


Figure 3.1

In babies, the skull bone is not one large vault as we see in adults. Rather, there are numerous bones (**Figure 3.1**). The places where these bones come together are called "sutures." Any or all of the cranial sutures can close prematurely. This is called craniosynostosis or just synostosis. It is accompanied by abnormalities of the facial skeleton. There are rare cases where other parts of the body, particularly the limbs, are affected, in addition.

In the past, these patients were socially ostracized and only reluctantly brought for treatment. Accordingly, such individuals led reclusive lives, with little ability to function normally in society.

In **Figure 3.2**, the most common type of synostosis is shown. This is sagittal synostosis, where the suture along the top of the head closes prematurely. It produces an abnormally shaped head termed scaphocephaly.

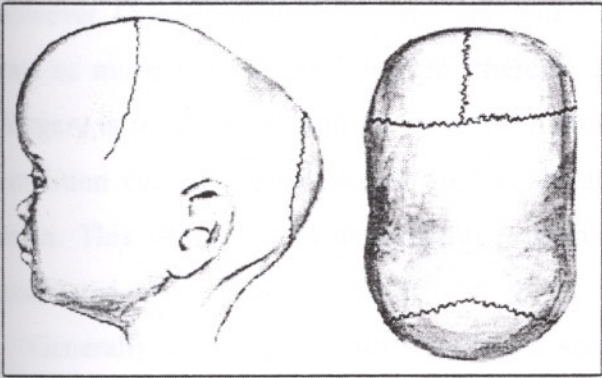


Figure 3.2

In **Figure 3.3**, the second most common disorder is shown. This occurs with one coronal suture fused (the suture that goes from left to right, just behind the hairline). The disorder is called plagiocephaly.

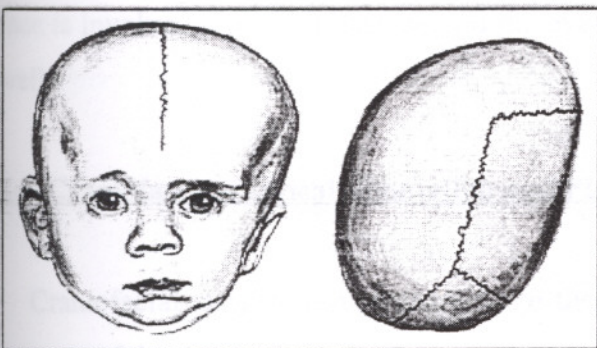
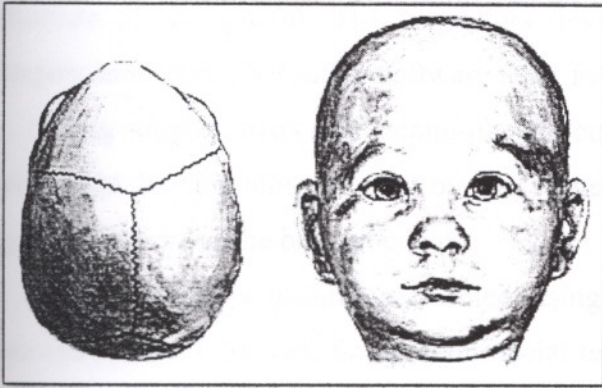


Figure 3.3

In **Figure 3.4**, trigonencephaly is shown. Here, the child is born with fusion of the suture down the middle of the forehead.



**Figure 3.4**

Fusion of one suture can be associated with mental retardation. Usually it is not, however. Operations in these cases are done for cosmetic (appearance) purposes. When two or more sutures are involved, there is a definite danger of mental retardation. Surgery is recommended in the first few months of life. Sometimes there can also be a condition called hydrocephalus. In this instance, extra fluid accumulates within the brain. This is due to an abnormality regarding circulation of this fluid and can be associated with mental retardation.

Generally speaking, craniosynostosis is seen in one out of 2,000 births. Deforming tumors and trauma are also quite common. In some cases, removal of a tumor will change the shape of the head and face. This requires bone and soft tissue reconstruction. In other instances, the same process can occur due to a traumatic event or injury. In craniosynostosis, skull growth is inhibited along the line of the fused suture. Growth occurs parallel to the suture. The purpose of surgery is to open fused sutures. When the face is involved, the skull base behind the eyes and nose have to be moved forward, as well.

### **3.4. Craniofacial Surgical Simulation and Planning**

Craniofacial surgery is used to change the bone structure of a patient's face. The purpose of this kind of surgery is to enhance the appearance patient or to restore normal function, when abnormal bone growth has resulted in restriction of body function or development.



In recent years promising work has been done to develop simulation software, which can model the soft tissue changes, that are the result of changes in the underlying bone structure of the patient. The simulators developed, are not realistic models of the surgery simulation, but rather software tools for modelling and planning surgery.

During surgery, parts of the craniofacial bones are separated from the remainder, and rearranged like building blocks, by the surgeon. Naturally, extensive planning takes place to determine the best procedure.

This planning is usually performed using X- ray cephalometric techniques and reconstructed CT images. Some craniofacial teams are using solid modelling software, but most use manual ad- hoc methods during the planning phase.

Treatment planning for complex craniofacial operations was initially based upon maxillofacial trauma experience, study of museum skulls with anomalies, conventional and polytomographic radiography, and large patient volume for a few surgeons. The development of computer assisted medical imaging, CT and MR, has made specific detailed craniofacial anatomy of affected individuals in vivo. Postprocessing of digital data acquired by CT or MR scanning (or both) into three dimensional surface or volumetric reformatted images or life sized models has made quantitative preoperative surgical planning possible. An accurate surgical "blueprint" can be generated from such images or models. Such blueprints are routinely used in many craniofacial centers (Lo et al. 1994).

Actualization of the blueprint prior to patient's operation is the objective of *surgical simulation*. Computer graphics workstations provide the platform for preoperative simulation by testing the blueprint on the digital image database rather than the patient. The outcome of the alternative procedures can be predicted, allowing selection of the one that seems to provide the best solution for the given patient. At times the selected procedure will be standard, and times it will be novel(Lo et al. 1994).

#### **3.4.1. The Basis for Surgical Simulation**

Preoperative planning for craniofacial surgery must encompass both functional and aesthetic considerations. Because many craniofacial operations are performed on infants and children, growth must be considered as well. The craniofacial operation should not only correct the dysmorphology at hand but also set the foundation for normal future

growth if possible. Ideally then, computer- assisted surgical simulation should include manipulation of both hard and soft tissues, prediction of the acute hard- and soft tissue outcomes, and prediction of surgical and growth- induced alterations of size and shape over time. Currently, the skull can be easily manipulated electronically and satisfactory accuracy is achieved. Although acute bone outcome prediction is quantitatively successful, soft tissue prediction is not. Long-term skeletal and soft tissue changes are not well simulated. Soft tissue outcome prediction and long- term temporal changes for all tissues remain the focus of current researches.

Comprehensive documentation of aberrant anatomy and function, through a multidisciplinary craniofacial team, precedes surgical simulation (Marsh and Vannier 1985). The surgical plan attempts, ideally, to normalize shape, symmetry, and dimensions of hard and soft tissues. Implementation of the surgical plan upon the digital database, rather than the patient, is surgical simulation (Lo et al. 1994).

Surgical simulation as a process, consists of three phases: development, validation and utilization (Figures 3.5 to 3.7). The developmental phase began in the mid 1980s and can be expected to continue as new simulation technology becomes available. During development, there is minimal interaction between the clinical environment and the imaging laboratory. During validation, surgical plans are simulated for patients who have already undergone the operation being simulated. The surgeon provides the imager with expert guidance, if the imager and the surgeon are not the same person, to fine-tune simulation until it coincides with the actual outcome. Although laboratory simulation- for example, phantom or cadavers- is important for verification of quantitative validity (Hildebolt et al. 1990), in vivo verification must precede clinical use. In vivo verification consists of application of the same operative blueprint to both digital database and the patient (the patient is operated upon following "traditional" computer assisted and other types of preoperative planning). Quantitative comparison is made between the simulation and the actual preoperative result. In the early phase of validation, discrepancies between the simulated and actual operative result lead to modifications of the simulation process. Once the simulation has been refined, the predicted and actual result will coincide. When this congruency occurs consistently, the simulation process has been validated and can be used prospectively as the definitive test of and guidance for the surgical plan.

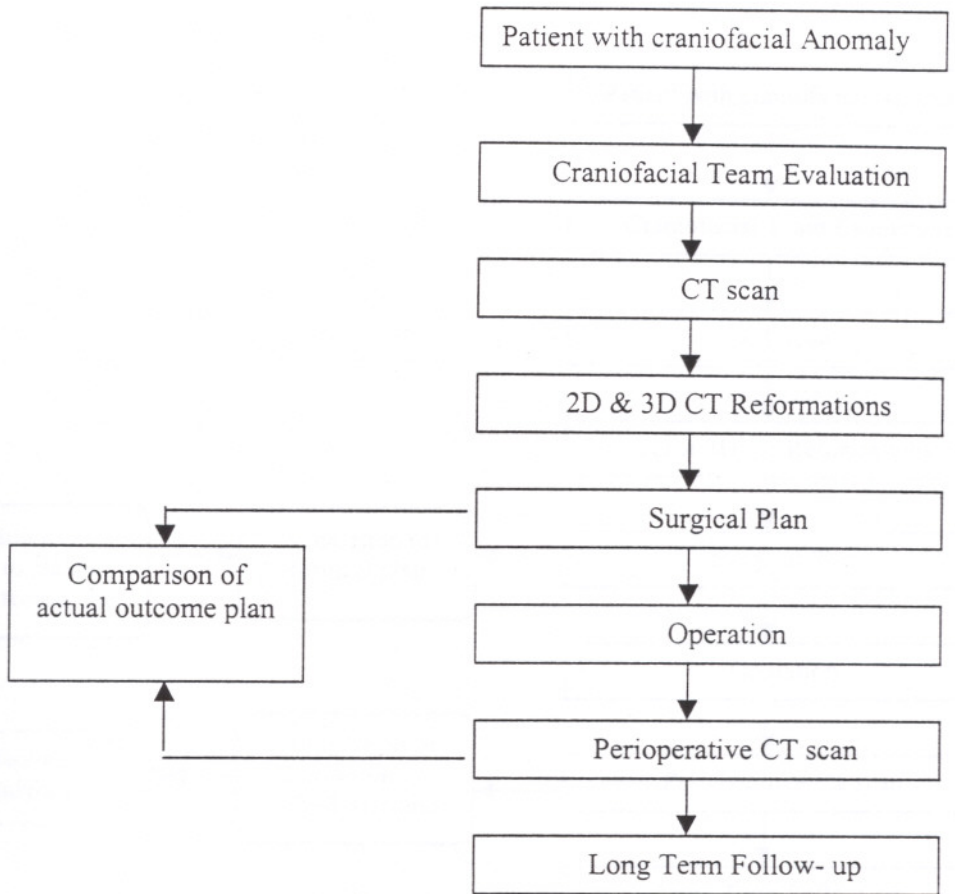


Figure 3.5 The development phase of surgical simulation consists of preoperative CT scan images with the conventionally derived preoperative plan. Familiarity with the congruencies and divergences between the two educates the planner(Lo et al. 1994).

Surgical simulation has been performed in two environments: digital graphics workstations and solid life- sized facsimiles. Although each environments has its proponents, both environments will likely to continue to be used for some time owing to unique capabilities of each. Digital graphics workstations allow the simulator unlimited interrogation of the database. Multiple simulation operations can be performed on the same database without degradation or destruction of that database. The major cost is the simulator's time. In addition, the workstation allows combinations of hard and soft tissues and can accommodate differential deformations of soft tissues following simulated surgery. The digital data format facilitates quantitative analysis of simulation and outcome. Facsimile models are intuitively more natural to practicing surgeons, who often prefer to handle objects in simulation- for example, orthognathic model surgery, because, they more closely resemble the real intraoperative experience. Structural conflict might be more easily discerned in solid model simulation than in electronic

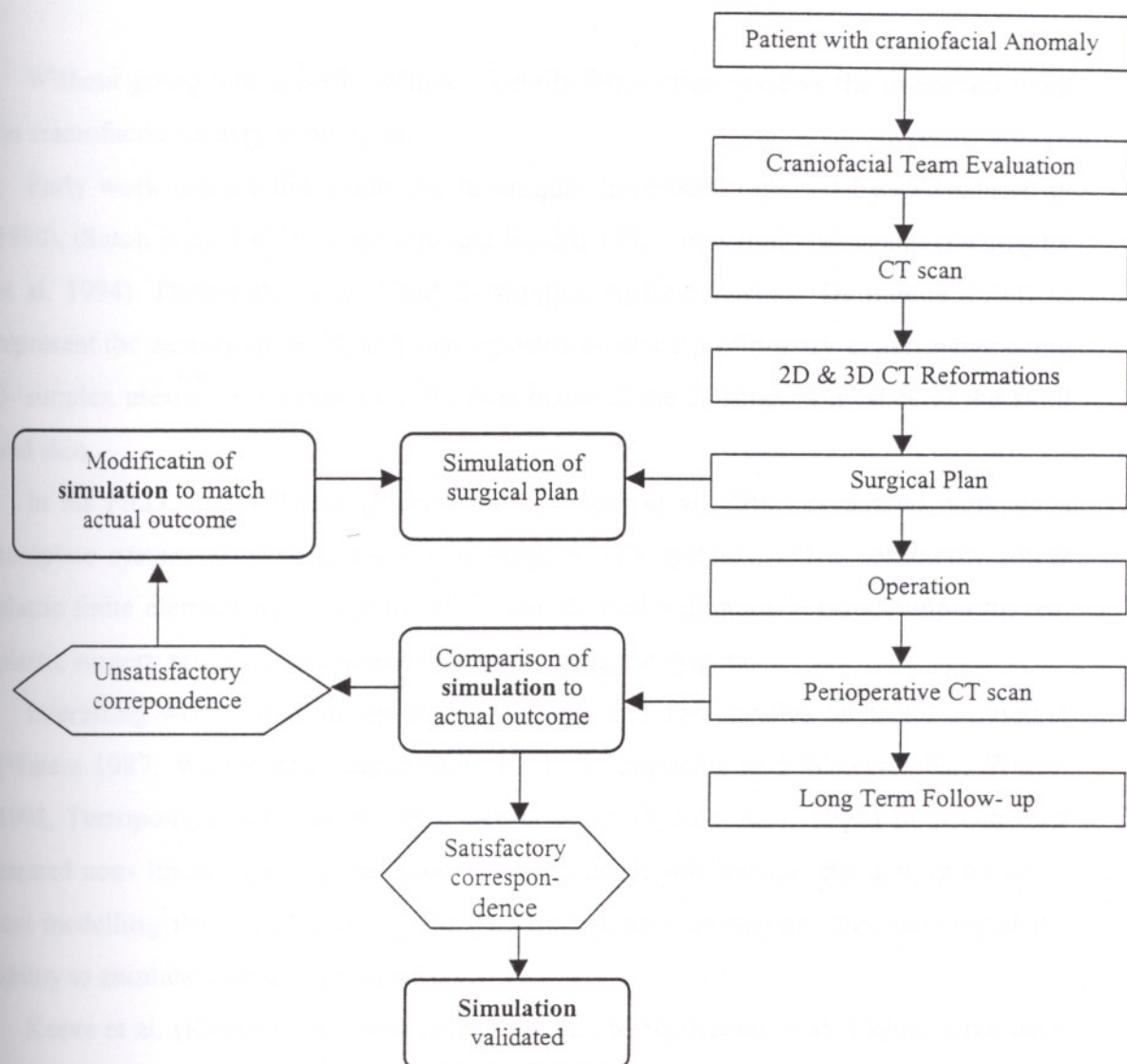


Figure 3.6 The validation phase of surgical simulation occurs after the operation has been performed. Archived preoperative CT digital data are reformatted for simulation of the operation. The simulated outcome is compared qualitatively and quantitatively with the actual outcome of the operation using archived preoperative CT data. The simulation is modified to maximize congruence between the simulation and the actual outcomes (Lo et al. 1994).

simulation. Rigid internal fixation plates can be prebent, or plates and other implants can be uniquely fabricated prior to the operation using solid models, thereby maximizing fit and minimizing intraoperative time. Destructive manipulation of models necessitates production of a new replica for each unique trial. The time and cost of model fabrication remains a limitation. Finally, models do not readily lend themselves to the quantitative comparisons necessary for simulation verification.

### 3.5. Previous Work

Without going into specific technical details this section reviews the important work on craniofacial surgery simulation.

Early work using solid modelling techniques have been reported by (Yasuda et al. 1990), (Satoh et al. 1992), (Caponetti and Fanelli 1993), (Lo et al. 1994) and (Delingette et al. 1994). Delingette et al. Used 2- simplex surface meshes (Delingette 1994) to represent the surface of skull, and also reported on some preliminary experiments using 3- simplex meshes as models of soft tissue between the 2- simplex meshes of the skull and skin.

In his Ph.D. thesis Pieper (Pieper 1992, Pieper et al. 1992) presented work on a complete system for simulating plastic surgery. His system used a volumetric linear elastic finite element model for the skin, and showed validation results comparing real plastic surgery results to predictions obtained using the system.

Interesting work has been reported by Waters and Terzopoulos on facial animation (Waters 1987, Waters and Terzopoulos 1991, Terzopoulos and Waters 1991, Waters 1992, Terzopoulos and Waters 1993, Lee et al. 1993, Lee et al. 1995) in which they created non- linear mass- spring models of the facial soft tissues. By adding muscles, and modelling the muscle activity for general human expressions, they developed the ability to animate human expressions.

Keeve et al. (Keeve et al. 1995a, Keeve et al. 1995b, Keeve et al. 1996a) have been strongly influenced by this work. They developed a similar system in which they created individualized models from CT scans and added the ability to perform surgical procedures on the bone structures using interactive computer graphics. By moving separated bone pieces and modelling the corresponding soft tissue deformation, obtained an estimate of the resulting facial changes for the particular surgery. Some validation results have been reported.

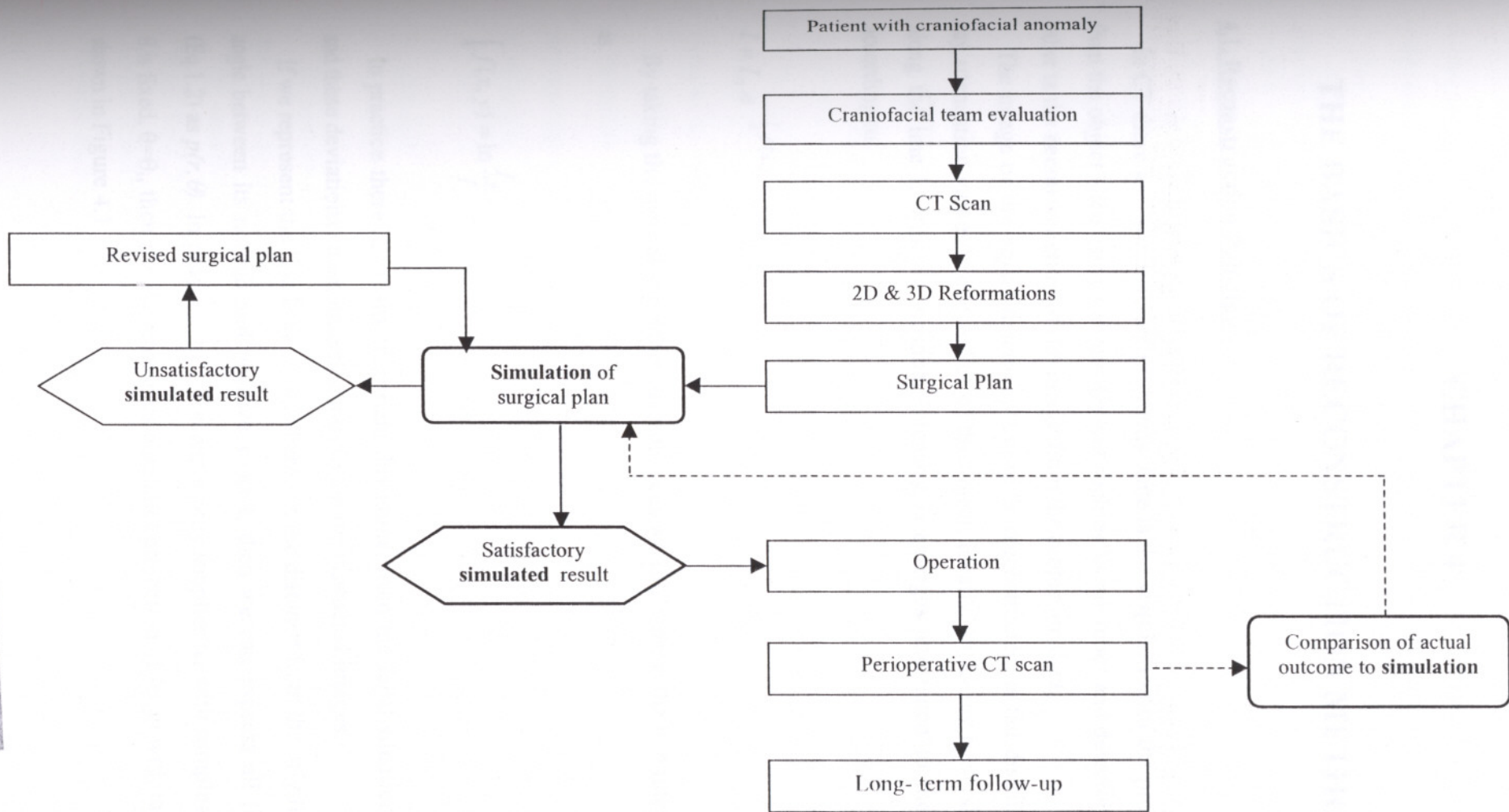
In (Keeve et al. 1995a, Keeve et al. 1995b, Keeve et al. 1996a) Keeve et al. Used the mass- spring models originally proposed by Waters and Terzopoulos with slight modifications. In later work (Keeve et al. 1996b, Keeve et al. 1996c) they have used finite element models of the soft tissue.

A similar system for modelling facial expressions around the mouth has also been developed by (Tanaka et al. 1995).

(Koch et al. 1996) have used commercially available tools to built a craniofacial simulation system. In this system soft tissue changes are modelled using a 2D finite element surface model of the skin. The skin model is connected to the skull using springs with spring constants depending on the soft tissue type they span. The soft tissue type is determined automatically by classification of the CT scan, which provides the 3D model of the patient.

Since only a *surface* model of the soft tissue is used, the work Keeve et al. Seems more convincing, although the calculation of the spring constants based on the gray level values of the CT scan, is an interesting new idea. Unfortunately, it is questionable whether sufficient information can be gathered from CT scans which normally optimized for bone visualization(Nielsen 1997).

In (Nielsen 1995, Nielsen and Cotin 1995) some preliminary work on craniofacial surgery simulation was reported by Nielsen. Using a volumetric elastic model proposed earlier by (Terzopoulos and Fleischer 1988) the deformation of the face of an individual was modelled for horizontal movement of the jaw.



**Figure 3.1** The utilization phase of surgical simulation uses preoperative simulation to test alternative procedures and to select the procedure to be executed. Comparative outcome analysis continues as a means of quality control.

İZMİR YÜKSEK TEKNOLOJİ ENSTİTÜSÜ  
 REKTÖRLÜĞÜ  
 Eğitimci Yardımcısı Dr. Öğr. Üyesi Dr. Öğr. Üyesi

## CHAPTER 4

### THE BASICS OF RECONSTRUCTION METHODS

#### 4.1. Reconstruction Principles

In CT there are two different problems. One is the **acquisition** of the projection data from the object slice using various devices such as X- ray tubes and detector arrays. The other is the **reconstruction** of the image from the projection data.

The image or the *object function*  $f(x,y)$  to be reconstructed, is the distribution of X- ray attenuation values inside a slice of the object. Ideally, an X- ray traversing the slice along the line L with the original intensity  $I_0$  emerges attenuated to the intensity I according to

$$I = I_0 \cdot e^{-\int_L f(x,y) dl} \quad (4.1)$$

By taking the natural logarithm on both sides we can express the line integral along L as

$$\int_L f(x,y) = \ln \frac{I_0}{I} \quad (4.2)$$

In practice there are many important deviations from the ideal situation in (Eq.4.2) and these deviations cause imperfections in the reconstructed images.

If we represent the line L by  $(r, \theta)$ , where  $r$  is the distance from the origin and  $\theta$  is the angle between its normal and positive  $x$ - axis, then we can express all line integrals (Eq.1.2) as  $p(r, \theta)$ . In practice the CT scanner only supplies us with samples of  $p(r, \theta)$ . If  $\theta$  is fixed,  $\theta = \theta_i$ , then  $p(r, \theta_i)$  represents a parallel projection of  $f(x,y)$  with the angle  $\theta_i$  as shown in Figure 4.1



The general problem can then be formulated as follows. Using a procedure  $\mathfrak{R}$ , a set of projections  $p(r, \theta)$  are obtained from the object function  $f(x, y)$  i.e.

$$p(r, \theta) = [\mathfrak{R}f](r, \theta) \quad (4.3)$$

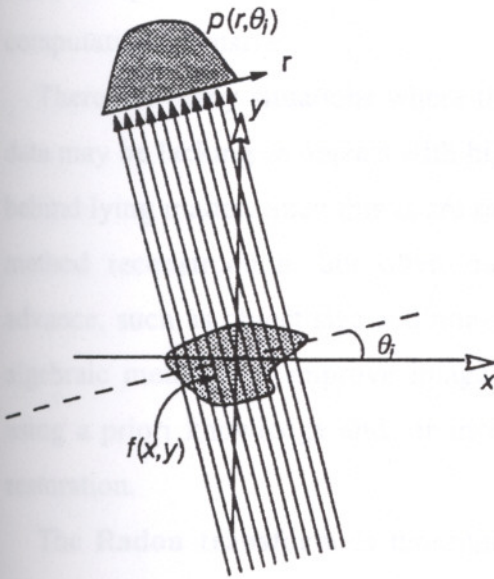


Figure 4.1 An object slice  $f(x, y)$  and one of its projections

Given  $p(r, \theta)$ , how can we obtain  $f(x, y)$ ? Obviously we need is the inverse  $\mathfrak{R}^{-1}$  of  $\mathfrak{R}$ . Reconstruction is a problem of *inversion*. The procedure  $\mathfrak{R}^{-1}$  should be applied according to

$$f(x, y) = [\mathfrak{R}^{-1} p](x, y) \quad (4.4)$$

All reconstruction methods and algorithms are implementations of  $\mathfrak{R}^{-1}$ . They can be divided in two main groups, transform algorithms and algebraic algorithms.

The **transform methods** solve the inversion problem by relying heavily on the Fourier Slice Theorem. Examples of such algorithms are the Filtered Backprojection Method (FBM) and Direct Fourier Methods (DFM). The Linogram Method is also a transform method. Developments of these algorithms are rooted in multidimensional

signal processing. An excellent overview of transform methods is given in (Lewitt 1983).

**Algebraic methods** are also called iterative methods. The solution is obtained from a first very crude guess by changing it stepwise until its projections comply with the original projections. The process may require several iterations. Although conceptually much simpler than the transform methods, this approach is normally much more computation intensive.

There are many situations where the input data are far from ideal. Some projection data may be lacking or objects with high density may absorb all X-ray energy and hide behind lying matter. Such things are rather disruptive for the image quality in transform method reconstruction, but often tractable by algebraic methods. Facts known in advance, such as object size and non-negative matter density may also be included in algebraic methods to improve image quality. Let us also remark that reconstruction using a priori knowledge and/ or incomplete data borders on the discipline of image restoration.

The **Radon transform** is thoroughly described in (Deans 1983, Toft 1996). The formula for the Radon transform of the 2D function  $f(x,y)$  is

$$p(r, \theta) = [\mathcal{R}f](r, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - r) dx dy \quad (4.5)$$

where the  $x \cos \theta + y \sin \theta - r = 0$  defines a line through  $f(x,y)$  and  $\delta$  is the Dirac impulse symbol. Thus the Radon transform  $[\mathcal{R}f](r, \theta)$  can be interpreted as the set of all line integrals of the function  $f(x,y)$ .

The inverse solution to the Radon transform was described by Johann Radon in 1917. In (Deans 1983), the Radon inversion formula is presented as

$$f(x, y) = [\mathcal{R}^{-1}p](x, y) = \frac{1}{2\pi} \int_0^{\pi} \int_{-\infty}^{\infty} \frac{\frac{1}{\pi} \frac{dp(r, \theta)}{dr}}{x \cos \theta + y \sin \theta - r} dr d\theta \quad (4.6)$$

This formula states that it is possible to obtain original function  $f(x,y)$  if all its projections  $p(r, \theta)$  are known. A straightforward digital implementation of (Eq. 4.6) will encounter difficulties since the denominator is zero for  $r=x\cos\theta+y\sin\theta$ .

**4.2. Line Integrals and Projections**

A projection is a set of line integrals. An example of physical phenomenon that generates almost perfect line integrals is the attenuation of X- rays as they propagate through biological tissue. Figure 4.1, shows a parallel projection. Figure 4.2 also shows a parallel projection, but with more detailed geometry and coordinate systems. Clearly; a parallel projection is a set of parallel line integrals.

A parallel projection for a special  $\theta=\theta_i$  is denoted  $p(r, \theta_i)$ . A set of projections, for all angles  $\theta$ , is denoted  $p(r, \theta)$  and also known as the Radon transform of  $f(x,y)$ .

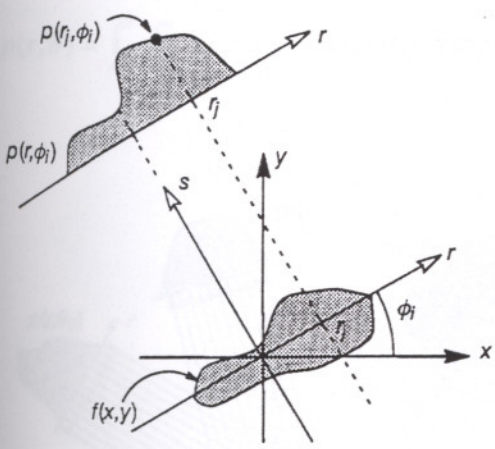


Figure 4.2 An object  $f(x,y)$  and one of its projections  $p(r, \theta_i)$ .

The  $(r,s)$  coordinate system is a rotated version of the  $(x,y)$  coordinate system as expressed by

$$\begin{bmatrix} r \\ s \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{4.7}$$

or similarly

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} r \\ s \end{bmatrix} \quad (4.8)$$

The set of projections  $p(r, \theta)$  in (Eq. 4.3) can then be formulated as

$$p(r, \theta) = \int_{-\infty}^{\infty} f(r \cos \theta - s \sin \theta, r \sin \theta + s \cos \theta) ds \quad (4.9)$$

or

$$p(r, \theta) = \int_{-\infty}^{\infty} f_r(r, s) ds \quad (4.10)$$

where  $f_r(r, s)$  is a rotated version of  $f(x, y)$ . By using the delta function we get

$$p(r, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - r) dx dy \quad (4.11)$$

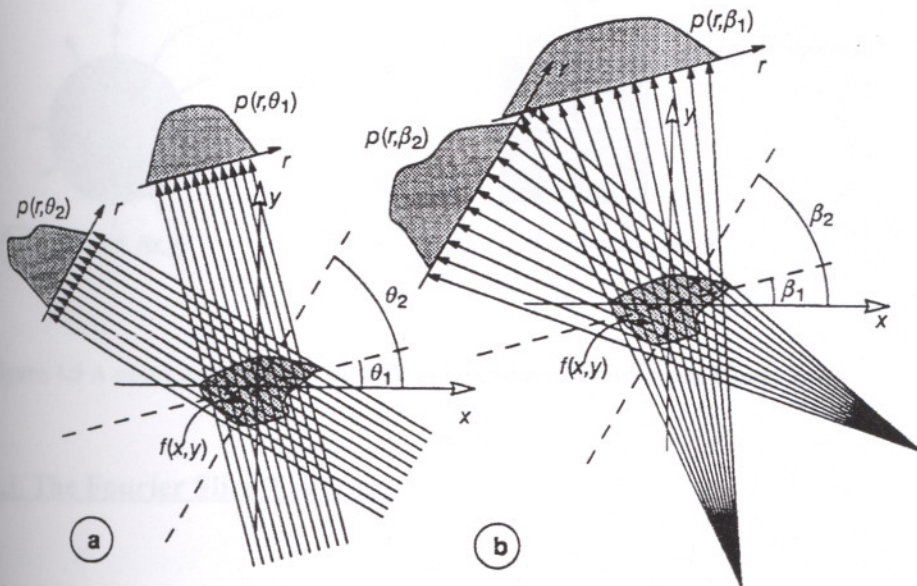


Figure 4.3 Projection geometries. (a) The geometry for parallel projections. (b) The geometry for fanbeam projections.

Another type of projection geometry is known as fanbeam projection, see Figure 4.3b. All modern commercial CT-scanners use fanbeam projection geometry.

The term **sinogram** is often used in this thesis. A sinogram is the collection of parallel projections of the object- disk taken at equidistant angles  $\theta$ , see Figure 4.4. The domain of the sinogram, seen as a 2D-function  $p(r, \theta)$ , is  $-R \leq r \leq R, -\pi/2 < \theta \leq \pi/2$ . The projection is a map of the projection data, i.e. of the Radon transform. The sinogram is cyclic in the  $\theta$ - coordinate. The term was first used by Edholm (Def.) and is motivated by the fact that a point in the object gives rise to a sinusoid in the sinogram, see Figure 4.5. The same point gives rise to a circle in the Radon space.

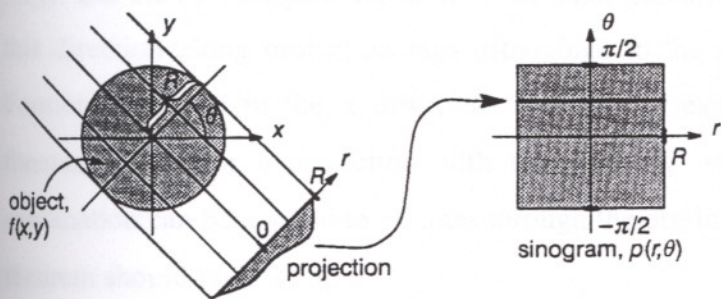


Figure 4.4 Collection of projections of an object disk in a sinogram.

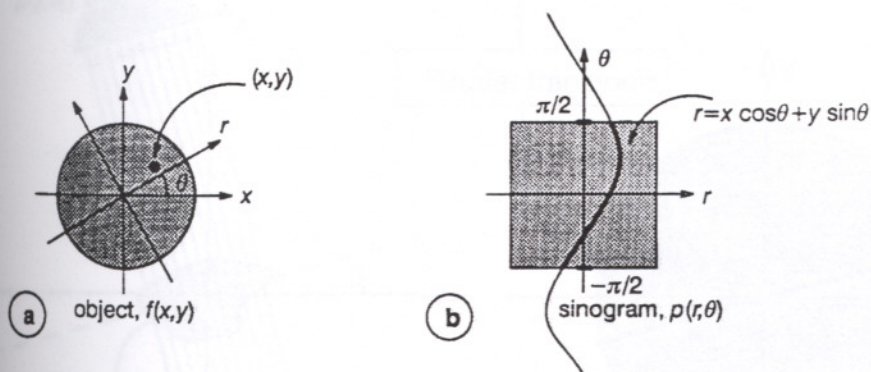


Figure 4.5 A point in the object disk gives rise to a sinusoid in the sinogram.

### 4.3. The Fourier Slice Theorem

An important property of the Radon transform is its correspondence with the Fourier transform. A projection of an object is formed by combining a set of line integrals through the object. The simplest projection, a parallel projection  $[\mathfrak{R}f](r, \theta)$ , is a collection of parallel line integrals, that is a set of „radon values along a line through the origin, as shown in Figure 4.6a.

**Theorem:** The Fourier transform  $P(R, \theta_i)$  of a parallel projection  $p(r, \theta_i)$  of an image  $f(x, y)$  taken at angle  $\theta_i$  is found in the two dimensional transform  $F(X, Y)$  on a line subtending the angle  $\theta_i$  with the X- axis.

The theorem states that the Fourier transform of  $p(r, \theta_i)$  gives the values of  $F(X, Y)$  along line AA in Figure 4.6a. An intuitive understanding of the projection slice theorem may be given by the following, see Figure 4.7. Consider the values along the X- axis in the Fourier domain,  $F(X, 0)$ . Here we find the zero frequencies in the y-direction from  $f(x, y)$ . But the zero frequencies in the y-direction can also be calculated by integrating in this direction along projection rays orthogonal to the x- axis and then taking the 1D Fourier transform in the x direction. And this is exactly what the projection slice theorem states for a projection with the angle  $\theta_i = 0$ . For symmetry reasons this explanation can be applied to all lines through the origin  $F(X, Y)$  and the projection slice theorem should hold for all  $\theta$ .

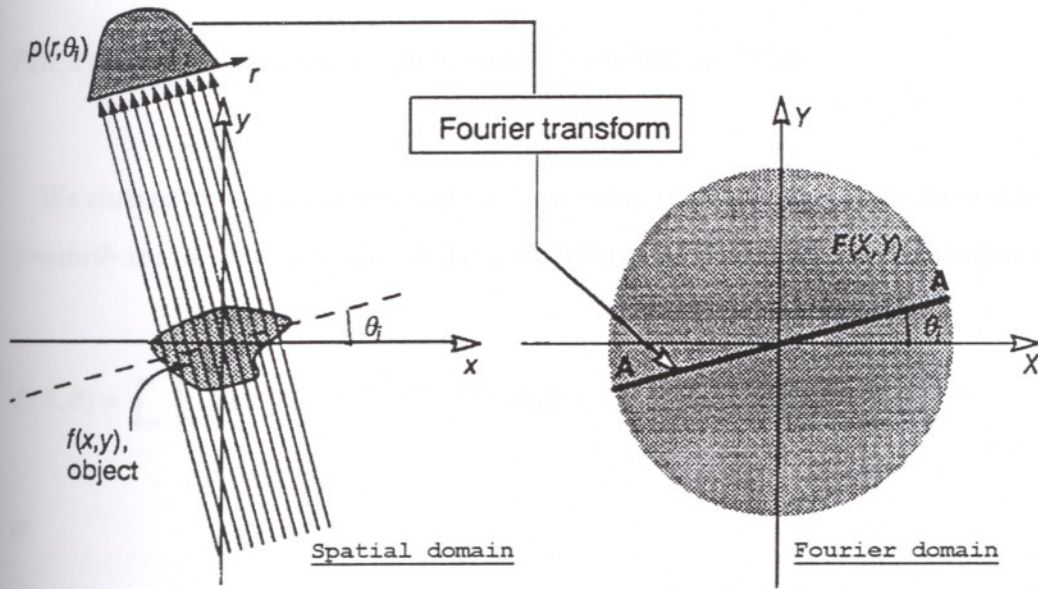


Figure 4.6 The projection slice theorem illustrated.

Mathematically the Fourier transform of  $p(r, \theta)$  in  $r$  is given by

$$P(R, \theta) = \int_{-\infty}^{\infty} p(r, \theta) e^{-j2\pi Rr} dr \quad (4.12)$$

or, when inserting Eq. 4.9

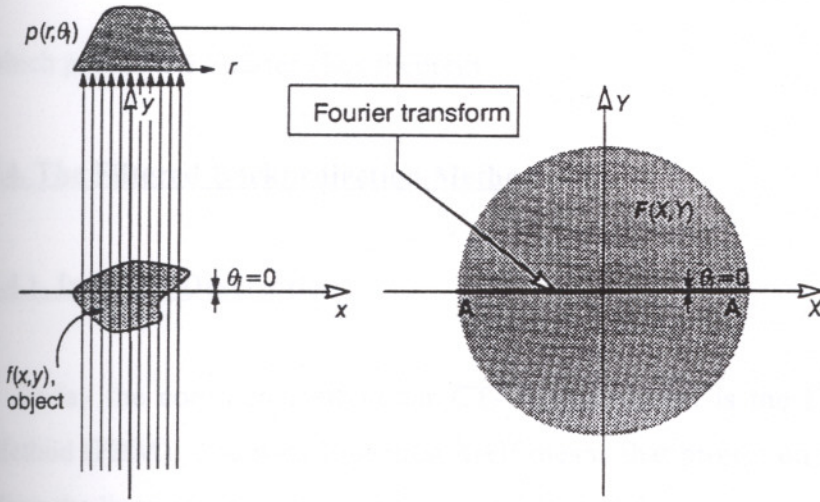


Figure 4.7 A special case of the projection slice theorem.

$$P(R, \theta) = \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} f(r \cos \theta - s \sin \theta, r \sin \theta + s \cos \theta) ds \right] e^{-j2\pi Rr} dr \quad (4.13)$$

We change to the  $(x,y)$  coordinate system using  $r = x \cos \theta + y \sin \theta$ ,  $x = r \cos \theta - s \sin \theta$ ,  $y = r \sin \theta + s \cos \theta$ , and by replacing the area differential  $dr ds$  with  $dx dy$ . The result is

$$P(R, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi R(x \cos \theta + y \sin \theta)} dx dy \quad (4.14)$$

or

$$P(R, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi [(R \cos \theta)x + (R \sin \theta)y]} dx dy \quad (4.15)$$

For any fixed value of  $\theta$ , the right hand side of this equation now represents values along a line through the origin of the two dimensional Fourier transform  $F(X,Y)$  and forming the angle  $\theta$  with the  $X$ - axis. The coordinate values along the line are  $(X,Y) = R(\cos \theta, \sin \theta)$ . Thus we can rewrite Eq. 4.15 as

$$P(R, \theta) = F(R \cos \theta, R \sin \theta) \quad (4.16)$$

which proves the Fourier slice theorem.

#### 4.4. The Filtered Backprojection Method

##### 4.4.1. Intuitive Description

Today the common method for CT- reconstruction is the Filtered Backprojection Method (FBM). The backprojection itself means that projection data are smeared back along the lines of their origin as shown in Figure 4.8.

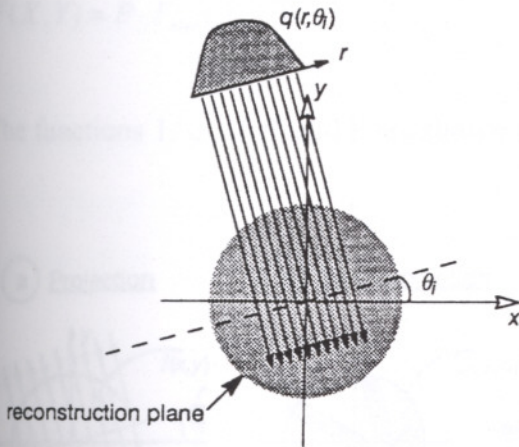


Figure 4.8 A filtered projection  $q(r, \theta_i)$  is smeared out (backprojected) over the reconstruction plane.

Projection followed by backprojection is an operation which causes some sort of low pass filtering in the image, see Figure 4.9. Projection  $p(r, \theta)$  are taken of a point- shaped object as shown to the left. Then these projections are backprojected over the reconstructed image area as shown in Figure 4.9b. The whole procedure is repeated for different equidistant  $\theta$ 's, not only for the four projections shown in Figure 4.9. We call the original function  $f(x, y)$  and the result image  $f_{blur}(x, y)$ . Since the object is a single point, it is easy to see that  $f_{blur}$  is not identical to  $f$  but to  $f$  convolved with the point spread function is  $1/\zeta$ , where  $\zeta$  is the distance from the original location of the point, i.e.



$$f_{blur}(x, y) = 1/\zeta * f(x, y) \quad (4.17)$$

The Fourier Transform of  $1/\zeta$  is  $1/P$  so that

$$F_{blur}(X, Y) = P \cdot F(X, Y) \quad (4.18)$$

Thus, to compensate for the blurring filter  $1/P$  inherent in the backprojection method, we can multiply  $F_{blur}(X, Y)$  with  $P$ , the inverse order of  $1/P$ . One possibility to reconstruct  $F(X, Y)$  is then to perform

$$F(X, Y) = P \cdot F_{blur}(X, Y) \quad (4.19)$$

The functions  $1/\zeta$ ,  $1/P$  and  $P$  are shown in Figure 4.10.

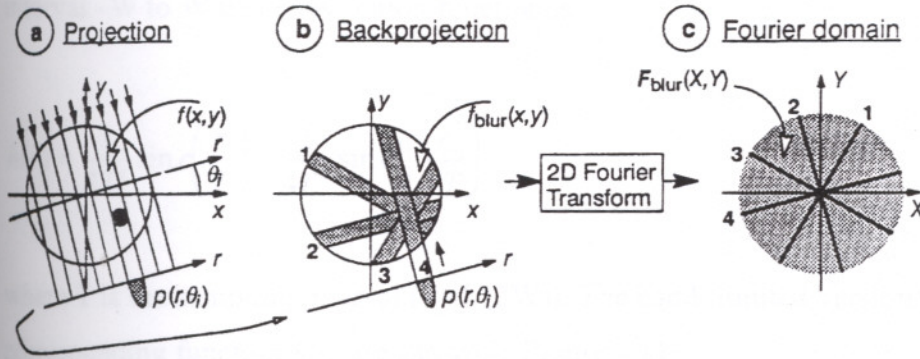


Figure 4.9 A set of projections followed by a set of backprojections causes some sort of low-pass filtering in the image. This is easy to see when the object consists of a single point.

Instead of doing the compensating filtering in the 2D Fourier domain as indicated by Eq. 4.19, we can get the same effect by filtering each projection with 1D ramp filter

$$H(R) = |R| \quad (4.20)$$

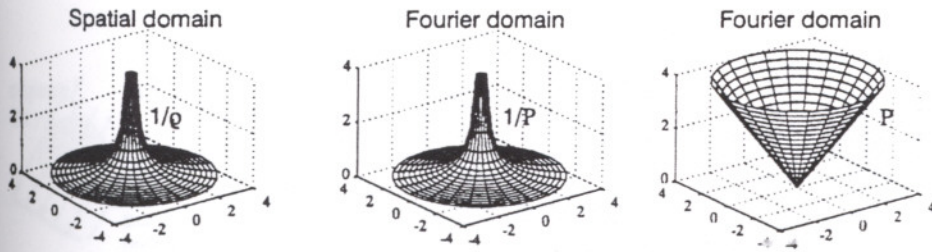


Figure 4.10 Projection followed by backprojection causes low-pass filtering in the image. The equivalent filter function is  $1/\zeta$  in the spatial domain or  $1/P$  in the Fourier domain. To compensate for the low-pass filtering we can multiply the image with  $P$  in the Fourier domain.

shown to the right in Figure 4.11. The reason is as follows. The backprojection in Figure 4.9b is a straightforward summation. In the Fourier domain this is also a summation albeit now of slices through the origin shown in Figure 4.9c. It is this 2D function that according to Eq.4.19 should be multiplied with  $P$ . However, if we multiply each contribution (i.e. the individual slices) with  $H(R) = |R|$ , we will get an identical result.

Alternatively, we can convolve each projection in the spatial domain with the inverse Fourier transform of  $H(R)$ , the 1D function  $h(r)$ . If we assume that  $H(R)$  is limited to the interval  $-W$  to  $W$  this convolution function is

$$h(r) = \frac{1}{2T^2} \text{sinc}\left(\frac{r}{T}\right) - \frac{1}{4T^2} \text{sinc}^2\left(\frac{r}{2T}\right) \quad (4.21)$$

where  $T$  is the sampling interval ( $T=1/(2W)$ ). The band-limited version of  $H(R)$  and the corresponding function  $h(r)$  are shown in Figure 4.11.

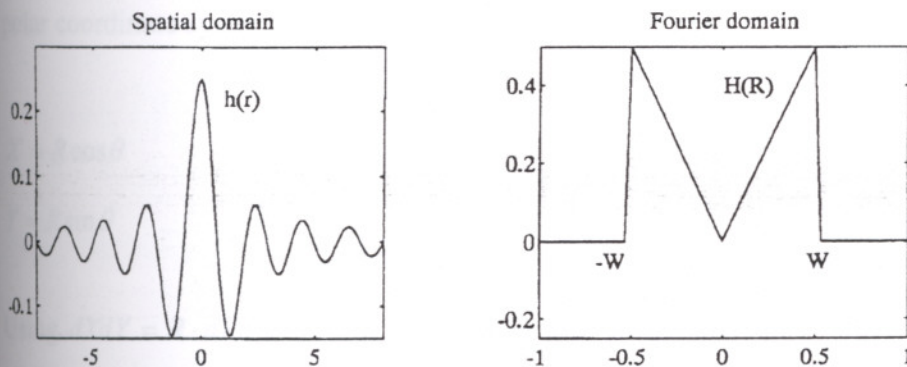


Figure 4.11 The band-limited version of the ramp-filter  $H(R) = |R|$  shown in both domains.

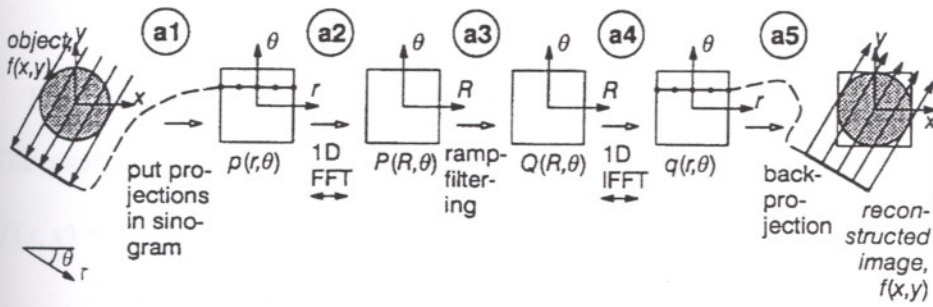


Figure 4.12 An overview of the FBM.

- A1) Take parallel projection around the object for a number of different equidistant angles  $\theta$  and collect them in a so called sinogram  $p(r,\theta)$  (the Radon transform).
- A2) Take the 1D Fourier transform of each projection to obtain  $P(R,\theta)$ .
- A3) Perform ramp- filtering according to:  $Q(R, \theta) = |R| \cdot P(R,\theta)$ .
- A4) Take the 1D inverse Fourier transform in the  $R$ - direction We now have filtered projections:  $q(r, \theta)$ .
- A5) Perform backprojection with each filtered projection  $q(r,\theta)$ .

An overview of the FBM is shown in Figure 4.12. Note that the steps a2), a3), and a4) can be replaced by a convolution with  $h(r)$ .

#### 4.4.2. Formal Proof

The 2- dimensional inverse Fourier transform of the object  $f(x,y)$  is given by

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(X, Y) e^{j2\pi(Xx + Yy)} dXdY \quad (4.22)$$

We exchange the rectangular coordinate system in the Fourier domain  $(X, Y)$ , to a polar coordinate system  $(R, \theta)$  by making the substitutions.

$$X = R \cos \theta \quad (4.23)$$

$$Y = R \sin \theta \quad (4.24)$$

Using  $dXdY = R \cdot dRd\theta$  we can now rewrite the formula as

$$f(x, y) = \int_0^{2\pi} \int_0^{2\pi} F(R \cos \theta, R \sin \theta) e^{j2\pi(x \cos \theta + y \sin \theta)} R dR d\theta \quad (4.25)$$

The integral can be split into two for  $0 \leq \theta \leq \pi$  and  $0 \leq (\theta + \pi) < \pi$ , respectively.

The result is

$$f(x, y) = \int_0^{\pi} \int_0^{\infty} F(R \cos \theta, R \sin \theta) e^{j2\pi R(x \cos \theta + y \sin \theta)} R dR d\theta + \\ + \int_0^{\pi} \int_0^{\infty} F(R \cos(\theta + \pi), R \sin(\theta + \pi)) e^{j2\pi R(x \cos(\theta + \pi) + y \sin(\theta + \pi))} R dR d\theta \quad (4.26)$$

Since

$$R \cos(\theta + \pi) = -R \cos \theta \quad (4.27)$$

$$R \sin(\theta + \pi) = -R \sin \theta \quad (4.28)$$

the Eq.4.26 yields

$$f(x, y) = \int_0^{\pi} \int_0^{\infty} F(R \cos \theta, R \sin \theta) e^{j2\pi R(x \cos \theta + y \sin \theta)} R dR d\theta + \\ + \int_0^{\pi} \int_0^{\infty} F(-R \cos \theta, -R \sin \theta) e^{j2\pi(-R)(x \cos \theta + y \sin \theta)} R dR d\theta = \\ = \int_0^{\pi} \left[ \int_{-\infty}^{\infty} F(R \cos \theta, R \sin \theta) |R| e^{j2\pi R(x \cos \theta + y \sin \theta)} dR \right] d\theta \quad (4.29)$$

Using  $r = x \cdot \cos \theta + y \cdot \sin \theta$ , we get

$$f(x, y) = \int_0^{\pi} \left[ \int_{-\infty}^{\infty} F(R \cos \theta, R \sin \theta) |R| e^{j2\pi Rr} dR \right] d\theta \quad (4.30)$$

Finally, we utilize the projection slice theorem Eq.4.16 to obtain

$$f(x, y) = \int_0^{\pi} \left[ \int_{-\infty}^{\infty} P(R, \theta) |R| e^{j2\pi Rr} dR \right] d\theta \quad (4.31)$$

Inside out Eq.4.31 is the same recipe as was illustrated in Figure 4.12. the first computation is taking Fourier transforms of the projections  $p(r, \theta)$  to obtain  $P(R, \theta)$ . The second one is

$$Q(R, \theta) = P(R, \theta) \cdot |R| \quad (4.32)$$

so that Eq.4.31 yields

$$f(x, y) = \int_0^{\pi} \left[ \int_{-\infty}^{\infty} Q(R, \theta) e^{j2\pi Rr} dR \right] d\theta \quad (4.33)$$

After 1D inverse Fourier transforms, the remaining step is the backprojection expressed by

$$f(x, y) = \int_0^{\pi} q(r, \theta) d\theta \quad (4.34)$$

From Eq. 4.7 we know that  $r = x \cos \theta + y \sin \theta$  so that Eq.4.34 can be rewritten as

$$f(x, y) = \int_0^{\pi} q(x \cos \theta + y \sin \theta, \theta) d\theta \quad (4.35)$$

This is the backprojection operation. See Figure 4.13. For every point  $(x, y)$  in the image there corresponds one certain  $r = x \cos \theta + y \sin \theta$  in the projection  $q(r, \theta)$ ,  $\theta$  fix. According to Eq.4.35 it is the value  $q(r, \theta)$  that should be accumulated to  $f(x, y)$ . The points in the image that should receive the same contribution  $q(r, \theta)$  are all found on the line  $r = x \cos \theta + y \sin \theta$  which is the line AA in Figure 4.13.

Thus, the value of the filtered backprojection,  $q(r, \theta)$ , will make the same contribution to all the points on the line AA. That is, in the reconstruction process, for every  $\theta$  the filtered projection  $q(r, \theta)$  are smeared out or backprojected over the image plane.

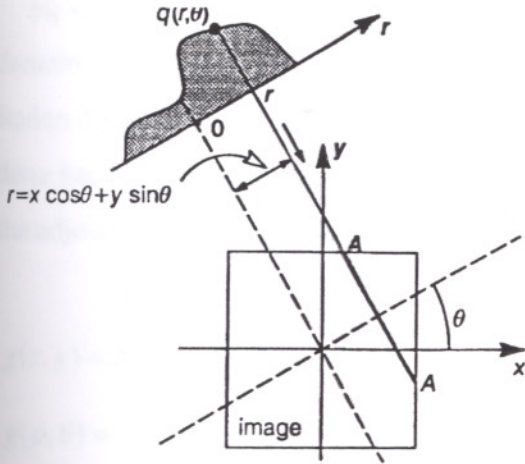


Figure 4.13 Backprojection takes place when, for a given  $\theta$ , a filtered projection  $q(r, \theta)$  is smeared out over the reconstruction plane along lines of constant  $r$ .

By insertion of Eq.4.9 in Eq.4.12 and 4.12 in 4.31, we get a fully unrolled formula Eq. 4.36 for all steps in the FBM.

$$f(x, y) = \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} f(r \cos \theta - s \sin \theta, r \sin \theta + s \cos \theta) ds \right] e^{-j2\pi Rr} dr \right] \left| R \right| e^{j2\pi Rr} dR \right] d\theta \quad (4.36)$$

#### 4.5. Filtering after Backprojection

It is also possible to make the backprojection, i.e., the adjoint Radon transform before the filtering (Deans 1993, Jain 1989). In this case another filter must be used (Toft 1996). For any function  $g(x, y)$  the Radon transform is given by

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\rho - x^* \cos \theta - y^* \sin \theta) dx^* dy^* \quad (4.37)$$

Rewriting  $g(x, y)$  using delta function, Eq.4.37 gives

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(x - x^*) \delta(y - y^*) dx^* dy^* \Rightarrow \quad (4.38)$$

$$\check{g}(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x^*, y^*) \delta(\rho - x^* \cos \theta - y^* \sin \theta) dx^* dy^* \quad (4.39)$$

Eq.4.39 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.

$$g(x, y) = \delta(x - x^*)\delta(y - y^*) \Rightarrow \quad (4.40)$$

$$\check{g}(\rho, \theta) = \delta(\rho - x^* \cos \theta - y^* \sin \theta) \quad (4.41)$$

$$\hat{g}(x, y) = \int_{-\pi}^{\pi} \check{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (4.42)$$

$$= \frac{1}{|(x - x^*) \sin \theta - (y - y^*) \cos \theta|} \Big|_{(x-x^*) \cos \theta + (y-y^*) \sin \theta = 0} \quad (4.43)$$

$$= \frac{1}{\left| -(x - x^*) \sin \arctan\left(\frac{x - x^*}{y - y^*}\right) - (y - y^*) \cos \arctan\left(\frac{x - x^*}{y - y^*}\right) \right|} \quad (4.44)$$

$$= \frac{\sqrt{1 + \left(\frac{x - x^*}{y - y^*}\right)^2}}{\frac{(x - x^*)^2}{y - y^*} + (y - y^*)} = \frac{1}{\sqrt{(x - x^*)^2 + (y - y^*)^2}} \quad (4.45)$$

It can be recognized that this is a two-dimensional convolution.

$$\hat{g}(x, y) = g(x, y) ** h(x, y) \text{ where } h(x, y) = \frac{1}{\sqrt{x^2 + y^2}} \quad (4.46)$$

Using the technique shown in Eq.4.39 it can be seen that Eq.4.46 is valid for any give  $g(x, y)$ , Eq.4.46 thus enables another inversion scheme. A two-dimensional Fourier transform of Eq.4.46 gives

$$\hat{G}(k_x, k_y) = G(k_x, k_y) H(k_x, k_y) \Rightarrow \quad (4.47)$$

$$G(k_x, k_y) = \frac{\hat{G}(k_x, k_y)}{H(k_x, k_y)} \quad (4.48)$$

From standard 2D Fourier transform table it can be found that

$$h(x, y) = \frac{1}{\sqrt{x^2 + y^2}} \leftrightarrow H(k_x, k_y) = \frac{1}{\sqrt{k_x^2 + k_y^2}} \quad (4.49)$$

This means that  $g(x, y)$  can be found as

$$\hat{g}(x, y) = \int_0^\pi \check{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (4.50)$$

$$\hat{G}(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \hat{g}(x, y) e^{-j2\pi(k_x x + k_y y)} dx dy \quad (4.51)$$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sqrt{k_x^2 + k_y^2} \hat{G}(k_x, k_y) e^{j2\pi(k_x x + k_y y)} dk_x dk_y \quad (4.52)$$

The reconstruction method presented is here called Filtering after Backprojection, which makes use of integration succeeded by a two-dimensional high pass filtering. This is an inversion algorithm very similar to Filtered Backprojection.

#### **4.6. The Direct Fourier Method**

Direct Fourier Methods of tomographic reconstruction have been investigated since the introduction of the technique in the early 1970's, although computerized tomograms employ the Filtered Backprojection Method 4.4 as means of the reconstruction. The main difficulty with DFM is the required interpolation of radially sampled data to the Cartesian grid in order to allow a 2D Fourier transform to be performed on the frequency plane data, and is at least part of the reason for the preference of FBM (Brantner et al. 1997).

A different approach is the Direct Fourier Method (DFM) (Shepp and Logan 1974, Stark et al. 1981), that is based on the projection theorem for Fourier transforms. This



theorem is known as the Fourier Slice Theorem (4.3), and relates the line integrals of a physical property to the 2D Fourier transform (FT) of the desired image. The major computational burden of this approach then becomes the FT. However, this can be achieved by exploiting the Fast Fourier Transform (FFT) algorithm, reducing the required computational time for 2D transforms.

The Fourier slice theorem demonstrates that the 2D FT of the desired image can be obtained by resampling a set of 1D FTs of the projection data. Fourier inversion of the 2D array yields the reconstruction. The resampling step is basically a coordinate transformation from the polar to the Cartesian grid in order to be able to apply a discrete Fourier transform.

This resampling step is quite difficult, and its accuracy is the crucial point of the DFM to improve reconstruction quality. There are several simple interpolation techniques, such as the nearest neighbour interpolation (NNB) and 2D linear interpolation, the bilinear interpolation (BI).

More sophisticated interpolation techniques evaluate each point in the complex frequency domain from many neighbouring samples, but are correspondingly more computationally intensive. (Brantner et al. 1997) Several different solutions have been suggested in recent publications that involve quite complex interpolation techniques. For example, Belleon and Lanzavecchia (Bellon and Lanzavecchia 1995) introduced a novel technique, the 'moving window Shannon reconstruction' (MWSR). This method uses a high number of coefficients to resample a single complex point of the 2D transform, i.e. each point is reconstructed from all samples encompassed by a window moving along the sampling interval. Matej and Bajla (Matej and Bajla 1990) suggested using 'hybrid spline- linear interpolation' (SPLI) which calculates, as a first step, de Boor's cubic spline interpolation in the radial direction; as a second step linear interpolation is performed in the angular direction. SPLI uses just four points to calculate the value of the desired point unlike MWSR.

A summary of DFM is shown in Figure 4.14, where the projections are stored immediately in the Radon space. This is quite natural because of the symmetrical relation between Radon and Fourier spaces as expressed in the Fourier Slice Theorem.

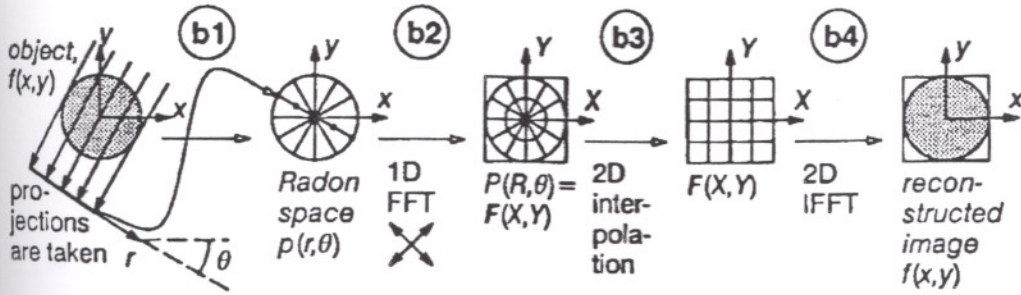


Figure 4.14 An overview of the DFM

B1) Take parallel projections  $p(r,\theta)$  around the object  $f(x,y)$  for a number of different equidistant angles. Put them intermediately in the Radon space.

B2) Take one 2D FFTs along the projection direction to arrive in  $F(X,Y)$ , the 2D Fourier transform of object.

B3) Do 2D interpolation in the  $(X,Y)$ - space to obtain values on a square grid.

B4) Perform 2D IFFT to receive the reconstructed image.

#### 4.7. The Linogram Method

The linogram method can be considered to be a direct Fourier method, which uses linogram sampling instead of sinogram sampling in the Radon and Fourier spaces. For details on the linogram sampling see (Magnusson 1993).

In the case of linogram sampling the distance of  $d$  along projection varies with  $\theta$  as

$$d(\theta) = \begin{cases} d(0^\circ) \cos \theta & -45^\circ \leq \theta \leq 45^\circ \\ d(0^\circ) \sin \theta & 45^\circ \leq \theta \leq 135^\circ \end{cases} \quad (4.53)$$

where  $d(0^\circ)$  is the sampling distance for the  $0^\circ$  projection and can be chosen to be the sampling interval in the sinogram case. In practise, this means that for all projection angles  $-45^\circ \leq \theta \leq 45^\circ$  the linear detector array lies fixed along the  $x$ -axis (Figure 4.15a) and for all projection  $45^\circ \leq \theta \leq 135^\circ$  the detector orientation is fixed along  $y$ -axis (Figure 4.15). This may be compared with sinogram sampling where the parallel projection data always are collected with a detector perpendicular to the ray direction.

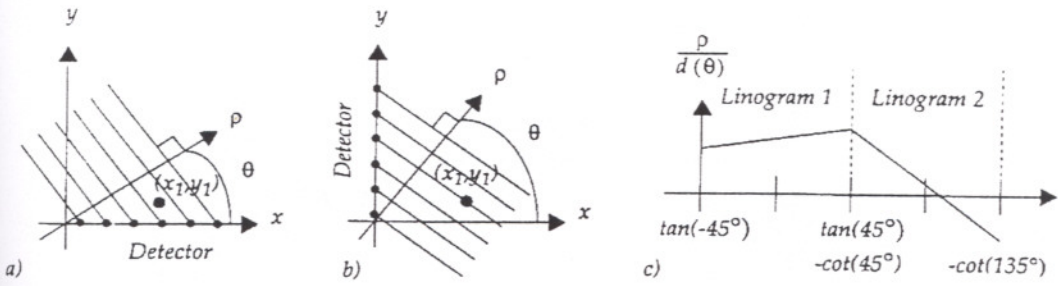


Figure 4.15 a) The detector orientation is fixed along the x- axis for all projections  $-45^\circ \leq \theta \leq 45^\circ$ .

b) The detector orientation is fixed along the y- axis for all projections  $45^\circ \leq \theta \leq 135^\circ$

c) An object point contributes to linogram data points along a line in both linograms.

The angular increment  $\Delta\theta$  is not constant as in the sinogram case, instead equidistance

$$\begin{cases} \Delta \tan \theta = 2/N & -45^\circ \leq \theta \leq 45^\circ \\ -\Delta \cot \theta = 2/N & 45^\circ \leq \theta \leq 135^\circ \end{cases} \quad (4.54)$$

The projection data organized in a Cartesian space is called a linogram. A linogram can be divided into two linograms with axis  $(\rho/d(\theta), \tan \theta), (\rho/d(\theta), -\cot \theta)$ , respectively. A specific point  $(x_1, y_1)$  contributes the Radon data along a line, hence the name linogram. (See Figure 4.15c)

Figure 4.16 attempts to visualize the difference between sinogram and linogram sampling. We note that the sinogram sampling produces on concentric circles in the Radon space (Figure 4.16), while the linogram samples are located on concentric sets of four semicircles (Figure 4.16).

The virtue of linogram projections comes forward in the Fourier domain. While the radially applied Fourier transform on each sinogram projection produces an identical pattern of concentric circles in the Fourier space (Figure 4.16), the linogram projections produce Fourier data in the form of concentric squares (Figure 4.16). Sampling along concentric squares in the Fourier domain was first proposed by (Mersereau 1973, Mersereau 1976).

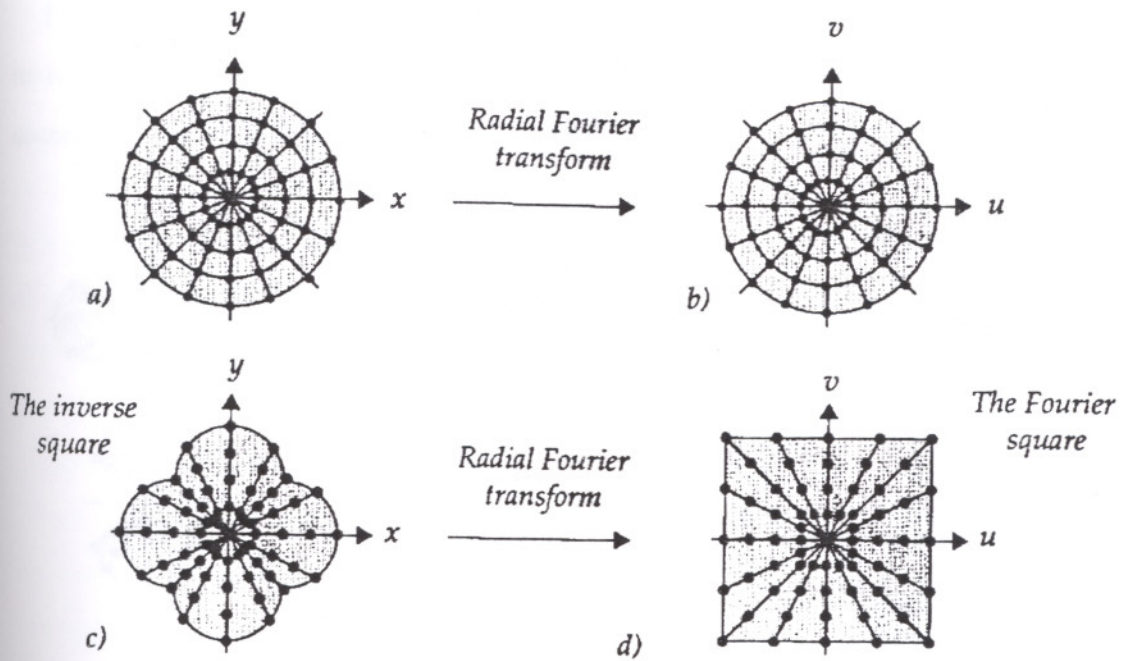


Figure 4.16 a) Sinogram sampling of the Radon space. b) The Fourier space corresponding to sinogram sampling. c) Linogram sampling of the Radon space. d) The Fourier space corresponding to linogram sampling.

Along each vertical or horizontal segment of such a square we find equidistant samples of Fourier data. Likewise the sampling distance  $D(\theta)$  is constant along every radial line. However as a direct consequence of (Eq.4.53) the sampling distance varies over  $\theta$  as

$$D(\theta) = \begin{cases} d(0^\circ) / \cos \theta & -45^\circ \leq \theta \leq 45^\circ \\ d(0^\circ) / \sin \theta & 45^\circ \leq \theta \leq 135^\circ \end{cases} \quad (4.55)$$

We use to refer to the Fourier space with the sample pattern of concentric squares as the Fourier square. The inverse square is the corresponding pattern in the Radon space consisting of four semicircles. The radial sampling distances in two spaces are inversely proportional.

Reconstruction by the direct Fourier method requires resampling in the Fourier space. Resampling from sinogram data (Figure 4.16) to a rectilinear grid requires 2D interpolation. Resampling from linogram data (Figure 4.16) on the other hand requires only 1D interpolation.

Figure 4.17 describes the complete linogram method. The projection data is divided in two linograms. The first one consists of the projections  $-45^\circ \leq \theta \leq 45^\circ$  and the second one consists of projections  $45^\circ \leq \theta \leq 135^\circ$ .

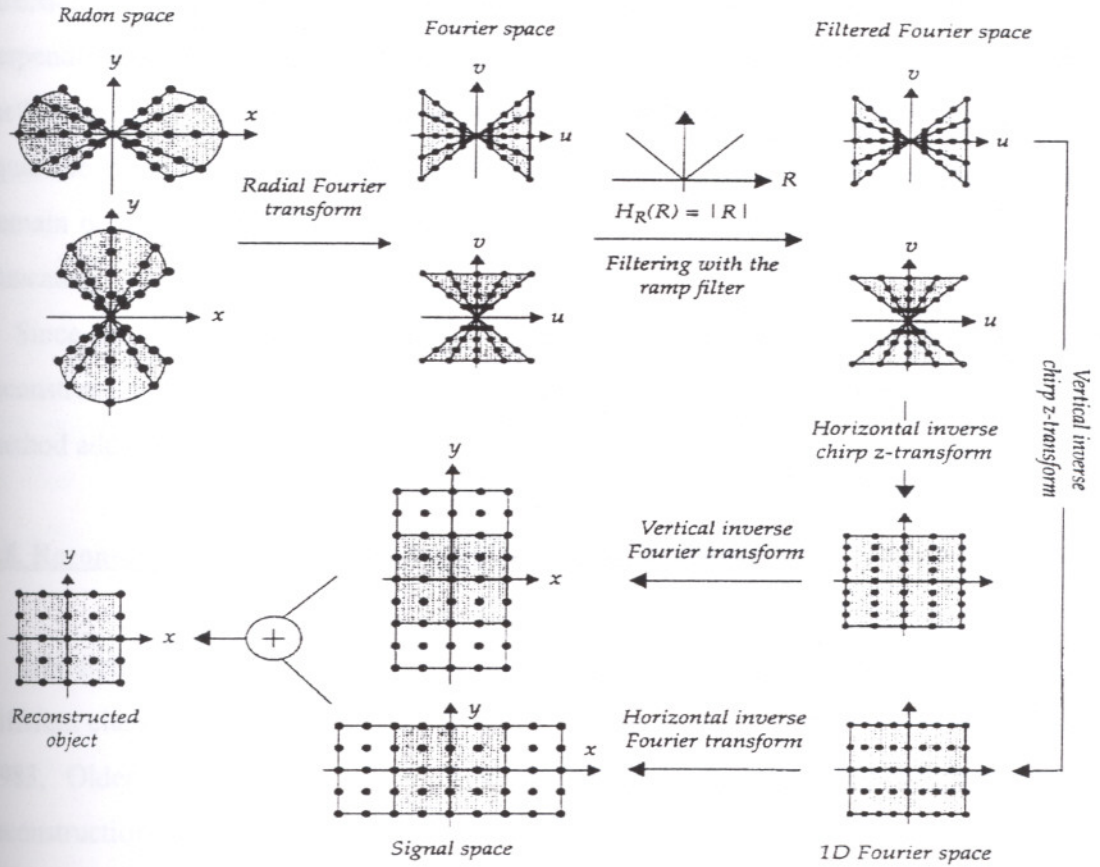


Figure 4.17 The linogram method.

With radial Fourier transforms of the projections in both linograms, we arrive to the Fourier space of the object. Due to the linogram sampling the data are sampled along concentric squares in the Fourier domain (i.e. if the two linograms are put together)

Instead of interpolation the basic linogram method employs the chirp-z transform described by (Rabiner et al. 1969). The inverse chirp-z transform computes the 1D inverse Fourier transform exactly in the required sample points. Therefore, and in spite of being a rightfully classified direct Fourier method, the basic linogram reconstruction is in fact performed without any interpolation at all.

In the inverse chirp- z transform all data points are weighted equally, which results in low pass filtering of the image, i.e. the lower frequencies are over represented. This is intuitively easy to understand, as the sample pattern becomes denser towards the origin

(Figure 4.16d). to compensate this low- pass filtering, ramp filtering is applied in the Fourier domain.

The inverse chirp- z transform is computed the 1D inverse Fourier transforms in one direction. To arrive in the spatial domain we apply another inverse Fourier transform, perpendicular to the one performed with the inverse chirp-z transform. The linogram method requires  $2N$  detector values in each projection to reach out the corners of the square to be reconstructed. Garbage from the ramp filtering is obtained in the signal domain outside the reconstructed square, as illustrated in Figure 4.17. This garbage is truncated.

Since the Radon data were divided into two linograms, each linogram delivers a reconstructed image obtained from half of the projections. The final step in the linogram method adds the two parts.

#### **4.8. Reconstruction Algorithms based on Linear Algebra**

Inversion of Radon transform need not be based on the direct inverse formulas. A different class of reconstruction schemes is based on linear algebra (Censor 1993, Deans 1983, Older and Johns 1993). This section presents the linear algebra based reconstruction theory.

##### **4.8.1. 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 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.

$$\begin{array}{ccc} \check{g}(\rho, \theta) = \mathfrak{R}g(x, y) & & \\ \downarrow & \downarrow & \downarrow \\ b & A & x \end{array} \quad (4.56)$$

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.,

$$b_i = b_{rT+t} = \check{g}(r, t) \quad (4.57)$$

and same technique is applied to the image, e.g.,

$$x_j = x_{nM+m} = g(m, n) \quad (4.58)$$

then using a series expansion shown in Eq.4.60 the Radon transform can be rewritten in the form shown in Eq.4.66. The vector dimensions are  $I = RT$  and  $J = MN$ , thus the transformation matrix A have RTMN elements, and will be large.

$$g(x, y) = \sum_m \sum_n g(m, n) \phi(x - x_m, y - y_n) \Rightarrow \quad (4.59)$$

$$\check{g}(r, t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left( \sum_m \sum_n g(m, n) \phi(x - x_m, y - y_n) \right) \delta(\rho_r - x \cos \theta_t - y \sin \theta_t) dx dy \quad (4.60)$$

$$= \sum_m \sum_n g(m, n) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \phi(x - x_m, y - y_n) \delta(\rho_r - x \cos \theta_t - y \sin \theta_t) dx dy \quad (4.61)$$

hence the matrix elements of the system matrix can be calculated as

$$a_{i,j} = a_{rT+t, nM+m} \quad (4.62)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \phi(x - x_m, y - y_n) \delta(\rho_r - x \cos \theta_t - y \sin \theta_t) dx dy \quad (4.63)$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \phi(x, y) \delta((\rho_r - x_m \cos \theta_t - y_n \sin \theta_t) - x \cos \theta_t - y \sin \theta_t) dx dy \quad (4.64)$$

$$= \hat{\phi}(\rho_r - x_m \cos \theta_t - y_n \sin \theta_t, \theta_t) \quad (4.65)$$

where the function  $\phi(\bullet)$  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.4.65 shows that the matrix element is the Radon transform of the expansion function, where the  $\rho$  parameter has been shifted according to the pixel position. Alternatives to the image pixel driven generation of matrix elements can be found in (Lewitt 1992).

The methods to be presented all rely on a linear dependence between two vectors; the known I- dimensional vector  $b$  ( $I=RT$ ), containing the sinogram, and the unknown J- dimensional vector  $x$  ( $J=M^2$ ). For tomography the vector  $b$  will contain the sinogram values wrapped into a vector using Eq.4.57, and  $x$  is the set of reconstructed pixels in the image formed as a vector using Eq.4.58.

Now the reconstruction problem will be written in a matrix vector formulation.

$$b = Ax \tag{4.66}$$

where  $A \in \mathbb{R}^{I \times J}$  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 Figure 4.18.

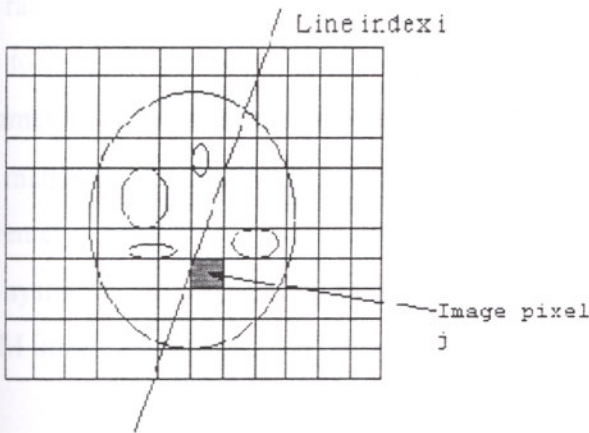


Figure 4.18 The matrix element  $a_{ij}$  can be considered as the weight factor between a certain sinogram value numbered by  $i$  and the image pixel  $j$ .

Compared to the Radon based direct reconstruction methods, 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 reconstruction.



- An irregular scanner geometry with, e.g., limited line orientation is very bad for direct reconstruction methods, and with 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  $A$  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.4.70.
- The system matrix  $A$  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 values 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  $A$  does not have a simple structure, hence the inversion of  $A$  have to use rather slow methods. The reason that  $A$  does not have a simple structure is partly due to easy sorting schemes shown in Eq.4.57 and 4.58. Another reason is that line parameters  $(\rho, \theta)$  does not have a very simple relation to the image coordinates  $(x, y)$ .
- The matrix  $A$  is normally very large, thus calculation of a generalized inverse of  $A$  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.

#### **4.8.2 Calculation of Matrix Elements**

The matrix  $A$  can be estimated in several ways. One very common approach is to use a nearest neighbour approximation. But a first order approximation better interpolation can also be used. As mentioned previous, the alternative methods can be found in (Lewitt 1992).

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.

#### 4.8.2.1. Pixel Oriented Nearest Neighbour Approximation

Around each pixel  $(x_m, y_n)$  is placed a square  $\Delta x * \Delta x$  wide. If the line with parameters  $(\rho_r, \theta_t)$  crosses the square then the matrix element  $a_{ij}$  is set to  $\Delta x$ , and else to 0. From Figure 4.19 it is easy to see that the test criterion can be written as

$$\rho' = \rho_r - x_m \cos \theta_t - y_n \sin \theta_t \quad (4.67)$$

$$|\rho' \cos \theta_t| < \frac{\Delta x}{2} \text{ and } |\rho' \sin \theta_t| < \frac{\Delta x}{2} \Rightarrow a_{rT+t, nM+m} = \Delta x \quad (4.68)$$

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 (Censor 1993) and zero otherwise. This will imply a general scaling of the solution  $x$ .

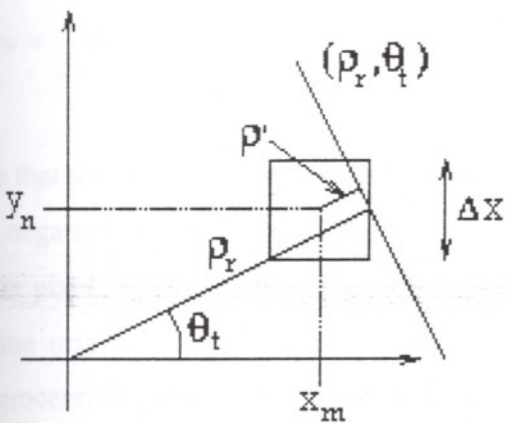


Figure 4.19 The line with index  $i$ , corresponding to  $(\rho_r, \theta_t)$ , crosses the square pixel centered at  $(x_m, y_n)$ .

### 4.8.2.2. 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 4.1 (Toft 1996), 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.4.72 and Algorithm 4.3 (Toft 1996).

### 4.8.2.3. The Sinc Interpolation Strategy

Another approach is to use the Eq.4.69, which is based on sinc interpolation scheme (Toft 1996).

$$I(\rho, \theta, x_m, y_n) = \frac{\Delta x}{\psi} \sin(\psi \min\left\{\frac{1}{|\sin \theta|}, \frac{1}{|\cos \theta|}\right\}) \quad (4.69)$$

This expression gives the elements of  $A$ .

$$a_{rT+n, nM+m} = \frac{\Delta x}{\psi} \sin(\psi \min\left\{\frac{1}{|\sin \theta|}, \frac{1}{|\cos \theta|}\right\}) \quad (4.70)$$

$$\psi = \frac{\pi}{\Delta x} (\rho_r - x_m \cos \theta_i - y_n \sin \theta_i) \quad (4.71)$$

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.4.70 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.

### 4.4.3. Duality Between Matrix Operations and the Radon Transform

Using the Radon transform scheme along with the matrix formalism as shown in Eq.4.66 implies ordinary matrix operations have equivalent Radon transform operations. Eq.4.66 is the Radon transform of discrete image  $g(m,n)$  into the full discrete parameter domain  $\tilde{g}(r,t)$ .

$$b = Ax \leftrightarrow \text{Radon Transform of full size parameter domain} \quad (4.72)$$

Iterative algorithms such as ART, use the scalar product between row number  $i$  of  $A$ , i.e.,  $a_i$  and the current reconstructed image  $x$ , i.e.,  $a_i^T x$ , which is the Radon transform of the image  $x$  into one specific sample in the parameter domain.

$$b_i = a_i^T x \leftrightarrow \text{Radon transform of one sample in the parameter domain} \quad (4.73)$$

Another common operator in iterative algorithms is the transpose of matrix. The operation  $\tilde{x} = A^T b$  is the backprojected discrete parameter domain into the image domain.

$$\tilde{x} = A^T b \leftrightarrow \text{Adjoint Radon transform of the sinogram into the image domain} \quad (4.74)$$

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 an adjoint operator, and in this case, the adjoint Radon transform is two times the backprojection operator, cf. Page 134 of (Deans 1983), and it is equivalent to the transpose of matrix, cf. Eq.4.35.

One well-known approach to solve set of equations with a non-square system matrix is to form normal equations.

$$A^T b = (A^T A)x \quad (4.75)$$

An interesting analysis is shown in (Kawata and Nalcioglu 1985) is that to the normal equations

$$x = (A^T A)^{-1} A^T b \quad (4.76)$$

can be interpreted in formalism of the direct reconstruction methods. The matrix  $A^T$  represent the backprojection operator, cf. Eq.4.74, and then  $(A^T A)^{-1}$  represents the filter in Eq.4.52. Likewise can it be shown that Filtered Backprojection can interpreted from Eq.4.74 if assuming full rank of A

$$x = (A^T A)^{-1} A^T b \quad (4.77)$$

$$= (A^T A)^{-1} A^T (AA^T)(AA^T)^{-1} b \quad (4.78)$$

$$= (A^T A)^{-1} (A^T A) A^T (AA^T)^{-1} b \quad (4.79)$$

$$= A^T (AA^T)^{-1} b \quad (4.80)$$

where  $(AA^T)^{-1}$  represents the filter in Eq.4.36 and  $A^T$  (again) represents the backprojection operator. Note that the assumption of full rank is not quite valid, due to the problems, such as zero frequency problem.

#### 4.4.4. Iterative Reconstruction using Algebraic Reconstruction Technique (ART)

A well-known way to solve the Eq.4.69 is named ART, which stands for Algebraic Reconstruction Technique. Many articles, (Censor 1983, Herman and Mayer 1993, Jain 1989) 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 the work of Johann Radon at that point, and later it was discovered that ART is a reinvention of the algorithm introduced by Kaczmarz in back in 1937.

The Basic operator required in ART is the scalar product between to certain row  $i$  of the system matrix,  $a_i$  and a solution vector  $\tilde{x}$

$$\tilde{b}_i = \sum_{j=1}^J a_{ij} \tilde{x}_j = a_i^T \tilde{x} \quad (4.81)$$

ART is formulated as an iterative reconstruction algorithm, where the solution vector in iteration  $k$  is updated by adding a scaled version of  $i$  of the system matrix.

$$x^{(k+1)} = x^{(k)} + \frac{b_i - a_i^T x^{(k)}}{a_i^T a_i} a_i \quad (4.82)$$

The main idea of ART is that the equation  $i$  is fulfilled in iteration  $k$ , which is easily shown:

$$a_i^T x^{(k)} = a_i^T x^{(k-1)} + \frac{b_i - a_i^T x^{(k-1)}}{a_i^T a_i} a_i^T a_i = b_i \quad (4.83)$$

If using the Radon transformation terminology, then ART will modify the reconstructed image in iteration  $k$ , in order to give correct Radon transform at  $(\rho_r, \theta_i)$ , where  $(r, t)$  is found from the sinogram sorting scheme shown in Eq.4.60. 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 (Jain 1989)  $i$  is 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 (Herman and Mayer 1993). Another very common and easy strategy is to choose  $i$  randomly using a uniform probability density function. Initially  $x^{(0)}$  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.

The ART algorithm can also be interpreted from a geometrical point of view. Figure 4.20 shows the use of ART in a two-parameter estimation problem with two projection lines. As shown in Eq.4.83 the current solution in iteration  $k$ , i.e.,  $x^{(k)}$  is projected perpendicularly (along the direction of  $a_i^T$ ) onto the hyperplane (in this case a line)

determined by  $b_{i(k)} = a_i^T x^{(k)}$ . As seen from Figure 4.20 the speed of convergence is very dependent on the angle between the hyperplanes.

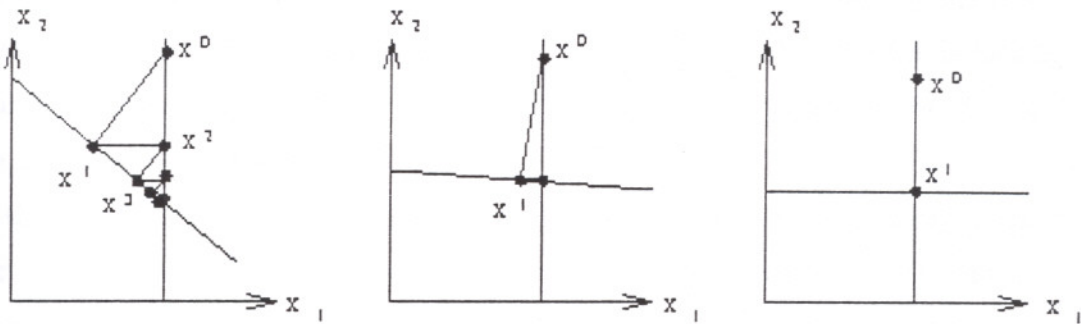


Figure 4.20 The different cases of iteration in a two parameter problem using ART. Depending on the orthogonality of hyperlines (here lines) the convergence can be slow (left most figure) or fast (right most figure)(Toft 1996).

Note that ART in each iteration only requires the scalar product between  $a_i$  and the current solution  $x^{(k)}$ , 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.4.84

$$x^{(k)} = x^{(k-1)} + \lambda_k \frac{b_i - a_i^T x^{(k-1)}}{a_i^T a_i} a_i^T \quad (4.84)$$

where  $\lambda_k$  can be set as a simple function of  $k$ , such as a linear function or an exponential decay. Even given the result Eq.4.83 it has experimentally been proven (Herman 1980), that setting  $\lambda_k$  to values different from one can improve the speed of convergence. It should be noted that the optimum value of  $\lambda_k$  is a function of  $k$ , the sinogram values, and the sampling parameters of the reconstructed image. In (Herman and Mayer 1993) it is shown that optimizing the value of  $\lambda_k$  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 4.8.5).

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 it is still known as a good reconstruction technique.

As shown in Algorithms 4.1 and 4.2 ART is very easy to implement. Only a scaler 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 be calculated in a function like it was shown in Section 4.8.2. In Algorithm 4.1 the denominator values  $a_i^T a_i$  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 4.2 which should be used several times.

---

ALGORITHM 4.1 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 4.2 THE ART ALGORITHM

---

```

For k=0 to K-1 //For K iteration 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

```

---



#### 4.8.4.1 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. There is a crude, but very easily implemented strategy is to restrict the solution after some iterations from a upper limit estimate on the solution in each pixel (Toft 1996), i.e., vector element or maybe just using a upper limit constant in each iteration. A non-negativity constraint can be imposed as shown in Algorithm 4.3, where the initialization part shown in Algorithm 4.1 has been removed. Other constraining schemes for ART can be found in (Censor 1993).

#### ALGORITHM 4.3 THE ART ALGORITHM WITH CONSTRAINTS

---

```
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
  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
End
End
```

---

#### 4.8.4.2 Initialization

Using iterative algorithms also implies that the solution vector  $x$  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. For the ART algorithm no restrictions are made concerning the initial value, but for other iterative algorithms such as EM the initial value has to be positive. Assuming that the solution is a constant, then by averaging in Eq.4.69 the value should be

$$x_j^0 = \frac{\sum_{i=1}^I b_i}{\sum_{j=1}^J \sum_{i=1}^I a_{i,j}} \forall j \quad (4.85)$$

This initialization requires that the system matrix is computed an additional time or can be combined with the first part of Algorithm 4.1. 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  $\Delta x$ , hence

$$\sum_{j=1}^J \sum_{i=1}^I a_{i,j} \approx IM\Delta x \Rightarrow x_j^0 = \frac{1}{IM\Delta x} \sum_{i=1}^I b_i \forall j \quad (4.85)$$

#### 4.4.5. 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. This lead many scientists to consider statistical approaches to derive reconstruction algorithms. One of the most prominent iterative reconstruction methods is Maximum Likelihood Reconstruction using the EM algorithm, which stands for Expectation Maximization. In the very famous articles (Shepp and Krustal 1978) by Shepp and Vardi and (Shepp et al. 1985) 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

$$L(x) = P(b|x) = \prod_{i=1}^I \frac{(b_i^*)^{b_i}}{b_i!} e^{-b_i^*} \quad (4.86)$$

where  $b^*$  contains the unknown mean values of  $b$ , i.e., the true sinogram without noise, and it is assumed that  $b^*$  fit the solution perfectly

$$b^* = Ax \quad (4.87)$$

where the coefficients of the system matrix are considered as transition probabilities normalized as

$$1 = \sum_{i=1}^I a_{i,j} \quad (4.88)$$

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  $l(x)$  by setting the derivatives of  $l(x)$  to zero

$$\frac{\partial l(x)}{\partial x_j} = -\sum_{i=1}^I a_{i,j} + \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}} = -1 + \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}} = 0 \quad (4.89)$$

In the literature an additional non-negativity constraint is imposed and it can be shown that this results in the Kuhn-Tucker conditions,

$$x_j \frac{\partial l(x)}{\partial x_j} = 0 \quad \forall j \quad \text{where} \quad x_j > 0 \quad (4.90)$$

$$\frac{\partial l(x)}{\partial x_j} \leq 0 \quad \forall j \quad \text{where} \quad x_j = 0 \quad (4.91)$$

where the first equation is used to formulate an iterative mapping scheme

$$0 = x_j \frac{\partial l(x)}{\partial x_j} = x_j \left( -1 + \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_{j'}} \right) \Rightarrow \quad (4.92)$$

$$x_j^{(k)} = x_j^{(k-1)} \frac{\sum_{i=1}^I a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_j^{(k-1)}} \quad (4.93)$$

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 (Carson and Lange 1985), and found to give very good results is used. It does not require the assumption shown in Eq.4.88, and works fine with the normalization used in Section 4.8.2.

$$x_j^{(k)} = \frac{x_j^{(k-1)}}{\sum_{i=1}^I a_{i,j}} \sum_{i=1}^I \frac{a_{i,j} b_i}{\sum_{j'=1}^J a_{i,j'} x_j^{(k-1)}} \quad (4.94)$$

Eq.4.94 is a very compact form to show the EM-algorithm, but it does not show that each iteration consists of four steps

$$b^f = Ax^{(k-1)} \quad (4.95)$$

$$b_i^g = \frac{b_i}{b_i^f} \quad (4.96)$$

$$x^b = A^T b^g \quad (4.97)$$

$$x_j^{(k)} = \frac{x_j^{(k-1)} x_j^b}{s_j} \quad (4.98)$$

where  $s_j = \sum_{i=1}^I a_{i,j}$ , forms a normalization vector that can be calculated once for all.

Eq.4.96 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.4.96 is a potential stability problem.

The EM algorithm is computationally demanding. Eq.4.95 shows that each iteration requires a forward projection (Radon transform) of the current solution, cf. Eq.4.72. The quotient in each point between the measured sinogram  $b_i$  and the forward projected solution  $b_i^f$ , i.e.,  $b^q$  is then backprojected into the image domain. Finally, in Eq.4.98, the next estimate of the solution is generated by multiplying for each index  $j$ , the current estimate of the solution  $x_j^{(k-1)}$ , 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.

In Algorithms 4.4 and 4.5 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 4.4 INITIALIZATION OF 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 shown in Algorithm 4.6, and then use Algorithm 4.5 for the remaining iterations. Note that in the first line of the Algorithm 4.5 the counter should then be For  $k = 1$  to  $K-1$ .

ALGORITHM 4.5 THE EM ALGORITHM

---

```

For k=0 to K-1 //For K iterations of EM
  For i=0 to I-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 I-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 4.6 THE FIRST ITERATION OF THE EM ALGORITHM

---

```

For i=0 to I-1 //For all vales 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 I-1 //For all values of row index
    la=a(i,j) //Store value in a 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

```

---

## CHAPTER 5

### THREE DIMENSIONAL RECONSTRUCTION

#### 5.1. Lines in a Three Dimensional Space

Lines in a three dimensional space cannot be written in a single 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

$$r = r_0 + s\tau \quad (5.1)$$

where  $s$  is the free parameter, and  $r_0$  is an offset vector. The vector  $\tau$  is a directional vector, which can be normalized to the unit length, i.e.,  $|\tau| = 1$ . The vector can, e.g., be described by

$$\tau = \begin{pmatrix} \cos \theta \cos \phi \\ \sin \theta \cos \phi \\ \sin \phi \end{pmatrix} \quad (5.2)$$

The base point vector (or offset vector)  $r_0$  is described by two parameters  $u$  and  $v$  using two directional vectors  $\alpha$  and  $\beta$ , both normalized to unit length.

$$r_0 = u\alpha + v\beta \quad (5.3)$$

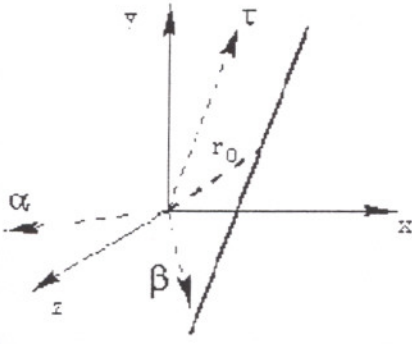


Figure 5.1 The  $(x,y,z)$  coordinate system and a line lying along to the  $\tau$  axis. The base point vector  $r_0$  can be written as a linear combination of  $\alpha$  and  $\beta$ .

It can be chosen that three vectors  $\tau$ ,  $\alpha$  and  $\beta$  form an orthogonal basis, and the last rotational degree of freedom can be used for specifying that the  $z$ - component of  $\alpha$  is zero. In (Clack et al. 1989),  $\alpha$  and  $\beta$  are defined as

$$\alpha = \begin{pmatrix} -\sin \theta \\ \cos \theta \\ 0 \end{pmatrix} \text{ and } \beta = \begin{pmatrix} -\cos \theta \sin \phi \\ -\sin \theta \sin \phi \\ \cos \phi \end{pmatrix} \quad (5.4)$$

It should be mentioned that other symbols of the vectors can be found in the literature, e.g., (Orlov 1975a, Orlov 1975b, Nattarer 1986). Following (Clack 1992), line integrals through a three- dimensional space can be expressed by the symbols defined in Eqs. 5.1 and 5.3.

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\tau + u\alpha + v\beta) ds \quad (5.5)$$

where  $(\theta, \phi, u, v)$  is four dimensional parameter domain. The mapping from the  $(x,y,z)$  domain to  $(s,u,v)$  is very useful

$$r = s\tau + u\alpha + v\beta \quad (5.6)$$

$$r = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos \theta \cos \phi & -\sin \theta & -\cos \theta \sin \phi \\ \sin \theta \cos \phi & \cos \theta & -\sin \theta \sin \phi \\ \sin \phi & 0 & \cos \phi \end{pmatrix} \begin{pmatrix} s \\ u \\ v \end{pmatrix} = Q_p \quad (5.7)$$



Using that the base vectors are orthogonal and normalized to unit length, hence the rotation matrix  $Q$  is unitary i.e.,

$$p = Q^{-1}r = Q^T r \quad (5.8)$$

One feature of matrix  $Q$  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 Eg 5.5 implies that a three dimensional function  $g(x,y,z)$  is transformed into a four dimensional parameter domain  $\check{g}(\theta, \phi, u, v)$ .

It has been chosen to denote the line integrals of  $g(r)$  with the symbol  $\check{g}$ . It can be argued that Eq. 5.5 is not a radon transform of the function  $g(x,y,z)$ . It is not, but it can be seen as a hybrid or generalized Radon transform for lines through dimensional space.

In the following, Eq. 5.5 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 uniquely determine which type of transformation used. From the definition of the 3D line Radon transform some basic rules are derived and shown in APPENDIX A. 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  $g(r)$  and a kernel  $h$  expressed in the coordinates  $p$  for a given combination of the angles  $\theta$  and  $\phi$ .

$$h(r) = h(p) = h(s, u, v) = \delta(u)\delta(v) \quad (5.9)$$

which can be seen from the following, where \*\*\* implies a three dimensional convolution in the parameters  $(s, u, v)$ .

$$\begin{aligned} \check{g}(\theta, \phi, u, v) &= g(r) *** \delta(u)\delta(v) \\ &= \int_{s'=-\infty}^{\infty} \int_{u'=-\infty}^{\infty} \int_{v'=-\infty}^{\infty} g(s'\tau + u'\alpha + v'\beta)\delta(u - u')\delta(v - v')ds' du' dv' \\ &= \int_{s'=-\infty}^{\infty} g(s'\tau + u\alpha + v\beta)ds' \end{aligned} \quad (5.10)$$

Besides the new integration variable  $s'$ , the results matches Eq. 5.5.

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

$$\check{g}(\rho, 0, \theta) = \int_{-\infty}^{\infty} g(s \cos \theta - \rho \sin \theta, s \sin \theta + \rho \cos \theta, z_0) ds \quad (5.11)$$

## 5.2. Fourier Slice Reconstruction in 3D

From the line integrals defined in Eq. 5.5, the function  $g(r)$  can be recovered using Fourier techniques, and the Fourier Slice Theorem is now derive for 3d line integrals by applying the two dimensional Fourier Transform to each of the  $(u, v)$  planes in Eq.5.5.

$$\begin{aligned} \check{G}(\theta, \phi, v_u, v_v) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \check{g}(\theta, \phi, u, v) e^{-j2\pi(uv_u + vv_v)} dudv \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(s\tau + u\alpha + v\beta) e^{-j2\pi(uv_u + vv_v)} dudvds \end{aligned} \quad (5.12)$$

which indicates a close connection to the three dimensional Fourier transform of the function  $g(\tau)$ .

$$G(v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(r) e^{-j2\pi rv} dr \quad (5.13)$$

$$g(r) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(v) e^{j2\pi rv} dv \quad (5.14)$$

where  $v$  is the three dimensional frequency vector.

Now, the integration parameters in Eq. 5.12. is changed into  $x, y$  and  $z$ , i.e.,  $r$ . It is used that  $\tau, \alpha$  and  $\beta$  are orthogonal.

$$r = s\tau + u\alpha + v\beta \Rightarrow \quad (5.15)$$

$$u = \alpha r \text{ and } v = \beta r \Rightarrow \quad (5.16)$$

$$uv_u + vv_v = r \cdot (v_u \alpha + v_v \beta) \Rightarrow \quad (5.17)$$

$$\check{G}(\theta, \phi, v_u, v_v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(r) e^{-j2\pi r \cdot (v_u \alpha + v_v \beta)} dr \quad (5.18)$$

These interesting results shows that the two-dimensional Fourier transform of the line integrals is the three dimensional Fourier transform of the function to be reconstructed (Clack 1992).

$$\check{G}(\theta, \phi, v_u, v_v) = G(v_u \alpha + v_v \beta) \quad (5.19)$$

which is a Fourier slice theorem for line integrals in a three-dimensional space. It implies that the function  $g(r)$  can be recovered by applying a two dimensional Fourier transform to the sinogram for all values of  $(\theta, \phi)$ , followed by a mapping of the spectrum, and finally recovering the desired volume by using a three dimensional inverse Fourier transform.

$$G(v_u \alpha + v_v \beta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \check{g}(\theta, \phi, u, v) e^{-j2\pi(uv_u + vv_v)} dudv \quad (5.20)$$

$$g(r) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G(v) e^{j2\pi v} dv \quad (5.21)$$

A non-trivial problem arises when implementing the mapping of the spectrum. Each frequency point  $v$  is mapped into  $v_u \alpha + v_v \beta$ , but  $v$  is three-dimensional vector and  $v_u \alpha + v_v \beta$  has four degrees of freedom. This implies that each  $v$  matches an infinite set of parameters  $(\theta, \phi, u, v)$ . One possible solution is to use weighted averages of the possible 4D frequency vectors for each value of  $v$ , and this problem is still an area of active research, though in (Stearns 1990) several aspects have been covered.

### 5.3. The 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  $(x, y, z)$  (Clack 1992). 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.

$$g(r) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(r_0) \delta(x - x_0) \delta(y - y_0) \delta(z - z_0) dx_0 dy_0 dz_0 \quad (5.22)$$

$$\equiv \int_{-\infty}^{\infty} g(r_0) \delta(r - r_0) dr_0 \quad (5.23)$$

This implies that the corresponding Radon transform is given by

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(r_0) \left( \int_{-\infty}^{\infty} \delta(s\tau + u\alpha + v\beta - r_0) ds \right) dr_0 \quad (5.24)$$

This shows that any function  $g(r)$  can be Radon transformed, if the point source can be transformed. Now this will be done.

$$g_p(r) = \delta(r - r_0) \Rightarrow \quad (5.25)$$

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} \delta(s\tau + u\alpha + v\beta - r_0) ds \quad (5.26)$$

$$= \int_{-\infty}^{\infty} \delta((s - s_0)\tau + (u - u_0)\alpha + (v - v_0)\beta) ds \quad (5.27)$$

$$= \int_{-\infty}^{\infty} \delta(\tilde{s}\tau + (u - u_0)\alpha + (v - v_0)\beta) d\tilde{s} \quad (5.28)$$

Here the unambiguous substitution  $r_0 = s_0\tau + u_0\alpha + v_0\beta$  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 if and only if  $u = u_0 = r_0 \cdot \alpha$  and  $v = v_0 = r_0 \cdot \beta$ , and because the base vectors are orthogonal the result can be expressed using the delta function.

$$\check{g}_p(\theta, \phi, u, v) = \delta(u - u_0)\delta(v - v_0) = \delta(u - r_0 \cdot \alpha)\delta(v - r_0 \cdot \beta) \quad (5.29)$$

This can also be found directly from Eq. 5.10. It is a very important result, which can be used to formulate a backprojection operator in 3D (Clack 1992).

$$\check{g} = \int_{\Omega} \check{g}(\theta, \phi, u = r \cdot \alpha, v = r \cdot \beta) d\Omega = \int_{\theta=0}^{\pi} \int_{\phi=-\psi}^{\psi} \check{g}(\theta, \phi, u = r \cdot \alpha, v = r \cdot \beta) \cos\phi d\phi d\theta \quad (5.30)$$

For convenience the integration over angles is written as a single integral with index  $\Omega$ . If the geometry is, e.g.,  $\Omega_{\psi}$  the integration becomes the last part of Eq. 5.30, where the term  $\cos\phi$  is the Jacobian, found when converting to spherical coordinates.

#### 5.4. 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 (Clack 1992). 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 in Eq. 5.5. into Eq. 5.30 gives

$$\check{g}(r) = \int_{\Omega} \int_{s=-\infty}^{\infty} g((r \cdot \alpha)\alpha + (r \cdot \beta)\beta + s\tau) ds d\Omega \quad (5.31)$$

Now it is utilized that the  $s$ - integration can be shifted along the  $\tau$ - axis, like it was done from Eq. 5.27 to 5.28. Here the offset is chosen to  $r \cdot \tau$ .

$$\check{g}(r) = \int_{\Omega} \int_{s=-\infty}^{\infty} g((r \cdot \alpha)\alpha + (r \cdot \beta)\beta + (r \cdot \tau)\tau + s\tau) ds d\Omega \quad (5.32)$$

$$= \int_{\Omega} \int_{s=-\infty}^{\infty} g(r + s\tau) ds d\Omega \quad (5.33)$$

Which can be recognized to be a convolution, thus a 3D Fourier transform can be applied on both sides.

$$\check{G}(v) = \int_{\Omega} \int_{s=-\infty}^{\infty} \int_{r=-\infty}^{\infty} g(r + s\tau) e^{-j2\pi v \cdot r} dr ds d\Omega \quad (5.34)$$

$$= \int_{\Omega} \int_{s=-\infty}^{\infty} \int_{r=-\infty}^{\infty} g(\tilde{r}) e^{-j2\pi v \cdot (\tilde{r} - s\tau)} d\tilde{r} ds d\Omega \quad (5.35)$$

$$= \int_{\Omega} \int_{s=-\infty}^{\infty} G(v) e^{j2\pi v \cdot \tau} ds d\Omega \quad (5.36)$$

$$= \int_{\Omega} G(v) \left( \int_{s=-\infty}^{\infty} e^{j2\pi v \cdot \tau} ds \right) d\Omega \quad (5.37)$$

$$= G(v) \int_{\Omega} \delta(v \cdot \tau) d\Omega \equiv \frac{G(v)}{H_a(v)} \quad (5.38)$$

This result is a 3D Filtering after Backprojection reconstruction method. The spectrum of the backprojected sinogram is multiplied with the filter  $H_a(v)$ , shown in Eq.5.39. Finally Eq. 5.14 is used to recover the desired volume.

$$H_a(v) = \frac{1}{\int_{\Omega} \delta(v \cdot \tau) d\Omega} \quad (5.39)$$

In the geometry  $\Omega_\psi$ , the filter can also be found on an analytical form. Several papers, e.g., (Colsher 1980, Clack 1992), have shown different filters, which again have been shown to only differ with a normalization constant. Due to the circular symmetry around z- axis, the filter can be calculated, with  $v_y$ , i.e., the y- component of the frequency vector, set to zero.

$$\int_{\Omega} \delta(v, \tau) d\Omega = \int_{\phi=-\psi}^{\psi} \int_{\theta=0}^{\pi} \delta(v_x \cos \theta \cos \phi + v_z \sin \phi) \cos \phi d\theta d\phi \quad (5.40)$$

$$= 2 \int_{\phi=-\psi}^{\psi} \int_{\theta=0}^{\pi/2} \delta(v_x \cos \theta + v_z \tan \phi) d\theta d\phi \quad (5.41)$$

$$= 2 \int_{\phi=-\gamma}^{\gamma} \frac{1}{|v_x \sin \theta_x|} d\phi \quad \text{where} \quad \begin{cases} \gamma = \min\{\psi, |\arctan(v_x / v_z)|\} \\ v_x \cos \theta_0 + v_z \tan \phi = 0 \end{cases} \quad (5.42)$$

$$= 2 \int_{\phi=-\gamma}^{\gamma} \frac{\cos \phi}{\sqrt{v_x^2 - (v_x^2 + v_z^2) \sin^2 \phi}} d\phi \quad (5.43)$$

The symmetry of the integral is used again to get the general results, i.e., in the last line  $|v_x|$  is substituted back to  $\sqrt{v_x^2 + v_y^2}$ , and the result is

$$H_a(v) = \begin{cases} \frac{v}{\pi} & \text{if } \sqrt{v_x^2 + v_y^2} < |v| \sin \psi \\ v \left( 2 \arcsin \left( \frac{|v|}{\sqrt{v_x^2 + v_y^2}} \sin \phi \right) \right)^{-1} & \text{otherwise} \end{cases} \quad (5.44)$$

Two things can be noted in Eq. 5.44. First, letting  $\psi \rightarrow \pi/2$  implies full angular coverage and Eq. 5.44 becomes very easy, i.e.,  $H_a(v) = v/\tau$ . On the other hand letting  $\psi \rightarrow 0$  implies that the filter becomes close to  $H_a(v) = \sqrt{v_x^2 + v_y^2} / (2\psi)$ . Neglecting the normalization constant  $2\psi$ , the filter can be recognized as the filter used in 2D Filtering after Backprojection.

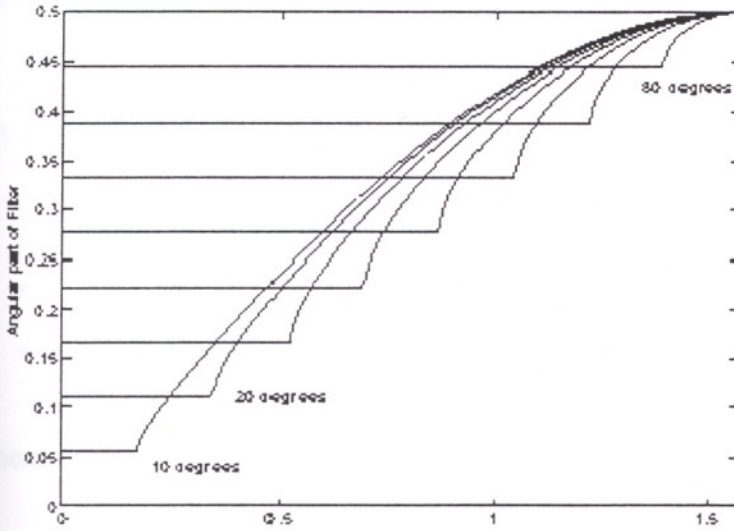


Figure 5.2 Normalized angular part of the filter as a function of  $\phi_v = \arcsin(v_z/|v|)$  for  $\psi = 10^\circ, 20^\circ, \dots, 80^\circ$ .

### 5.5. Filtered Backprojection of Line Integrals in 3D

Analogous to the two-dimensional Filtered Backprojection method, the filtering can be done before 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  $(\theta, \phi)$ . After the filtering shown in Eq. 5.45, the backprojection operator is used as shown in Eq. 5.47.

$$\bar{g}(\theta, \phi, u, v) = h_b(\theta, \phi, u, v) ** \check{g}(\theta, \phi, u, v) \quad (5.45)$$

$$= \int_{u'=-\infty}^{\infty} \int_{v'=-\infty}^{\infty} h_b(\theta, \phi, u - u', v - v') \check{g}(\theta, \phi, u', v') du' dv' \quad (5.46)$$

$$g(r) = \int_{\Omega} \bar{g}(\theta, \phi, r \cdot \alpha, r \cdot \beta) d\Omega \quad (5.47)$$

The criterion that the filter in 3D Filtered Backprojection will have to satisfy can be derived by choosing  $g(r)$  as a point source and requiring that Eqs. 5.45 and 5.47 are self-consistent, i.e.,

$$g(r) = \delta(r) \Rightarrow \check{g}(\theta, \phi, u, v) = \delta(u)\delta(v) \Rightarrow \quad (5.48)$$

$$\delta(r) = \int_{\Omega} h_b(\theta, \phi, r \cdot \alpha, r \cdot \beta) d\Omega \quad (5.49)$$

The last equation can also be viewed in the 3D Fourier domain

$$1 = \int_{\Omega} \left( \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_b(\theta, \phi, r \cdot \alpha, r \cdot \beta) e^{-j2\pi v \cdot r} dr \right) d\Omega \quad (5.50)$$

$$= \int_{\Omega} \left( \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_b(\theta, \phi, u, v) e^{-j2\pi(sv \cdot \tau + uv \cdot \alpha + uv \cdot \beta)} dp \right) d\Omega \quad (5.51)$$

$$= \int_{\Omega} H_b(\theta, \phi, v \cdot \alpha, v \cdot \beta) \int_{-\infty}^{\infty} e^{-j2\pi s v \cdot \tau} ds d\Omega \quad (5.52)$$

$$= \int_{\Omega} H_b(\theta, \phi, v \cdot \alpha, v \cdot \beta) \delta(v \cdot \tau) d\Omega \quad (5.53)$$

where 2D Fourier transform of the filter has been used

$$H_b(\theta, \phi, v_u, v_v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_b(\theta, \phi, u, v) e^{-j2\pi(uv_u + vv_v)} dudv \quad (5.54)$$

In a given geometry, several filters are valid (Clack 1992), 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 (Clack 1992) a criterion is given which makes Filtered Backprojection in a sense equivalent to Filtering after Backprojection.

$$H_b(v_u, v_v) = H_a(v_u \alpha + v_v \beta) \quad (5.55)$$

Defining  $v_x = v_u \alpha_x + v_v \beta_x$  and  $v_y = v_u \alpha_y + v_v \beta_y$ , then Eqs. 5.44 and 5.55 imply that one valid filter is

$$H_b(v_u, v_v) = \begin{cases} \frac{\sqrt{v_u^2 + v_v^2}}{\sqrt{v_x^2 + v_y^2}} & \text{if } \sqrt{v_x^2 + v_y^2} < |v| \sin \psi \\ \frac{\pi}{\sqrt{v_u^2 + v_v^2}} & \text{otherwise} \\ 2 \arcsin \left( \frac{\sqrt{v_u^2 + v_v^2}}{\sqrt{v_x^2 + v_y^2}} \sin \psi \right) & \end{cases} \quad (5.56)$$

In the geometry  $\Omega_{\pi/2}$ , the filter is very simple



$$H_b(\theta, \phi, v_u, v_v) = \frac{1}{\pi} \sqrt{v_u^2 + v_v^2} \quad (5.57)$$

The  $\Omega_{\pi/2}$  filter can be validated by inserting Eq. 5.57 in 5.53.

$$\int_{\Omega_{\pi/2}} H_b(\theta, \phi, v \cdot \alpha, v \cdot \beta) \delta(v \cdot \tau) d\Omega \quad (5.58)$$

$$= \frac{1}{\pi} \int_{\Omega_{\pi/2}} \sqrt{(v \cdot \alpha)^2 + (v \cdot \beta)^2} \delta(v \cdot \tau) d\Omega \quad (5.59)$$

$$= \frac{1}{\pi} \int_{\Omega_{\pi/2}} \sqrt{(v \cdot \alpha)^2 + (v \cdot \beta)^2 + (v \cdot \tau)^2} \delta(v \cdot \tau) d\Omega \quad (5.60)$$

$$= \frac{1}{\pi} \int_{\Omega_{\pi/2}} |v| \delta(v \cdot \tau) d\Omega = 1 \quad (5.61)$$

In the last line the result from Eqs. 5.40- 5.44 are used, with  $\psi = \pi/2$ .

# CHAPTER 6

## IMPLEMENTATION DETAILS

### 6.1. Numerical Implementation of 2D Direct Reconstruction Algorithms

In this section the implementation of Filtered Backprojection, Filtering after Backprojection and The Fourier Theorem are shown. All of these algorithms can be based on Fourier transform, which is reviewed in the Chapter 4.

#### 6.1.1. 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 the spectra (Fourier Slice theorem). For digital signals, the discrete Fourier transform (DFT) can be used to estimate the spectrum, and in practise, the spectrum is estimated by the Fast Fourier Transform.

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

$$G(f) = \int_{-\infty}^{\infty} g(t)e^{-j2\pi ft} dt \approx \Delta t \sum_{n=0}^{N-1} g_s(n)e^{-j2\pi mn/N} = G_s(m) \quad (6.1)$$

hence,  $G_s(m)$  will approximate the continuous spectrum

$$\Delta t G_s(m) \approx G\left(\frac{m}{N\Delta t}\right) \quad (6.2)$$

The discrete spectrum is in Eq.6.1 computed using the DFT of  $g_s(n)$ . With the same approach the inverse Fourier transform is commonly approximated by

$$g_a(t) = \int_{-\infty}^{\infty} G_a(f) e^{j2\pi ft} df \approx \frac{1}{N\Delta t} \sum_{m=0}^{N-1} G_d(m) e^{j2\pi mn/N} = g_d(n) \quad (6.3)$$

This technique can easily be generalized to two or more dimensions.

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.,

$$G_s(m) = G_s(m + N) \quad (6.4)$$

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 behaviour 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.

#### 6.1.1.1. The FFT Applied for Filtering

Now the practical use of the DFT for implementation of the operator  $\mathfrak{F}_{\rho_1}^{-1}$  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  $\theta_i$ , i.e.,  $h(r) = \check{g}(\theta_i, \rho_r)$ , where the values of  $\rho$  lies symmetrically around 0. In order to get the phase of the spectrum correct, the array  $g(n)$  used for the DFT must be filled as shown in Figure 6.1. Note the wrapping so negative values of  $\rho$  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.

A large number of zeros give 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 Figure 6.2 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.

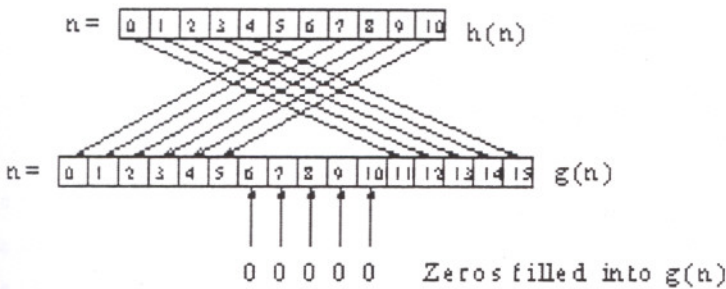


Figure 6.1 Filling an array  $h(n)$  into a larger array  $g(n)$ , when using radix-2 FFT.

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 6.1.5.1. 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.

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  $g(n)$

$$G(m) = G(-m)^* = G(M - m)^* \quad (6.5)$$

where  $*$  denotes the complex conjugate and  $g(n) = g(n + N)$ . This symmetry is easily proved from the DFT definition shown in Eq.6.1, 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.

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 (Press et al.). 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.

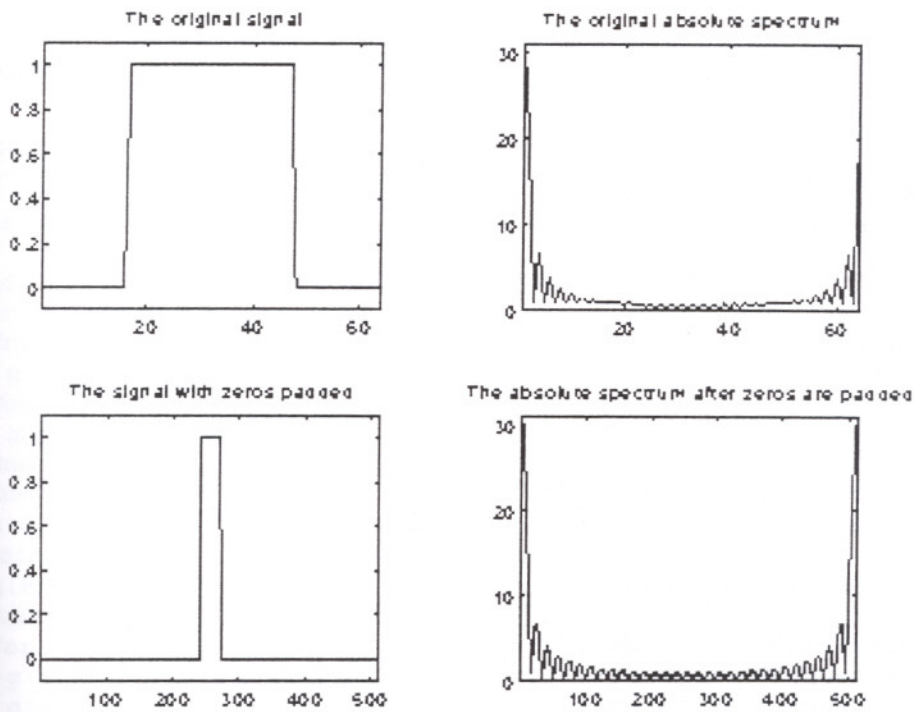


Figure 6.2 Upper left shows a square and the upper right shows the corresponding absolute spectrum. The frequency ranges from 0 to the sampling frequency (last half is the negative frequencies). Lower left shows the square where the zeros have been padded, and the lower right shows the corresponding absolute spectrum, which appears more smooth.

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.,  $|v|$ . This filter is approximated in many ways, but the structure of the FFT based filtering is the same, as shown in Algorithm 6.1. The algorithm does not demonstrate the implementation of the extra speedup gained by exploiting Eq.6.5, hence the array  $g(n)$  has complex entries. Note also that the algorithm does not split the signal values as it was shown in Figure 6.1, 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  $(g \text{ radon}(t, r))$ , which is done for sake of memory efficiency.

The filter calculated in line three of Algorithm 6.1 (sampling of Eq.6.2) 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  $|v|$  at low frequencies, and the

difference is pronounced as the frequency  $\nu$  approaches half of the sampling frequency,

$$\text{denoted by } \nu_u = \frac{1}{2\Delta\rho}.$$

ALGORITHM 6.1 FILTERING USING DFT

---

```

Set P to upper power of two (P>=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_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 radix2 FFT
  g(0)=0 //Handle zero frequency
specially //Handling half sampling
  g(P/2)=g(p/2)*f(p/2)
frequency
  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_radon(t,r)=real(g(r)) //Move the real part
  End
End
End
End

```

---

Some of the filters reported in the literature (Jain 1989) are given below, where it is only 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 6.1.5.5. Many windows can be presented, but here only four examples are given.

**The Cropped Ram-Lak/ Ramp Filter** Sample the filter  $|\nu|$  until  $\nu_l$ , i.e., the filter is

$$H(\nu)_{\text{Ram-Lak}} = |\nu| \tag{6.6}$$

and for  $v_l \leq |v| < v_u$  the filter is set to zero. The ramp filter is widely used but will amplify noise if  $v_l$  is chosen too high.

**The Shepp-Logan Filter** A sinc window is multiplied to the ramp filter

$$H(v)_{\text{Shepp-Logan}} = |v| \frac{1}{2} \frac{\sin\left(\frac{\pi v}{2v_l}\right)}{\left(\frac{\pi v}{2v_l}\right)} \quad (6.7)$$

and for  $v_l \leq |v| < v_u$  the filter is set to zero.

**The Hann Filter** A Hann Window is multiplied to the filter.

$$H(v)_{\text{Hann}} = |v| \left( 1 + \cos\left(\frac{\pi v}{v_l}\right) \right) \quad (6.8)$$

and for  $v_l \leq |v| < v_u$  the filter is set to zero.

**The Generalized Hamming Filter**

$$H(v)_{\text{GeneralizedHamming}} = |v| \left( \alpha + (1 - \alpha) \cos\left(\frac{\pi v}{v_u}\right) \right) \quad (6.9)$$

where typical values of  $\alpha$  are 0.5- 0.54. Again for  $v_l \leq |v| < v_u$  the filter is set to zero.

**Stochastic Filters** In Section 10.8 of (Jain 1989), Jain derives a parameterized filter, which is shaped to the actual noise level.

Figure 6.3 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.

In the top part Figure 6.4 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.

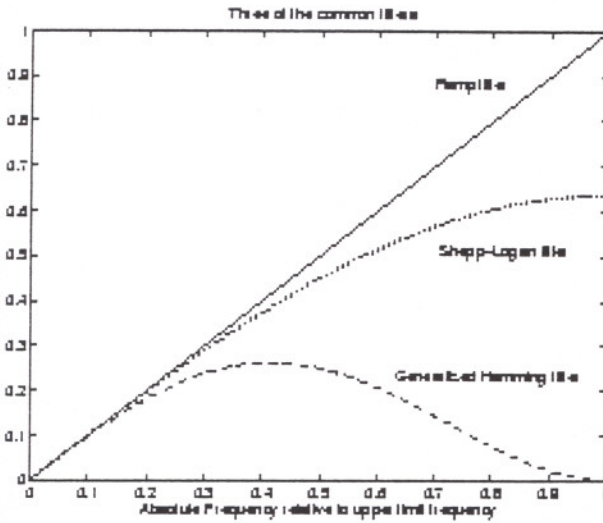


Figure 6.3 The amplitude of the Ram-Lak filter, the Shepp-Logan filter, and the generalized Hamming filter using  $\alpha=0.5$ , all three as a function of frequency normalized to the upper limit frequency,  $\nu_f$ .

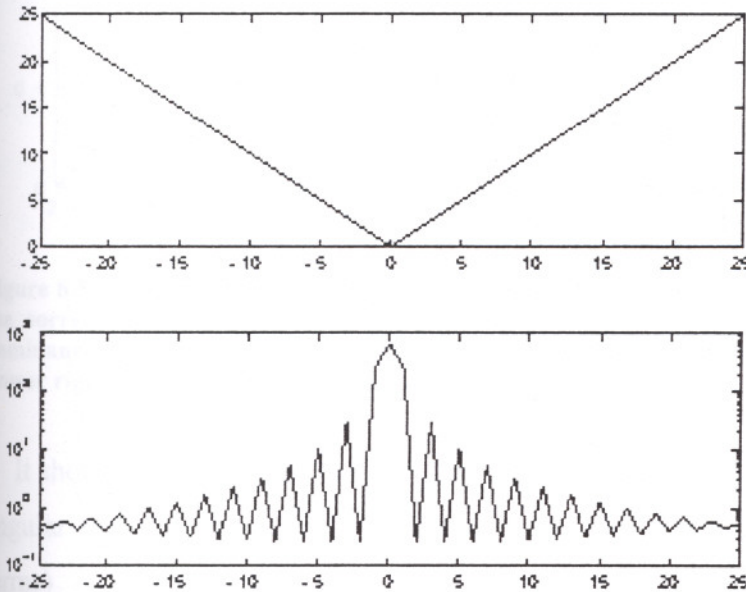


Figure 6.4 Upper shows 51 samples of a ramp filter as a function of frequency. Upper right shows the absolute value of the corresponding spectrum, found from a DFT of the same length. Here no windowing has been used to reduce the cyclical behaviour of the DFT.

Figure 6.5 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 Figure 6.7, be backprojected to demonstrate the full



reconstruction algorithm of Filtering after Backprojection. From Figure 6.5 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.

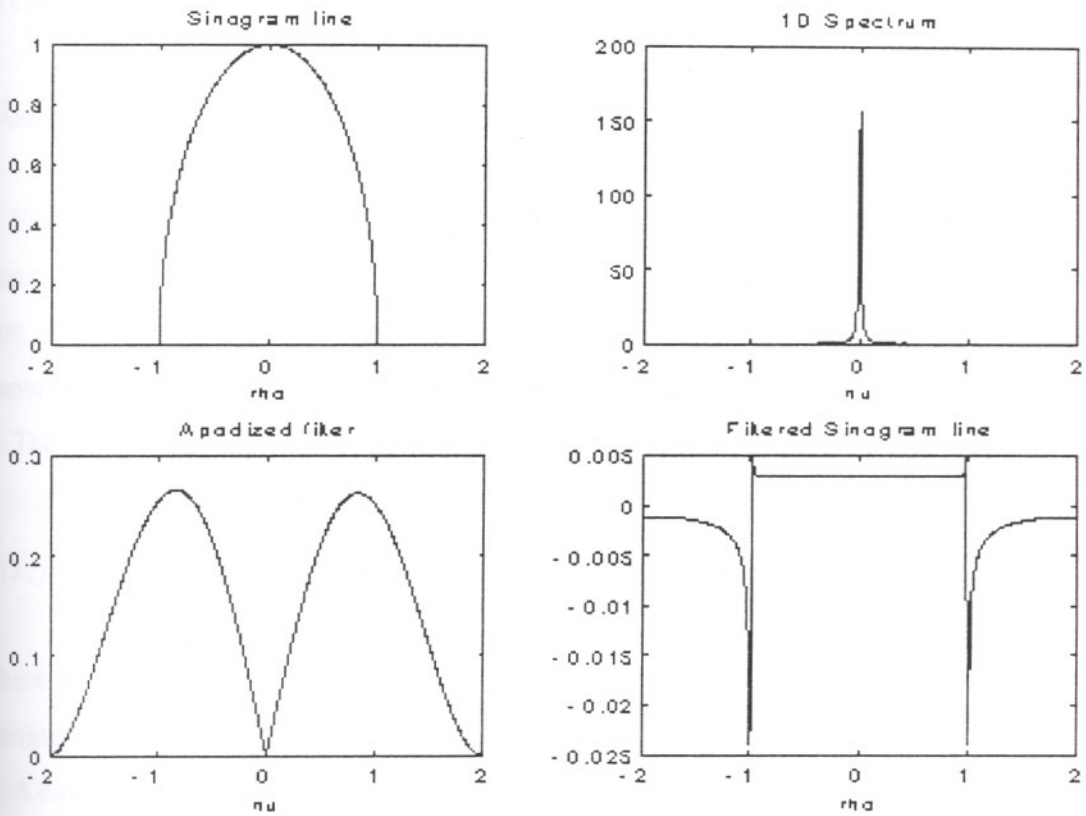


Figure 6.5 Upper left shows the sinogram for a fixed value of  $\theta$  and varying  $\rho$ . Upper right shows the corresponding discrete spectrum, and it can be noted that there is heavy low frequency dominance. Lower left shows the filter, which here is the ramp multiplied with a Hann window. Lower right shows the filtered sinogram part.

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)

$$O_{Filtering} = O(TR \log R) \tag{6.10}$$

where  $R$  should be replaced by the smallest power of two larger or equal to  $R$  if a radix-2 FFT is used.

The 1D filtering could also have been done by convolving the sinogram with the proper impulse response, which is infinitely long as indicated in Figure 6.4, 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.

### **6.1.2. 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 6.1.1.1 and in this section the implementation of the backprojection part is described.

The backprojector operator was found in Eq.4.35.

$$g(x, y) = \int_0^\pi \bar{g}(x \cos \theta + y \sin \theta, \theta) d\theta \quad (6.11)$$

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.6.11 is

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} \bar{g}(x_m \cos \theta_t + y_n \sin \theta_t, \theta_t) \quad (6.12)$$

where a one dimensional interpolation must be used in  $\rho$  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**

$$g(x_m, y_n) \approx \Delta\theta \sum_{t=0}^{T-1} \bar{g}([r^*]_t), \quad \text{where} \quad r^*(m, n; t) = \frac{x_m \cos \theta_t + y_n \sin \theta_t - \rho_{\min}}{\Delta\rho} \quad (6.13)$$



---

**ALGORITHM 6.2 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 6.3 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 6.3 it is not checked whether the value of  $rl$  correspond 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  $\rho$  direction by padding zeros, in order to fulfil

$$0 < \frac{x_m \cos \theta_t + y_n \sin \theta_t - \rho_{\min}^*}{\Delta \rho} < R - 1 \quad \forall (x_m, y_n, \theta_t) \Rightarrow R^* > \sqrt{2}(M - 1) \frac{\Delta x}{\Delta \rho} + 1 \quad (6.18)$$

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  $\rho_{\min}$  to maintain a symmetrical sampling of  $\rho$

$$\rho_{\min}^* = -\Delta \rho \frac{R^* - 1}{2} \quad (6.19)$$

The given implementation in Algorithm 6.2 and 6.3 will require two matrices of size  $MT$  and the computational load is lowered significantly.

The discrete implementation of Filtered Backprojection is schematically shown in Figure 6.6 using first filtering and then backprojection.

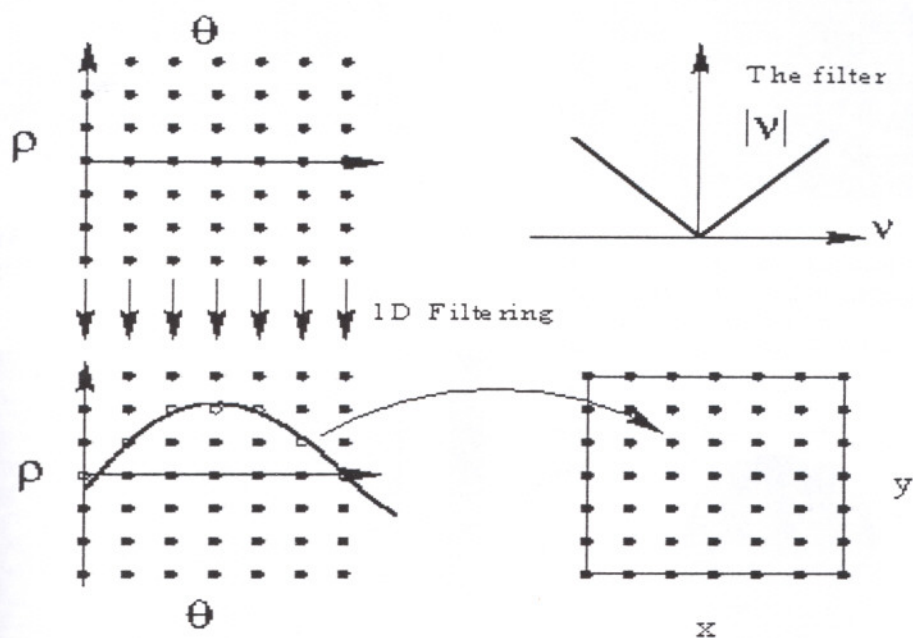


Figure 6.6 The discrete implementation of Filtered Backprojection. At the left the sinogram is filtered and then the filtered sinogram is by backprojection mapped into the reconstructed image.

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. Figure 6.5 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  $\theta_i$ . From 5 angular samples in the sinogram, i.e.,  $T = 5$ , Figure 6.7 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 Figure 6.8, 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 Figure 6.9, the artifacts are reduced in amplitude compared to Figure 6.8, and with 100 angular samples, shown in Figure 6.10, the disc is nearly recovered perfectly.

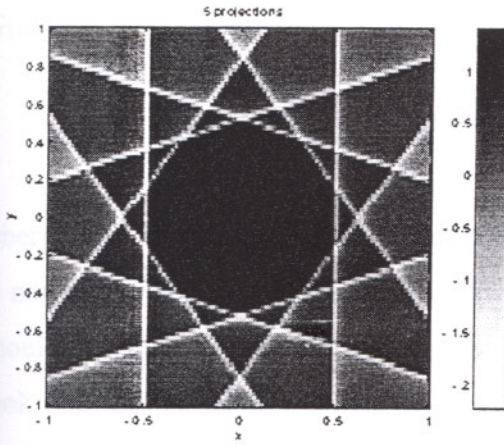


Figure 6.7 Filtered Backprojection from a sinogram with 5 angular samples ( $T=5$ )

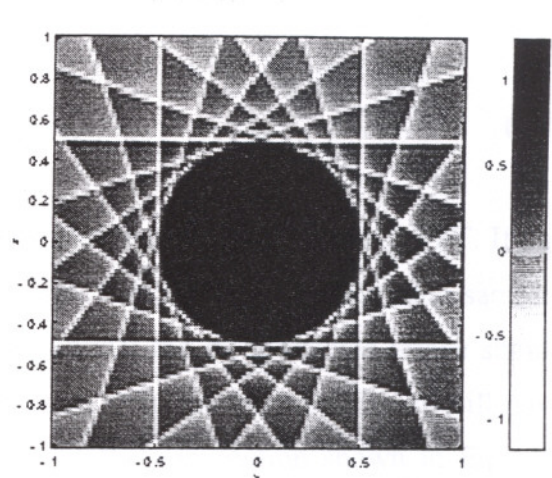


Figure 6.8 Filtered Backprojection from a sinogram with 10 angular samples ( $T=10$ )

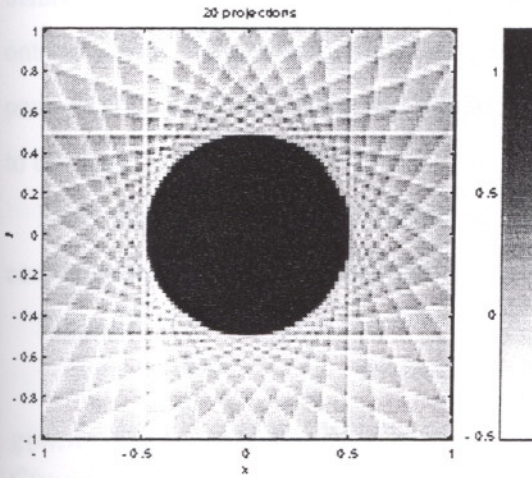


Figure 6.9 Filtered Backprojection from a sinogram with 20 angular samples ( $T=20$ )

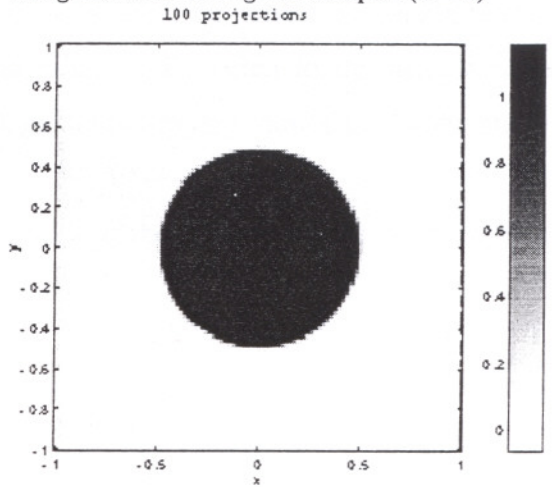


Figure 6.10 Filtered Backprojection from a sinogram with 100 angular samples ( $T=100$ )

### 6.1.3. Implementation of Filtering after Backprojection

Implementation of Filtering after Backprojection requires a discrete implementation of the backprojection operator, as shown in Section 6.1.2. After the backprojection the matrix  $g(m,n)$  must be high pass filtered. The implementation resembles the one shown in Subsection 6.1.1.1, but 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. (Press et al. 1992), or by using a one dimensional FFT on first all of the rows and then all the columns of the image, which is possible because the discrete spectrum can be written

$$G(u, v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m, n) e^{-j2\pi(mu/M + nv/N)} \quad (6.20)$$

$$= \sum_{m=0}^{M-1} \left[ \sum_{n=0}^{N-1} g(m, n) e^{-j2\pi nu/M} \right] e^{-j2\pi mv/N} \quad (6.21)$$

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.4.46. 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 complex conjugate symmetry for a real valued signal, as shown in

Figure 6.11.

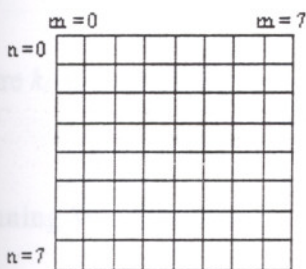
$$g(m, n) = g(m + M, n) = g(m, n + N) \quad (6.22)$$

$$G(u, v) = G(m + M, n) = G(m, n + N) \quad (6.23)$$

$$g(m, n) = g(m, n)^* \Rightarrow G(u, v) = G(-u, -v)^* = G(M - u, N - v)^* \quad (6.24)$$

The symmetry of the spectrum can be exploited for reducing the memory requirements due to a rather odd storage strategy needed.

$g(m, n)$  real



$G(u, v)$  complex

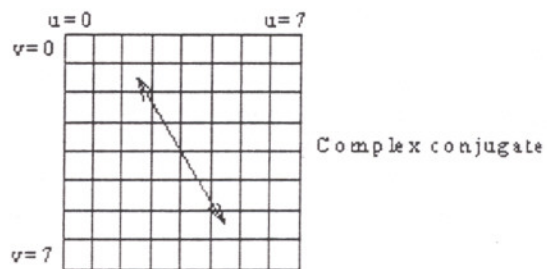


Figure 6.11 A real valued image gives a spectrum with complex conjugate pairs.

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  $\sqrt{k_x^2 + k_y^2}$ .

$$k_x \rightarrow \frac{u}{M\Delta x} \quad \text{and} \quad k_y \rightarrow \frac{v}{N\Delta x} \quad (6.25)$$

$$\sqrt{k_x^2 + k_y^2} \rightarrow \sqrt{\left(\frac{u}{M\Delta x}\right)^2 + \left(\frac{v}{N\Delta x}\right)^2} \quad (6.26)$$

If addressing the negative frequencies  $k_x \neq 0$ ,  $u$  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  $k_l$

$$H(k_x, k_y) = \begin{cases} \sqrt{k_x^2 + k_y^2} & \text{if } \sqrt{k_x^2 + k_y^2} < k_l \\ 0 & \text{else} \end{cases} \quad (6.27)$$

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



$$H(k_x, k_y) = \begin{cases} \frac{1}{2} \sqrt{k_x^2 + k_y^2} \left( 1 + \cos \left( \pi \frac{\sqrt{k_x^2 + k_y^2}}{k_l} \right) \right) & \text{if } \sqrt{k_x^2 + k_y^2} < k_l \\ 0 & \text{else} \end{cases} \quad (6.28)$$

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.

#### **6.1.4. Implementation of the Fourier Slice Theorem**

The basis of the Fourier Slice Theorem is given in Section 4.3. In the implementation, the discrete spectrum of the sinogram is calculated for each of the angular samples like it was shown in Sub-section 6.1.1.1. Note that here the phase is important, hence the shifting shown in Figure 6.1 must be considered. This spectrum is considered as polar samples, and must be mapped onto a quadratic frequency grid, as shown in Figure 6.12. 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.

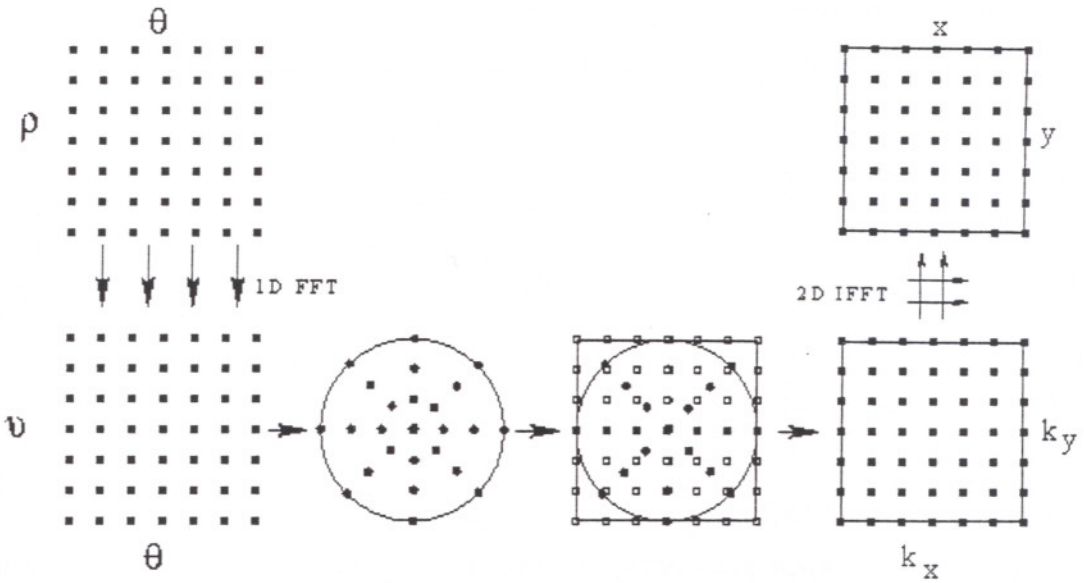


Figure 6.12 Following from upper left. At first the discrete spectrum is computed. The spectrum is then considered to be polar, and mapped onto a quadratic grid in the frequency domain using two-dimensional interpolation. Finally, the 2D spectrum is inverted into the reconstructed image using 2D inverse FFT.

The complexity of this implementation, where the FFT is used to computing the spectra is given by

$$O_{Forward1DFFTofSinogram} = O(TR \log R) \tag{6.29}$$

$$O_{Inverse2DFFTofSpectrum} = O(M^2 \log M) \tag{6.30}$$

$$O_{FourierSliceTheorem} \approx O(M^2 \log M) \tag{6.31}$$

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. 6.31 indicates that the implementation of the Fourier Slice Theorem is of a lower order than Filtered Backprojection and Filtering after Backprojection.

Next some of the problems with this implementation are discussed. If omitting the important shifting problems, illustrated in Figure 6.1, the polar and the quadratic spectrum can match in three different ways as shown in Figure 6.13. 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  $\Delta x = \Delta y$ .

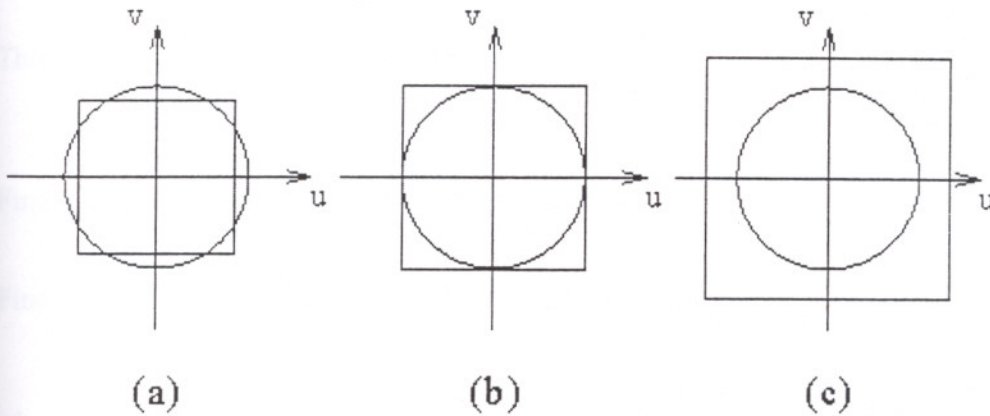


Figure 6.13 Three ways that the polar and the quadratic spectrum can match.

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

$$\max |k_x| = \frac{1}{2\Delta x} < v_{\max} = \frac{1}{2\Delta \rho} \Leftrightarrow \Delta x > \Delta \rho \quad (6.32)$$

In the second case (b) the entire 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. Figure 6.14 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.

$$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = \tilde{\nu} \begin{pmatrix} \cos \tilde{\theta} \\ \sin \tilde{\theta} \end{pmatrix} \quad (6.33)$$

Then the four nearest neighbours are found.

$$\text{Find the integer } t = \left\lfloor \frac{\tilde{\theta}}{\Delta\theta} \right\rfloor \Rightarrow \theta_t \leq \tilde{\theta} < \theta_{t+1} \text{ and set } \omega_\theta = \frac{\tilde{\theta} - \theta_t}{\Delta\theta} \quad (6.34)$$

$$\text{Find the integer } r = \left\lfloor \frac{\tilde{\nu}}{\Delta\nu} \right\rfloor \Rightarrow \nu_r \leq \tilde{\nu} < \nu_{r+1} \text{ and set } \omega_\nu = \frac{\tilde{\nu} - \nu_r}{\Delta\nu} = \Delta\rho(\tilde{\nu} - \nu_r) \quad (6.35)$$

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

$$\begin{aligned} G(k_x, k_y) = & (1 - \omega_\theta) \left( (1 - \omega_\nu) G(\nu_r, \theta_t) + \omega_\nu G(\nu_{r+1}, \theta_t) \right) \\ & + \omega_\theta \left( (1 - \omega_\nu) G(\nu_r, \theta_{t+1}) + \omega_\nu G(\nu_{r+1}, \theta_{t+1}) \right) \end{aligned} \quad (6.36)$$

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  $\nu$  the distance between samples is  $\Delta\nu = \frac{1}{R\Delta\rho}$ , cf. Eq. 6.2. 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.6.32, one criterion is that the sampling interval in the radial parameter  $\nu$  must match the one in the rectangular grid, which is a reasonable tradeoff, i.e.,

$$\Delta k_x = \Delta k_y = \frac{1}{M\Delta x} = \Delta\nu = \frac{1}{R\Delta\rho} \Rightarrow M = R \quad (6.37)$$

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 (Carlson et al. 1995) illustrated in Figure 6.15 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.

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.

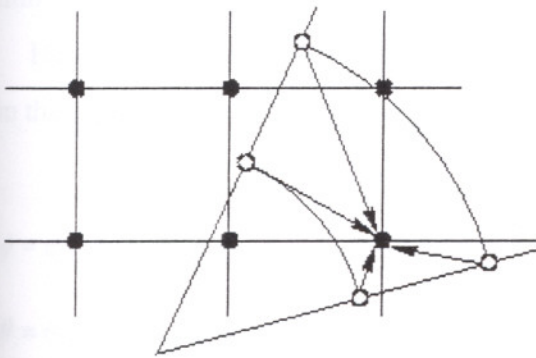


Figure 6.14 Strategy 1: A weighted sum of the four closest polar samples is used to estimate the spectrum on the quadratic grid.

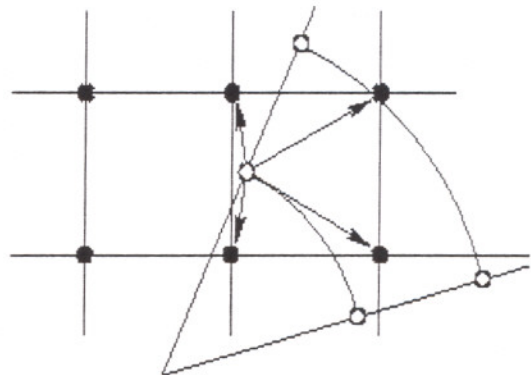


Figure 6.15 Strategy 2: Any of the samples on the polar grid are distributed with certain weights to the quadratic grid.

## 6.2. Implementation of 3D Direct Reconstruction Algorithms

Going from 2D to 3D reconstruction one major difference is the size of the inversion problem. A practical reconstruction program (either iterative or direct) will include several steps of filtering, Radon transform, and the backprojection is the most time consuming.

The program will require a uniformly sampled sinogram as it will shown in Eq.6.38, based on  $\Omega_{\psi}$ - geometry and the reconstructed volumes will also be sample uniformly.

The value of  $\Psi$  is left to the user. The package also includes the program names "3D\_RadonAna" which has been developed for generating a volume and the corresponding four dimensional sinogram from a set of scaled, rotated, and translated primitives. This program is based on properties shown in Appendix A. The usage of the program is shown in Appendix B.

### 6.2.1. 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.5.56, or the backprojected volume, cf. Eq.5.44. The implementation is a straight forward generalization of the 2D reconstruction implementation discussed in Chapter 4, 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  $\Omega_\psi$ , and the parameters in the parameter domain are sampled uniformly

$$\begin{aligned}
 \phi &= \phi_p = -\psi + p \frac{2\psi}{P-1}, & p &= 0, 1, \dots, P-1 \\
 \theta &= \theta_t = t \frac{\pi}{T}, & t &= 0, 1, \dots, T-1 \\
 u &= u_i = u_{\min} + i\Delta u, & i &= 0, 1, \dots, I-1 \\
 v &= v_j = v_{\min} + j\Delta v, & j &= 0, 1, \dots, J-1
 \end{aligned} \tag{6.38}$$

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

$$\begin{aligned}
 x &= x_k = x_{\min} + k\Delta x, & k &= 0, 1, \dots, K-1 \\
 y &= y_l = y_{\min} + l\Delta y, & l &= 0, 1, \dots, L-1 \\
 z &= z_m = z_{\min} + m\Delta z, & m &= 0, 1, \dots, M-1
 \end{aligned} \tag{6.39}$$

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 reprojection technique.

### 6.2.1.1. Implementation of the Backprojection Operator

The backprojection operator can be approximated by a sum, here shown using a nearest neighbour approximation

$$\hat{g}(r_{k,l,m}) = \int_{\theta=0}^{\pi} \int_{\phi=-\psi}^{\psi} \check{g}(\theta, \phi, u = r \cdot \alpha, v = r \cdot \beta) \cos \phi d\phi d\theta \quad (6.40)$$

$$\approx \Delta\theta\Delta\phi \sum_{p=0}^{P-1} \cos \phi_p \sum_{t=0}^{T-1} \check{g}\left(t, p, \left[ \frac{r_{k,l,m} \cdot \alpha_{p,t} - u_{\min}}{\Delta u} \right], \left[ \frac{r_{k,l,m} \cdot \beta_{p,t} - v_{\min}}{\Delta v} \right] \right) \quad (6.41)$$

In Algorithm 6.4 the implementation of the backprojection operator is shown, when using nearest neighbour interpolation.

It has been shown that all of the reconstruction methods are heavily based on projections of the directional vectors  $\alpha$ ,  $\beta$ , and  $\tau$  onto the volume coordinates  $r$ . In order to speed up the reconstruction algorithms, the vectors  $\alpha = (\alpha_x, \alpha_y, \alpha_z)^T$ ,  $\beta = (\beta_x, \beta_y, \beta_z)^T$ , and  $\tau = (\tau_x, \tau_y, \tau_z)^T$ , should be computed once, for all values of  $(p, t)$ . This can be done rather efficiently and will only require six matrices. In Algorithm 6.4 the arrays are assumed given, e.g.,  $\alpha_x(\tau; p) = \alpha_x(\theta_t, \phi_p)$ . For further optimization, the double loop of all possible angles  $(\theta, \phi)$  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.

For each value of  $(t, p)$ , Eq.6.40 requires that the  $u$ - and the  $v$ - values lies within the bounds shown in Eq.6.38. 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 For the expansion, it can easily be shown that

$$\left\lfloor \frac{-\max |r_{k,l,m}| - u_{\min}}{\Delta u} \right\rfloor \leq i \leq \left\lceil \frac{\max |r_{k,l,m}| - u_{\min}}{\Delta u} \right\rceil \quad (6.42)$$

$$\left\lfloor \frac{-\max |r_{k,l,m}| - v_{\min}}{\Delta v} \right\rfloor \leq i \leq \left\lceil \frac{\max |r_{k,l,m}| - v_{\min}}{\Delta v} \right\rceil \quad (6.43)$$

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.6.40. In principle this is viable, but might require more computations compared to the reduction being offered by this alteration.

The number of loops shown in Algorithm 6.4 illustrates that the complexity of backprojection is high, namely

$$O_{3D\text{Backprojection}} = O(KLMNPT) \quad (6.44)$$

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.

---

ALGORITHM 6.4 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) //Calculate u value
          i=round((u-u_min)/Delta_u) //Calculate i index
          If 0<=i<l
            v=x(k)*beta_x(t,p)+y(l)*beta_y(t,p)+
              z(m)*beta_z(t,p) //Calculate v value
            j=round((v-v_min)/Delta_v)
            If 0<=j<J
              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
End

```

---



### 6.2.1.2. Implementation of the Radon Transform Operator

An implementation of the Radon transform defined in Eq.5.5 will now be shown. This operator is needed for the Kinahan & Rogers reprojection methods and for any of the iterative methods presented in Chapter 4.

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\tau + r_0) ds, \quad \text{where } r_0 = (x_0, y_0, z_0)^T = u\alpha + v\beta \quad (6.45)$$

$$= \int_{-\infty}^{\infty} g(s\tau_x + x_0, s\tau_y + y_0, s\tau_z + z_0) ds \quad (6.46)$$

In order to avoid the problems with too high slopes, the line integral is projected onto the axis with maximum absolute component of the directional vector relative to its sampling interval. Assume that the projection is made onto the x- axis

$$\frac{\tau_x}{\Delta x} > \frac{\tau_y}{\Delta y} \quad (6.47)$$

Then the 3D Radon transform can be written

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} \frac{1}{|\tau_x|} g(x, xy^{(x)} + y_0^{(x)}, xz^{(x)} + z_0^{(x)}) dz \quad (6.48)$$

where

$$y^{(x)} = \frac{\tau_y}{\tau_x} \quad \text{and} \quad y_0^{(x)} = y_0 - y^{(x)}x_0 \quad (6.49)$$

$$z^{(x)} = \frac{\tau_z}{\tau_x} \quad \text{and} \quad z_0^{(x)} = z_0 - z^{(x)}x_0 \quad (6.50)$$

where the exponent  $(x)$  merely means with respect to  $x$ .

A simple discretization, which will require a few calculations each time, can now be derived from Eqs.6.39, 6.46, 6.49, and 6.50.

$$\check{g}(\theta, \phi, u, v) \approx \Delta x \sum_{k=0}^K g(k, \left[ \frac{x_k y^{(x)} + y_0^{(x)} - y_{\min}}{\Delta y}, \left[ \frac{x_k z^{(x)} + z_0^{(x)} - x_{\min}}{\Delta z} \right] \right]) \quad (6.51)$$

$$= \Delta x \sum_{k=0}^K g(k, [a_y^{(x)} k + b_y^{(x)}], [a_z^{(x)} k + b_z^{(x)}]) \quad (6.52)$$

where  $a_y^{(x)} = \frac{\tau_y}{\tau_x} \frac{\Delta x}{\Delta y}$  and  $b_y^{(x)} = \frac{\tau_y}{\tau_x} \frac{(x_{\min} - x_0)}{\Delta y} + \frac{y_0 - y_{\min}}{\Delta y}$  (6.53)

$$a_z^{(x)} = \frac{\tau_z}{\tau_x} \frac{\Delta x}{\Delta z} \quad \text{and} \quad b_z^{(x)} = \frac{\tau_z}{\tau_x} \frac{(x_{\min} - x_0)}{\Delta z} + \frac{z_0 - z_{\min}}{\Delta z} \quad (6.54)$$

With respect to the high slope problem, Eq.6.47 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.6.52, will not lead to a step in  $l$  or  $m$  greater than 1.

The discrete implementation of Radon transform is also quite demanding, due to the complexity

$$O_{3DRadonTransform} = O(PTIJK) \quad (6.55)$$

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  $\tau$  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 6.5 the implementation of the discrete 3D Radon transform is shown using a nearest neighbour approximation. The algorithm only shows the case where Eq.6.47 is fulfilled. The algorithm uses the 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 6.5 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.6.47 is fulfilled
        Set a_y,b_y,a_z,b_z from Eqs.6.52,6.53
        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
```

### 6.3. Examples Using Direct Reconstruction Algorithms

Here, reconstruction of a phantom is examined. In Figure 6.1 is shown the noise free sinogram corresponding to the image shown Figure 6.17

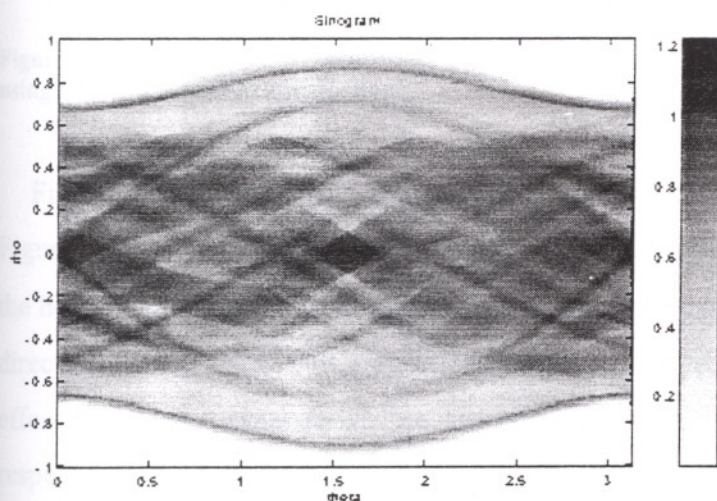


Figure 6.16 Sinogram of the phantom

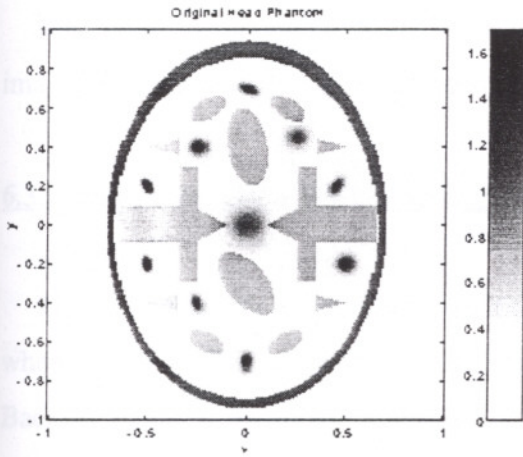


Figure 6.17 The original head phantom

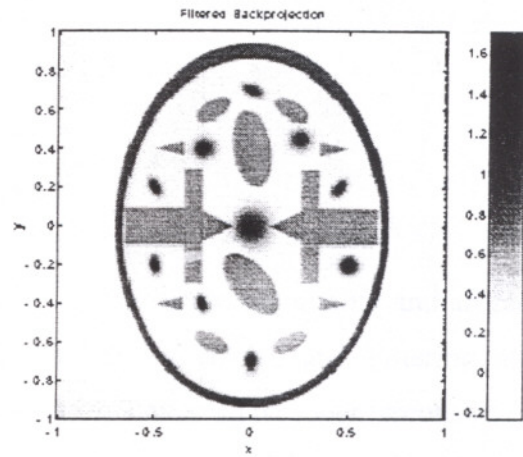


Figure 6.18 The reconstructed head phantom using filtered backprojection

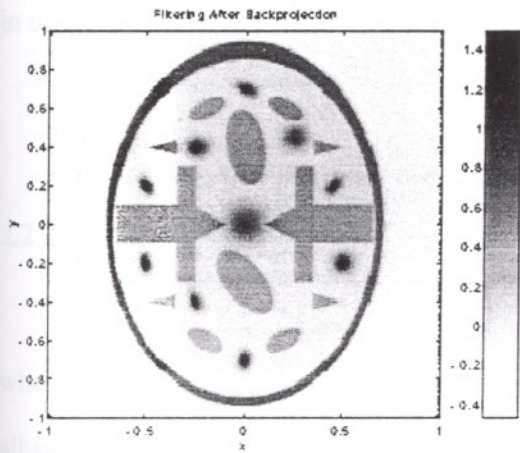


Figure 6.19 The reconstructed head phantom using filtering after backprojection

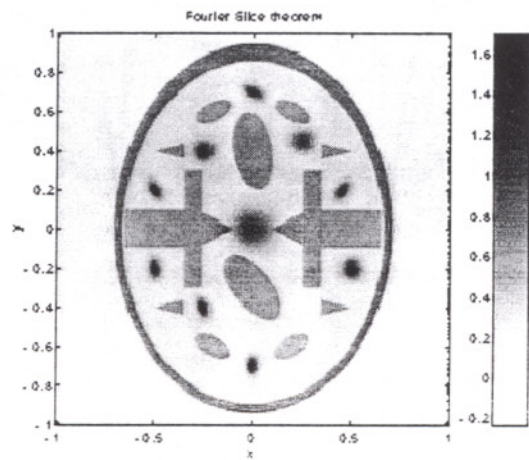


Figure 6.20 The reconstructed head phantom using Fourier slice theorem

First, Filtered Backprojection has been used to reconstruct the image as shown in Figure 6.18. 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  $\rho$ -direction. When using Filtering after backprojection, as shown in Figure 6.19, the ring effect has been 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 Figure 6.20. The reconstructed image appears to be comparable to the two backprojection methods, but more "texture" can be found in the area outside of 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.

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.

### 6.3.1. Reconstruction With Varying Image Size

The next example addresses qualification of the reconstruction quality and artifacts when using a Fourier Slice algorithm, Filtered Backprojection, and Filtering after Backprojection. From the sinogram in Figure 6.16, image have been reconstructed with varying 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 in Figure 6.17.

A modified  $L^2$  - measure of misfit is given in Eq.6.56.

$$L2 = \sqrt{\frac{\sum_{m,n} (g_{m,n} - g_{m,n}^{ref} - \bar{g} + \bar{g}^{ref})^2}{\sum_{m,n} (g_{m,n}^{ref} - \bar{g}^{ref})^2}} \quad (6.56)$$

where  $g_{m,n}^{ref}$  is the reference image (the original) and the bars indicate the average over all samples.

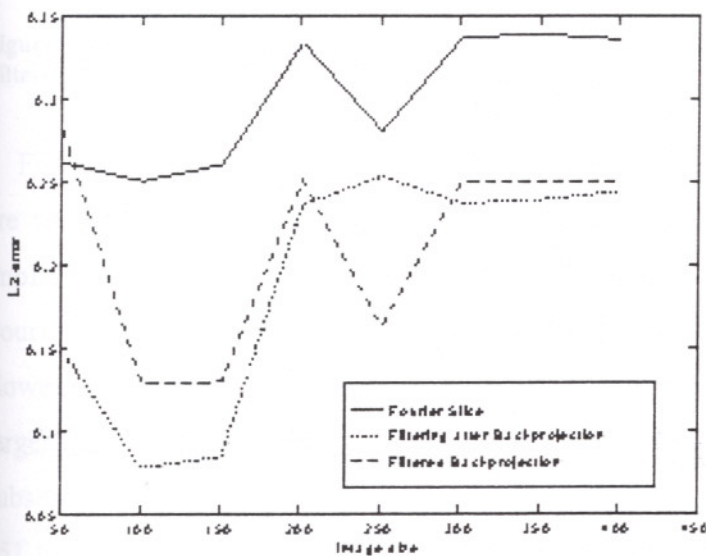


Figure 6.21  $L_2$  error as a function of the reconstructed image size  $M$  for Filtered Backprojection, Filtering after Backprojection, and a Fourier slice implementation.

Figure 6.21 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. For Filtered Backprojection, Figure 6.22 shows that the error has several reasons. It is obvious that the step edges in the image are not reconstructed perfectly, due to the use of 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 Figure 6.18 will also add the total error.

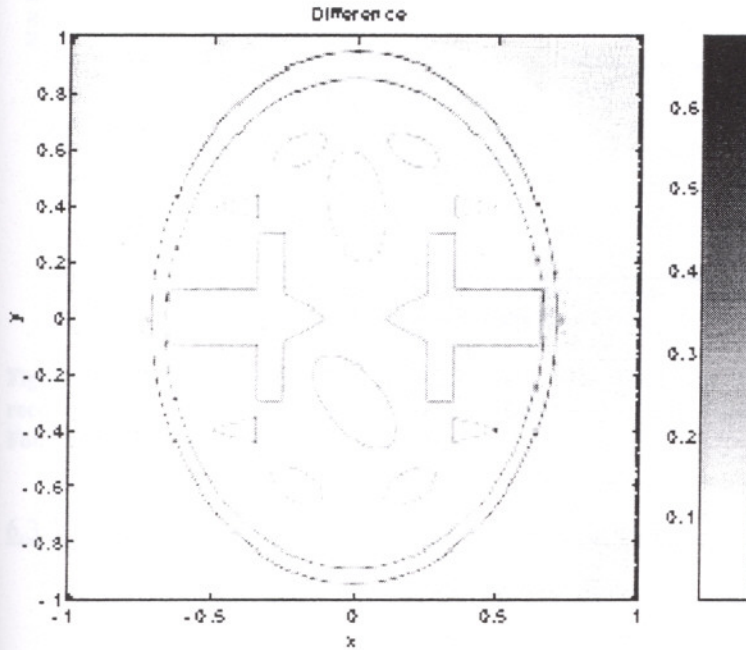


Figure 6.22 The absolute error between the original image and the reconstructed image using Filtered Backprojection.

For a Pentium 120 MHz (Linux system) the time needed to reconstruct the images are shown in Figure 6.23. It is clear 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  $256 \times 256$  image is generated in all three cases (151, 201, 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.

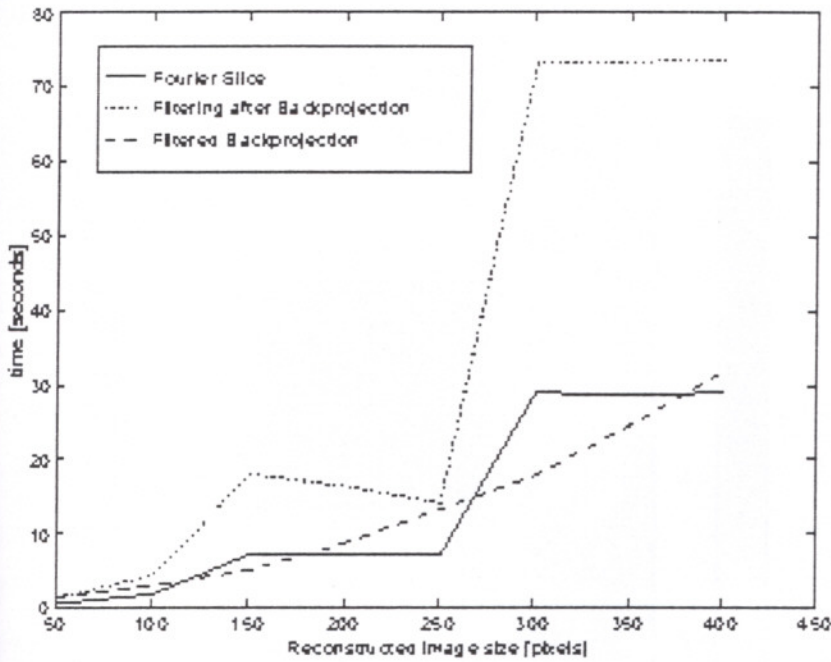


Figure 6.23 Time usage for reconstructing the sinogram in seconds as a function of the reconstructed image size  $M$  for Filtered Backprojection, Filtering after Backprojection, and the Fourier Slice Theorem implementation.

### 6.3.2. Reconstruction into an Oversampled Image

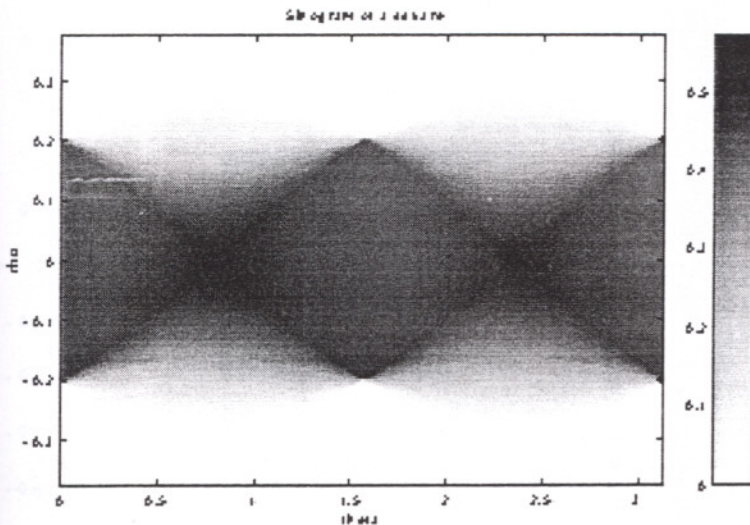


Figure 6.24 Sinogram of a square with  $R=251$ ,  $T=200$ , and  $\Delta\rho=0.0025$ .

In order to illustrate the problems with the Fourier Slice Theorem if  $\Delta x > \Delta\rho$ , (cf. Eq.6.32) 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 Figure 6.24. Then two reconstruction methods have been used, namely the Fourier Slice Theorem in Figure 6.25, and Filtering after Backprojection Figure 6.26.

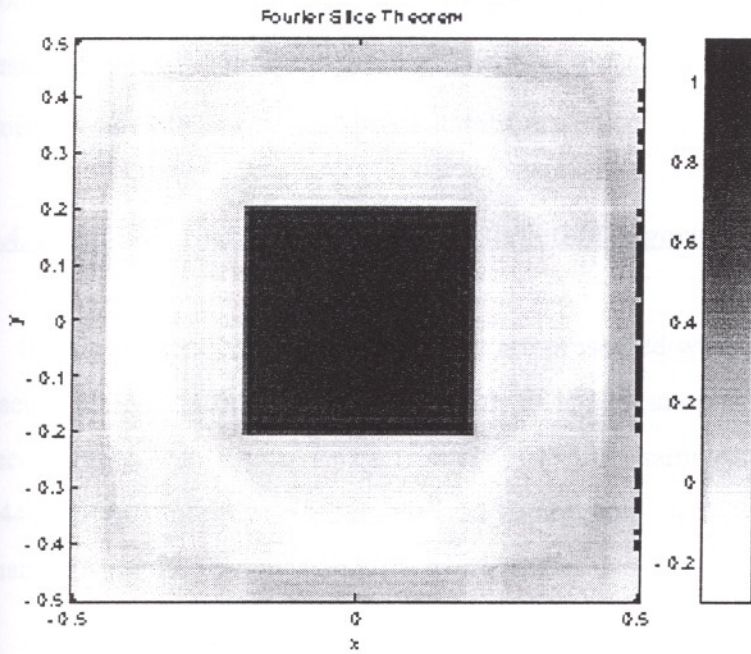


Figure 6.25 Reconstructed image with  $M=101$  and  $\Delta x=0.01$  using Fourier Slice Theorem.

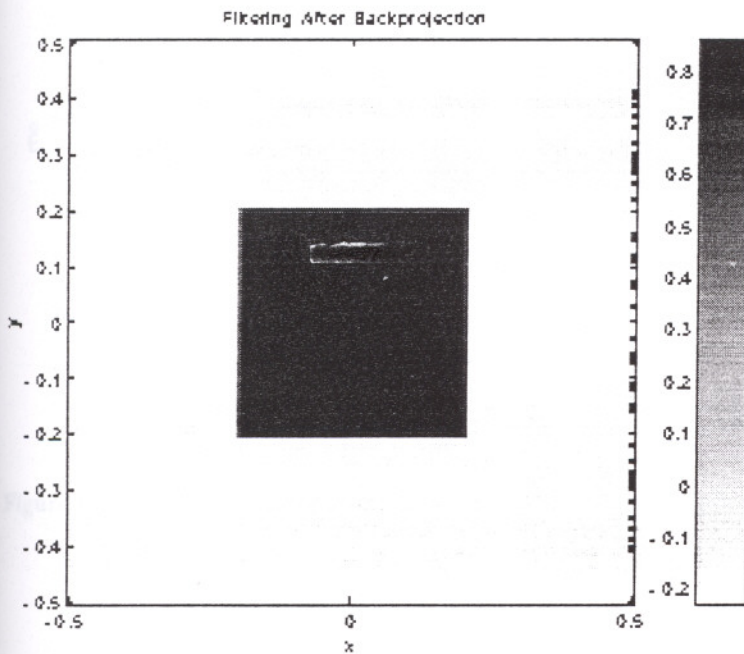


Figure 6.26 Reconstructed image with  $M=101$  and  $\Delta x=0.01$  using Filtering after Backprojection.



It can be seen that Filtering after Backprojection has no problems. Using Fourier Slice Theorem severe artifacts can be seen, due to aliasing of the spectrum.

As a result, Filtered Backprojection is normally precise, but somehow slow, with complexity of  $O(M^3)$ , where  $M$  is the number of samples in one direction of the resulting image. The Fourier Slice Theorem gives a fast algorithm,  $O(M^2 \log M)$ , but does not have the same numerical stability.

#### 6.4. Examples Using Iterative Reconstruction Algorithms

In this subsection a set of examples are presented when iterative methods have been used. In Figure 6.27 a sinogram with  $281 \times 336$  samples is shown. The sinogram is reconstructed into an image with  $301 \times 301$  samples. The system matrix has  $94416 \times 90601$  elements of which 0.23% are non-zero when modelling then system matrix using the Radon transform of a square.

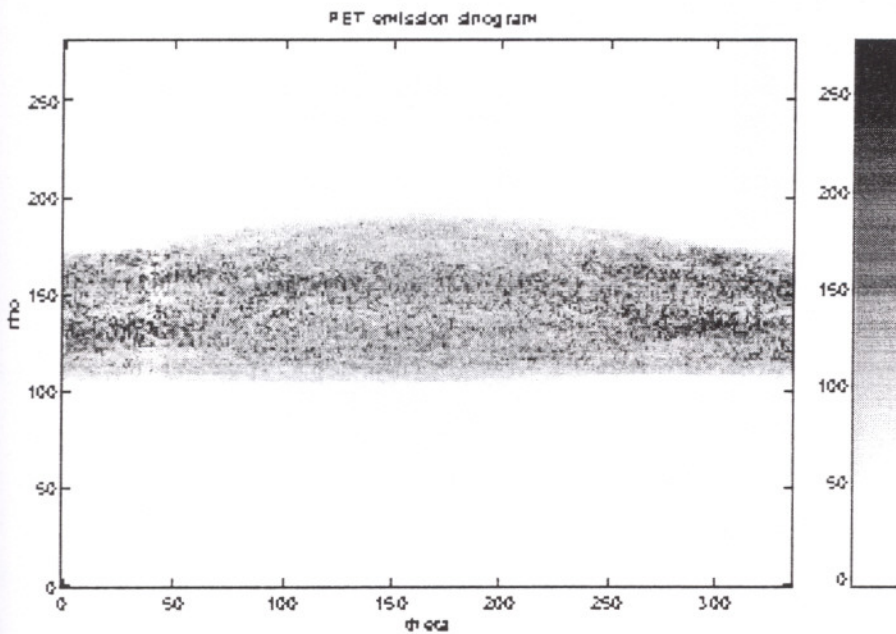


Figure 6.27 A slice of a sinogram of human brain.

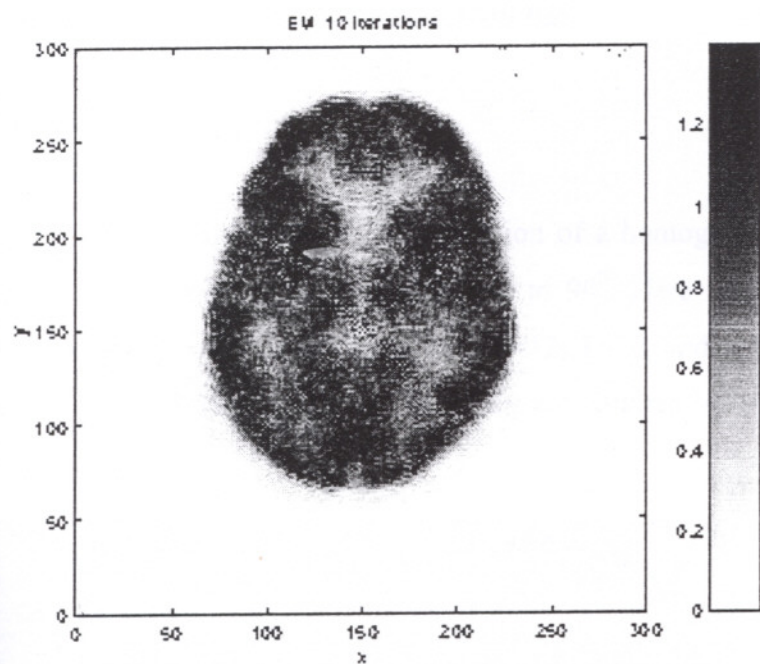


Figure 6.28 The reconstructed image after 10 iterations of EM.

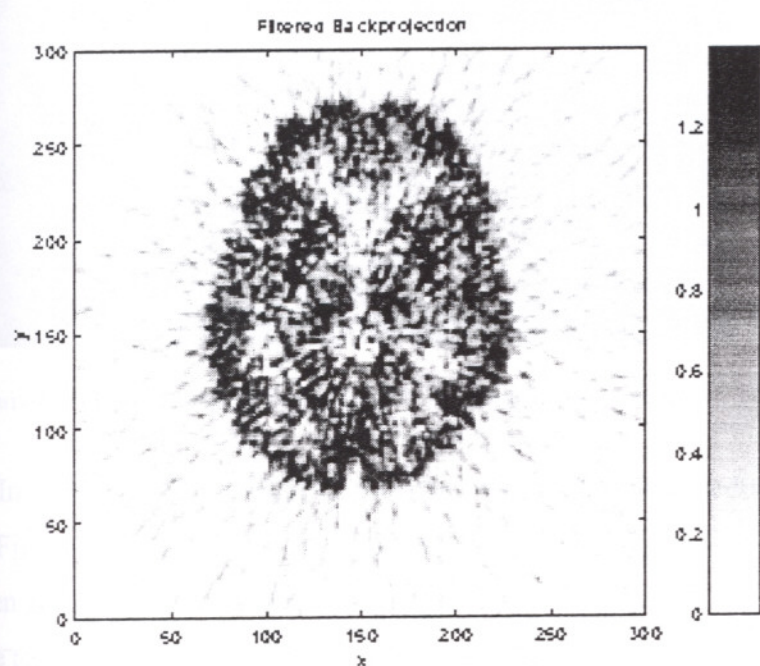


Figure 6.29 The reconstructed image using Filtered Backprojection with a ramp filter.

## 6.5. Examples of 3D Reconstructed Volumes

### 6.5.1. Reconstruction of a Ball

The first example concern reconstruction of a homogeneous ball, cf. A.2.1. In this case the axial limiting angle  $\Psi$  was set to  $90^0$ . Here the reconstructed volume has  $51 \times 51 \times 51$  voxels and the sinogram uses  $P=12$ ,  $T=10$ , and  $51 \times 51$  samples in each of the  $(u,v)$  planes. Both the volume and the sinogram requires MBytes of memory.

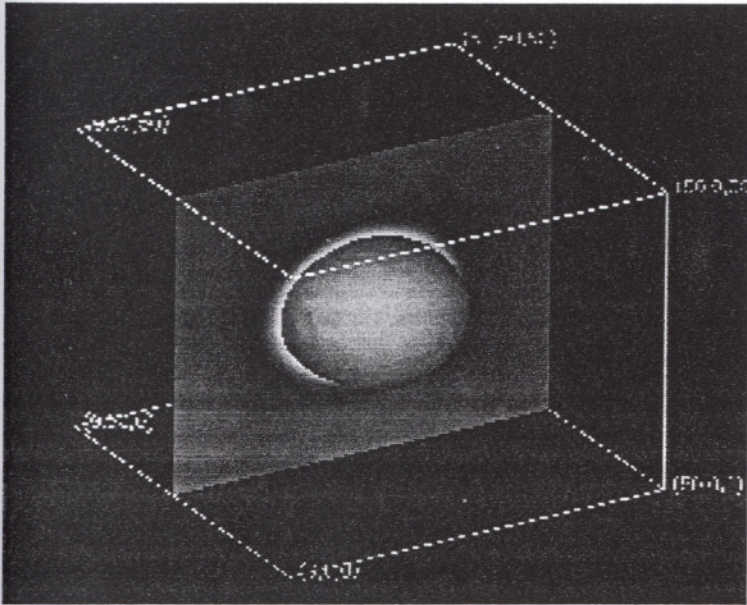


Figure 6.30 Reconstructed ball using 3D Filtered Backprojection.

In Figure 6.30, 3D Filtered Backprojection has been used to reconstruct the ball, and in Figure 6.31 3D Filtering after Backprojection. The figures indicate that the ball has been recovered well, though the edges of the ball are blurred. This could be expected, due to low number of samples in both domains.

Next the iterative methods ART and EM have been used to reconstruct the same ball. For ART some artifacts were found, while EM is successful. Here the figures has been omitted due to the high similarity shown above.

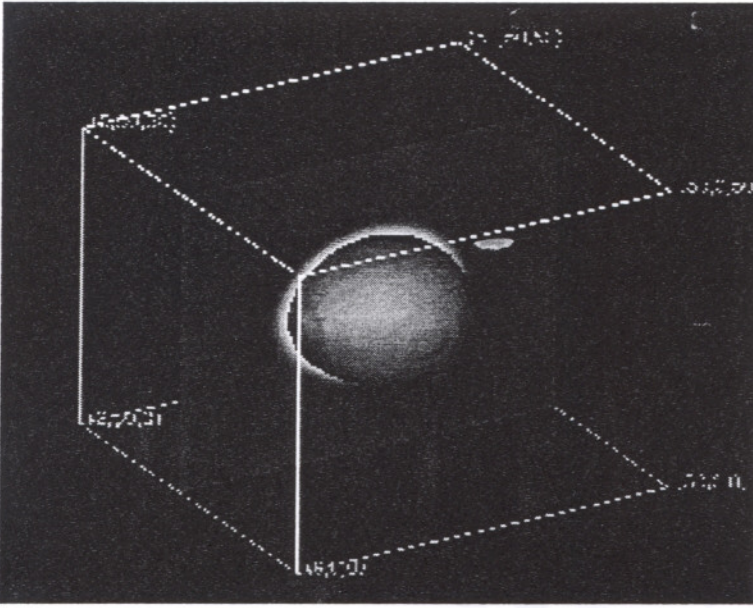


Figure 6.31 Reconstructed ball using 3D Filtering after Backprojection

### 6.5.2. Reconstruction of the Mickey Phantom

In this example, the axial acceptance has been reduced to merely  $\Psi=90^\circ$ . The volume has centered around  $(0,0,0)$  with  $\Delta x=\Delta y=\Delta z=0.5$  and  $K=L=M=71$ , which requires 1.4MBytes 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.

Using 3D Filtering after Backprojection, the reconstructed volume can be visualized with Polyr and Geomview. Figure 6.32 shows the result which looks like original Mickey Mouse. In Figure 6.33 and Figure 6.34 the reconstructed central  $(x,y)$  and  $(y,z)$  planes are shown. The figures use individual color scale according to the minimal and maximal value. This volume has the value 0.2 inside the "skull" and a small "tumor"-ball with radius fo 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 .The sinogram was backprojectde onto a  $128*128*128$  volume in order to use radix-2 FFT filtering.

Next Filtered Backprojection used to reconstruct the same phantom. Figure 6.35 and Figure 6.36 show the central  $(x,y)$  and  $(y,z)$  planes for Filtered Backprojection.

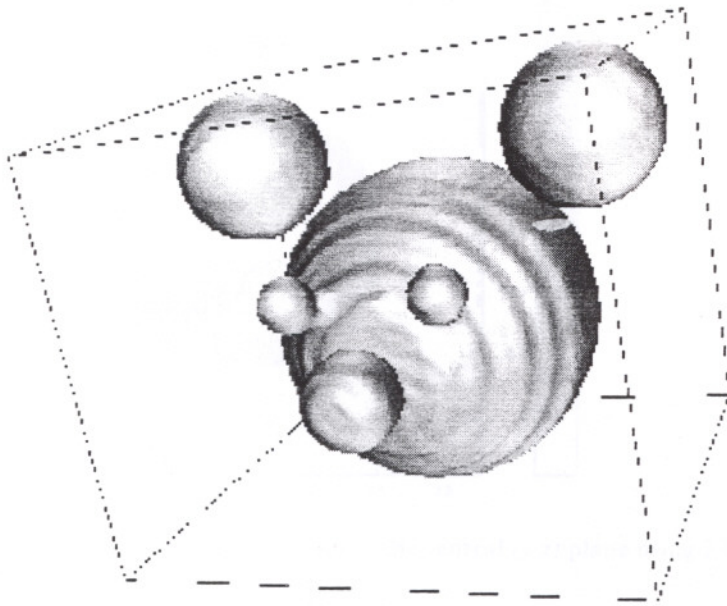


Figure 6.32 The Mickey phantom

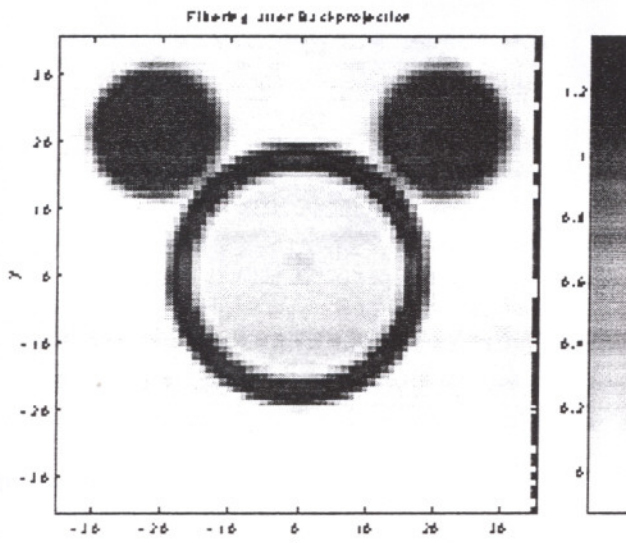


Figure 6.33 Reconstructed phantom in the central (x,y) plane using Filtered Backprojection.

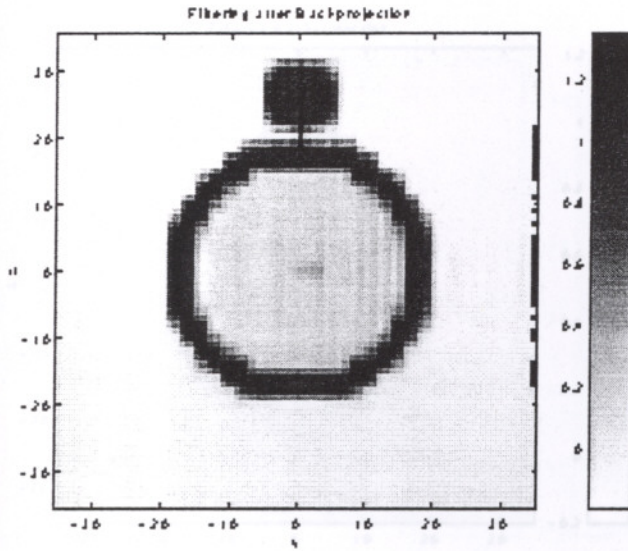


Figure 6.34 Reconstructed phantom in the central (y,z) plane using Filtering after Backprojection.

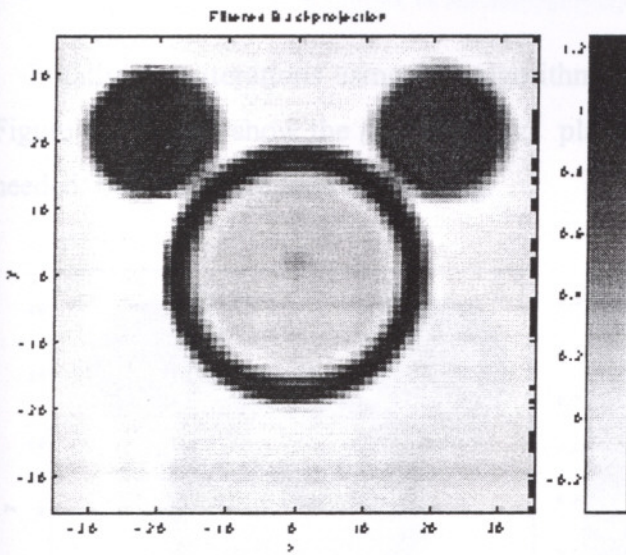


Figure 6.35 Reconstructed phantom in the central (x,y) plane using Filtered Backprojection.

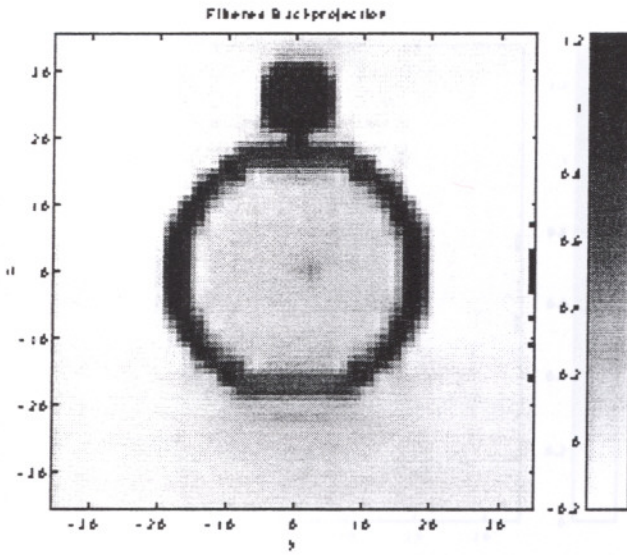


Figure 6.36 Reconstructed phantom in the central  $(y,z)$  plane using Filtered Backprojection.

Finally, five iterations using EM algorithm has been demonstrated. Figure 6.37 and Figure 6.38 again show the  $(x,y)$  and  $(y,z)$  planes. It is obvious that more iterations are needed, but it is more time consuming.

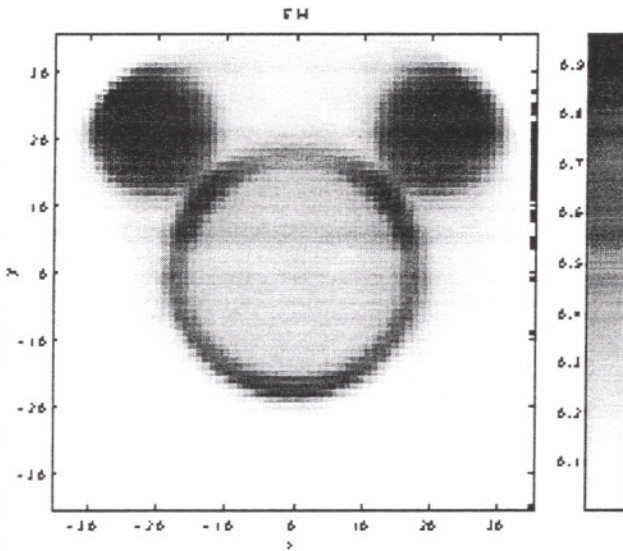


Figure 6.37 Reconstructed phantom in the central  $(x,y)$  plane using five- iterations of EM.

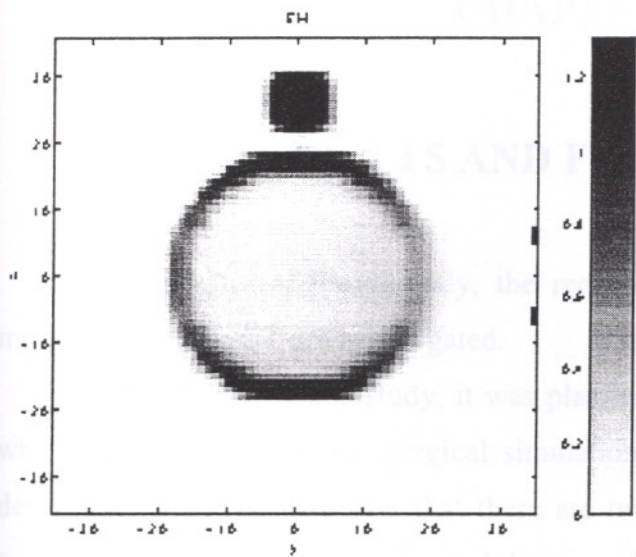


Figure 6.38 Reconstructed phantom in the central  $(y,z)$  plane using five iterations of EM.



## CHAPTER 7

### RESULTS AND FUTURE WORK

In this work, most prominently, the reconstruction methods of medical imaging modalities such as CT are investigated.

At the beginning of this study, it was planned to have a complete software system, which performs craniofacial surgical simulation and also planning. While going into details of the subject, we saw that there are two different steps. The first one is the reconstruction of medical images and the second and complementary one is to process these reconstructed images in cooperation with surgeons, who are specialist in craniofacial surgery.

As was stated before, this thesis although named as “Craniofacial Computer- Aided Surgical Planning and Simulation”, we have worked on just reconstruction step, and a software package, which consists of implementations of several reconstruction techniques, was written.

The next step for who plan to work on this subject will be to understand the functions of structures in the head and face, and the purposes of craniofacial surgery, which is especially important for planning procedure.

Identification of bones, the interconnections of these bones and the influences on each other would be included in simulation phase. The definitions, techniques are briefly described in Chapter 3 for both simulation and planning phases.

Medical image registration may also be viewed as a further work, which provides a complete image that consists of both hard and soft tissues. In this case, it would be possible to show the to the patient how he/ she would look like after the operation.

## REFERENCES

- Bellon P., Lanzavecchia S., "A Direct Fourier Method (DFM) for X-ray Tomographic Reconstructions and the Accurate Simulations of Sinograms", *Int. Journal of Bio-Medical Computing*, **Vol:38**, (1995).
- Brantner S., Young R.C.D., Budgett D., Chatwin C.R., "High-Speed Tomographic Reconstruction Employing Fourier Methods", *Real Time Imaging*, **Vol:3**, (1997).
- Bro-Nielsen M., "Modelling Elasticity in Solids Active Cubes- Applications to Simulated Operations", *Proc. Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, (1995).
- Bro-Nielsen M., Cotin S., "Soft Tissue Modelling in Surgery Simulation for Prediction of Results of Craniofacial Operations & Steps Towards Virtual Reality Training Systems", *Proc. 3<sup>rd</sup> Int. Workshop on Rapid Prototyping in Medicine & Computer-Assisted Surgery*, (1995).
- Bro-Nielsen M., *Medical Image Registration and Surgery Simulation*, Technical University of Denmark, Department of Mathematical Modelling, (1997), **(Ph.D. Thesis)**
- Caponetti L., Fanelli A.M., "Computer Aided Simulation for Bone Surgery", *IEEE Computer Graphics & Applications*, **November**, (1993).
- Carlson J. 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.
- Carson R.E., Lange K., "The EM Parametric Image Reconstruction Algorithm", *Journal of the American Statistical Association*, **Vol: 389**, (1985).

Censor Y., "Finite Series- Expansion Reconstruction Methods", *Proc. Of IEEE*, **Vol: 71**, (1993).

Cho Z.H., Jones J.P., Singh M., *Foundations of Medical Imaging*, John Wiley & Sons, Inc, (1993).

Clack R., Townsend D., Defrise M., "An Algorithm for Three- Dimensional Reconstruction Incorporating Cross- Plane Rays", *IEEE Trans. Med. Imag.*, **Vol: 8, No: 1**, (1989).

Clack R.. "Towards a Complete Description of Three- Dimensional Filtered Backprojection.", *Physics in Medicine & Biology*, **Vol: 37, No:3**, (1992).

Colsher J.G., "Fully Three Dimensional Positron Emission Tomography", *Physics in Medicine and Biology*, **Vol:25, No:1**, (1980).

Deans S.R., *The Radon Transform and some of its Applications*, Krieger Publishing Company, 2<sup>nd</sup> ed., (1993).

Deans S.R., *The Radon Transform and some of its Applications*, Wiley – Interscience, (1983).

Delingette H., "Simplex Meshes: A General Representation for 3D Shape Reconstruction", *Proc. Computer Vision and Pattern Recognition (CVPR '94)*, (1994).

Delingette H., Subsol G., Cotin S., Pignon J., "A Craniofacial Surgery Simulated Testbed", *INRIA Tech. Rep.*, **No:2199**, (1994).

Grangeat P., "Mathematical Framework of Cone Beam 3D Reconstruction via the First Derivative of the Radon Transform", *Mathematical Methods in Tomography, Lecture Notes in Mathematics*, **No.1497**, (1991).

- Hardy T.L., "Computerized Atlas for Functional Stereotaxis Robotics and Radiosurgery", *Visualization in Biomedical Computing*, Vol: 2359 of SPIE, (1994).
- Herman G.T., *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography*, (Academic Press, 1980).
- Herman G.T., Mayer L.B., "Algebraic Reconstruction Techniques can be Made Computationally Efficient", *IEEE Trans. Med. Imag.*, Vol: 12, No:3, (1993).
- Hildebolt C.F., Vannier M.W., Knapp R.H., "Validation Study of Skull Three-Dimensional Computerized Tomography Measurements", *Am J Phys. Anthropol.*, Vol:82, (1990).
- Jain A.K. *Fundamentals of Digital Image Processing*. Prentice Hall, 2<sup>nd</sup> ed., (1989).
- Kawata S., Nalcioglu O., "Constrained Iterative Reconstruction by the Conjugate Gradient Method", *IEEE Tran. Med. Imag.*, Vol: 4, No:2, (1985).
- Keeve E., Girod E., Girod B., "Computer- Aided Craniofacial Surgery", *Proc. Computer Assisted Radiology (CAR' 96)*, (1996).
- Keeve E., Girod S., Girod B., "Craniofacial Surgery Simulation", *Visualization in Biomedical Computing (VBC' 96)*, (1996).
- Keeve E., Girod S., Jauch J., Girod B., "Anatomy Based Modelling of Human Facial Tissue for Craniofacial Surgery Simulation", *Proc. Visualization' 95- Dynamics and Complexity*, (1995).
- Keeve E., Girod S., Pfeifle P., Girod B., "Anatomy Based Facial Tissue Modelling Using the Finite Element Method", *Visualization' 96*, (1996).

- Keeve E., Girod S., Pfeifle P., Girod B., "Interactive Craniofacial Surgery Planning by 3D Simulation and Visualization", *Proc. Int. Workshop on Rapid Prototyping in Medicine & Computer-Assisted Surgery*, (1995).
- Koch R.M., Gross M.H., Carls F.R., von Büren D.F., Fankhauser G., Parish Y.I.H., "Simulating Facial Surgery Using Finite Element Models", *SIGGRAPH'96*, (1996).
- Kok A., Slaney M., *Principles of Computerized Tomographic Imaging*, IEEE Press, (1987).
- Kudo H., Saito T., "Feasible Cone Beam Scanning Methods for Exact Reconstruction in Three Dimensional Tomography", *J. Opt. Soc. Am. A*, **Vol.7, No.12**, (Dec. 1990).
- Lee Y., Terzopoulos D., Waters K., "Constructing Physical-Based Facial Models of Individuals", *Proc. Graphics Interface '93*, (1993).
- Lee Y., Terzopoulos D., Waters K., "Realistic Modelling of Facial Animation", *Proc. SIGGRAPH'95*, (1995).
- Lewitt R.M., "Alternatives to Voxels for Image Representation in Iterative Reconstruction Algorithms", *Phys. Med. Biol.*, **Vol: 37, No: 2**, (1992).
- Lewitt R.M., "Reconstruction Algorithms: Transform Methods", *Proceedings of IEEE*, **Vol. 71, No. 3**, (March 1983).
- Lo L.J., Marsh J.L., Vannier M.W., and Patel V.V., "Craniofacial Computer Assisted Surgical Planning and Simulation", *Clin. Plast. Surg.* **Vol.21, No:4**, (1994).
- Magnusson M., *Linogram and Other Direct Fourier Methods for Tomographic Reconstruction*. Dept. of Electrical Eng, Linköping University, (1993), **Ph.D. Thesis**.
- Marsh J.L., Vannier M.W., *Comprehensive Care for Craniofacial Deformities*, CV Mosby, (St. Louis, 1985).

Matej S., Bajla I., " A High Speed Reconstruction from Projections Using Direct Fourier Method with Optimized Parameters- an Experimental Analysis", *IEEE Trans. Med. Imaging*, **Vol: 9**, (1990).

Mersereau R.M., "Direct Fourier Transform Techniques in 3D Image Reconstruction", *Comput. Biol. Med.*, **Vol: 6**, (1976).

Mersereau R.M., "Recovering Multidimensional Signals from Their Projections", *Computer Graphics and Image Processing 1*, (1973).

Nattarer F., *The Mathematics of Computerized Tomography*, John Wiley & Sons, 2<sup>nd</sup> edition, 1986.

Older J.K., Johns P.C., "Matrix Formulation of Computed Tomography Reconstruction", *Physics in Medicine and Biology*, **Vol: 38**, (1993).

Orlov S.S., "Theory of Three Dimensional Reconstruction. The Recovery Operator", *Sov. Phys. Crystallogr.*, **Vol: 20, No:4**, (1975).

Orlov S.S., "Theory of Three Dimensional Reconstruction.1.Conditions for a Complete Set of Projections.", *Sov. Phys. Crystallogr.*, **Vol: 20, No:3**, (1975).

Pieper S.D., *Computer Aided Plastic Surgery*, MIT, (1992), (Ph.D. thesis)

Pieper S.D., Rosen J., Zeltzer D., "Interactive Graphics for Plastic Surgery", *Computer Graphics*, **Vol: 26, No: 2**, (1992).

Press W.H., Teulovsky S.A., Vetterling W.T., Flannery B.P., *Numerical Recipes in C: The Art of Scientific Computing*, The Press Syndicate of the University of Cambridge, (1992).

Rabiner L.R., Schafer R.W., Rader C.M., "The chirp- z Transform Algorithm", *IEEE Trans. On Audio and Electroacoustics*, Vol: AU-17, No:2,(1969).

Satava R.M., "Virtual Reality Surgical Simulator", *Surgical Endoscopy*, Vol:7, (1990).

Satoh J., Ciyokura H., Kobayashi M., Fujino T., "Simulation of Surgical Operation Based on Solid Modelling", in T. L. Kunii Editor, "Visual computing, integrating computer graphics with computer vision", Tokyo (Japan), *Computer Graphics Society*, (1992).

Shepp L.A., Krystal J.B., "Computerized Tomography: The New Medical X- ray Technology", *Am. Math. Monthly*, Vol:85,(1978).

Shepp L.A., Logan B., "The Fourier Reconstruction of a Head Section", *IEEE Tran. Nucl. Sci.*, Vol: 21,(1974).

Shepp L.A., Vardi Y., Kaufman L., "A Statistical Model for PET", *Journal of American Statistical Association*, Vol: 80,(1985).

*Sinogram*. This term was introduced in a poster presentation at the 1975 meeting on Image Processing for 2-D and 3-D Reconstruction from Projections at Stanford, CA. The material appeared in a collection post- deadline papers for that meeting. PD5- Tomogram Construction by Photographic Techniques. Paul Edholm and Bertil Jacobson.

Stark H., Woods J., Paul I., Hingorani R., " Direct Fourier Reconstruction in Computer Tomography", *IEEE Accoust Speech Signal Process.*, Vol:2,(1981).

Stearns R.H., *Accelerated Image Reconstruction for a Cylindrical Positron Tomograph Using Fourier Domain Methods*, MIT, (1990), (Ph.D. Thesis).

Subsol G., Thirion J.P., Ayache N., "A General Scheme for Automatically Building 3D Morphometric Anatomical Atlases: Application to a Skull Atlas", *INRIA Tech. Rep.*, No: 2586, (1995).

Tanaka L., Kobayashi M., Fujimo T., Chiyokura H., "Simulation for Facial Lip Expression Using the Facial Muscle Model", *Proc. Computer Assisted Radiology (CAR '95)*, (1995).

Terzopoulos D., Fleischer K., "Deformable Models", *The Visual Computer*, Vol: 4, (1988).

Terzopoulos D., Waters K., "Analysis and Synthesis of Facial Image Sequences Using Physical and Anatomical Models", *IEEE Trans. On Pattern Analysis and Machine Intelligence*, Vol: 15, No: 6, (1993).

Terzopoulos, D., Waters K., "Techniques for Realistic Facial Modelling and Animation", *Computer Animation*, (1991), Springer-Verlag.

Toft P., *The Radon Transform, Theory and Implementation*, Dept. of Mathematical Modelling, Technical University of Denmark, (1996), **Ph.D. Thesis**.

Waters K., "A Physical Model of Facial Tissue and Muscle Articulation, Derived from Computer Tomography Data", *Proc. Visualization in Biomedical Computing (VBC'96)*, (1992).

Waters K., "A Muscle Model for Animating Three Dimensional Facial Expression", *Computer Graphics*, Vol: 21, No: 4, (1987).

Waters K., Terzopoulos D., "Modelling and Animating Faces Using Scanned Data", *J. Visualization Comput. Animation*, Vol: 2, No: 4, (1991).



Yasuda T., Hashimoto Y., Yokoi S., Toriwaki J.I., "Computer System for Craniofacial Surgical Planning Based on CT Images", *IEEE Trans. Medical Imaging*, Vol: 9, No:3, (1990).

# APPENDIX A

## A PROPERTIES OF THE 3D LINE NORMAL RADON TRANSFORM

### A.1. Basic Properties of the 3D Line Radon Transform

$$\check{g}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(s\tau + u\alpha + v\beta) ds \quad (\text{A.1})$$

#### A.1.1. Linearity

The first crucial property is linearity of the transform.

$$g(r) = \sum_i K_i g_i(r) \Rightarrow \check{g}(\theta, \phi, u, v) = \sum_i K_i \check{g}_i(\theta, \phi, u, v) \quad (\text{A.2})$$

where  $K_i$  are arbitrary constants.

#### A.1.2. Translation

It is assumed that

$$\begin{aligned} h(r) &= g(r - r_0) \Rightarrow \\ \check{h}(\theta, \phi, u, v) &= \int_{-\infty}^{\infty} g(s\tau + u\alpha + v\beta - r_0) ds \end{aligned} \quad (\text{A.3})$$

Here the translation  $r_0$  are resolved after the basis vectors, i.e.,

$$r_0 = s_0\tau + u_0\alpha + v_0\beta \quad (\text{A.4})$$

This can be done in an unambiguous way

$$s_0 = r_0 \cdot \tau \quad u_0 = r_0 \cdot \alpha \quad v_0 = r_0 \cdot \beta \quad (\text{A.5})$$

This is inserted in Eq.A.3

$$\bar{h}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g((s - s_0)\tau + (u - u_0)\alpha + (v - v_0)\beta) ds \quad (\text{A.6})$$

$$= \int_{-\infty}^{\infty} g(\bar{s}\tau + (u - u_0)\alpha + (v - v_0)\beta) d\bar{s} \Rightarrow \quad (\text{A.7})$$

$$\bar{h}(\bar{\theta}, \bar{\phi}, u, v) = \bar{g}(\bar{\theta}, \bar{\phi}, u - r_0 \cdot \alpha, v - r_0 \cdot \beta) \quad (\text{A.8})$$

Note that the angular parameters  $\theta$  and  $\phi$  are not changed.

### A.1.3. Rotation and Scaling

Rotation and scaling can be controlled by means of a general transformation matrix  $A$

$$h(r) = g(Ar) \quad (\text{A.9})$$

The diagonal elements of the 3\*3 matrix control the individual scaling and off-diagonal elements are controlling the rotation. The corresponding Radon transform is given by

$$\bar{h}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g(A(s\tau + u\alpha + v\beta)) ds \quad (\text{A.10})$$

$$= \int_{-\infty}^{\infty} g(sA\tau + uA\alpha + vA\beta) ds \quad (\text{A.11})$$

This will in general alter the directional vector  $\tau$  of the line, i.e., a new direction vector is needed

$$A\tau = \gamma\tilde{\tau} \quad |\tilde{\tau}| = 1 \quad \gamma \geq 0 \quad (\text{A.12})$$

The vector  $\tilde{\tau}$  defines cf. Eq.5.2 two corresponding angles  $\tilde{\theta}$  and  $\tilde{\phi}$ , which again defines the two remaining vectors  $\tilde{\alpha}$  and  $\tilde{\beta}$ , cf. Eq.A.12 is inserted into Eq.A.11.

$$\bar{h}(\theta, \phi, u, v) = \frac{1}{\gamma} \int_{-\infty}^{\infty} g(s\tilde{\tau} + A(u\alpha + v\beta)) ds \quad (\text{A.13})$$

The next step is to resolve the line offset parameters after the three new tilde vectors, which again can be done unambiguously.

$$A(u\alpha + v\beta) = s_0 \tilde{\tau} + u_0 \tilde{\alpha} + v_0 \tilde{\beta} \quad (\text{A.14})$$

This implies that the 3D line Radon transform is given by

$$\tilde{h}(\theta, \phi, u, v) = \frac{1}{\gamma} \int_{-r}^r g((s + s_0)\tilde{\tau} + u_0\tilde{\alpha} + v_0\tilde{\beta}) ds \quad (\text{A.15})$$

$$= \frac{1}{\gamma} \int_{-r}^r g(\tilde{s}\tilde{\tau} + u_0\tilde{\alpha} + v_0\tilde{\beta}) d\tilde{s} \quad (\text{A.16})$$

$$= \frac{1}{\gamma} \tilde{g}(\tilde{\theta}, \tilde{\phi}, u_0, v_0) \quad (\text{A.17})$$

where four tilde parameters are given by

$$\tilde{\tau} = \begin{pmatrix} \cos \tilde{\phi} \cos \tilde{\theta} \\ \cos \tilde{\phi} \sin \tilde{\theta} \\ \sin \tilde{\phi} \end{pmatrix} = \frac{1}{|A\tau|} A\tau, \quad \tilde{\alpha} = \begin{pmatrix} -\sin \tilde{\theta} \\ \cos \tilde{\theta} \\ 0 \end{pmatrix} \text{ and } \tilde{\beta} = \begin{pmatrix} -\sin \tilde{\phi} \cos \tilde{\theta} \\ -\sin \tilde{\phi} \sin \tilde{\theta} \\ \cos \tilde{\phi} \end{pmatrix} \quad (\text{A.18})$$

$$\gamma = |A\tau|, \quad u_0 = \tilde{\alpha} \cdot A \cdot (cu + \beta v), \quad v_0 = \tilde{\beta} \cdot A \cdot (cu + \beta v) \quad (\text{A.19})$$

The matrix A can be partitioned in three rotational matrices and one scaling matrix

$$A_z A_y A_x S \quad (\text{A.20})$$

where  $A_y$  rotates in the  $y$ - $z$  coordinates, e.g.

$$A_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \psi_x & \sin \psi_x \\ 0 & -\sin \psi_x & \cos \psi_x \end{pmatrix} \quad (\text{A.21})$$

and  $A_z$  rotates in the  $x$ - $z$  coordinates, e.g.

$$A_x = \begin{pmatrix} \cos \psi_x & 0 & \sin \psi_x \\ 0 & 1 & 0 \\ -\sin \psi_x & 0 & \cos \psi_x \end{pmatrix} \quad (\text{A.22})$$

and finally  $A_z$  rotates in the  $x$ - $y$  coordinates, e.g.

$$A_z = \begin{pmatrix} \cos \psi_z & \sin \psi_z & 0 \\ -\sin \psi_z & \cos \psi_z & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (\text{A.23})$$

The scaling matrix  $S$  here is chosen to be the lat in Eq.A.20 because the interpretation of the scaling is by far easier when applied directly on the base function  $g(r)$ . The matrix can be chosen to

$$S = \begin{pmatrix} \frac{1}{S_x} & 0 & 0 \\ 0 & \frac{1}{S_y} & 0 \\ 0 & 0 & \frac{1}{S_z} \end{pmatrix} \quad (\text{A.24})$$

The scaling parameters in the diagonal are chosen so that they directly relate to length in the new function  $h(r)$ . The total rotation and scaling matrix defined in Eq.A.20 multiplied together becomes

$$A = \begin{pmatrix} \frac{\cos \psi_x \cos \psi_y}{S_x} & \frac{\cos \psi_y \sin \psi_x}{S_y} & \frac{\sin \psi_y}{S_z} \\ -\frac{\cos \psi_z \sin \psi_x + \cos \psi_x \sin \psi_y \sin \psi_z}{S_x} & \frac{\cos \psi_x \cos \psi_z - \sin \psi_x \sin \psi_y \sin \psi_z}{S_y} & \frac{\cos \psi_y \sin \psi_z}{S_z} \\ \frac{\sin \psi_x \sin \psi_z - \cos \psi_x \cos \psi_z \sin \psi_y}{S_x} & -\frac{\cos \psi_x \sin \psi_x \sin \psi_y + \cos \psi_x \sin \psi_z}{S_y} & \frac{\cos \psi_y \cos \psi_x}{S_z} \end{pmatrix} \quad (\text{A.25})$$

## A.2. Analytical Radon Transform of Primitives

Along 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.

### A.2.1. The Ball

The first primitive is the unit ball, i.e.,

$$g_{ball}(x, y, z) = \begin{cases} 1 & \text{for } x^2 + y^2 + z^2 < 1 \\ 0 & \text{for } x^2 + y^2 + z^2 \geq 1 \end{cases} \quad (\text{A.26})$$

Due to the rotational symmetry of the primitive, the corresponding Radon transform cannot depend on the two angular parameters  $\theta$  and  $\phi$ , thus the Radon transform is derived in a rotated coordinate system with  $\tau$  lying along the x- axis and the vector  $r_0 = u\alpha + v\beta$  along the y- axis. As illustrated in Fig.A.1, the Radon transform is merely the distance between the two intersection points between the line and the ball.

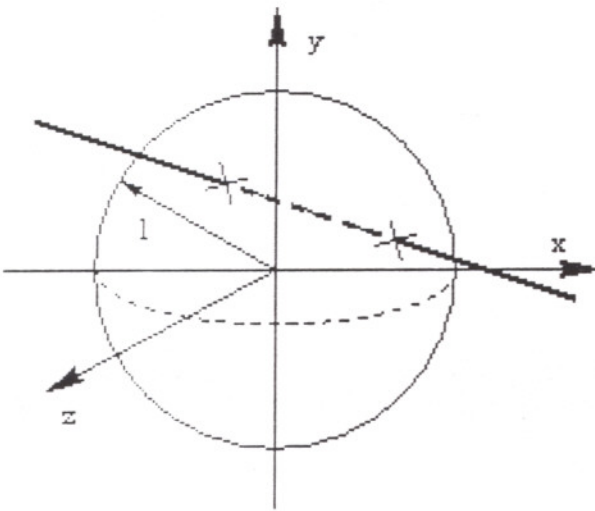


Figure A.1 The unit ball with radius 1 and the Radon transform is the length between the two intersection points between the line and the surface of the ball.

The length of  $r_{ij}$  is  $\sqrt{u^2 + v^2}$ , which implies that the Radon transform can be calculated as

$$\check{g}_{ball}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} g_{ball}(s, \sqrt{u^2 + v^2}, 0) ds \Rightarrow \quad (\text{A.27})$$

$$\check{g}_{ball}(\theta, \phi, u, v) = \begin{cases} 2\sqrt{1 - u^2 - v^2} & \text{for } u^2 + v^2 < 1 \\ 0 & \text{for } u^2 + v^2 \geq 1 \end{cases} \quad (\text{A.28})$$

### A.2.2. The Gaussian bell

The Gaussian bell is another primitive, where the Radon transform can be found analytically.

$$g_{gauss}(x, y, z) = \exp(-x^2 - y^2 - z^2) \quad (\text{A.29})$$

Again the function has rotational symmetry, hence the Radon transform does not depend on the angular parameters  $\theta$  and  $\phi$ , and the Radon transform can, e.g., be derived as

$$\check{g}_{gauss}(\theta, \phi, u, v) = \int_{-\infty}^{\infty} e^{-s^2 - u^2 - v^2} ds \Rightarrow \check{g}_{gauss}(\theta, \phi, u, v) = \sqrt{\pi} e^{-u^2 - v^2} \quad (\text{A.30})$$

# APPENDIX B

## B THE USAGE OF RECONSTRUCTION TOOLS

A software package has been developed for 3D reconstruction of sinograms (line integrals), as shown in Chapter 5. The programs described in this Appendix are compiled both on Linux, and Windows98 operating systems. In Section B.1 the package of “Recon3D” is explained, which can generate a volume and the corresponding four- dimensional sinogram from a set of scaled, rotated, and translated primitives.

### B.1. The 3D Reconstruction Program “Recon3D”

The program “Recon3D” uses the following parameters

- Sinogram [String] Name of the file containing the 4D sinogram. The file must have the name same as the .ini file, and the 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 the extension .f3f (binary format), or .a4f (ASCII format).
- Function [String] Parameter specifying the function to be used.
  - ♦ FB Direct reconstruction using Filtered Backprojection, result in Volume .
  - ♦ FAB Direct reconstruction using Filtering After Backprojection, result in Volume
  - ♦ CS Direct reconstruction using Fourier Slice theorem (Experimental). Result in Volume .
  - ♦ Forward Radon transform of volume into Sinogram.
  - ♦ ART Iterative reconstruction using ART. Result in Volume .
  - ♦ EM Iterative reconstruction using EM. Result in Volume .



- Iterations [Integer] For EM the number of iterations before the iteration ends. For ART the number of full iterations, i.e., the number of actual ART iterations are Iterations times the number of rows in the system matrix.
- Iterative\_Par\_1 [Float] In ART the initial weight parameter  $\lambda$ .
- Iterative\_Par\_2 [Float] In ART the final weight parameter  $\lambda$ .
- 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 will choose tri-linear interpolation technique.
- 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  $x$  samples in the volume. Only needed if OrgFile is not specified.
- Ysamples [Integer] The number of  $y$  samples in the volume. Only needed if OrgFile is not specified.
- Zsamples [Integer] The number of  $z$  samples in the volume. Only needed if OrgFile is not specified.

An example of valid ini- file for "Recon3D" is shown below. The ini- file is used to reconstruct a sinogram `El.ini.f4f` into `El.ini.FB.f3f` with the same sampling parameters as contained in the `El.ini.f3f`. Here the Filtered Backprojection is used.

```
Sinogram=El_Sino.f4f
Volume=El_Vol.FB.f3f
OrgFile=El_Vol.f3f
Function=FB
Xmin=-2
DeltaX=0.08
XSamples=51
Iterations=5
```

## **B.2. The Analytical Sinogram Program “3D RadonAna”**

The program “3D\_RadonAna” uses the following parameters.

- 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 is used.
- Zmin [Float] (optional) The minimum value of  $z$  in the volume. If not specified, Xmin is used.
- DeltaX [Float] The sampling interval of  $x$  in the volume.
- DeltaY [Float] (optional) . The sampling interval of  $y$  in the volume. If not specified, DeltaX is used.
- DeltaZ [Float] (optional) The sampling interval of  $z$  in the volume. If not specified, DeltaX is used.
- XSamples [Integer] The number of  $x$  samples in the volume.
- YSamples [Integer] (optional) The number of  $y$  samples in the volume. If not specified, XSamples is used.
- ZSamples [Integer] (optional) The number of  $z$  samples in the volume. If not specified, XSamples is used.
- USamples [Integer] The number of  $u$  samples in the sinogram.
- VSamples [Integer] (optional) The number of  $v$  samples in the sinogram. If not specified, USamples is used.

- DeltaU [Float] The sampling interval of  $u$  in the sinogram.
- DeltaV [Float] The sampling interval of  $v$  in the sinogram. If not specified DeltaU is used.
- PhiSamples [Integer] The number of  $\phi$  samples in the sinogram. This number will be distributed from  $\phi_{\min} = -\psi = -PhiLimit$  to  $-\phi_{\min} = \psi = PhiLimit$ .
- PhiLimit [Float] The axial acceptance angle  $\Psi$  (measured in degrees).
- GenerateVolume If set to 0 then the volume will not be generated, and if 1 it will be.
- GenerateSinogram If set to 0 then the sinogram will not be generated, and if 1, then it will be.
- NumberOfShapes Number of primitives used.

The ordering of parameters shown above is 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

- Shapei [String] where  $i$  is 0 to NumberOfShapes
- 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  $x$ -axis.
- offy [Float] Shift of of the primitive in the  $y$ -axis.
- offz [Float] Shift of of the primitive in 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  $x$ -axis.
- scaly [Float] Scaling of the primitive in direction of  $y$ -axis.
- scalz [Float] Scaling of the primitive in direction of  $z$ -axis.
- power [Float] Scaling of the value of primitive.

### B.3. The Functions used in “Recon3D”

- Recon3ddeneme2.cpp

This is the central program, which contains routines to call the main calculation routines. It reads the specified IniFile.ini file and executes one of the following functions:

```
ffl Central Slice (CS) experimental!  
ffl Filtered BackProjection (FB).  
ffl Filtering after BackProjection (FAB).  
ffl Algebraic Reconstruction Technique (ART).  
ffl Expectation Maximization (EM).  
ffl Convert between ASCII and Binary (Convert).  
ffl Numerical Radon Transform (RT).
```

- CSreconstruct3D

Synopsis : void CSreconstruct3D(void)  
Description : The function implements the Fourier Slice Theorem  
Usage : CSreconstruct()

- FABreconstruct3D

Synopsis : void FABreconstruct3D(void)  
Description :The function implements filtering after backprojection.  
Usage : FABreconstruct()

- FBreconstruct3D

Synopsis : void FBreconstruct3D(void)  
Description : The function implements filtered backprojection.  
Usage : FBreconstruct()

- ART3D

Synopsis : void ART3D(void)  
Description : The function implements 3D ART.  
Usage : ART3D()

- EM3D

Synopsis : void EM3D(void)

Description : The function implements 3D EM.

Usage : EM3D()

- Forward3D

Synopsis : void Forward3D(void)

Description : The function implements a numerical calculation of the line integrals.

Usage : Forward3D()

Note : Uses the OrgFile to specify the 4D Sinogram.

- Convert

Synopsis : void Convert(void)

Description : The function converts the Volume and Sinogram from ASCII to binary or opposite depending on the types. The function writes in the same filenames with opposite extensions.

Usage : Convert()

#### **B.4. Functions Used in “3D\_RadonAna”**

This program generates Volumes and corresponding four-dimensional sinograms of line integrals.

Currently two basic functions can be used. Either a uniform ball or a Gaussian 3D bell. Both can be scaled, rotated, and shifted and put together with an arbitrary number of objects.

- init

Synopsis : void init(void)

Description : The function will initialize and read the .ini-file.

- MakeVolume

Synopsis : void MakeVolume(void)

Description : The function will generate the volume.

- MakeSinogram

Synopsis : void MakeSinogram(void)

Description : The function will generate the 4D-Sinogram.

## B.5. Additional Functions used in both Packages

- `GenerateTrigDef`

Synopsis : `TrigDef * GenerateTrigDef(`  
          `int ThetaSamples, PhiSamples, Usamples, VSamples,`  
          `float Thetamin, Phimin, Umin, Vmin,`  
          `float DeltaTheta, DeltaPhi, DeltaU, DeltaV)`

Description : This function returns a `TrigDef` with proper trigonometric arrays defined. The structure is used for 3D reconstruction from line integrals.

Usage :

```
MyTrigDef=GenerateTrigDef(ThetaSamples,PhiSamples,USamples,VSamples,  
Thetamin,Phimin,Umin,Vmin, DeltaTheta,DeltaPhi,DeltaU,DeltaV);
```

- `GenerateXYZ`

Synopsis : `XYZDef * GenerateXYZ(`  
          `int Xsamples, YSamples, ZSamples,`  
          `float Xmin, Ymin, Zmin,`  
          `float DeltaX, DeltaY, DeltaZ)`

Description : This function returns a `XZYZDef` with proper arrays defined for sampling the volume domain. The structure is used for 3D reconstruction from line integrals.

Usage :

```
MyXYZDef=GenerateXYZ(XSamples,YSamples,ZSamples, Xmin,Ymin,Zmin,  
DeltaX,DeltaY,DeltaZ);
```

- `GenerateXYZ_fromMultiStruct`

Synopsis : `XYZDef *GenerateXYZ_fromMultiStruct(`  
          `MultiStruct *MyMultiStruct)`

Description : The function generates the `XYZDef` from a three dimensional `MultiStruct`.

Usage : `MyXYZ=GenerateXYZ fromMultiStruct(MyMultiStruct);`

Note :See `GenerateXYZ`.

- `GenerateXYZ_fromIniFile`  
 Synopsis : `XYZDef *GenerateXYZ_fromIniFile()`  
 Description :The function generates the XYZDef from data from the IniFile  
 Usage : `MyXYZ=GenerateXYZ fromIniFile();`  
 Note :See GenerateXYZ
- `GenerateTrigDef_fromMultiStruct`  
 Synopsis :`TrigDef *GenerateTrigDef_fromMultiStruct(MultiStruct *MyMultiStruct)`  
 Description :The function generates the TrigDef from a four dimensional MultiStruct.  
 Usage :`MyTrig=GenerateTrigDef_fromMultiStruct(MyMultiStruct);`  
 Note :See GenerateTrigDef
- `CSrebin3D_TL`  
 Synopsis :`MultiStruct * CSrebin3D_TL(MultiStruct*,TrigDef*,XYZDef*);`  
 Description :Calculates the two-dimensional Fourier transformation of a sinogram (for each combination of angles). This sinogram is distributed in the Fourier space of the reconstructed volume with tri linear interpolation. (Central Slice ifea).  
 Usage :`RecVol=CSrebin3D Tl(Sinogram,MyTrig,MyXYZ);`  
 Reconstructs the volume from the sinogram using central slice theorem for lines in a three dimensional space.
- `CSrebin3D_NN`  
 Synopsis : `MultiStruct * CSrebin3D_NN(MultiStruct*,TrigDef*,XYZDef*);`  
 Description : Calculates the two-dimensional Fourier transformation of a sinogram (for each combination of angles). This sinogram is distributed in the Fourier space of the reconstructed volume. (Central Slice ifea).  
 Usage : `RecVol=CSrebin3D NN(Sinogram,MyTrig,MyXYZ);`  
 Reconstructs the volume from the sinogram using central slice theorem for lines in a three dimensional space.
- `NumericalRadon3D_Restore`  
 Synopsis : `MultiStruct* NumericalRadon3D—Restore(MultiStruct *MyRadon, MultiStruct *InVolume, TrigDef *MyTrig)`

Description :The function will numerically calculate line integrals through a three dimensional volume. The sampling of the Radon domain is given through MyTrig.

Usage :NumericalRadon3D(MyRadon,InVolume,MyTrig);

Note :Uses bilinear interpolation. Requires that the Sinogram (MyRadon) is allocated. See also NumericalRadon3D.

- NumericalRadon3D

Synopsis :MultiStruct\* NumericalRadon3D(MultiStruct \*InVolume,TrigDef \*MyTrig)

Description : The function will numerically calculate line integrals through a three dimensional volume. The sampling of the Radon domain is given through MyTrig.

Usage : MyMultiStruct=NumericalRadon3D(InVolume,MyTrig);

Note : See NumericalRadon3D Restore.

- NumericalRadon3D\_OneSample\_NN

Synopsis : int NumericalRadon3D—OneSample—NN(VolWeight \*,TrigDef \*,XYZdef \*, int,int,int,int)

Description :The function will numerically calculate line integrals through a three dimensional volume ONLY at one samples in the parameter domain. The function will number of samples and in the first parameter the indices and weight factors corresponding to the voxels. The sampling of the Radon domain is given through MyTrig. uint, vvint, thetaint and phiint determines the sample in the parameter domain. The function will return in VolWeight, which must be externally be allocated with at least a length of maxfXSamples, YSamples, ZSamplesg.

Usage :NoOfSamples=NumericalRadon3D OneSample NN(MyWeight, MyTrig, MyXYZ, int thetaint, int phiint int uint, int vvint);

Note :Uses currently nearest neighbor interpolation. See also NumericalRadon3D.

- NumericalRadon3D\_OneSample\_BL

Synopsis :int NumericalRadon3D—OneSample—BL(VolWeight \*,TrigDef \*,XYZdef \*, int,int,int,int)



Description : The function will numerically calculate line integrals through a three dimensional volume ONLY at one samples in the parameter domain. The function will number of samples and in the first parameter the indices and weight factors corresponding to the voxels. The sampling of the Radon domain is given through MyTrig. uuint, vvint, thetaint and phiint determines the sample in the parameter domain. The function will return in VolWeight, which must be externally be allocated with at least a length of maxfXSamples, YSamples, ZSamplesg.

Usage : NoOfSamples=NumericalRadon3D OneSample BL(MyWeight, MyTrig, MyXYZ, int thetaint, int phiint int uuint, int vvint);

Note : Uses bilinear interpolation. See also NumericalRadon3D.

- BackProject3D\_Restore

Synopsis : void BackProject3D—Restore(MultiStruct \*MyVol, MultiStruct \*My4D, XYZDef \*MyXYZ, TrigDef \*MyTrig)

Description : The function will backproject a 4D set of line integrals through a 3D volume. (See Chapter 5.)

Usage : BackProject3D Restore(MyVolume,My4D,MyXYZ,MyTrig);

Note : This function requires that the volume has been allocated elsewhere.

- BackProject3D

Synopsis : MultiStruct \*BackProject3D( MultiStruct \*My4D, XYZDef \*MyXYZ, TrigDef \*MyTrig)

Description : The function will backproject a 4D set of line integrals through a 3D volume. (See Chapter 5)

Usage : My3D=BackProject3D(My4D,MyXYZ,MyTrig);

Note : See also BackProject3D Restore

Searches for an entry 'Function' in IniBuffer, and returns its value in Function.

- MultReStore

Synopsis : void MultReStore(float \*p1,float \*p2)

Description : The function will multiply the two complex numbers A and B and store the result at the location of A. Here  $A = p1[0] + i p1[1]$  and  $B = p2[0] + i p2[1]$ .

Usage : MultReStore(arr1,arr2);

Preforms the multiplication  $arr1=arr1*arr2$ .

- `MultNew`

Synopsis : `void MultNew(float *p1,float *p2,float *p3)`

Description : and B, so that  $C = AB$ . The result is stored at the location of C. Here  $A = p1[0] + i p1[1]$ ,  $B = p2[0] + i p2[1]$  and  $C = p3[0] + i p3[1]$ .

Usage : `MultNew(arr1,arr2,arr3);`

Performs the multiplication  $arr3=arr1*arr2$ .

- `FindSignalIndexInMultiStruct`

Synopsis : `int FindSignalIndexInMultiStruct(int *Indices,int *Samples,int Dimensions)`

Description : Returns index in `MultiStruct->Signal` for the multidimensional element with indices `Indices`. Applies to `MultiStruct` of type `RealArray`.

Usage : `Myindex=int FindSignalIndexInMultiStruct(Indices,Samples,Dim)`

Note : See `FindIndicesInMultiStruct` for the opposite function.

- `FindIndicesInMultiStruct`

Synopsis : `void FindIndicesInMultiStruct(int *Indices,int *Samples, int index,int Dimensions)`

Description : Returns indices in for the multidimensional element with index `Index` in `MultiStruct->Signal`. Applies to `MultiStruct` of type `RealArray`.

Usage : `FindSignalIndexInMultiStruct(Indices,Samples,index,Dim)`

Note : See `FindIndexInMultiStruct` for the opposite function.

- `SliceMultiStruct`

Synopsis : `MultiStruct * SliceMultiStruct(MultiStruct *InMultiStruct,int * CutVector)`

Description : The function returns a slice of `InMultiStruct` corresponding to `CutVector` which is of same dimension. If the *i*'th element of `CutVector` is negative that dimension is unchanged. If positive elements with that position is chosen. The number of non-negative elements determine the dimension of the returned `MultiStruct`.

Usage : `MyMulti=SliceMultiStruct(InMultiStruct,CutVector)`

- `SubtractMultiStruct`

Synopsis : `MultiStruct * SubtractMultiStruct(MultiStruct *,MultiStruct *)`;

Description : This function will subtract the second MultiStruct from the first. It is checked whether the two are of same size.

Usage : DifNewMultiStruct=SubtractMultiStruct(MyMult1,MyMult2);

- L2\_measure

Synopsis : float \* L2—measure(MultiStruct \*,MultiStruct \*);

Description : This function will return the variance estimate of the difference, i.e., the squareroot of the second order measure of misfit between the two MultiStructs. It is checked whether the two are of same size.

Usage : MyL2=L2 measure(MyMult1,MyMult2);

- StatMultiStruct

Synopsis : void StatMultiStruct(MultiStruct \*,float \*, float\*);

Description : This function will return the mean estimate and the deviance estimate of the MultiStruct.

Usage : StatMultiStruct(MyMulti,&MyMean,&MyDev);  
where MyMean, MyDev are of type float.

- ReadFIF

Synopsis : Image \*ReadFIF(char \*FileName);

Description : The function reads a picture in .fif format (raw float with header) from the file 'FileName.fif'. All memory allocation is done internally. A pointer to the new image is returned.

Usage : Test = ReadFIF("Test");

Reads the image Test.fif and return the pointer in Test.

- WriteFIF

Synopsis : void WriteFIF(Image \*MyImage);

Description : The function writes a picture in .fif format. The name of the saved image is determined by MyImage->FileName. The file extension is appended to the filenames in all the 'write' routines.

Usage : WriteFIF(Test);

Write the image Test to a .fif file .

- WriteMultiStruct

Synopsis : void WriteMultiStruct(MultiStruct \*MyMultiStruct)

Description :The function will save the MultiStruct to the disc under the filename contained in the structure. A file extension .f<N>f is added, where <N> is the dimension.

Usage : WriteMultiStruct(MyMultiStruct);

- ReadMultiStruct

Synopsis : MultiStruct \* ReadMultiStruct(char \*FileName,int Dim)

Description : The function will read a MultiStruct with filename FileName.f<Dim>f.

Usage : MyMultiStruct=ReadMultiStruct("My3D",3);

This will read My3D.f3f.

- Iterative\_ART\_3D

Synopsis :MultiStruct \* Iterative—ART—3D(MultiStruct \*,XYZDef \*,float)

Description : This function will estimate the reconstructed volume using ART in a three dimensional version. The function will return the reconstructed volume. The input parameters are: The 4D sinogram, a XYZDef defining the sampling of the volume and the last parameter determines the number of iterations relative to the number of samples in the 4D sinogram.

Usage : MultiStruct \* Iterative\_ART\_3D(MultiStruct \*MyRadon, XYZDef \*MyXYZ, float maxiterations)

- Iterative\_EM\_3D

Synopsis : MultiStruct \* Iterative\_EM\_3D(MultiStruct \*,XYZDef \*,float)

Description :This function will estimate the reconstructed volume using EM in a three dimensional version. The function will return the reconstructed volume. The input parameters are: The 4D sinogram, a XYZDef defining the sampling of the volume and the last parameter determines the number of iterations relative to the number of samples in the 4D sinogram.

Usage : MultiStruct \* Iterative\_ART\_3D(MultiStruct \*MyRadon, XYZDef \*MyXYZ, float maxiterations)

- WindowVolume

Synopsis : void WindowVolume(MultiStruct \*My3D,int width)

Description : The function windows a volume with a Hann window in all three directions. The parameter width determines how many samples are altered measured from the edges.

Usage : WindowVolume(InVolume,5);

- Filter4DSinogram

Synopsis : void Filter4DSinogram(MultiStruct \*My4D, TrigDef \*MyTrig, float PhiMin, int WindowParam, float Relative\_filtercutoff)

Description : The function will apply a highpass filter to the images  $(u, v)$  for each  $(\theta, \phi)$ . The parameter PhiMin determines the limiting  $\Psi$ -angle in the geometry  $\Omega_\Psi$ . The parameter WindowParam controls the windowing. If set to 0 no windowing is done. If set to 1 pre-windowing is done. A cubic Hann- window is used to reduce the edge artifacts.

Usage : Filter4DSinogram(My4D,MyTrig,PhiMin,0);

- FilterVolume

Synopsis : void FilterVolume(MultiStruct \*My3D,float psi)

Description : The function will filter a given volume with a high pass filter. The function is intended for a filtering after backprojecting algorithm.

Usage : FilterVolume(InVolume,-MySinogram->psi\_min);

- ComplexNdimFFT

Synopsis : void ComplexNdimFFT(float \*data, unsigned long \*nn, int ndim, int isign);

Description : Calculates the n-dimensional Fourier transformation of a n-dimensional data structure in data. (See (Press et al.1989) pp. 523-524 for a complete description). Data are stored in sequence in data. The dimensions in dim, where dim[1] is the size in the first direction, and so on. . .

Usage : ComplexNdimFFT(&Test[-1],&dim[-1],3, \_FFT);

Calculates the 3-dimensional FFT of the image stored in data.

Note :Be sure that dimensions is powers of two. See also NewNdimFFT.

- BoxFilterVolume

Synopsis : void BoxFilterVolume(MultiStruct \*My3D,int hwidth)

Description : The function filters a volume with a cubic box filter with width equal  $2*hwidth+1$  in all three directions.

Usage : WindowVolume(InVolume,5);

- NdimFFT

Synopsis : void NdimFFT(float \*data, int \*nn, int ndim, int isign);

Description : Calculates the n-dimensional Fourier transformation of a n-dimensional data structure in data. (See (Press et al. 1989) pp. 523-524 for a complete description). Used by FFTImage. Data are stored in sequence in data. The dimensions in dim, where dim[0] is the size in the first direction, and so on. . . This function is meant to give a simpler interface (no old-fortran shift problem) to ComplexNdimFFT.

Usage : NdimFFT(Test,dim,3, FFT);

Calculates the 3-dimensional FFT of the image stored in data.

Note : Be sure that dimensions is powers of two. See also ComplexNdimFFT.