Linköping Studies in Science and Technology
Dissertation No. 672

Cone-Beam Reconstruction Using Filtered Backprojection

Henrik Turbell

INSTITUTE OF TECHNOLOGY
LINKÖPINGS UNIVERSITET

Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden

Linköping, February 2001

ISBN 91-7219-919-9
ISSN 0345-7524

Printed in Sweden by UniTryck, Linköping 2001

To my parents

Abstract

The art of medical computed tomography is constantly evolving and the last years have seen new ground breaking systems with multi-row detectors. These tomographs are able to increase both scanning speed and image quality compared to the single-row systems more commonly found in hospitals today. This thesis deals with three-dimensional image reconstruction algorithms to be used in future generations of tomographs with even more detector rows than found in current multi-row systems.

The first practical algorithm for three-dimensional reconstruction from cone-beam projections acquired from a circular source trajectory is the FDK method. We present a novel version of this algorithm that produces images of higher quality. We also formulate a version of the FDK method that performs the backprojection in $\mathcal{O}(N^3 \log N)$ steps instead of the $\mathcal{O}(N^4)$ steps traditionally required.

An efficient way to acquire volumetric patient data is to use a helical source trajectory together with a multi-row detector. We present an overview of existing reconstruction algorithms for this geometry. We also present a new family of algorithms, the PI methods, which seem to surpass other proposals in simplicity while delivering images of high quality.

The detector used in the PI methods is limited to a window that exactly fits the cylindrical section between two consecutive turns of the helical source path. A rebinning to oblique parallel beams yields a geometry with many attractive properties. The key property behind the simplicity of the PI methods is that each object point to be reconstructed is illuminated by the source during a rotation of exactly half a turn. This allows for fast and simple reconstruction.

Acknowledgements

I would like to take this opportunity to thank all those who have contributed to this thesis, directly or indirectly.

First and foremost, I would like to thank my supervisor Prof. Per-Erik Danielsson who introduced me to the field and whose enthusiasm has been a daily source of inspiration. Much of the work in this thesis is based on his ideas. It has been a privilage to have a supervisor who always keeps the door open and is eagerly willing to discuss the work.

Prof. Paul Edholm, Dr. Maria Magnusson-Seger, and Jan Eriksson, Tec. Lic., for letting me participate in their research on helical reconstruction and for many interesting discussions where their knowledge and experience always were of great help.

Dr. Roland Proksa, Dr. Michael Grass, and Dr. Thomas Köhler at Philips Research Laboratories in Hamburg, for showing an early interest in the PI method and taking the time to explain many of the real-world objectives and restraints in medical imaging. The thesis shows several examples of the symbiotic relationship developed over the years, where the Linköping algorithms have been improved upon by the Hamburg lab and vice versa.

Prof. Gabor Herman, Dr. Samuel Matej, Dr. Robert M. Lewitt, and Bruno Motta de Carvalho at the Medical Image Processing Group at Pennsylvania University for their hospitality and for sharing their deep knowledge in algebraic reconstruction techniques during my stay in Philadelphia in the autumn of 1998. Although this month of research did not result in any conclusive results, it gave me an invaluable perspective on image reconstruction.

Prof. Michel Defrise not only for publishing mind-boggling algorithms with firm theoretical foundations but also for giving creative feedback on an early draft of the thesis.

Jens, our system engineer, and his predecessors, for providing a reliable computer system.

All past and present members of Image Processing Laboratory for a most enjoyable working environment.

Qingfen for all her food, love, and understanding during the final stages of the work.

The financial support from the Swedish Research Council for Engineering Sciences (281-95-470) and from Philips Medical Systems is gratefully acknowledged.

Contents

1

Introduction

Computed tomography (CT) is a technique for imaging cross-sections of an object using a series of X-ray measurements taken from different angles around the object. It has had a revolutionary impact in diagnostic medicine and has also been used successfully in industrial non-destructive testing applications. In 1972 Hounsfield patented the first CT scanner and he was awarded a Nobel Prize together with Cormack for this invention in 1979. Ever since, new developments have led to faster scanning, better dose usage and improved image quality.

An important part in this story of success has been the development of new efficient image reconstruction algorithms. Although the problem of image reconstruction in its purest mathematical form was solved by Johann Radon in 1917, the field is steadily evolving and gives rise to a seemingly never ceasing flow of new algorithms.

1.1 Modern Computed Tomography

There exist many texts on the history of tomography. Webb (1990) gives a detailed survey of classical tomography, i.e. techniques used before the computerised version was invented, and also discusses the first CT-related patents. Kalender (2000) presents the main developments thereafter. In this section we will focus on the on the developments in CT during the last ten years.

The so-called slip-ring technique was introduced in CT around 1990. Previously, power support to the X-ray tube and connectors for tapping detector data

Figure 1.1 *A modern tomograph (by courtesy of Philips Medical Systems).*

required a set of long cables hooked to the rotating gantry. To avoid strangulation, this gantry had to be accelerated and decelerated between data captures of two successive slices. In a slip-ring system, such as the one shown in Figure 1.1, the cables are replaced by slipping contacts for the power and cable-free optical connections for the measurement data. Alternative solutions include on-gantry rechargeable batteries and short-term memories, which are recharged and tapped, respectively, over fixed connectors between two imaging events.

In any case, due to the slip-ring technology, the X-ray source could be given a continuous and well-controlled rotation velocity, which today has been increased to over two rotations per second. The table with the patient is translated continuously to provide for exposure of successive slices. Relative to the patient, the source is then moving in a helix with a rather small pitch. In these single-row helical CT-systems the detector still consists of a single row of detector elements arranged along a circular arc centred in the source. The first reconstruction algorithms for this setup are also inherited from the previous planar fan-beam situation, but some precautions have to be taken. For instance, the fan-beam projections available for reconstruction of an image slice are no longer stemming from rays that are contained in the slice.

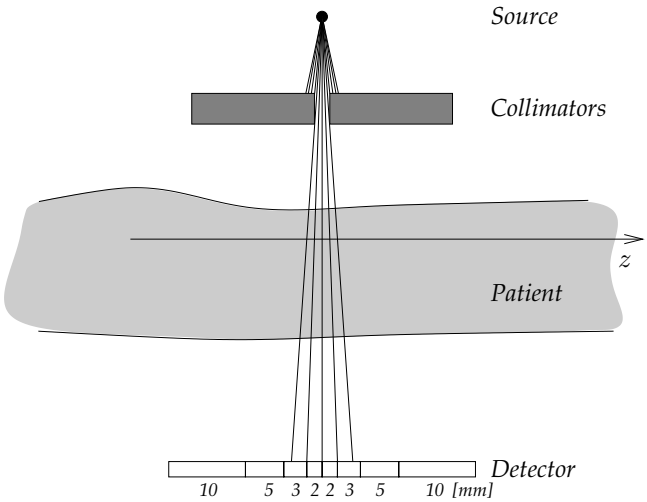


Figure 1.2 *By collimating the beam and by adding rows together, several slice widths are possible using this 8-row adaptive array detector. (The scale is heavily distorted.)*

Even if the slip-ring allowed for a considerable increase in speed, it soon became obvious that the next generations of CT-machines would include multi-row detectors. One manufacturer introduced double-row detectors several years ago, but the real breakthrough seems to have happened in 1998 with at least three new machines coming out on the market. These machines are capable to collect data from four rows in parallel with an approximately corresponding gain in speed. With such a relatively low number of rows the rays hitting the multi-row detector are only somewhat more divergent in the translation direction than the single-row rays. As a consequence, the first generation reconstruction algorithms for multi-row data rely heavily on single-row counterparts.

Figure 1.2 shows the detector for one of these state-of-the art tomographs. It consists of eight rows of varying widths. The detector signals are delivered to pre-amplifiers, high-precision AD-converters and pre-processing DSPs. Presently, these components constitute a data-rate bottle-neck and no commercially available systems today can handle data from more than four rows. This number is expected to double within the near future. To exploit all eight rows in Figure 1.2 signals from several detector elements can be added before AD-conversion. The detector rows have different widths which together with collimation allow for several different configurations listed in Table 1.1. Other manufacturers have detector rows of equal widths, or two different widths, and corresponding configuration schemes to efficiently obtain four detector rows for different slice widths.

Total Width	Detector Rows Utilized	Slice Width
2 mm	$2+2$ collimated to $1+1$	1 mm
4 mm	$2+2$	2 mm
8 mm	$3+2+2+3$ collimated to $2+2+2+2$	2 mm
10 mm	$(3+2)+(2+3)$	5 mm
20 mm	$5+(3+2)+(2+3)+5$	5 mm
20 mm	$(5+3+2)+(2+3+5)$	10 mm
40 mm	$10+(5+3+2)+(2+3+5)+10$	10 mm

Table 1.1 *Possible configurations using the detector in Figure 1.2. All widths are given as measured on the detector.*

The multi-row systems provide important flexibility and trade-off between dose, speed, signal-to-noise ratio, and z-resolution by changing the table speed, collimation of the cone-beam, and adjusting the X-ray-tube current. The speed of the new multi-row CT-systems is impressive. With a slice width of $\Delta z = 2$ mm and 2.5 rotations per second, a volume covering 200 mm in the z-direction is acquired in around ten seconds. Most patients have no problem to keep their breaths for ten seconds, which means that breathing artifacts can be avoided, or at least alleviated in full body scanning. One manufacturer even has a visible "egg-clock" so that the patients know how much longer they should hold their breaths. The rotation speed is kept constant since the rotating gantry, constantly subjected to heavy forces of several G, is designed to work optimally at a certain speed. The variable parameters are therefore the translation speed of the patient and the detector row width.

These high rotation speeds not only allow for high patient throughput, but also open up the possibility for *gated* tomography, where motion artifacts when imaging the heart can be reduced since each phase of the cardiac cycle is measured from a projection angle interval long enough to allow for reconstruction. The reconstruction algorithm uses the E.C.G. of the patient during the scan to determine the cardiac phase. A recent example of gated tomography utilising a multi-row detector is the work of Pan and Shen (2000). Other examples of medical application where fast scanning is a virtue are volume flow analysis and dynamic studies of a contrast agent.

The medical importance of multi-row scanners is not only reflected in the numerous clinical studies using such scanners presented at scientific meetings such as the annual meeting of the Radiological Society of North America (RSNA) but also in articles in more popular magazines such as the cover story by Kopecky, Buckwalter, and Sokiranski (1999) in Diagnostic Imaging.

1.2 Cone-Beam Reconstruction

Although the projections are obtained from a cone-beam rather than a fan-beam, the so-called multi-slice reconstruction algorithms in contemporary machines are firmly rooted in two-dimensional reconstruction of planar objects. The multi-slice algorithms can be divided into short-scan and full-scan algorithms. The short-scan algorithms are dominating and these are preferably classified according to how redundant data are handled, using smooth sinogram windowing, complementary rebinning, or parallel rebinning, as will be explained below.

It is well known that the contemporary reconstruction algorithms for multi-row data are far from optimal. Being adaptations and approximations of the two-dimensional case, they are not giving full value in image quality for the extensive measuring effort. Also, any further increase of the pitch and the cone-angle will make them obsolete. In fact, several new, fully three-dimensional helical cone-beam reconstruction algorithms are already waiting to take over. Most of these are using three-dimensional backprojection and they are the topic of the second half of this thesis. It might be surprising that none of these algorithms as of yet have been employed commercially. One probable reason is that the algorithms used today enable re-use of existent optimised hardware specially designed for two-dimensional filtered backprojection. Another one is the natural tendency to move slowly on new and unbroken paths.

Three-dimensional reconstruction algorithms are conveniently divided into exact and non-exact algorithms. An exact algorithm is mathematically correct in the following sense. Let ε be the largest allowable deviation between the object function and the reconstructed result. Provided noise-free projection data, captured with sufficient density along the source trajectory with a detector array having sufficient detector element density, an exact algorithm is able to deliver the wanted result for any given ε. Non-exactness can depend on several things. Missing data is a common reason. In the two-dimensional case it is easy to understand that if parallel projection data are lacking in a certain angular interval, certain frequency components of the object function are simply missing and therefore exact reconstruction is impossible. However, non-exactness may also be due to deliberate simplifications of an otherwise exact reconstruction algorithm. The gain in speed and simplicity may outweigh the advantage of exactness.

The first successful fully three-dimensional algorithm is non-exact and due to Feldkamp, Davis, and Kress (1984). It is presented in Section 3.2. The source trajectory is a circle. See Figure 1.3. Each horizontal row of detector values is ramp-filtered just as if they were projections of a two-dimensional object. The filtered projection data are then backprojected along the original rays, a procedure we call three-dimensional backprojection in the sequel. The mid-slice of the object will be reconstructed exactly. Object points further away from this plane will exhibit arti-

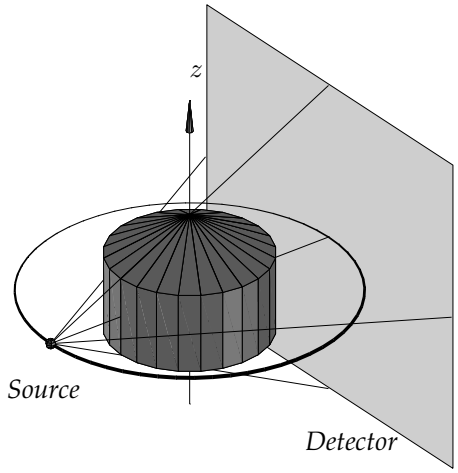


Figure 1.3 *A circular source trajectory. Image points exposed from all projection angles are found within a truncated double cone.*

facts with growing cone-angles. An object which extends above the given detector can also be reconstructed with this algorithm, although the reconstructible part is limited to the region where the object points are exposed in all projections. This region is a truncated double cone as shown in Figure 1.3. Truncation of projections in the transaxial direction is not allowed.

The exact algorithm due to Grangeat (1987) is an important milestone in three-dimensional reconstruction. Mathematically, the algorithm is a generalisation of two-dimensional filtered backprojection in the sense that it performs an inverse three-dimensional Radon transform. This requires a first processing step where the ray sums, line integrals, captured by the two-dimensional detector are converted into three-dimensional Radon data, which means plane integrals. For parallel X-ray projections, such a conversion is easy. Each line integral in the detector plane produces one plane integral. The divergent cone-beam projections pose a greater problem, the solution of which is the core of the thesis of Grangeat (1987). This algorithm requires non-truncated projections. This is unfortunate since a detector that covers the patient from top to toe is unlikely to materialise.

Ideally, a helical cone-beam reconstruction algorithm should be able to output results in a continuous fashion in near synchronism with the input data. Intuitively, this seems quite feasible. As soon as an object point has left the section of exposure for good, it could appear that there is no more information to measure for this point. However, for exact reconstruction, this intuitive conclusion seems to be at odds with the well-known Tuy-Smith sufficiency condition (Tuy

1983), which states that all planes which intersect the object must be visited by the source. This dilemma is sometimes called the long object problem. Recent efforts by Defrise, Noo, and Kudo (1999) and Schaller, Noo, Sauer, Tam, Lauritsch, and Flohr (1999) presented in Sections 4.1.3 and 4.1.3 have solved the long object problem.

1.3 Main Contributions

The main contribution of this thesis is a collection of new algorithms for circular and helical cone-beam reconstruction. These new algorithms are:

- the P-FDK method in Section 3.3.1
- the FDK-SLANT method in Section 3.3.3
- the FDK-FAST method in Section 3.4
- the PI-ORIGINAL method in Section 4.3.1
- the PI-SLANT method in Section 4.3.2
- the PI-2D method in Section 4.3.3
- the PI-FAST method in Section 4.3.4

The algorithm development has often been a team effort and it is difficult to afterwards pinpoint which ideas originated from whom. Most of the ideas resulting in the circular methods were by Henrik Turbell whereas the PI methods mainly originate from ideas by Professor Per-Erik Danielsson.

A substantial part of the material in the thesis is a review of algorithms published by other research groups world-wide. These sections hopefully contribute as an introduction to cone-beam reconstruction using filtered backprojection and as a unified source of references for further details. Chapter 6 contains a list on the specific contributions in the review sections.

1.4 Outline of the Thesis

Table 1.2 shows one possible taxonomy of filtered backprojection algorithms with one-dimensional ramp-filtering. This table also reflects the organisation of the thesis.

Chapter 2 starts with an introduction to two-dimensional reconstruction algorithms where the basic concepts of the following chapters are explained. One approach to the relatively new area of fast backprojection is presented in Section 2.3.

Type	BP	Full-Scan	Short-Scan		
			Parallel Rebinning	Complementary Rebinning	Smooth Sinogram Windowing
Circular	**2D**	Section 2.2.2	Sections 2.2.1 and 2.2.3	Section 2.2.3	Parker (1982), Section 2.2.3
	3D	FDK (Feldkamp et al. 1984), Section 3.2	**P-FDK** (Turbell 1999); HT-FDK (Grass et al. 2000a), Section 3.3.1; **FDK-SLANT**, Section 3.3.3; **FDK-FAST** (Turbell and Danielsson 1999b), Section 3.4	*No known implementation*	Zeng and Gullberg (1990), Section 3.3
Helical	**2D** *to parallel slices*	360°LI, Section 4.2.1	Schaller et al. (1998), Section 4.2.1	180°LI, Taguchi and Aradate (1998), Hu (1999), Section 4.2.1	CB-SSRB (Noo et al. 1998), Section 4.2.1
	2D *to nutating slices*	*No possible implementation*	Larson et al. (1998), Heuscher (1999), ASSR (Kachelriess et al. 2000a), Section 4.2.2; **PI-2D** (Turbell and Danielsson 2000), Section 4.3.3	*No known implementation*	*No known implementation*
	3D	Kudo and Saito (1991), Wang et al. (1993), Yan and Leahy (1992), Section 4.2.3; n-PI (Proksa et al. 2000), Section 4.3.5	IHCB (Silver 1997), Section 4.2.3; **PI-ORIGINAL** (Danielsson et al. 1998a), Section 4.3.1; **PI-SLANT** (Turbell and Danielsson 1999a), Section 4.3.2; **PI-FAST**, Section 4.3.4	CFBP (Schaller et al. 1996), Section 4.2.3	Wang et al. (1994); IHCB (Silver 1998); SS-FDK (Noo et al. 1998), Section 4.2.3

Table 1.2 *Categorisation of filtered backprojection algorithms using one-dimensional ramp-filtering. The first row correponds to two-dimensional reconstruction, whereas the other four rows correspond to three-dimensional reconstruction. The contributions of this thesis are written in bold.*

Chapter 3 deals with reconstruction from cone-beam data where the source trajectory is limited to a circle. The chapter starts with a short presentation on exact cone-beam reconstruction in general. The approximate FDK algorithm of Feldkamp et al. (1984) is then presented. New rebinning and filtering approaches for FDK are proposed in Section 3.3. The idea of fast backprojection is applied to cone-beam data in Section 3.4. The resulting method, FDK-FAST, has some shortcomings and the section may be temporarily skipped.

In Chapter 4 we present new and review existing algorithms for helical cone-beam reconstruction. The exact methods are presented in Section 4.1. Space only permits a brief and non-formal presentation, but many important concepts for the following sections are introduced. The main focus of the chapter is on the approximate algorithms in Section 4.2. The efficient and approximate PI-ORIGINAL is presented in Section 4.3.1. New extensions and variations on this algorithm are then presented in Sections 4.3.2 – 4.3.5.

When evaluating reconstruction algorithms it is useful to have synthetically generated projection data as input. This provides the evaluator full control over the geometry, noise, and related properties which are non-perfect and calibration-dependent when using data from a real tomograph. A new method for the generation of projection data from discrete data sets is discussed and evaluated in Chapter 5.

1.5 Publications

Certain results of the thesis have previously been published by:

- Turbell (1997)
  Contains a description of the software used in the thesis to generate synthetical projection data.
- Danielsson, Edholm, Eriksson, Magnusson-Seger, and Turbell (1998a), also published as Danielsson, Edholm, Eriksson, Magnusson-Seger, and Turbell (1998b)
  Patent application with the first presentation of PI-ORIGINAL.
- Turbell and Danielsson (1998)
  Presents PI-ORIGINAL and some of its properties in a new way.
- Turbell and Danielsson (1999b)
  Presents P-FDK and FDK-FAST.
- Turbell (1999)
  Licentiate thesis. Approximately half of the material in this Ph.D.-thesis is a revision of the licentiate thesis.

- Turbell and Danielsson (1999a)
  Presents PI-SLANT and PI-2D.

- Clasén (1999), supervised by Turbell
  A Master thesis on the use of teture mapping hardware for acceleration of three-dimensional backprojection.

- Turbell and Danielsson (2000)
  Review of methods for helical cone-beam reconstruction.

- Köhler, Turbell, and Grass (2000)
  Presents a new method for line integration through voxel volumes.

Omitted from the list are some articles presented only at national symposia.

Parts of the work presented in the thesis has resulted in the following patent applications:

- "Computer Tomography Method with Helicoidal Scanning of an Examination Area"
  International patent application WO 99/36885, filing date 1999-07-22
  Describes PI-ORIGINAL, Section 4.3.1.

- "Computertomographie-Verfahren mit helixfoermiger Relativbewegung"
  European patent application 00203160, filing date 2000-09-12
  Describes PI-SLANT and PI-2D, Sections 4.3.2 and 4.3.3.

- "Computertomographie-Verfahren mit helixfoermiger Relativbewegung"
  German patent application 10061120.6, filing date 2000-07-12
  Describes PI-FAST, Section 4.3.4.

2

Two-Dimensional Reconstruction

The theory of reconstruction of two-dimensional functions from their projections is well-known. Detailed presentations can be found in the books of Kak and Slaney (1987), Herman (1980) and Natterer (1986) among others. Several solutions to the reconstruction problem, such as transform-based, algebraic, and statistical, have been proposed and used in practice. In this brief introduction we will concentrate on the transform-based methods in general and on the filtered backprojection method in particular. Extensions of this method will be the base for the new algorithms for three-dimensional reconstruction presented in Chapters 3 and 4.

2.1 Projections and the Fourier Slice Theorem

Let $f(x, y)$ represent the density of a two-dimensional object to be measured. The applied intensity I_0 from the X-ray tube is then attenuated by the object along the line L according to the well-known formula

$$I_{\text{out}} = I_0 e^{-\int_L f(x,y)\, dl} \tag{2.1}$$

By taking the logarithm of the relative attenuation, we obtain a line integral value of the object function as

$$\int_L f(x, y)\, dl = -\ln \frac{I_{\text{out}}}{I_0} \tag{2.2}$$

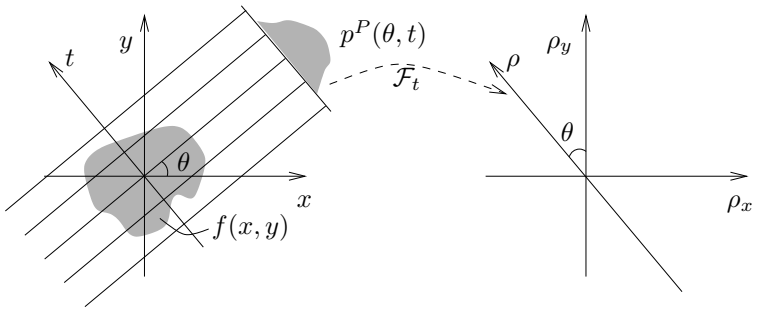


Figure 2.1 *The Fourier slice theorem. Left: A parallel projection $p^P(\theta,t)$ of the object $f(x,y)$. Right: The one-dimensional Fourier transform of the projection is found along a radial line of the two-dimensional Fourier transform of the object.*

Hence, X-ray measurements may be considered as line-integration values after the simple manipulation in (2.2).

Primary measurements in the form of line-integrals are the basis for all transform-based CT reconstruction techniques. It should be observed, however, that (2.2) describes an ideal situation. In a practical case, phenomena like beam-hardening, partial occlusion, detector sensitivity etc., changes (2.2) to an approximation. Nevertheless, for the sole purpose of discussing and investigating reconstruction algorithms (2.2) is a widely accepted model.

Consider the line L as belonging to a set of parallel lines constituting a projection shown in Figure 2.1. At projection angle θ and detector position t, the line-integral (2.2) can be written as

$$p^P(\theta,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(y\cos\theta - x\sin\theta - t)\,dxdy \tag{2.3}$$

The superscript P in (2.3) indicates that the projection is parallel. It is illuminating to imagine the projection values in the Cartesian (θ,t)-space. Edholm and Jacobsson (1975) named this space the *sinogram* since each object point contributes to projection values along a sinusoidal curve in this space.

The Fourier slice theorem states that the values of the one-dimensional Fourier transform of a parallel projection is found along a radial line in the two-dimensional Fourier transform of the object. See Figure 2.1.

$$\mathcal{F}_t p^P(\theta,\rho) = \mathcal{F}_2 f(-\rho\sin\theta, \rho\cos\theta) \tag{2.4}$$

The Fourier transforms of parallel projections from an angular interval of length π cover the complete two-dimensional Fourier space of the object. The object func-

tion may then be obtained by the straight forward inverse transformation

$$\begin{aligned} f(x,y) &= \mathcal{F}_2^{-1}\mathcal{F}_2 f(x,y) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathcal{F}_2 f(\rho_x,\rho_y) e^{j2\pi(x\rho_x+y\rho_y)}\, d\rho_x d\rho_y \end{aligned} \tag{2.5}$$

Algorithms based on (2.5) are called direct Fourier methods. A problem when using (2.5) in a practical implementation is that the Fourier slice theorem gives us samples on a polar grid while the standard inverse Fourier transform requires data on a rectangular grid. One solution to the problem is a technique known as gridding first formulated in the context of medical imaging by O'Sullivan (1985) and another is the use of linograms as investigated by Magnusson-Seger (1993). The main advantage with direct Fourier reconstruction is that no step in the algorithm requires more than $\mathcal{O}(N^2 \log N)$ calculations.

2.2 Filtered Backprojection

The most commonly used algorithm for tomographic reconstruction is filtered backprojection (FBP). As the name suggests it consists of two steps, filtering of projection data followed by backprojection (BP) . The latter can be seen as the dual, or in a more strict mathematical sense the adjoint, of projection. Instead of projecting density values to a projection value, a projection value is backprojected, or smeared out, over the image points along the ray.

Although filtered backprojection has proven very popular in real implementations of image reconstruction, it is not without its shortcomings. One is the computational complexity that slows down the reconstruction process. This is partly rectified by the fast backprojection presented in Section 2.3. Image quality can also be a problem. Small metallic objects may induce streaking artifacts on the reconstructed image. Compared to so-called algebraic reconstruction techniques, filtered backprojection is not so adaptable to missing data and partial occlusion effects. However, algebraic techniques are typically more computationally expensive and are beyond the scope of this thesis.

The computation steps in filtered backprojection depend on the ray geometry and the following two sections present the algorithm for parallel beams and fan-beams respectively. These sections were inspired by similar, but more extensive, treatments by Kak and Slaney (1987) and Schaller (1998).

2.2.1 Parallel-Beam Reconstruction

By inserting (2.4) and changing the integration variables from $d\rho_x d\rho_y$ to the polar $|\rho| d\rho d\theta$, (2.5) can be simplified to

$$\begin{aligned} f(x,y) &= \frac{1}{2}\int_0^{2\pi}\int_{-\infty}^{\infty} \mathcal{F}_t p^P(\theta,\rho) e^{j2\pi\rho(y\cos\theta - x\sin\theta)}\,|\rho| d\rho d\theta \\ &= \frac{1}{2}\int_0^{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p^P(\theta,t) e^{-j2\pi\rho t}\,dt\, e^{j2\pi\rho(y\cos\theta - x\sin\theta)}\,|\rho| d\rho d\theta \\ &= \frac{1}{2}\int_0^{2\pi}\int_{-\infty}^{\infty} p^P(\theta,t)\int_{-\infty}^{\infty} |\rho| e^{j2\pi\rho(y\cos\theta - x\sin\theta - t)}\,d\rho dt d\theta \\ &= \int_0^{2\pi}\int_{-\infty}^{\infty} p^P(\theta,t) g_\infty^P(y\cos\theta - x\sin\theta - t)\,dt d\theta \\ &= \underbrace{\int_0^{2\pi} \underbrace{(p^P * g_\infty^P)}_{\text{Filtering}}(\theta, y\cos\theta - x\sin\theta)\,d\theta}_{\text{Backprojection}} \end{aligned} \tag{2.6}$$

where the distribution

$$g_\infty^P(t) = \frac{1}{2}\int_{-\infty}^{\infty} |\rho| e^{j2\pi\rho t}\,d\rho \tag{2.7}$$

is often called a ramp-filter due to its shape in the Fourier domain. The right hand side of (2.6) is a recipe for the two steps involved in filtered backprojection. First the projection data are ramp-filtered. An image point is then reconstructed by integrating the filtered projections along a sinusoidal curve in the sinogram over all projection angles.

While the theoretical understanding of image reconstruction requires continuous mathematics, practical algorithms only work on sampled data. Most continuous variables therefore have to be translated to a corresponding discrete indexing variable. Thus, the detector which is $2t_{\text{max}}$ length units (l.u.) long is sampled at N_t equidistant position with a sampling distance of Δt. Similarly, we have N_θ projection angles sampled at a $\Delta\theta$ radian interval. As Oppenheim and Willsky (1997), we will denote sampled functions with brackets. The relationship between the continuous projection $p^P(\theta,t)$ and the sampled $p^P[i,k]$ is then given by

$$p^P[i,k] = p^P(\theta_i, t_k) \tag{2.8}$$

where

$$\theta_i = i\Delta\theta, \quad \Delta\theta = \frac{2\pi}{N_\theta}, \quad i = 0,\ldots,N_\theta - 1 \tag{2.9}$$

and

$$t_k = (k + O_k + 0.5)\Delta t - t_{\text{max}}, \quad \Delta t = \frac{2t_{\text{max}}}{N_k}, \quad k = 0,\ldots,N_t - 1 \tag{2.10}$$

where O_k is an offset constant to be discussed shortly in connection with Figure 2.3.

The sampled projections are filtered with a band-limited version of the ramp-filter $g_\infty^P(t)$, defined as

$$g^P(t) = \int_{-\frac{1}{2\Delta t}}^{\frac{1}{2\Delta t}} |\rho| e^{j2\pi\rho t}\, d\rho = \frac{1}{2(\Delta t)^2}\,\mathrm{sinc}\,\frac{2t}{2\Delta t} - \frac{1}{4(\Delta t)^2}\,\mathrm{sinc}^2\,\frac{t}{2\Delta t} \tag{2.11}$$

The sampled values of this filter are given by

$$g^P[k] = \Delta t \cdot g^P(k\Delta t) = \begin{cases} \frac{1}{4\Delta t}, & k = 0 \\ 0, & k \text{ even} \\ -\frac{1}{k^2\pi^2\Delta t}, & k \text{ odd} \end{cases} \tag{2.12}$$

Note that we have inserted a scaling factor Δt. This enables us to express the filtering of discrete data as the discrete convolution

$$\tilde{p}^P[n,k] = p^P[n,k] * g^P[k] \tag{2.13}$$

since

$$\begin{aligned} p^P(\theta_n, t_k) * g^P(t) &= \int p^P(\theta_n, t_k - t') g^P(t')\, dt' \\ &\approx \sum_{k'} p^P[n, k-k'] g^P(k'\Delta t)\Delta t \\ &= \sum_{k'} p^P[n, k-k'] g^P[k] = p^P[n,k] * g^P[k] \end{aligned} \tag{2.14}$$

For efficiency reasons, the filtering is often performed as a multiplication in the Fourier domain instead of this spatial convolution.

The reconstruction result is produced for $N_x \times N_y$ pixels with midpoints placed on a Cartesian grid around the origin as specified by

$$\begin{aligned} x_{i_1} &= (i_1 + 0.5)\Delta x - x_{\max}, \quad \Delta x = \frac{2x_{\max}}{N_x}, \quad i_1 = 0, \ldots, N_x - 1 \\ y_{i_2} &= (i_2 + 0.5)\Delta y - y_{\max}, \quad \Delta y = \frac{2y_{\max}}{N_y}, \quad i_2 = 0, \ldots, N_y - 1 \end{aligned} \tag{2.15}$$

Only pixels which are illuminated from all projection angles are reconstructed correctly, and they constitute the field of view (FOV). These pixels are are placed inside the circle of radius $R_{\text{FOV}} = t_{\max}$ around the origin.

The ramp-filtered projections $\tilde{p}^P[n,k]$ are backprojected as

$$f_{\text{FBP-P}}(x,y) = \frac{2\pi}{N_\theta} \sum_{n=0}^{N_\theta - 1} \tilde{p}^P[n, k(x,y,\theta)] \tag{2.16}$$

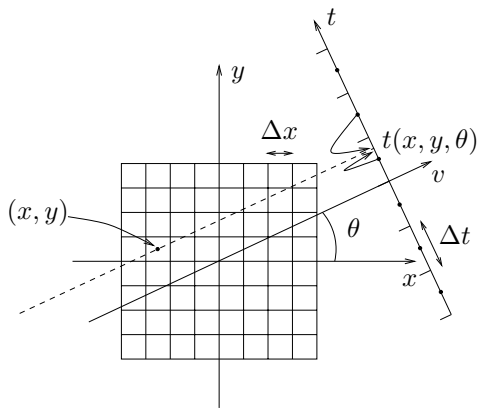


Figure 2.2 *Backprojection in the sampled geometry. A linear interpolation gives the filtered projection value to be delivered to the pixel* (x, y).

where the index $k(x, y, \theta)$ of the ray intersecting the pixel (x, y) can be derived from the continuous intersection coordinate

$$t(x, y, \theta) = y \cos\theta - x \sin\theta \tag{2.17}$$

using (2.10) to translate between the continuous and indexing variable. This ray does not usually coincide exactly with the positions of the sampled rays, calling for an interpolation in (2.16). See Figure 2.2. We will later have use of the coordinate

$$v(x, y, \theta) = x \cos\theta + y \sin\theta \tag{2.18}$$

along the ray.

The measured projection data extend over a full projection angle interval of 2π. Note from (2.3) that the ray (θ_1, t_1) gives the same measurement as $(\theta_1 + \pi, -t_1)$, rendering one of the two projections θ_1 and $\theta_1 + \pi$ redundant. However, by offsetting the detector sampling pattern a quarter unit, i.e. setting $O_k = 0.25$ in (2.10), the rays of the projections at θ_1 and $\theta_1 + \pi$ will interleave and form a beam where the rays are $\Delta t/2$ length units apart. This image quality enhancement trick is called *quarter offset* and is often used in practice when data is collected over a full turn.

For quarter offset to work optimally, the interlaced sampling patterns should be merged into one set before filtration as shown in Figure 2.3. The filter and backprojection interpolation function may then be designed for the double resolution, while the backprojection of the merged projection data is performed over the interval $\theta \in [0, \pi]$.

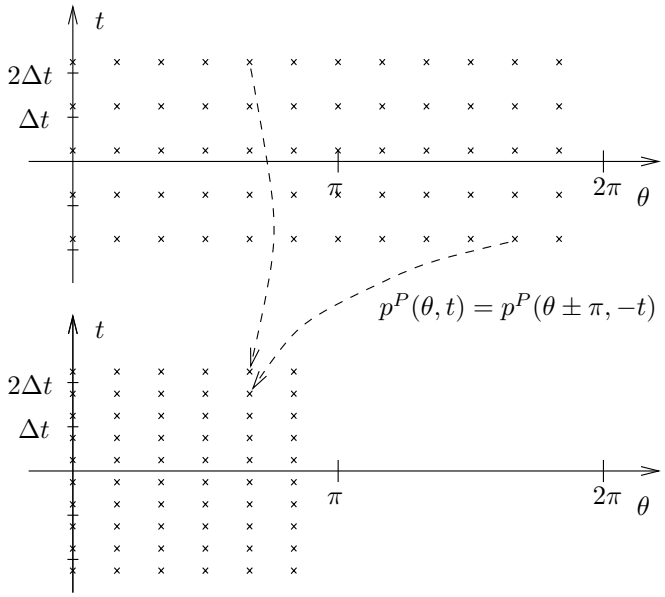


Figure 2.3 *By merging the two halves of the quarter offsetted projections, the resolution in the t-direction is doubled. Top: before merging. Bottom: after merging.*

2.2.2 Fan-Beam Reconstruction

A modern tomograph does not create parallel beams but a set of rays diverging from one and the same point, the X-ray source. See Figure 2.4. The source moves in the (x, y)-plane along a circle of radius R. At projection angle β it is positioned at the coordinate $(-R\cos\beta, -R\sin\beta)$. The detector may be flat or curved. In medical tomography it is usually placed on an circular arc with its origin on the source. From a physical point of view, this shape has the attractive property that all detector elements are on the same distance from the source. The rays form a *fan* and each ray is described by its angle to the central ray, the *fan-angle*, γ. The projection values are written as $p(\beta, \gamma.$. The sampling positions along β and γ are defined by

$$\begin{aligned} &\beta_j = j\Delta\beta, \quad \Delta\beta = \frac{2\pi}{N_\beta}, \quad j = 0, \ldots, N_\beta - 1 \\ &\gamma_o = (o + O_o + 0.5)\Delta\gamma - \gamma_{\max}, \quad \Delta\gamma = \frac{2\gamma_{\max}}{N_\gamma}, \quad o = 0, \ldots, N_\gamma - 1 \end{aligned} \tag{2.19}$$

Lakshminarayanan (1975) and Herman and Naparstek (1977) modified the filtered backprojection formula for the parallel geometry into a formula for the fan-beam geometry. The algorithm differs on the following three points:

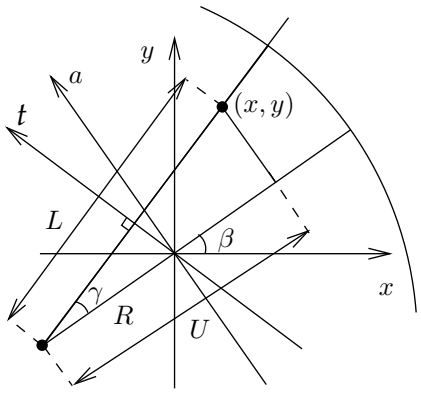


Figure 2.4 *A divergent projection.*

- The first step is a pre-weighting with $\cos\gamma$.
- The rampfilter

$$g_\infty(\gamma) = \left(\frac{\gamma}{\sin\gamma}\right)^2 g_\infty^P(\gamma) \tag{2.20}$$

  is a modified version of the original $g_\infty^P(t)$ for parallel projections. Together with the pre-weighting this yields

$$\tilde{p}(\beta,\gamma) = (p(\beta,\gamma)\cos(\gamma)) * g_\infty(\gamma) \tag{2.21}$$

- The backprojection

$$f_{\text{FBP}}(x,y) = \int_0^{2\pi} \frac{R^2}{L(x,y,\beta)^2}\tilde{p}(\beta,\gamma(x,y,\beta))\,d\beta \tag{2.22}$$

  includes a space dependent factor, L^{-2}, calculated from the source-point distance

$$L(x,y,\beta) = \sqrt{(R + x\cos\beta + y\sin\beta)^2 + (-x\sin\beta + y\cos\beta)^2} \tag{2.23}$$

  The ray intersecting the pixel in (2.22) is given by

$$\gamma(x,y,\beta) = \arctan\frac{-x\sin\beta + y\cos\beta}{R + x\cos\beta + y\sin\beta} \tag{2.24}$$

If a *flat* detector is used instead the rays are not sampled equiangularly but equidistantly along the a-axis in Figure 2.4. This places the detector on the axis of rotation, the z-axis, which of course is physically infeasible since the object

to be measured should be positioned there, and we may therefore refer to the detector as a *virtual* detector. The detector coordinate on an actual detector placed further away from the source is then obtained by a pure scaling of a. We write the projection data as $p^F(\beta, a)$ where the superscript F stands for flat. They are related to the the equiangular projections as

$$p^F(\beta, a) = p(\beta, \arctan\frac{a}{R}) \quad \text{and} \quad p(\beta, \gamma) = p^F(\beta, R\tan\gamma) \tag{2.25}$$

Reconstruction by filtered backprojection for this geometry requires that the data are pre-weighted and ramp-filtered as

$$\tilde{p}^F(\beta, a) = \left(p^F(\beta, a)\frac{R}{\sqrt{R^2+a^2}}\right) * g^P(a) \tag{2.26}$$

Note that the ramp-filter is the same as for the parallel beam. The backprojection is similar to (2.22) but with a different weighting function.

$$f_{\text{FBP-F}}(x, y) = \int_0^{2\pi} \frac{R^2}{U(x, y, \beta)^2}\tilde{p}^F(\beta, a(x, y, \beta))\, d\beta \tag{2.27}$$

The detector position is given by

$$a(x, y, \beta) = R\frac{-x\sin\beta + y\cos\beta}{R + x\cos\beta + y\sin\beta} \tag{2.28}$$

The weighting in (2.27) is a function of the distance $U(x, y, \beta)$ between the source and the line parallel with the detector that intersects the image point (x, y) as shown in Figure 2.4. This distance can be expressed as

$$U(x, y, \beta) = R + x\cos\beta + y\sin\beta \tag{2.29}$$

2.2.3 Short-Scan Reconstruction

The previous two sections presented two main categories of beam geometries, parallel and fan-beam. The relationship between the two parameterisations (θ, t) and (β, γ), seen in Figure 2.4, can be expressed as

$$\theta = \beta + \gamma, \quad t = R\sin\gamma \tag{2.30}$$

or equivalently

$$\beta = \theta - \gamma, \quad \gamma = \arcsin\frac{t}{R} \tag{2.31}$$

If a flat detector is used in the fan-beam case the γ above should be replaced by $\gamma = \arctan\frac{a}{R}$.

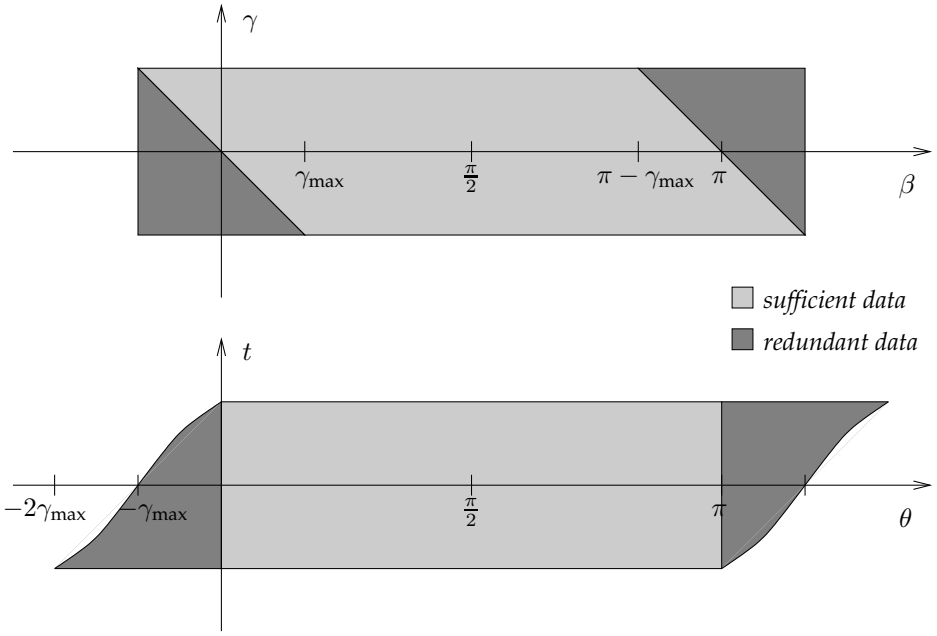


Figure 2.5 *Data measured in short-scan acquisition. Top: fan-beam sinogram for data from a curved detector. Bottom: sinogram for parallel data containing rebinned fan-beam data. The lightly shaded parts indicate sufficient data for reconstruction.*

When we derived the filtered backprojection formula in (2.6) we saw that parallel data from a projection interval $\theta \in [0, \pi]$ is sufficient for reconstruction. Using (2.31) we deduce that this corresponds to a fan-beam projection interval of $\beta \in [-\gamma_{\max}, \pi + \gamma_{\max}]$ as shown in Figure 2.5. Such a projection interval is referred to as short-scan acquisition as opposed to full-scan where the interval is $\beta \in [0, 2\pi]$. The short-scan projection interval can of course start at any angle as long as it is of length $\pi + 2\gamma_{\max}$. A short-scan acquisition measures some redundant rays at the beginning and ending of the projection interval. The redundancy increases with the fan-angle $\gamma_{\max}$.

Zeng and Gullberg (1991) list three possible approaches to reconstructing images from short-scan fan-beam data. They all play important roles in Chapter 4 on helical cone-beam reconstruction. A good understanding of the two-dimensional case is necessary for the discussions of the cone-beam case. The three methods are named parallel rebinning, complementary rebinning, and smooth sinogram windowing.

Parallel Rebinning

Using (2.31), a parallel projection set can be constructed from the measured fan-beam data as

$$p^P(\theta, t) = p(\theta - \arcsin\frac{t}{R}, \arcsin\frac{t}{R}) \tag{2.32}$$

This construction is referred to as *rebinning*, but it is not simply a rearrangement of the data since an interpolation is always necessary. Redundant information in the input data is discarded. The reconstruction is then made using the parallel filtered backprojection algorithm, described by (2.13) and (2.16), over the interval $\theta \in [0, \pi]$.

Complementary Rebinning

In the above discussion on quarter offset we noted that parallel data contains duplicate projection values as

$$p^P(\theta, t) = p^P(\theta \pm \pi, -t) \tag{2.33}$$

Combining this with (2.31), we derive a similar relation for fan-beam data as

$$p(\beta, \gamma) = p^P(\beta + \gamma, R\sin\gamma) = p^P(\beta + \gamma \pm \pi, -R\sin\gamma) = \\ p(\beta + 2\gamma \pm \pi, -\gamma) \tag{2.34}$$

which can be used to fill out the missing parts of a short-scan fan-beam sinogram. Because more than half of the fan-beam sinogram is measured, the missing part

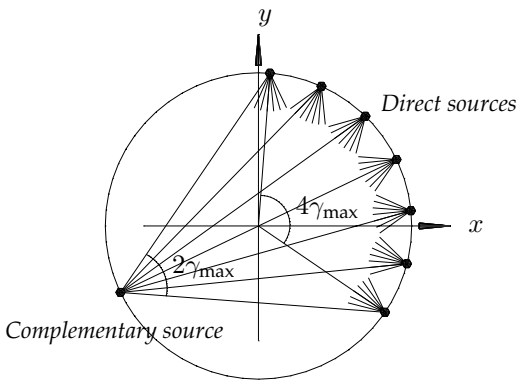


Figure 2.6 *A complementary projection is obtained from the original direct projections. [Based on a similar illustration by Schaller (1998)]*

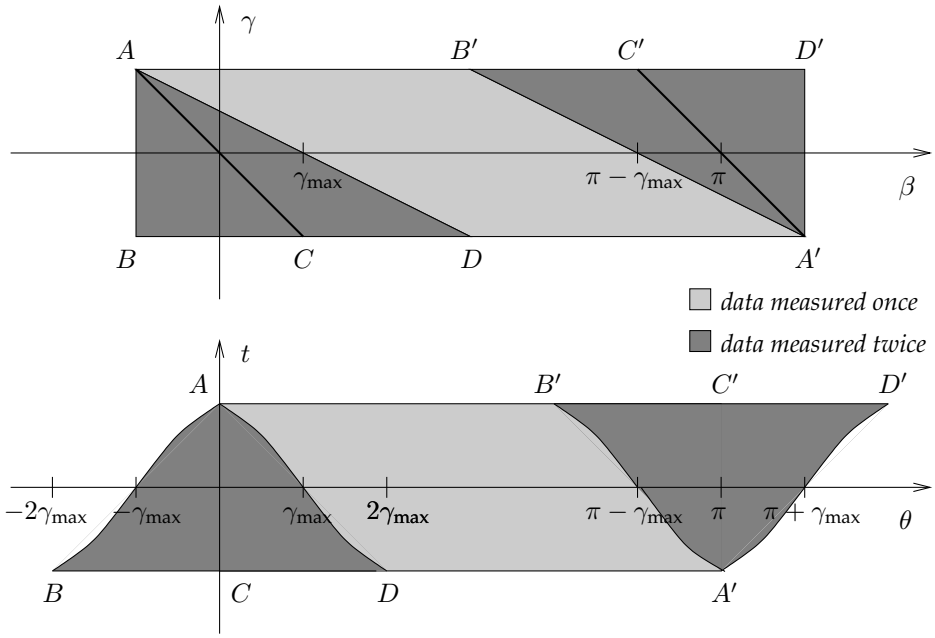


Figure 2.7 *Smooth sinogram windowing. Primed and unprimed letters correspond to identical rays, measured in differnt directions. The darkly shaded areas consist of data measured twice. Top: fan-beam sinogram. Bottom: parallel beam sinogram.*

only extends over $\pi + \gamma_{\text{max}} \leq \beta \leq 2\pi - \gamma_{\text{max}}$. The data for this region are found in the lightly shaded parallelogram $AB'A'D$ in Figure 2.7.

The complementary rebinning is followed by a full-scan fan-beam filtered backprojection as described by (2.21) – (2.23). The set of rays for a specific projection angle β constructed from (2.34) still form a fan as shown in Figure 2.6, referred to as a *complementary* projection. The rays from a complementary projection all stem from a *complementary source*. We will refer to the original non-complementary rays as *direct*.

Complementary rebinning can be performed without any interpolation, provided that $\Delta\gamma = \Delta\beta/2$. This sampling situation is illustrated in Figure 2.6. For a maximum fan-angle of $\gamma_{\text{max}} = \pi/6$, this implies that $N_\beta = 6N_\gamma$. Actual tomographs on the market take substantially fewer projections per turn.

(a) Full-scan reconstruction

(b) Short-scan reconstruction, binary windowing before rampfiltering

(c) Short-scan reconstruction, smooth windowing before rampfiltering

(d) Short-scan reconstruction, smooth windowing after rampfiltering

Figure 2.8 *Fan-beam reconstruction of the Shepp-Logan phantom.* $N_\beta = 250$, $\gamma_{max} = 30°$. *Greyscale interval [1.0, 1.04].*

Smooth Sinogram Windowing

The short-scan fan-beam data can be windowed before filtering and backprojection so that the projection values of rays measured twice are weighted and normalised to unity. The simplest window function is binary, which removes all data outside a non-redundant area of the sinogram. Examples of such non-redundant areas in Figure 2.7 are $AC'A'C$, $AB'A'B$, and $AD'A'D$. However, if the binary truncated data is filtered and backprojected, heavy artifacts will appear due to filtering over the sharp edge along the borders of the non-redundant area. See Figure 2.8(b). Parker (1982) suggested the following smooth windowing function to eliminate these artifacts:

$$w(\beta,\gamma) = \begin{cases} \sin^2\left(\frac{\pi}{4}\frac{\beta+\gamma_{\max}}{\gamma_{\max}-\gamma}\right) & ADB : -\gamma_{\max} \le \beta \le \gamma_{\max} - 2\gamma \\ 1 & AB'A'D : \gamma_{\max} - 2\gamma \le \beta \le \pi - \gamma_{\max} - 2\gamma \\ \sin^2\left(\frac{\pi}{4}\frac{\pi+\gamma_{\max}-\beta}{\gamma_{\max}+\gamma}\right) & B'D'A' : \pi - \gamma_{\max} - 2\gamma \le \beta \le \pi + \gamma_{\max} \end{cases} \quad (2.35)$$

This function has the value 0 along the lines AB and $D'A'$, the value 0.5 along the lines AC and $A'C'$ and the value 1 along the lines AD and $B'A'$. It furthermore has continuous partial derivatives over the complete region. The only discontinuities are in the points A and A'. However, if the detector width is somewhat larger than the radius of the measured object, these discontinuities cause no problem. Again, let us emphasise that it is necessary to apply the weight function *before* rampfiltering. Figures 2.8(c) and (d) show the substantial difference in image quality for these two cases.

None of the three approaches are perfect. The interpolation necessary in the two rebinning approaches introduces errors, no matter how carefully it is per-

formed. The smooth sinogram windowing always affects the filtering somewhat, no matter how smooth it is. The parallel rebinning approach discard redundant information, which affects the noise behaviour of the reconstruction. An important practical advantage of the parallel rebinning approach is that it is possible to drastically optimise software or hardware implementations of the parallel backprojection for speed.

It is possible to perform a variation of parallel rebinning where no interpolation is used, but the data is purely re-organised. Due to the $\sin\gamma$-term in (2.30), the resulting parallel beam then consists of non-equidistantly sampled parallel rays, sampled somewhat more densely at its outer parts. Besson (1996) calls this beam a fan-parallel beam. The reconstruction approach is then to filter and backproject these beams and thereby obtaining a somewhat sharper image thanks to the avoided interpolation. Besson (1996) showed that there exists no shift-invariant ramp-filter that can be applied to the fan-parallel beam, but he later presented an approximate shift-invariant filter (Besson 1998) that works well.

2.3 Fast Backprojection

An $N \times N$ image requires $\mathcal{O}(N^3)$ backprojection summation steps which makes the backprojection the most time-consuming part of the reconstruction process. Brandt, Mann, Brodski, and Galun (1999), Basu and Bressler (2000), Brady (1998) and Nilsson (1996) have seemingly independently invented fast backprojection algorithms, which decrease this complexity to $\mathcal{O}(N^2 \log N)$. Here, we will give an introduction of the fast backprojection formulated by Danielsson (1997). A more extensive treatment is given by Ingerhed (1999).

2.3.1 Links and the Basic Step

The main idea behind fast backprojection is to perform the summation in (2.16) step by step by recursively splitting it into intermediate sums, which may then be used in several computations in the next step. Nilsson (1996, 1997) describes this recursive process with the help of the image space, but as Brady (1998) and Danielsson (1997), we prefer to present the algorithm in the projection space, the sinogram. Here, backprojection consists of a summation of filtered projection values along a sinusoid for each pixel to be computed. The divide-and-conquer strategy of the algorithm is to approximate this summation as the sum of summations along shorter curves, which in their turn have been calculated from even shorter curves, and so on. Danielsson (1997) introduced the term *link* for these sinusoidal curves. We will only consider backprojection for parallel beams, since it is cumbersome to include the L^{-2}-factor necessary in fan-beam backprojection.

We denote the link between the grid points (θ_{n_1}, t_{k_1}) and (θ_{n_2}, t_{k_2}) as

$$(\theta_{n_1}, t_{k_1}; \theta_{n_2}, t_{k_2}) \tag{2.36}$$

and its associated value as

$$\tilde{I}[n_1, k_1; n_2, k_2] \tag{2.37}$$

We use the tilde to indicate that the link value is calculated from ramp-filtered data. The value will be computed from the values of shorter links in the basic step of the algorithm described below.

Geometrically, we may regard a point in the projection space as a ray in the image space, and hence the two end-points of the link represent two rays that intersect at an image point. Specifically, the link $(\theta_{n_1}, t_{k_1}; \theta_{n_2}, t_{k_2})$ corresponds to the image point

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -\sin\theta_{n_1} & \cos\theta_{n_1} \\ -\sin\theta_{n_2} & \cos\theta_{n_2} \end{pmatrix}^{-1} \begin{pmatrix} t_{k_1} \\ t_{k_2} \end{pmatrix} \tag{2.38}$$

Note that the two end points of the sinusoid uniquely define the link provided that

$$\theta_{n_2} \neq \theta_{n_1} + i\pi, \quad i \in \mathbb{Z} \tag{2.39}$$

The image point (x, y) will describe a sinusoid in the sinogram. The link value $\tilde{I}(n_1, k_1; n_2, k_2)$ should ideally be identical to

$$\int_{\theta_{n_1}}^{\theta_{n_2}} \tilde{p}(\theta, y\cos\theta - x\sin\theta)\, d\theta \tag{2.40}$$

with (x, y) given by (2.38). As mentioned, we will formulate a recursive calculation of the link value instead of using (2.40) directly.

We define the length of a link as the θ-difference of its end points. We allow ourselves to indicate the length in both radians (e.g. a $\frac{\pi}{2}$-link) and indexing units (e.g. a 2-link).

The basic step of the algorithm calculates a link value from the values of four links of half the length. Consider the link $(\theta_{n_1}, t_{k_1}; \theta_{n_2}, t_{k_2})$ in Figure 2.9. We will use four half-length links from θ_{n_1} to $\theta_{n_{\text{mid}}}$ and from $\theta_{n_{\text{mid}}}$ to θ_2, where n_{mid} is the index of the middle projection angle given by

$$n_{\text{mid}} = \frac{n_1 + n_2}{2} \tag{2.41}$$

If we choose the number of projections N_θ to be a power of 2 it can be easily shown that n_{mid} is always an integer. We derive the index of the t-position of the link for

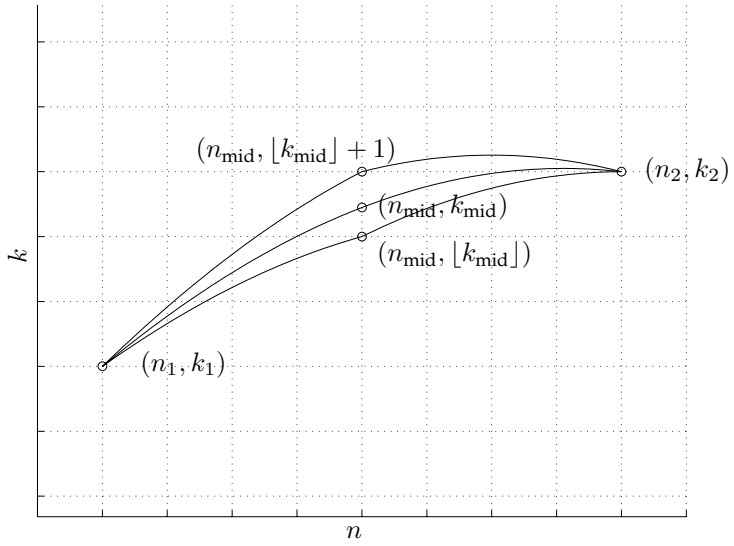


Figure 2.9 *The basic step, a link value is constructed from the values of four links of half the length.*

this middle projection angle using (2.17) as

$$k_{\text{mid}} = \frac{k_1 + k_2}{2\cos(\frac{\theta_{n_1} - \theta_{n_2}}{2})} \tag{2.42}$$

Note that all links have their end-points on the grid points of the discrete projection space, whereas k_{mid} may be non-integer, which calls for an interpolation. Using linear interpolation we formulate the basic equation of the algorithm as

$$\begin{aligned} \tilde{I}[n_1, k_1; n_2, k_2] = w \cdot \big(\tilde{I}[n_1, k_1; n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor] + \tilde{I}[n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor; n_2, k_2]\big) + \\ w' \cdot \big(\tilde{I}[n_1, k_1; n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor + 1] + \tilde{I}[n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor + 1; \check{n}_2, k_2]\big) \end{aligned} \tag{2.43}$$

where the interpolation weights w and w' are simply calculated as

$$w = 1 - w', \quad w' = k_{\text{mid}} - \lfloor k_{\text{mid}} \rfloor \tag{2.44}$$

The correspondence between (2.43) and Figure 2.9 should be obvious. Danielsson (1997) showed that the relative intercept relation between the shorter links and the long link are constant throughout the $[\theta_{n_1}, \theta_{n_2}]$ interval, which motivates (2.44) even further. It was also shown what follows from (2.42), namely that t_m is identical for all links with identical $t_1 + t_2$ and identical $\theta_1 - \theta_2$. If a table is used to store the pre-computed interpolation weights, this observation can be used to reduce the size of that table from $\mathcal{O}(N_\theta^2 N_t^2)$ to $\mathcal{O}(N_\theta N_t)$.

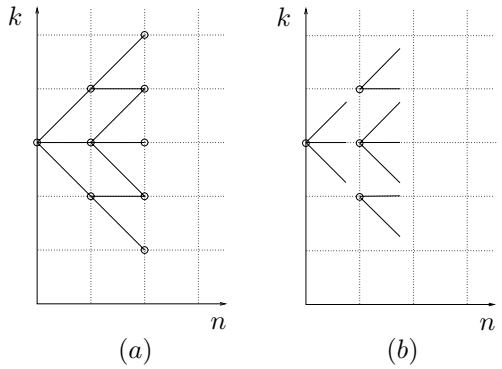


Figure 2.10 *The* 1*-links needed for a* 2*-tree. (a) The value of a* 1*-link is calculated as the average of the end-point values. (b) the value of a* 1*-link is equal to the value of its left end-point.*

Neglecting the link curvature, we may regard the short links as line segments. This simplifies the calculation of k_{mid} in (2.42) to $k_{\text{mid}} = (k_1 + k_2)/2$, resulting in rational interpolation weights w and w', many of which are zero-valued. These attractive weights speed up the basic step for short links considerably. Experiments by Danielsson and Ingerhed (1997) have shown that this simplification does not influence the image quality when used for links shorter than approximately $\sqrt{N_\theta}$, which means that half of the $\lg N_\theta$ steps in the backprojection can take advantage of this simplification.

2.3.2 A Complete Algorithm

The algorithm starts with calculating all the necessary 1-links. At first sight, the value $\tilde{I}[n_1, k_1; n_1 + 1, k_2]$ could be assigned the average of the filtered projection data at the link end-points as $(\tilde{p}^P[n_1, k_1] + \tilde{p}^P[n_1 + 1, k_2])/2$, illustrated in Figure 2.10(a). But since the end-points coincide with end-points of the links to the left and right, it is sufficient and more efficient to only consider one end-point per link, shown in Figure 2.10(b), giving

$$\tilde{I}[n_1, k_1; n_1 + 1, k_2] = \tilde{p}^P[n_1, k_1] \tag{2.45}$$

for all k_2.

The next step of the algorithm constructs 2-links using the basic step described in (2.43). After $\lg N_\theta - 2$ steps, we have constructed the necessary $\frac{N_\theta}{4}$-links (or equivalently, $\frac{\pi}{2}$-links), which are the longest links we are able to produce due to the restriction in (2.39).

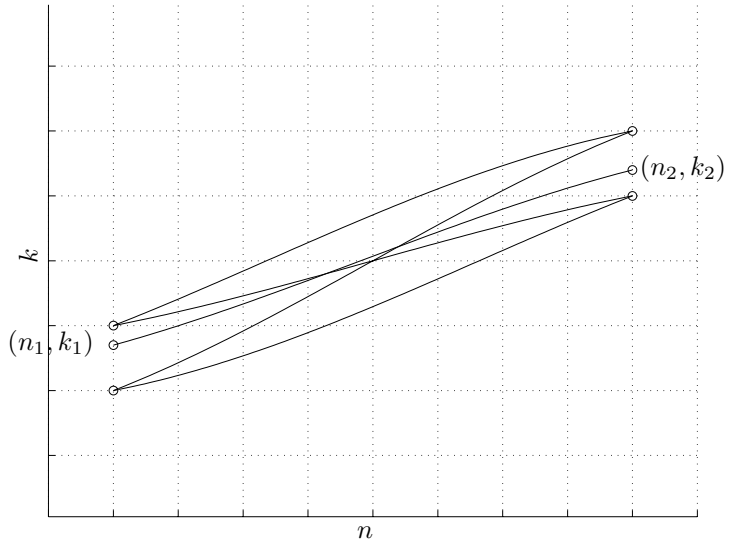


Figure 2.11 *A contribution to the pixel value is obtained by interpolation of the values of four links of length* $\frac{\pi}{2}$.

If we set the pixel resolution $\Delta x = \Delta y$ equal to the detector resolution Δt and the offset $O_k = 0$ in (2.10), the k-axis coincides with the i_1- or the i_2-axes for $\theta = 0, \pi/2, \ldots, 2\pi$. The final step of the algorithm is then a summation without any interpolation of four $\frac{\pi}{2}$-links per pixel.

With quarter offset or arbitrary detector and pixel resolutions, an interpolation in the t-direction has to be performed, utilising the four $\frac{\pi}{2}$-links per $\frac{\pi}{2}$-interval shown in Figure 2.11. The interpolation used is

$$\begin{aligned} f_{\text{FAST}}[i_1, i_2] = & w_1 w_2 \tilde{I}[n_1, \lfloor k_1 \rfloor; n_2, \lfloor k_2 \rfloor] + \\ & w_1 w_2' \tilde{I}[n_1, \lfloor k_1 \rfloor; n_2, \lfloor k_2 \rfloor + 1] + \\ & w_1' w_2 \tilde{I}[n_1, \lfloor k_1 \rfloor + 1; n_2, \lfloor k_2 \rfloor] + \\ & w_1' w_2' \tilde{I}[n_1, \lfloor k_1 \rfloor + 1; n_2, \lfloor k_2 \rfloor + 1] + \\ & \text{[similar expressions for the other three quadrants]} \end{aligned} \tag{2.46}$$

where

$$\begin{aligned} w_1 &= 1 - w_1', \quad w_1' = k_1 - \lfloor k_1 \rfloor \\ w_2 &= 1 - w_2', \quad w_2' = k_2 - \lfloor k_2 \rfloor \end{aligned} \tag{2.47}$$

Since an interpolation is performed in every step of the algorithm, the net result differs from traditional backprojection where the interpolation is only made

once for each point along the sinusoid. This *iterative interpolation* results in an actual interpolation kernel that is shift-variant and wider than the linear kernel of (2.44). This actual kernel was first discussed in Danielsson (1997) and experiments by Ingerhed and Danielsson (1998) show that it introduces a slight smoothing in the resulting image. This is generally unwanted but in the special application of Kreuder, Grass, Rasche, Braunisch, and Dössel (1999) it decreased a ripple effect apparent when traditional backprojection was used. We will examine the actual kernel and its implications on image quality closer in Section 3.4, when we extend the algorithm to cone-beam data. Brandt, Mann, Brodski, and Galun (1999) describe a post-processing de-blurring method that reduces the smoothing due to iterative interpolation.

2.3.3 Complexity Analysis

The links of the same length, l, starting from the same point, (θ_{n_1}, t_{k_1}), form a *tree*. The *length* of a tree is the length of its links. The *width* of a tree is its number of links, which is determined by the maximum and minimum slope of the sinusoids in the actual region of the projection space. At the borders of the projection space, the trees may be heavily pruned. In order for the interpolation in the basic step to work, the shorter trees have to be somewhat wider than the longer ones. Neglecting these effects we may assume that all trees employed in a certain step of the algorithm have the same width. Typically, the 1-trees have a width of 3 or 5, the 2-trees a width of 5 or 9, etc. With a slight but conservative approximation we may assume that the 1-trees consist of c links, the 2-trees consist of $2c$ links, the 4-trees of $4c$ links, and so on.

Type	Quantity	ADD	MULT	MFLOPs	Links per tree
Pixels	51468	15	16	1595508	
128-links	207412	3	2	1037060	202.6
64-links	296376	3	2	1481880	144.7
32-links	327184	3	2	1635920	79.9
16-links	347104	3	2	1735520	42.4
8-links	377408	3	2	1887040	23.0
4-links	442752	3	2	2213760	13.5
2-links	571136	3	2	2855680	8.7
Total	3416732			14442368	

Table 2.1 *Calculation of FLOPs when* $N = N_t = N_x = N_y = N_\theta/2 = 256$.

There are half as many 2-trees as 1-trees since there are half as many n-positions for them to start from. The total number of links created in step $i = 2$ to step $i = \lg N_\theta - 1$ is then

$$\sum_{i=2}^{\lg N_\theta - 1} \underbrace{N_t N_\theta 2^{1-i}}_{\text{Number of trees}} \cdot \underbrace{c2^{i-1}}_{\text{Links per tree}} = cN_t N_\theta (\lg N_\theta - 2) \in \mathcal{O}(N^2 \log N) \tag{2.48}$$

Note that because of the technique introduced by (2.45) the 1-links come without any extra storage space besides the original projection data.

The final performance of an algorithm is dependent on numerous factors, which makes it difficult to compare two different algorithms. Even so, it seems worthwhile to estimate the number of floating point operations (FLOPs) on the projection data for the algorithm. The 1-links require no calculations, whereas each basic step needs 3 additions and 2 multiplications, and the final pixel interpolation needs 4 multiplications and 3 additions per $\frac{\pi}{2}$-segment followed by 3 additions to sum the segments, totalling 16 multiplications and 15 additions.

Traditional backprojection with linear interpolation needs 2 multiplications and 2 additions for each summation step, and if only the pixels inside the circle of radius $t_{\max}$ are reconstructed, the total number of backprojection FLOPs becomes

$$4 \quad \underbrace{N_x N_y \frac{\pi}{4}}_{\text{Pixels inside FOV}} \quad N_\theta = 2\pi N^3 \tag{2.49}$$

We have implemented the fast backprojection using a top-down approach as explained later in Section 3.4.1, which performs a maximal pruning so that no unnecessary link values are calculated. Table 2.1 shows the number of links used by the program and how we estimate the number of FLOPs needed. It also shows the average number of links per tree, which, just as the theory predicted, is almost doubled in each step. Table 2.2 shows estimates made in a similar manner for different values of N. The number of FLOPs for fast backprojection is according to this table approximately $35N^2 \log N$. This is in good correspondence with the experiments of Ingerhed (1999).

N	32	64	128	256	512	1024
Trad.	$2 \cdot 10^5$	$2 \cdot 10^6$	$1 \cdot 10^7$	$1 \cdot 10^8$	$8 \cdot 10^8$	$6 \cdot 10^9$
Fast	$1 \cdot 10^5$	$7 \cdot 10^5$	$3 \cdot 10^6$	$1 \cdot 10^7$	$6 \cdot 10^7$	$3 \cdot 10^8$

Table 2.2 *Number of FLOPs on projection data in traditional and fast two-dimensional backprojection as a function of $N = N_x = N_y = N_t = N_\theta/2$.*

3

Circular Cone-Beam Reconstruction

The two-dimensional algorithms discussed in the previous chapter can reconstruct a *slice* of the measured object. If a volume segment needs to be reconstructed, the complete procedure must be performed slice-by-slice with a small movement of the object or of the source-detector system between each slice. A more efficient acquisition setup for volumetric CT is to use a two-dimensional detector. The rays then form a cone with its base on the detector and its apex on the source. An X-ray source naturally produces a cone of rays, so cone-beam acquisition not only increases the scanning speed, but also makes better use of the emitted rays otherwise removed by collimation.

3.1 Exact Methods

Given noise-free data, an exact reconstruction algorithm produces a result where the difference to the real object can be made arbitrarily small by increasing the detector resolution and the number of projections. The derivation of the two-dimensional filtered backprojection in (2.5) proves the exactness of that particular algorithm. As this section will show, there also exist exact algorithms for cone-beam data. In some cases these exact algorithms have some important shortcomings and we will therefore also look at approximate algorithms, starting from Section 3.2 and onwards.

The following subsections will discuss a simple condition on when exact reconstruction is possible. It turns out that the circular geometry does not fulfil this

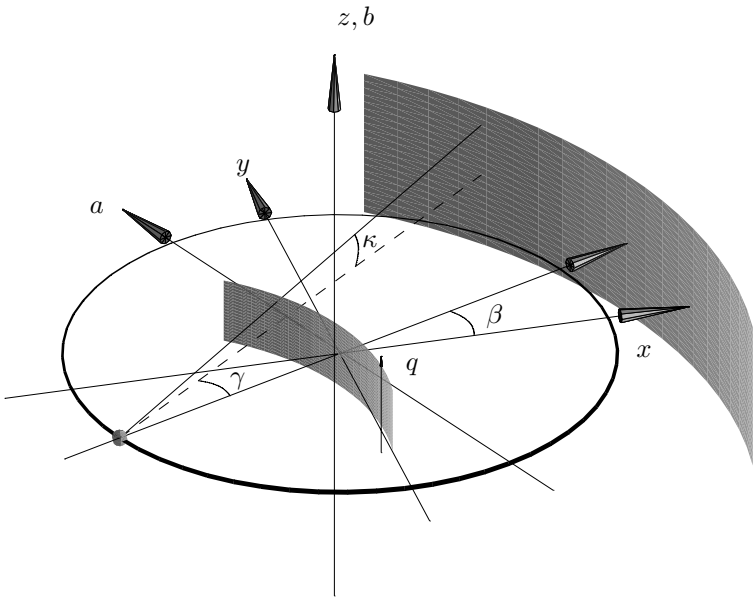


Figure 3.1 *The cone-beam geometry with the cylindrical (γ, q) detector. The axes of the planar (a, b) detector are also drawn.*

condition. A supplementary trajectory, such as a line or an extra circle, is required. Hence, in a strict sense this section on exact methods does not quite belong to this chapter on circular cone-beam reconstruction.

3.1.1 The Cone-Beam Geometry

For a cone-beam we define the projection angle β and fan-angle γ as before in (2.19). See Figure 3.1. The data is normally collected either on a planar or cylindrical detector. Both detector shapes are used in both medical and industrial applications. Cylindrical detectors are used in ordinary medical tomographs. We will show how the different reconstruction algorithms can be modified to work on data from either of the two detector shapes.

The actual radius of the cylindrical detector is not essential for our discussion and we choose to set it equal to the source trajectory radius, R, which results in the detector coordinates (γ, q) in Figure 3.1. Similiarly, we place a planar detector plane (a, b) on the axis of rotation so that the b-axis of this detector coincides with the z-axis. Both of these detectors may therefore be considered as virtual. The data from the two detectors are placed in $p(\beta, \gamma, q)$ and $p^F(\beta, a, b)$ respectively.

The sampling of the rows of the cylindrical detector is defined as

$$q_m = (m + 0.5)\Delta q - q_{\max}, \quad \Delta q = \frac{2q_{\max}}{N_q}, \quad m = 0, \ldots, N_q - 1 \tag{3.1}$$

where $2q_{\max}$ is the height of the virtual detector. The sampling of the planar detector is defined in a similar way. The relationship between the two detector coordinate systems is given by

$$a = R\tan\gamma \quad \text{and} \quad b = \frac{q}{\cos\gamma} \tag{3.2}$$

The cone-angle, κ is defined as

$$\kappa = \arctan\frac{q}{R} = \arctan\frac{b}{\sqrt{R^2 + a^2}} \tag{3.3}$$

It is constant along the rows of the cylindrical detector, but varying along the rows of the planar detector.

3.1.2 The Three-Dimensional Radon Transform

Exact three-dimensional reconstruction algorithms are usually based on the three-dimensional Radon transform. A three-dimensional Radon value is a plane integral in the object domain. Each plane can be represented by a unique point in the object space. This point is the intersection of the plane and its normal passing the origin. The three-dimensional Radon space consists of all Radon values placed at the corresponding points.

The Radon values of all planes intersecting the object have to be known in order to perform an exact reconstruction. The Tuy-Smith sufficiency condition (Tuy 1983) states that exact reconstruction is possible if all planes intersecting the object also intersect the source trajectory at least once. This condition makes intuitive sense, since the source must be positioned in a plane in order to measure its integral.

We observe that the circular trajectory does not satisfy the Tuy-Smith condition since a plane parallel to the trajectory may intersect the object but not the trajectory. It is therefore necessary to extend the trajectory with an extra circle or line if exact reconstruction is required. The available Radon data from a circular trajectory is confined within a torus shown in Figure 3.2. The area with missing Radon data is usually referred to as a shadow zone.

Note that the arguments above assume that the plane integrals are taken over complete planes, i.e. planes with no boundaries and infinite extensions. This is only practically possible if the object has a limited extension and the detector fully covers the projection of the object and the attenuation values outside the object can

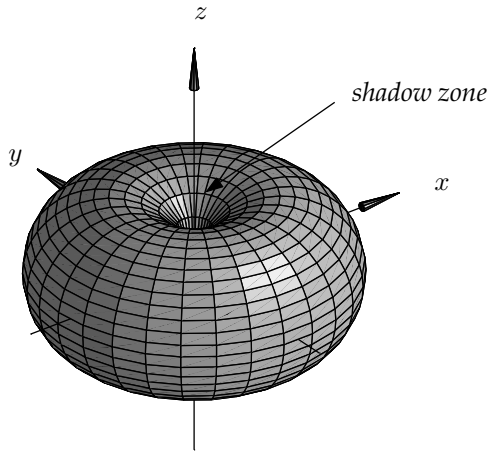


Figure 3.2 *The circular trajectory measures Radon values within a torus. The unmeasured data are positioned in a shadow-zone around the z-axis.*

be assumed to be zero. Such projection data are *untruncated*. In medical tomography it is not feasible to build a detector fully covering the patient in all directions. The detector delivers untruncated data in the fan-direction, but truncated data in the longitudinal direction. In Section 4.1 we discuss possibilities for exact cone-beam reconstruction with truncated data, but for now we assume the data to be untruncated.

3.1.3 Reconstruction Algorithms

If parallel projection data are available, a plane integral is obtained from one-dimensional integration of projection data along a line on the planar detector. Unfortunately, as we have seen, the source produces divergent cone-beams and a plane integral is not equal to a one-dimensional integral of divergent projection data. Grangeat (1987) showed how the radial derivative of the Radon transform can be calculated from the divergent projections and how an exact reconstruction can proceed from these intermediate values.

Defrise and Clack (1994) reformulated Grangeat's method into a filtered back-projection where the filtering is two-dimensional and shift-variant. One advantage of this approach is that each projection image may be filtered and backprojected once it has been acquired. The Grangeat method has to fully construct the radial derivative of the Radon space using all projection data before the actual reconstruction can start. Jacobson (1996) gives a lucid presentation of the above methods together with investigations of how they may be efficiently implemented using linogram techniques.

3.2 The FDK Method

Feldkamp, Davis, and Kress (1984) describe an approximate reconstruction algorithm for circular cone-beam tomography. We will call this algorithm the FDK method. It is approximate in the sense that the reconstruction result will deviate somewhat from the measured object regardless of the measurement resolution. For moderate cone-angles, these differences are however small and often acceptable. The simplicity of the FDK method has made it the most used algorithm for cone-beam reconstruction. Another important advantage, discussed in Section 3.2.3, is that FDK, in contrast to the exact methods, handles data truncated in the longitudinal direction.

3.2.1 Reconstruction from Planar Detector Data

In the original form, FDK assumed the data to come from a planar detector. The reconstruction is a filtered backprojection very similar to the two-dimensional algorithm presented in Section 2.3. Unlike the exact cone-beam filtered backprojection of Defrise and Clack (1994) where the filtering is two-dimensional, the projection data in the FDK method are handled row by row. In addition to the convolution with the ramp-filter $g^P(\gamma)$, a pre-weighting factor, dependent on both the fan-angle and on the cone-angle, is applied, yielding

$$\tilde{p}^F(\beta, a, b) = \left(\frac{R}{\sqrt{R^2 + a^2 + b^2}} p^F(\beta, a, b) \right) * g^P(a) \tag{3.4}$$

The pre-weighting factor is geometrically interpreted as the cosine of the angle between the ray and the central ray of the projection. It can be factorised into the two cosine factors of the fan- and cone-angle as

$$\frac{R}{\sqrt{R^2 + a^2 + b^2}} = \frac{R}{\sqrt{R^2 + a^2}} \frac{\sqrt{R^2 + a^2}}{\sqrt{R^2 + a^2 + b^2}} = \cos\gamma \cos\kappa \tag{3.5}$$

The pre-weighted and filtered projections are backprojected into the reconstruction volume as

$$f_{\text{FDK}}(x, y, z) = \int_0^{2\pi} \frac{R^2}{U(x, y, \beta)^2} \tilde{p}^F(\beta, a(x, y, \beta), b(x, y, z, \beta))\, d\beta \tag{3.6}$$

where $a(x, y, \beta)$ is given by (2.28) and

$$b(x, y, z, \beta) = z \frac{R}{R + x\cos\beta + y\sin\beta} \tag{3.7}$$

In the discrete case, the integral is naturally replaced by a sum over the projection angles. A two-dimensional interpolation of the filtered projection data is then required for each term of the backprojection sum.

The factor U^{-2} in (3.6) is identical to the factor (2.29) in fan-beam backprojection. Hence, it is independent of the z-coordinate of the voxel and only depends on the distance between the source and the reconstructed voxel projected onto the central ray as

$$U(x, y, \beta) = R + x\cos\beta + y\sin\beta \tag{3.8}$$

It is worth noting that this cone-beam backprojection is identical to the one used in the exact method by Defrise and Clack (1994).

The mapping between the planar detector and a slice to be reconstructed is a projection transform. Such a transform can be described by a 4×4 projection matrix using homogeneous coordinates (Foley et al. 1990). This facilitates incremental updates of a and b for fixed increments of x, y, and z in the inner loop of the backprojection. Schaller, Karolczak, Engelke, Wiesent, and Kalender (1998) present an implementation that utilises such incremental updates.

Projection transformations of two-dimensional data sets have been implemented in hardware for acceleration of computer graphics. In this context, the operation is known as *texture mapping*. The U^{-2}-factor in (3.6) can be handled by a graphics hardware feature called z-depth cueing, which normally is used to make objects in a scene appear darker as they move away from the observer. Cabral, Cam, and Foran (1994) implemented the backprojection in the FDK method using texture mapping hardware as:

Algorithm 3.1 *FDK backprojection using texture mapping hardware*

1: **for all** slices **do**
2: setup depth queing as U^{-2}
3: setup perspective texture coordinate matrix
4: **for all** projection angles j **do**
5: setup the rotation β_j
6: render a circle with the filtered projection data of projection j as the texture
7: accumulate the rendered circle to the slice
8: **end for**
9: **end for**

The hardware performs the actual backprojection on line 6 almost instantaneously for a complete projection. Cabral et al. (1994) claim to reconstruct a 128^3 volume in 3.5 seconds, but present no evaluation of the image quality.

Clasén (1999) has made further experiments to see if off-the-shelf hardware aimed at the gaming consumer market could be used for backprojection. He concludes that the limited bit-depth of the currently available graphics cards is an important limiting factor. Although the texture pixels are stored using 24 bits

each, only 8 bits are allocated for each colour channel. All interpolations are performed on the colour channel separately, restricting the actual bit-depth to 8 bits. Clasén shows that this bit-depth is sufficient for acceptable reconstructions of high contrast objects, such as industrial parts or bone. For low contrast objects, like the human body, it is on the other hand far from sufficient. He further notes that most graphics cards not can perform the accumulation step on line 7 internally. The backprojected data then has to be transferred back to the main memory and accumulated by the CPU for each projection. This requires fast bus communication or a shared memory in order not to slow down the process considerably. None of the alternative geometries that will be presented in Sections 3.2.2–3.3.2 can be performed with a 4×4 matrix multiplication of homogeneous coordinates. Therefore hardware acceleration only works with data from a planar detector.

Mueller (1998) describes hardware-accelerated backprojection for algebraic reconstruction techniques. The reconstructed images show that the image quality is unacceptable for low contrast medical reconstruction.

3.2.2 Reconstruction from Cylindrical Detector Data

The above algorithm (3.4)-(3.6) applies to systems that use a planar detector such as an image intensifier or a digital flat-panel detector. If the detector sampling deviates from the planar case the projection data can be interpolated onto a Cartesian grid on the planar detector. As an alternative, Schaller, Flohr, and Steffen (1997) have reformulated the FDK-method for the case of a cylindrical detector centred on the source as in Figure 3.1.

In a detailed analysis in an appendix of his thesis, Schaller (1998) shows that an algorithm identical to the FDK method should perform filtering not along straight lines but along slightly bent curves on the cylindrical detector. As an approximation for the sake of simplicity, this is dispensed with in the proposed algorithm by Schaller (1998), which we call C-FDK. Instead the filtering is performed along the horizontal rows of the cylindrical detector. The computations in C-FDK then become very similar to the FDK method in (3.4)–(3.8), starting with ramp-filtering and pre-weighting as

$$\tilde{p}(\beta, \gamma, q) = \left(\cos\gamma \frac{R}{\sqrt{R^2 + q^2}} p(\beta, \gamma, q) \right) * g(a) \tag{3.9}$$

followed by backprojection

$$f_{\text{C-FDK}}(x, y, z) = \int_0^{2\pi} \frac{R^2}{L(x, y, \beta)^2} \tilde{p}(\beta, \gamma(x, y, \beta), q(x, y, z, \beta))\, d\beta \tag{3.10}$$

where $\gamma(x, y, \beta)$ is given by (2.24) and

$$q(x, y, z, \beta) = z \frac{R}{\sqrt{(R + x\cos\beta + y\sin\beta)^2 + (x\sin\beta - y\cos\beta)^2}} \tag{3.11}$$

The factor $L(x, y, \beta)$ is independent of the z-value and is defined by (2.23).

3.2.3 Properties

Defrise and Clack (1994) noted that their algorithm reduces and becomes identical to FDK reconstruction when the source trajectory is a single circle. This implies that the FDK method handles the available projection data in a sound manner. The result will be identical, apart from numerical implementation differences, with a Radon based approach. However, it is still bound to be an approximate result since the circular trajectory does not fulfil the Tuy-Smith condition.

The Defrise-Clack algorithm assumes the unmeasured Radon data in the shadow zone of Figure 3.2 to be zero. Grangeat (1991) fills the shadow zone by interpolation from the measured data on the surface of the torus. Experiments by Grangeat (1991) and Jacobson (1996) show that this improves the reconstruction quality considerably. So despite the fact that the FDK method uses all available Radon data in a correct way, we should not assume that it is impossible to achieve better image quality by modifying it.

The Radon-based methods only work for untruncated projections, something which in many applications, such as medical, is unfeasible to obtain. An important advantage of the FDK method is that it works well with truncated projections since the filtering is only performed in the untruncated fan-direction.

Despite its approximate character, Feldkamp et al. (1984) observed the following three properties of the FDK method:

- it is exact in the mid-plane, $z = 0$. This is obvious since it is identical to the fan-beam reconstruction presented in Section 2.2.2 in this plane. The reconstruction becomes more and more erroneous for planes further away from the mid-plane.

- it is exact for objects homogeneous in the z-direction, i.e. when $f(x, y, z) = f(x, y)$. This can be intuitively understood if we first note that the pre-weighting dependent on the cone-angle in (3.4) is equal to $\cos\kappa$. Thus, the longer attenuation path of a ray with a high cone-angle is weighted down as if it was measured in the mid-plane. When the object is homogeneous, the pre-weighted projections of different rows will therefore be identical, i.e. $\tilde{p}(\beta, \gamma, q) = \tilde{p}(\beta, \gamma)$ and the algorithm will reduce to the two-dimensional fan-beam reconstruction of Section 2.2.2.

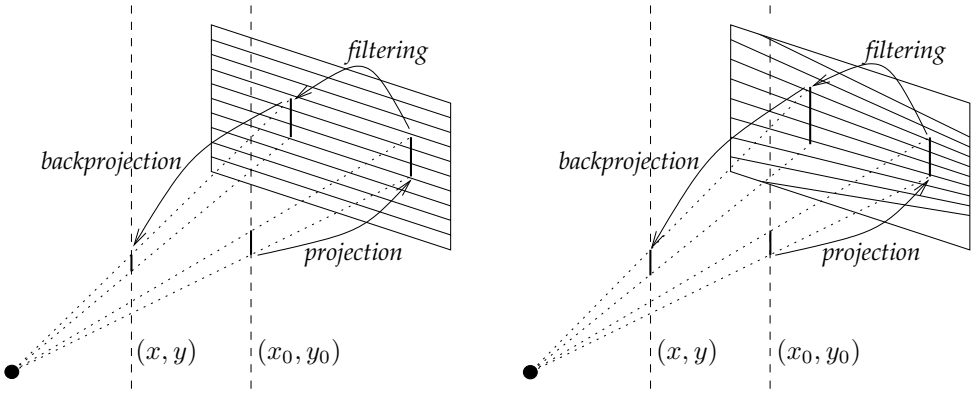


Figure 3.3 *How the line density along the line at (x_0, y_0) contributes to the line intregral at (x, y). Left: FDK-filtering. Right: Slanted filtering.*

- the integral value $\int f(x, y, z)\, dz$ is preserved. This is due to the fact that all three-dimensional Radon data at $z = 0$, i.e. all plane integrals of the planes perpendicular to the mid-plane, can be computed from the projections obtained with the single circular scan. A proof, following the proof by Feldkamp, Davis, and Kress (1984), is given in Appendix A.

The third property implies that the main distortion to be expected is blurring in the axial direction. Linearity and the three properties furthermore imply that this blurring only occurs for the parts of the object that are not z-homogeneous.

We can get more intuitive insight into the third property by studying the reconstruction of a short line segment parallel to the z-axis. Let us examine how the arbitrary line segment at (x_0, y_0) shown in Figure 3.3(a) contributes to the reconstructed image points along an arbitrary line at (x, y) parallel to the z-axis for a specific projection angle. The line segment is magnified when projected onto the detector. The projection signal is then transported along the filtering direction on the detector and scaled with the ramp-filter. The geometrical demagnification in the backprojection sets the length of the line segment at (x, y) that will get a contribution. If we view the situation from above and integrate all values along the z-axis we can compare the situation with two-dimensional fan-beam filtered backprojection. The proof in Appendix A shows that the pre-weighting and U^{-2}-factor in the FDK method compensate for both magnification and demagnification so that the line integral is reconstructed correctly.

We will now examine if the three properties above also hold for C-FDK. If we project the filtering directions of C-FDK onto a planar detector we see that the filtering is performed along curves. Such a curve, for a given constant q, is

described by

$$b(a,q) = q\cos\frac{R}{\sqrt{R^2+a^2}} \tag{3.12}$$

Thus the question is whether this new filtering direction alters the conditions for the three properties. The first two properties hold because of the same reasons as for the original FDK:

- The mid-plane is reconstructed exactly since (3.9) and (3.10) simplify to the two-dimensional (2.21) and (2.22) when $z = q = 0$. The filtering for the mid-plane is performed along the same line as in FDK.

- The pre-weighting in (3.9) makes the projection values for z-homogeneous objects independent of the row q which simplifies C-FDK to exact two-dimensional fan-beam reconstruction.

The third property does not hold for C-FDK. Figure 3.3(b) shows a situation with non-parallel filtering directions. Although the filtering directions in this figure are different from the filtering directions of C-FDK, the figure will still apply to our argument. The critical point is that the filtering directions are non-parallel and that the density of filtering lines, or curves, therefore will vary over the detector. Again, consider the reconstruction of the line segment at (x_0, y_0). The number of filtering curves that will contribute in the backprojection to the line at (x, y) will be proportional to the filtering curve density in the b-direction at the point of projection. The length of the segment at (x, y) that will receive backprojected results will be inversely proportional to the filtering curve density at the point of backprojection. These two curve densities will not necessarily be equal for non-parallel filtering curves, resulting in a shorter or longer segment at (x, y) that receives backprojection contributions compared to the FDK case above. The integral along z might therefore be different and the third property will not hold for C-FDK.

In a more formal study in Appendix A the line segment at (x_0, y_0) is shortened into a point. It is then showed that a pre-filtering and post-filtering density compensation can modify C-FDK so that the third property holds. To this end the projection values should be divided with the local filtering curve density before ramp-filtering and multiplied with the local filtering curve density after ramp-filtering. However, if these two compensation steps are incorporated into the algorithm it is clear that the second property will no longer hold. The first property is also in jeopardy but will hold as long as the filtering curves are horizontal towards the centre of the detector.

3.3 Variations of the FDK Method

The U^{-2}-factor in (3.6) complicates the innermost loop of the backprojection and is the main obstacle in formulating a fast backprojection algorithm for cone-beam data. In Section 3.3.1 we therefore propose a rebinning of the projection data into a geometry that rids this factor from the backprojection formula. The resulting beam geometry is parallel in the fan direction but divergent in the cone direction and the algorithm exploiting this geometry is named P-FDK, as in parallel FDK.

By modifying the filtering directions of P-FDK Grass, Köhler, and Proksa (2000a) have formulated an algorithm, T-FDK, with attractive geometrical properties as well as a seemingly increased image quality. A more elaborate filtering scheme is introduced in the previously unpublished FDK-SLANT algorithm presented in Section 3.3.3.

3.3.1 The P-FDK Method

Cone-beam data acquired from a circular trajectory cannot be rebinned to truly parallel beam data. However, we may apply the two-dimensional parallel rebinning of each row separately. This has been previously proposed for the helical source trajectory geometry by Schaller, Flohr, and Steffen (1997). To the best of our knowledge, this rebinning was first proposed by Turbell (1999) for the circular geometry.

Planar detector

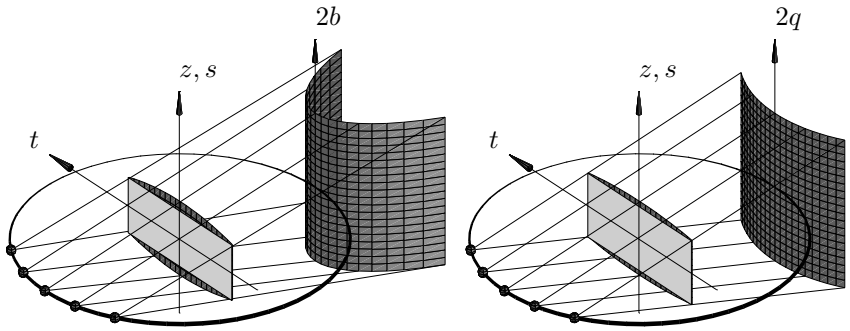
From the projection data measured on a planar detector the rebinned data is obtained as

$$p^{FP}(\theta, t, b) = p^{F}(\theta - \arcsin\frac{t}{R}, \frac{tR}{\sqrt{R^2 - t^2}}, b) \tag{3.13}$$

Note that the row coordinate b is left unchanged. The result is rebinned projection data, the rays of which are parallel when seen from above, along the z-axis. See Figure 3.4(a). The rays of a specific row b intersect the virtual planar detector (t, s) on the z-axis along the curve segments

$$s(b, t) = b \cdot \left(1 - \frac{t^2}{R^2}\right) \tag{3.14}$$

To the right in Figure 3.4(a) we also show a virtual detector where all rays of a specific row b intersect the detector at the same z-value. When moved to the centre of rotation this detector is positioned along the curve $v = t^2/\sqrt{R^2 - t^2}$.



(a) Original data from planar detector.

(b) Original data from cylindrical detector.

Figure 3.4 *The beam geometry after parallel rebinning.*

The reconstruction algorithm P-FDK for the cylindrical detector case proceeds with pre-weighting and filtering of the rebinned data according to

$$\tilde{p}^{FP}(\theta,t,b)=\left(\frac{R^2}{\sqrt{R^4+R^2b^2-b^2t^2}}p^{FP}(\theta,t,b)\right)*g^P(t) \tag{3.15}$$

where the pre-weighting is the cosine of the cone-angle and $g^P(t)$ is the ordinary ramp-filter for parallel beams. The backprojection formula lacks the U^{-2}-factor in (3.6) and is written as

$$f_{\text{P-FDK}}(x,y,z)=\int_0^{2\pi}\tilde{p}^{FP}(\theta,t(x,y,\theta),b(x,y,z,\theta))\,d\theta \tag{3.16}$$

where the detector coordinates of the projected voxel (x,y,z) are

$$t(x,y,\theta)=y\cos\theta-x\sin\theta \tag{3.17}$$

as in (2.17), together with

$$b(x,y,z,\theta)=\frac{zR^2}{v(x,y,\theta)\sqrt{R^2-t(x,y,\theta)^2}+R^2-t(x,y,\theta)^2} \tag{3.18}$$

where

$$v(x,y,\theta)=x\cos\theta+y\sin\theta \tag{3.19}$$

The derivation of $b(x,y,z,\theta)$ can be made with the help of Figure 3.5.

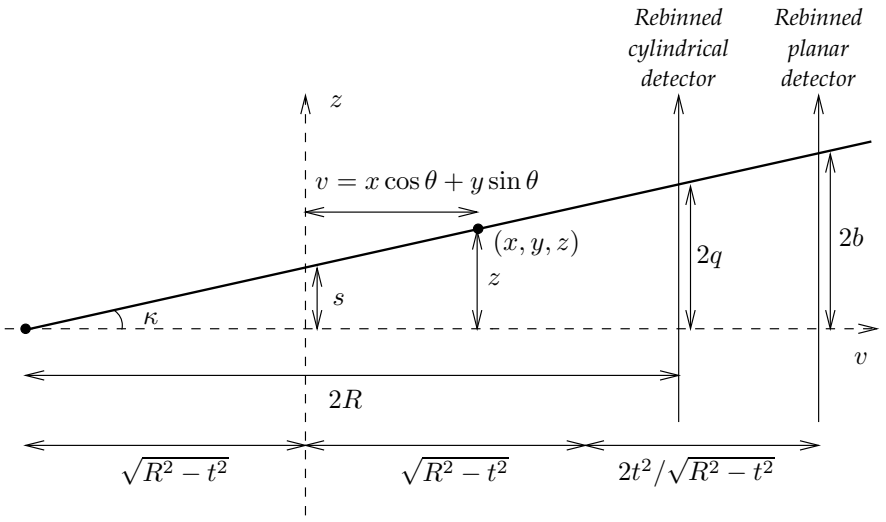


Figure 3.5 *Projection of the point (x, y, z) onto the detectors in the rebinned geometry. The figure is drawn parallel to the (v, z)-plane.*

Cylindrical detector

P-FDK can also be formulated for data from a cylindrical detector. Keeping the row coordinate q unchanged, the rebinned data is obtained as

$$p^{CP}(\theta, t, q) = p(\theta - \arcsin\frac{t}{R}, \arcsin\frac{t}{R}, q) \tag{3.20}$$

The resulting beam, shown in 3.4(b), intersects the virtual planar (t, s) detector along the curves

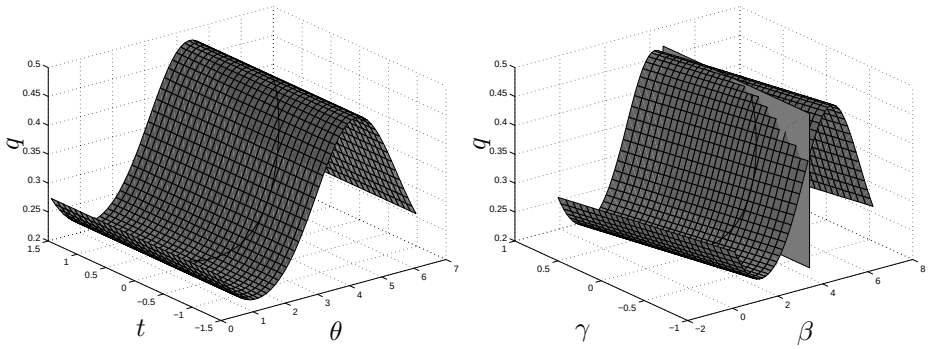
$$s(q, t) = q\sqrt{1 - (t/R)^2} \tag{3.21}$$

which are less curved than for the planar detector case in (3.14). The detector where all rays of a specific row q intersect the detector at the same z-value is placed on a cylinder of radius R, having its axis parallel to the z-axis and centred arbitrarily far away on the v-axis. The reconstruction continues with pre-weighting and filtering

$$\tilde{p}^{CP}(\theta, t, q) = \frac{R}{\sqrt{R^2 + q^2}} p^{CP}(\theta, t, q) * g^P(t) \tag{3.22}$$

followed by backprojection

$$f_{\text{P-FDK}}(x, y, z) = \int_0^{2\pi} \tilde{p}^{CP}(\theta, t(x, y, \theta), q(x, y, z, \theta))\, d\theta \tag{3.23}$$



(a) Rebinned projection space $p^{CP}(\theta, t, q)$ (b) Original projection space $p(\beta, \gamma, q)$

Figure 3.6 *The data used for reconstruction of the image point (x, y, z) lie on a surface in the rebinned projection space. The data used for ramp-filtering are found on the intersection of the surface and a (γ, q)-plane in the original projection space.*

where

$$q(x, y, z, \theta) = \frac{zR}{\sqrt{R^2 - t(x, y, \theta)^2} + v(x, y, \theta)} \tag{3.24}$$

with $v(x, y, \theta)$ defined as in 3.19. The intersection height $q(x, y, z, \theta)$ in (3.24) is seen in Figure 3.5.

Discussion

Experiments show no difference in artifacts for FDK and P-FDK. The algorithms are however not identical as the following argument will show. We will investigate what projection data is used to reconstruct an arbitrary object point (x, y, z) using P-FDK for a cylindrical detector. The three-dimensional backprojection uses the filtered data $p^{CP}(\theta, t, q)$ along the curve

$$\{(\theta, t, q) \,|\, \theta \in [0, 2\pi), t = y\cos\theta - x\sin\theta, q = \frac{zR}{\sqrt{R^2 - t^2} + x\cos\theta + y\sin\theta}\} \tag{3.25}$$

drawn in Figure 3.6(a). These filtered data are constructed by convolution as in (3.22) and hence all data on the *surface*

$$\{(\theta, t, q) \,|\, \theta \in [0, 2\pi), t \in [-t_{\text{max}}, t_{\text{max}}], \\ q = \frac{zR}{\sqrt{R^2 - (y\cos\theta - x\sin\theta)^2} + x\cos\theta + y\sin\theta}\} \tag{3.26}$$

shown in Figure 3.6(a) will be used for the object point. These projection values are also found in the original cone-beam projection space on a surface found by transformation using the rebinning equations (2.30). This surface is shown in Figure 3.6(b). The filtering in the parallel projection space corresponds to filtering along the curved intersection between the surface and a (γ, q)-plane. Since the shape of the surface depends on (x, y, z), it is not possible to formulate the P-FDK algorithms as a pure filtered backprojection of the cone-beam data. The filtering then has to be different for different object points.

The vertical divergence of the oblique parallel beam makes it impossible to utilise quarter offset optimally and merge data from two opposite projections.

3.3.2 The T-FDK, HT-FDK, and S-FDK Methods

Seen on the planar virtual detector, the (t, s)-plane, the filtering in P-FDK is performed along curves given by (3.14) or (3.21). Grass, Köhler, and Proksa (2000a) proposed to rebin the projection data to horizontal lines on the (t, s)-plane. We denote this projection data as $p^P(\theta, t, s)$. It can be interpolated form $p^{FP}(\theta, t, b)$ using (3.14) or from $p^{CP}(\theta, t, q)$ using (3.21). For the ramp-filtering to work, the data needs to be untruncated. Projection data outside the lightly shaded rectangles in Figure 3.4 are therefore discarded. The discarded area can also be found on the original detector, shown in Figure 3.7(a). Collimation of the beam to this area can be used to reduce the dose. The resulting beam after parallel rebinning resembles a tent with a rectangular base. The algorithm is consequently named T-FDK as in tent-FDK[1].

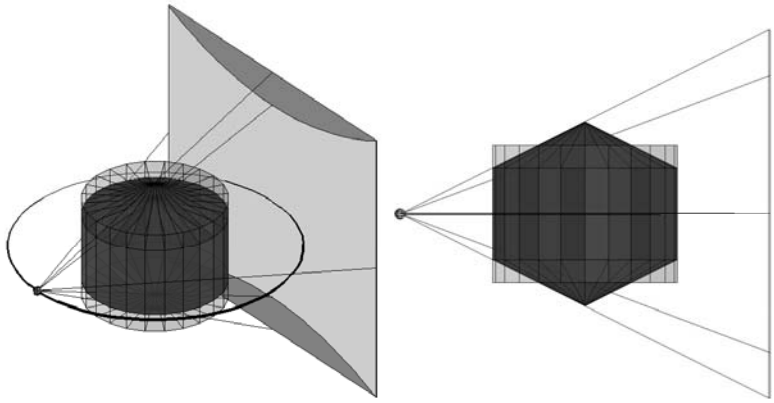
The algorithm includes pre-weighting, ramp-filtering and three-dimensional backprojection. For projection angle θ the voxel (x, y, z) gets a backprojection contribution from the filtered detector value at the s-coordinate derived from Figure 3.5 as

$$s(x, y, z, \theta) = \frac{z\sqrt{R^2 - t(x, y, \theta)^2}}{\sqrt{R^2 - t(x, y, \theta)^2} + v(x, y, \theta)} \tag{3.27}$$

where $t(x, y, \theta)$ and $v(x, y, \theta)$ were defined in (3.17) and (3.19).

T-FDK has an unexpected image quality improvement in the reconstructions which will be examined in Section 3.3.4. It furthermore has a geometry which makes it simple to determine if a voxel receives backprojection contributions from all projection angles, at least half of the projection angles, or fewer than half of the projection angles. See Figure 3.7. The full-scan voxels are confined within a truncated double cone as in FDK and P-FDK. Due to the collimation, the height of this cone is a factor $\cos^2 \gamma_{\text{max}}$ smaller than for FDK if a planar detector is used and a factor $\cos \gamma_{\text{max}}$ smaller if a cylindrical detector is used. The half-scan voxels are

[1] As a consequence of German humour, T-FDK is also known as *Zeltkamp* reconstruction.



(a) Perspective view (b) Side view

Figure 3.7 *FDK and P-FDK are able to reconstruct object points within the dark diamond shape. HT-FDK is able to reconstruct object points within the light cylinder.*

placed within the cylinder circumscribing the double cone. Grass et al. (2000a) propose to reconstruct all full-scan voxels with $360°$ backprojection and all short-scan voxels with $180°$ backprojection. The algorithm is called hybrid tent FDK reconstruction (HT-FDK). The hybrid approach allows for a larger reconstruction volume.

Grass, Köhler, and Proksa (2000b) investigate how the reconstruction result from HT-FDK is improved if the short-scan voxels are backprojected for more than $180°$. A smooth weighting function is applied such that the weights from two opposing projections add to unity. An example of such a function is shown in Figure 3.10. This gives a better signal to noise behaviour and also suppresses certain artifacts.

If a volume larger than the cylinder mentioned above needs to be reconstructed the source trajectory has to be extended. The natural extension is a helical trajectory as described in Chapter 4. A shortcoming of the helical geometry in medical applications is that voxels only illuminated from the beginning or ending parts of the helix are unreconstructable and therefore exposed to radiation in vain. An alternative to the helix is to use a number of circular trajectories. The patient is then moved a certain distance Z along the z-axis between each circular exposure.

Köhler, Proksa, and Grass (2000) show that the beam geometry in HT-FDK is propitious for this sequential circular source trajectory. The proposed reconstruction algorithm is called sequential FDK (S-FDK). If the detector is high enough

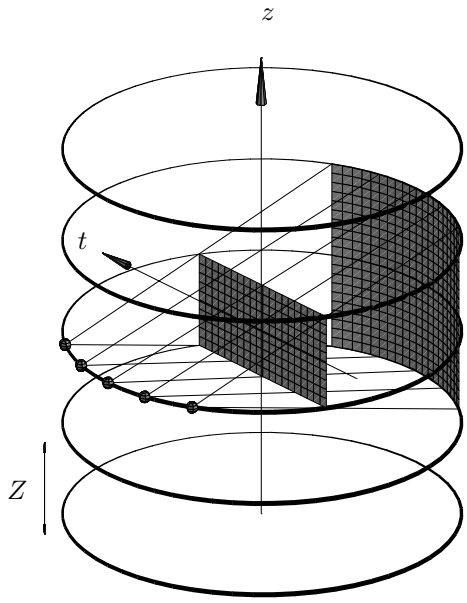
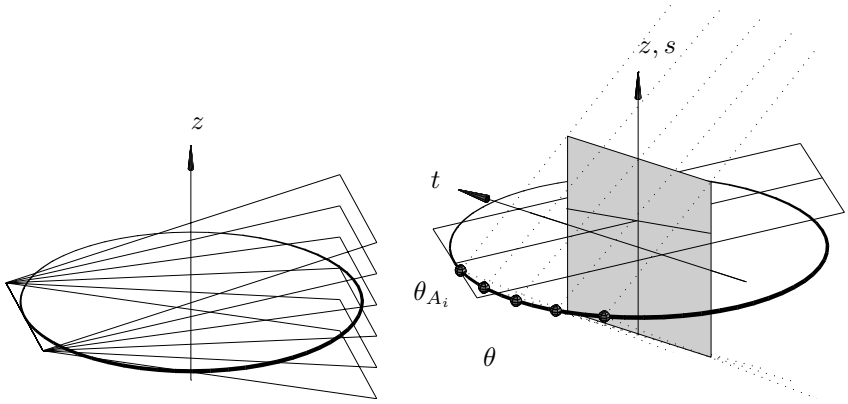


Figure 3.8 *The beam geometry in S-FDK.*

to make the height of the virtual rectangular detector on the (t, z)-plane exactly Z, all voxels will get one backprojection contribution from each projection angle, not more nor less. The voxels inside the double cones get their contributions from the closest source trajectory circle whereas the other voxels get contributions from either this circle or the adjacent circle. The fact that the backprojection intervals from two adjacent circles are complementary in this sense can be understood from Figure 3.8. Projected onto the cylinder on which the circular source trajectories are placed, the rectangular detector window fills up an area limited by the two adjacent trajectory circles. If and only if an arbitrary image point (x, y, z) is not illuminated from the closest circle at a certain projection angle θ, it will be illuminated from the neighbouring circle at projection angle $\theta \pm \pi$. All voxels are therefore illuminated for exactly $360°$. Both the geometry and illumination arguments for S-FDK are highly related to the PI methods discussed in Section 4.3.

3.3.3 The FDK-SLANT Method

In the original derivation of the FDK method Feldkamp et al. (1984) considered an object point at a projection angle. By viewing the situation as a two-dimensional filtered backprojection in the plane containing both the source and the object point and intersecting the planar detector along a vertical line, they derived the weighting and filtering in the FDK method. For the next projection angle a new plane



(a) A set of book pages

(b) Intersection between a book page and the detector plane

Figure 3.9 *The intersection between the (t, s)-detector plane and the book page determines the filtering direction.*

was considered. Inspired by the successful nutating slice algorithms for helical cone-beam reconstruction described in Section 4.2.2 we will derive a variation of FDK by fixing the plane discussed above and not letting it move along the projection direction. The resulting algorithm is called FDK-SLANT.

The FDK algorithm would be exact if projection data from a circular or elliptical source trajectory on the plane in the derivation were available. All image points in the field of view on this plane would then participate in all filtering events without any contamination from image points outside the plane. This is clearly not feasible with projection data restricted to a single circular trajectory. Our objective is to formulate an algorithm where the same image points involved in the filtering for one projection angle also are, as much as possible, filtered together for the other projection angles. This will only be possible for a short-scan algorithm. Section 2.2.3 discussed three ways of handling short-scan data and we choose to use the parallel rebinning of P-FDK to restrict the backprojection for each object point to half a turn.

Figure 3.9(a) shows a set of planes intersecting the source trajectory at projection angle θ_{A_i} at the point $(x, y, z) = (-R\cos\theta_{A_i}, R\sin\theta_{A_i}, 0)$ and having horizontal intersections with the planar detector for this projection angle. The planes are divergent as a set of pages in a half-opened book. The intersection point with the source trajectory is called the *anchor point* for these book pages.

In contrast to the two-dimensional case, using data only from a short-scan interval in the cone-beam case will not utilise all Radon data obtained from the

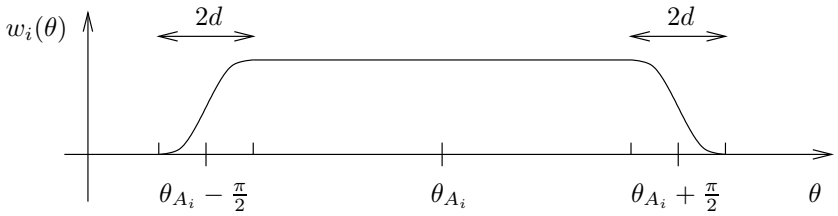


Figure 3.10 *The data is smoothly windowed to a θ-interval of length $\pi + 2d$.*

full-scan. A full cone-beam scan is not redundant in the same way as a full fan-beam scan. In order to make use of all projection data in a full scan, we therefore propose to use N_A books with corresponding anchor points evenly distributed around the cirle. The placement of the anchor points will affect the artifacts of the reconstructed image. With several books, the artifacts can be reduced by averaging the results from all books after backprojection.

The filtering directions are given by the intersections of the book pages at anchor point θ_{A_i} with the planar virtual (t, s)-detector at projection angle θ. See Figure 3.9(b). These intersections are lines. We parametrise the lines with l such that $l = s$ at the centre of the detector where $t = 0$. Projection data $p^P(\theta, t, s)$ from the rebinning steps of T-FDK are interpolated along each column of the detector into the new detector space (θ, t, l) as

$$p^{S_i}(\theta, t, l) = p^P(\theta, t, l - t\frac{l\sin(\theta - \theta_{A_i})}{R}) \tag{3.28}$$

The interpolation for each θ and t is performed from a one-dimensional uniform sampling pattern to another sparser or wider uniform sampling pattern. The chirp-z transform (Oppenheim and Schafer 1975) could be used to perform this interpolation ideally and efficiently via the Fourier domain.

An small extra margin of angular length d is added to each side of the π projection angle interval. This margin is used to introduce a smooth window $w_i(\theta)$ in the θ-direction, shown in Figure 3.10. The rebinned data are pre-weighted, ramp-filtered and smoothly windowed as

$$\tilde{p}^{S_i}(\theta, t, l) = \left(\frac{\sqrt{R^2 - t^2}}{\sqrt{R^2 + s^2 - t^2}} p^{S_i}(\theta, t, l) * g^P(t)\right) w_i(\theta) \tag{3.29}$$

where the window is given by

$$w_i(\theta) = \begin{cases} \cos^2(\frac{\pi}{2} \cdot \frac{\theta_{A_i} - \frac{\pi}{2} - d - \theta}{2d}) & \theta_{A_i} - \frac{\pi}{2} - d \leq \theta < \theta_{A_i} - \frac{\pi}{2} + d \\ 1.0 & \theta_{A_i} - \frac{\pi}{2} + d \leq \theta < \theta_{A_i} + \frac{\pi}{2} - d \\ \cos^2(\frac{\pi}{2} \cdot \frac{\theta - \theta_{A_i} - \frac{\pi}{2} - d}{2d}) & \theta_{A_i} + \frac{\pi}{2} - d \leq \theta \leq \theta_{A_i} + \frac{\pi}{2} + d \\ 0 & \text{otherwise} \end{cases} \tag{3.30}$$

The backprojection is made over the $\pi + 2d$ interval as

$$f_{\text{FDK-SLANT}\,i}(x,y,z) = \int_{\theta_{A_i}-\frac{\pi}{2}-d}^{\theta_{A_i}+\frac{\pi}{2}+d} \tilde{p}^{S_i}(\theta, t(x,y,\theta), l(x,y,z,\theta))\, d\theta \tag{3.31}$$

where $t(x,y,z)$ is given by (3.17) and $l(x,y,z,\theta)$ is derived using (3.27) as

$$\begin{aligned} l(x,y,z,\theta) &= s(x,y,z,\theta)\frac{R}{R - t(x,y,\theta)\sin(\theta - \theta_{A_i})} \\ &= \frac{zR\sqrt{R^2 - t(x,y,\theta)^2}}{(\sqrt{R^2 - t(x,y,\theta)^2} + v(x,y,\theta)) \cdot (R - t(x,y,\theta)\sin(\theta - \theta_{A_i}))} \end{aligned} \tag{3.32}$$

where $v(x,y,\theta)$ is given by (3.19). The contribution from all N_A sets of book pages are finally summed as

$$f_{\text{FDK-SLANT}}(x,y,z) = \frac{2}{i}\sum_{i=1}^{N_A} f_{\text{FDK-SLANT}\,i}(x,y,z) \tag{3.33}$$

The main point of the algorithm is to keep the filtering events as much as possible as an interaction between voxels in the same book page. The book pages are kept fixed during the source-detector rotation. Clearly, an unwanted non-homogeneous handling of projection data from different projection angles is unavoidable. At the anchor point angles the projections of the pages almost take the shapes of straight lines but the projections will be more dispersed for projection angles that differ from these two anchor points. An unwanted abrupt change in the filtering occurs at $\theta = \theta_{A_i} \pm \frac{\pi}{2}$. The smoothing interval of length $2d$ has been introduced to make the change less abrupt.

The projection interval per book page could be made shorter than $180°$. The filtering curves would then fit the book pages better. However, the reconstruction on each book page would be from an incomplete projection interval. It is interesting to note that FDK-SLANT becomes identical to T-FDK when the interval approaches zero. An alternative way of motivating the filtering direction in T-FDK is to consider the reconstruction volume as consisting of a number of slices parallel to the (x,y)-plane. Using the same filtering direction arguments as we will do for the PI-SLANT method in Section 4.3.2, the horizontal parallel filtering directions in T-FDK are optimal for this set of planes.

3.3.4 Experimental Results

The three-dimensional Shepp-Logan phantom, defined in Appendiz B.2, is a simple model of a skull. We have synthetically generated noise-free cone-beam projections on a planar detector using the program `take` by Müller-Merbach (1996)

Source trajectory radius	$R = 2.0$ length units (l.u.)
Projections per turn	$N_\beta = N_\theta = 512$
Detector rows	$N_b = 232$
Detector row height	$\Delta b = 0.01$ l.u.
Detector elements per row	$N_a = N_t = 232$
Maximum fan-angle	$\gamma_{\max} = \arctan \frac{N_a \Delta a}{2R} = 30.1°$
Maximum cone-angle at $a = 0$	$\kappa_{\max} = \arctan \frac{N_b \Delta b}{2R} = 30.1°$
Reconstruction grid	$N_x \times N_y \times N_z = 232 \times 232 \times 232$
Reconstruction grid resolution	$\Delta x = \Delta y = \Delta z = 0.01$ l.u.
Simulated rays per detector element	18
Anchor points in FDK-SLANT	$N_A = 4$
Smoothing width in FDK-SLANT	$d = 10.5°$

Table 3.1 *Experiment parameters for the three-dimensional Shepp-Logan phantom.*

and Turbell (1997). The source was modelled as a small square of the same width as a detector element. To reduce aliasing, each detector element value was calculated as the mean of eighteen line integrals from different positions on the source to different positions on the detector element. The reconstruction parameters are given in Table 3.1.

The reconstruction results are shown in Figures 3.12 and 3.14. FDK has a clear drop in image level for voxels far away from the mid-slice. This is a well-known shortcoming of FDK. T-FDK shows less drop and FDK-SLANT even less. The profiles along the z-axis in Figure 3.14 illustrate this phenomenon clearly. None of the methods reconstruct the top and bottom bone of density 2.0 correctly but FDK-SLANT performs better than the other two.

In a second experiment the voxelised head phantom, defined in Appendix B.3, was reconstructed using the parameters of Table 3.2. The reconstruction results in Figure 3.13 show that the T-FDK reconstruction has a smaller intensity drop for this phantom as well. FDK-SLANT shows even smaller drop, but also introduces substantial artifacts. These are unfortunately not reduced much by introducing more anchor points or a longer smoothing interval d.

These simple experiments indicate the superiority of T-FDK and FDK-SLANT over FDK. A more extensive image quality comparison between the three algorithms is under way.

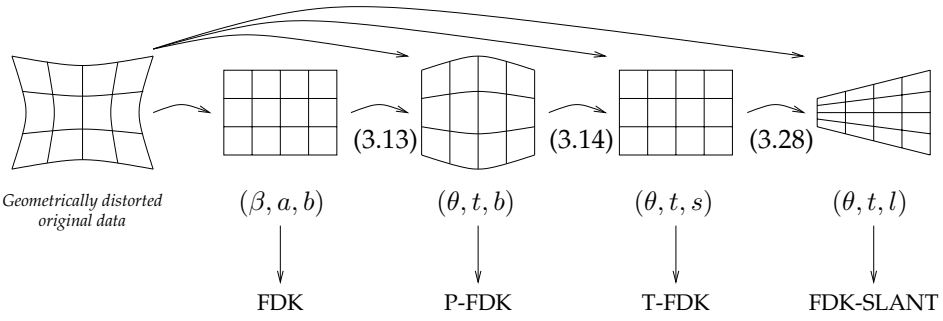


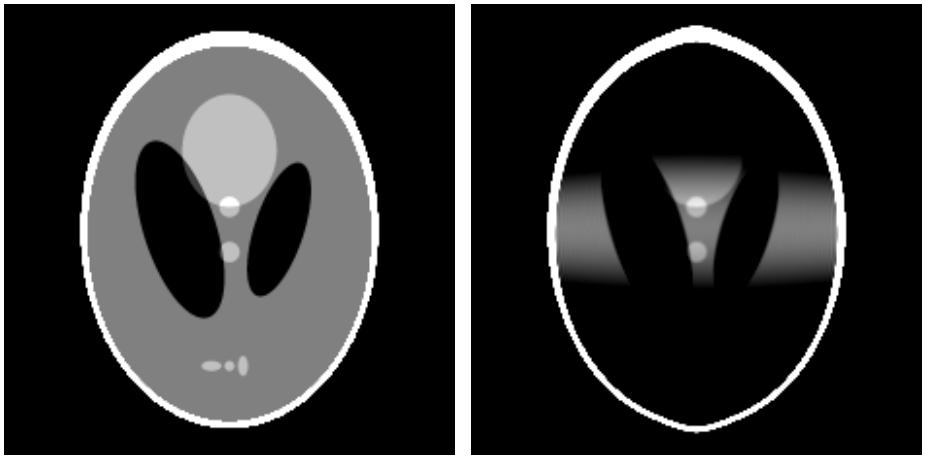
Figure 3.11 *The chain of resamplings can be short-cut.*

3.3.5 Discussion

As it is formulated here, FDK-SLANT requires untruncated projections. If the slanted lines spread outside the measured detector window, such that the rebinning equation (3.28) generates values of s outside the measured range, it is not clear what to do. Extrapolation of data from the top and bottom detector could be one solution, but this needs to be experimentally verified.

The reconstruction results for FDK-SLANT could probably be slightly improved if the shape of the book pages were optimised in a similar manner to the nutating surfaces described in Section 4.2.2. The planar book pages used in FDK-SLANT are certainly not optimal in this sense, but do allow for relatively simple computation steps. Although the helical geometry will be shown to be much better suited for the concept of nutating surfaces, we believe that PI-SLANT shows that the concept can be used for the circular geometry as well.

Figure 3.11 shows the chain of rebinning steps for the proposed methods. This chain can be believed to introduce extra interpolation steps in the algorithms, which could affect the image quality. However, it is important to note that systems an image intensifier detector have a built-in geometrical distortion. Calibration methods are commonly used to determine this distortion and rectify it by a two-dimensional resampling to a true Cartesian grid before the reconstruction commences. See for example the work of Reimann and Flynn (1992). This extra interpolation step could be dispensed with if the geometrical distortion equations were combined with the resampling equations in the new FDK methods. The values on the (θ, t, s) or (θ, t, l) grid could then be obtained directly from the original data by a single three-dimensional interpolation. The only extra interpolation introduced for the new FDK methods compared to the original is then one-dimensional in the angular direction from β to θ.



(a) Phantom

(b) FDK

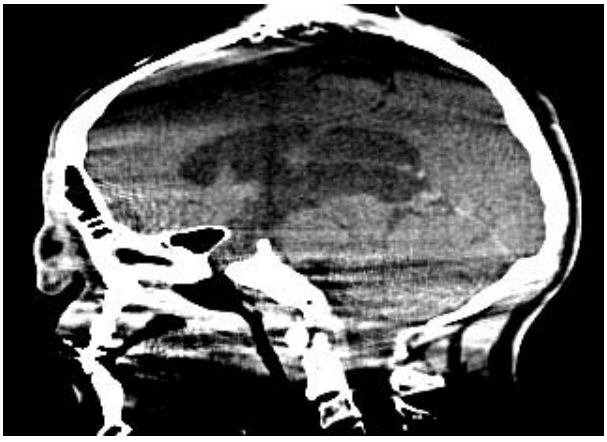
(c) T-FDK

(d) FDK-SLANT

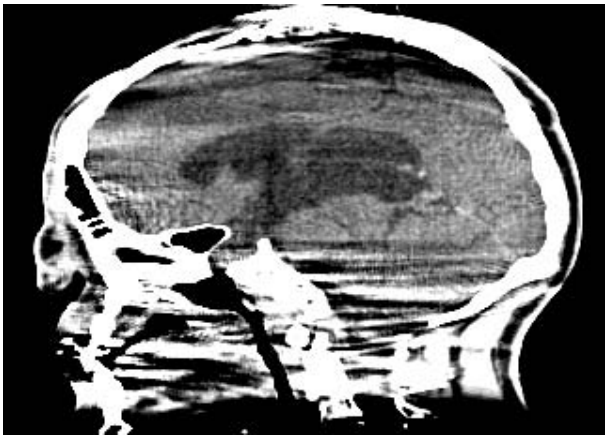
Figure 3.12 *Reconstruction of the three-dimensional Shepp-Logan phantom. The* (x, z)*-slice at* $y = -0.25$ *is shown. Parameters as in Table 3.1. Greyscale interval* $[1.0, 1.04]$.

(a) Phantom

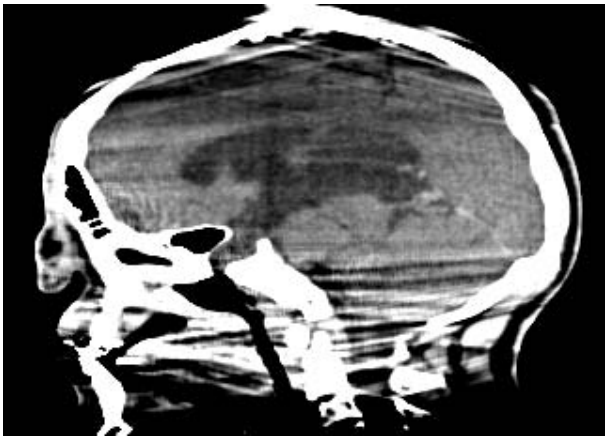
(b) FDK



(c) T-FDK



(d) FDK-SLANT, $N_A = 2$, $d = 0$.



(e) FDK-SLANT, $N_A = 8$, $d = 10°$.

Figure 3.13 *Reconstructed (y, z)-slice through the voxelized head phantom with a 64-row detector. Parameters as in Table 4.4. Greyscale interval* $[980, 1080]$.

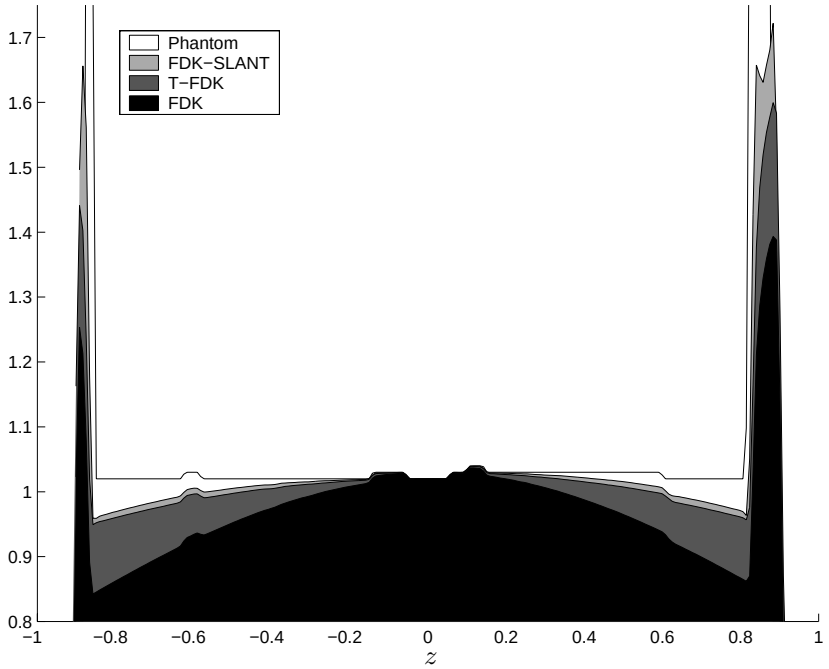


Figure 3.14 *Profile of the Shepp-Logan phantom along the z-axis at* $(x, y) = (0, -0.25)$. *Parameters as in Table 3.1.*

Source trajectory radius	$R = 230.0\text{mm}$
Projections per turn	$N_\beta = N_\theta = 512$
Detector elements per row	$N_a = N_t = 321$
Detector element width	$\Delta a = \frac{270}{321}\,\text{mm} \approx 0.84\,\text{mm}$
Detector rows	$N_b = 321$
Detector row height	$\Delta b = \frac{270}{321}\,\text{mm} \approx 0.84\,\text{mm}$
Maximum fan-angle	$\gamma_{\max} = \arctan\frac{N_a \Delta a}{2R} \approx 30.4^\circ$
Maximum cone-angle at $a = 0$	$\kappa_{\max} = \arctan\frac{N_b \Delta b}{2R} \approx 30.4^\circ$
Reconstruction grid	$N_x \times N_y \times N_z = 320 \times 320 \times 230$
Reconstruction grid resolution	$\Delta x = \Delta y = \Delta z = \frac{221}{320}\,\text{mm} \approx 0.69\,\text{mm}$
Simulated rays per detector element	18

Table 3.2 *Experiment parameters for the voxelized head phantom.*

3.4 The FDK-FAST Method

Armed with the oblique parallel backprojection of P-FDK and the general principles for fast backprojection, it would seem to be a straight forward task to formulate a fast backprojection for the FDK method. However, some fundamental differences in degrees of freedom for lines in two and three dimensions will make the extension from two dimensions non-trivial.

Most material of this section is taken from Turbell (1999) which contains the first formulation and implementation of a complete cone-beam backprojection in $\mathcal{O}(N^3 \log N)$ time. Nilsson (1997) briefly deals with the problem of a fast FDK backprojection algorithm, but focuses on how long a piece of fan-beam source path could be used without having to introduce the L^{-2}-factor in the backprojection. He claims somewhat prematurely that the fast FDK method trivially reduces to fan-beam reconstruction.

3.4.1 Links and the Basic Step

The main idea behind our fast cone-beam backprojection is identical to the two-dimensional case in Section 2.3: recursive summation of link values. However, since the wanted summation curves are now defined in three dimensions, the divide-and-conquer strategy is less obvious.

In Section 2.3, the fast two-dimensional backprojection algorithm starts with calculating the 2-links, and then continues with longer and longer links until, eventually, the pixel values are calculated. This may be called a *bottom-up* approach. We also mentioned pruning of trees without discussing how it was done. Approximate coarse pruning rules can be formulated, but the iterative and discrete nature of the algorithm makes it difficult to predict exactly which shorter link are necessary for constructing the longer links. But since most of the blindly computed links will be used no major inefficiency occurs. Unfortunately, in three dimensions, pruning is essential while the approximate pruning rules become more complex. We therefore feel compelled to introduce the following *top-down* approach, applicable to both two- and three-dimensional backprojection, to decide in advance which link values to calculate.

Starting with the pixels (voxels), we decide which $\frac{\pi}{2}$-links are necessary for performing the interpolation in (2.46). The indexes of these links are tabulated, but their values can of course not yet be calculated. The next step is to decide and tabulate which $\frac{\pi}{4}$-links are necessary to perform the basic step for the tabulated $\frac{\pi}{2}$-links. The process goes on with shorter and shorter links, ending when the necessary 2-links have been tabulated. Preferably, the interpolation weights obtained from (2.44) are also stored in the tables at the same time. Note that all these table entries only depend on the scanning geometry, not on the object to be

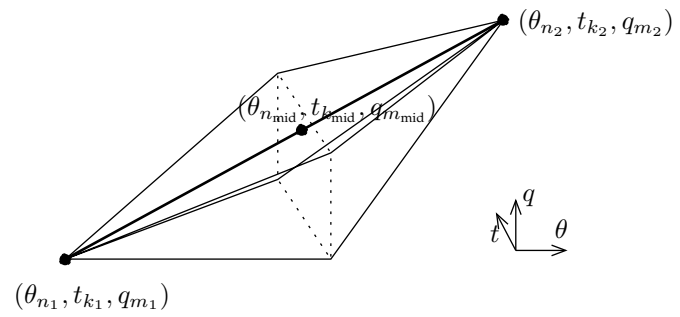


Figure 3.15 *The basic step. The value of a link is calculated from the values of eight links of half the length. The links are drawn as lines for simplicity, but are in reality curved.*

measured, and they may therefore be pre-calculated once and for all. Unfortunately, the tables are large. Section 3.4.2 contains a discussion on the size of the tables.

The link value calculation is simple once the tables are fabricated. It follows the previous bottom-up approach, but does not include any mid-point or interpolation weight calculations, only the computation of the basic equation (2.43).

The core of the algorithm is a straightforward generalisation of the basic step in two dimensions, but this time we calculate the value of a new link from eight links of half the length. See Figure 3.15. For this basic step to work we need to know the midpoint $(\theta_{\text{mid}}, t_{\text{mid}}, q_{\text{mid}})$ of a given link $(\theta_{n_1}, t_{k_1}, q_{m_1}; \theta_{n_2}, t_{k_2}, q_{m_2})$. If we project the two end-point rays $(\theta_{n_1}, t_{k_1}, q_{m_1})$ and $(\theta_{n_2}, t_{k_2}, q_{m_2})$ onto the (x, y)-plane (see Figure 3.16), their intersection $(x_{\text{mid}}, y_{\text{mid}}, 0)$ is given by the solutions to the following system of equations

$$\begin{pmatrix} x_{\text{mid}} \\ y_{\text{mid}} \end{pmatrix} = \begin{pmatrix} -\sin\theta_{n_1} & \cos\theta_{n_1} \\ -\sin\theta_{n_2} & \cos\theta_{n_2} \end{pmatrix}^{-1} \begin{pmatrix} t_{k_1} \\ t_{k_2} \end{pmatrix} \tag{3.34}$$

as in the two-dimensional case in (2.38). The first two indexes of the mid-point are identical to the two-dimensional case (2.41) and (2.42), repeated here for convenience, namely

$$n_{\text{mid}} = \frac{n_1 + n_2}{2} \tag{3.35}$$

and

$$k_{\text{mid}} = \frac{t_{k_1} + t_{k_2}}{2\cos(\frac{\theta_{n_1} - \theta_{n_2}}{2})} \tag{3.36}$$

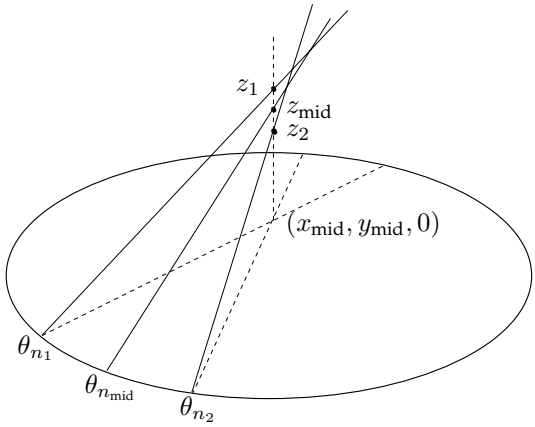


Figure 3.16 *The two rays corresponding to the link end points, do not necessarily intersect. The mid-point* $(\theta_{n_{mid}}, t_{k_{mid}}, q_{m_{mid}})$ *is derived by letting the ray corresponding to the mid-point intersect the point* $(x_{mid}, y_{mid}, z_{mid})$.

The problem is the third index m_{mid}. Consider the following facts. The two end-points of a link correspond to two rays in the image domain. Unlike the two-dimensional case, these two rays almost never intersect in three dimensions as illustrated in Figure 3.16. The mapping between links and image points is therefore ambiguous, but we propose to resolve the ambiguity with the following technique.

First we calculate z_{mid}, as shown in Figure 3.16, as

$$z_{\text{mid}} = \frac{z_1 + z_2}{2} \tag{3.37}$$

where

$$z_1 = q_{m_1} \frac{\sqrt{R^2 - t_{k_1}^2} + v_1}{R} \tag{3.38}$$

and

$$z_2 = q_{m_2} \frac{\sqrt{R^2 - t_{k_2}^2} + v_2}{R} \tag{3.39}$$

finally giving us

$$q_{\text{mid}} = \frac{z_{\text{mid}} R}{\sqrt{R^2 - t_{k_{\text{mid}}}^2} + x \cos\theta_{n_{\text{mid}}} + y \sin\theta_{\text{mid}}} \tag{3.40}$$

Figure 3.15 illustrates how the new link value is obtained from the two-dimensional interpolation

$$
\begin{aligned}
&I[n_1, k_1, m_1; n_2, k_2, m_2] = \\
&w_t w_q \big(I[n_1, k_1, m_1; n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor, \lfloor m_{\text{mid}} \rfloor] + I[n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor, \lfloor m_{\text{mid}} \rfloor; n_2, k_2, m_2] \big) + \\
&w_t w'_q \big(I[n_1, k_1, m_1; n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor, \lfloor m_{\text{mid}} \rfloor + 1] + I[n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor, \lfloor m_{\text{mid}} \rfloor; n_2, k_2, m_2] \big) + \\
&w'_t w_q \big(I[n_1, k_1, m_1; n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor + 1, \lfloor m_{\text{mid}} \rfloor] + I[n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor + 1, \lfloor m_{\text{mid}} \rfloor; n_2, k_2, m_2] \big) + \\
&w'_t w'_q \big(I[n_1, k_1, m_1; n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor + 1, \lfloor m_{\text{mid}} \rfloor + 1] + I[n_{\text{mid}}, \lfloor k_{\text{mid}} \rfloor + 1, \lfloor m_{\text{mid}} \rfloor + 1; n_2, k_2, m_2] \big)
\end{aligned}
\tag{3.41}
$$

where the weights, if bi-linear interpolation is used, are

$$
\begin{aligned}
w_t &= 1 - w'_t, \quad w'_t = k_{\text{mid}} - \lfloor k_{\text{mid}} \rfloor \\
w_q &= 1 - w'_q, \quad w'_q = m_{\text{mid}} - \lfloor m_{\text{mid}} \rfloor
\end{aligned}
\tag{3.42}
$$

Since the end-point rays of a π-link would not intersect even when projected onto the mid-plane, we do not construct any links longer than $\frac{\pi}{2}$. The voxel values are evaluated as the sum of four interpolations of $\frac{\pi}{2}$-links in a similar manner to the two-dimensional case. The interpolation in the t-direction may be skipped if quarter-offset is not used and the detector and voxel resolutions match, whereas the q-interpolation may not. Each one of the four interpolations utilises 16 $\frac{\pi}{2}$-links according to

$$
\begin{aligned}
&w_{t_1} w_{t_2} w_{q_1} w_{q_2} \cdot I[n_1, \lfloor k_1 \rfloor, \lfloor m_1 \rfloor; n_2, \lfloor k_2 \rfloor, \lfloor m_2 \rfloor] + \\
&w_{t_1} w_{t_2} w_{q_1} w'_{q_2} \cdot I[n_1, \lfloor k_1 \rfloor, \lfloor m_1 \rfloor; n_2, \lfloor k_2 \rfloor, \lfloor m_2 \rfloor + 1] + \\
&\qquad\qquad \vdots \\
&w'_{t_1} w'_{t_2} w'_{q_1} w'_{q_2} \cdot I[n_1, \lfloor k_1 \rfloor + 1, \lfloor m_1 \rfloor + 1; n_2, \lfloor k_2 \rfloor + 1, \lfloor m_2 \rfloor + 1]
\end{aligned}
\tag{3.43}
$$

with the bi-linear weights

$$
\begin{aligned}
w_{t_1} &= 1 - w'_{t_1}, \quad w'_{t_1} = k_1 - \lfloor k_1 \rfloor \\
w_{t_2} &= 1 - w'_{t_2}, \quad w_{t_2} = k_2 - \lfloor k_2 \rfloor \\
w_{q_1} &= 1 - w'_{q_1}, \quad w'_{q_1} = m_1 - \lfloor m_1 \rfloor \\
w_{q_2} &= 1 - w'_{q_2}, \quad w_{q_2} = m_2 - \lfloor m_2 \rfloor
\end{aligned}
\tag{3.44}
$$

This concludes the reconstruction.

3.4.2 Complexity Analysis

With the introduction of tabulation of links, we need to analyse both the space complexity, i.e. the amount of memory needed, and the time complexity of the algorithm. We start with the latter.

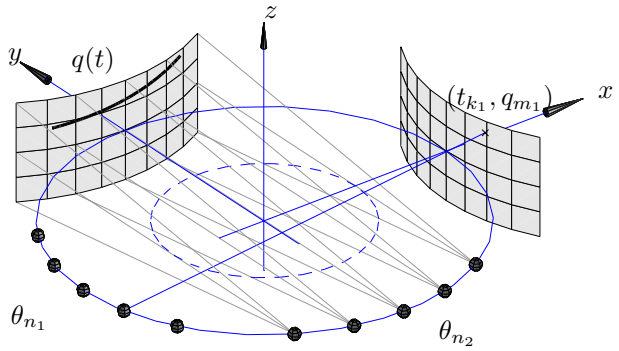


Figure 3.17 *The branch endpoints of the $\pi/2$-tree with root in $(\theta_{n_1}, t_{k_1}, q_{m_1})$ are geometrically described as the projection of the ray $(\theta_{n_1}, t_{k_1}, q_{m_1})$ onto the detector at projection angle $\theta_{n_2} = \theta_{n_1} + \pi/2$. We only regard the part of the ray that passes through the cylindrical FOV, indicated with a dashed circle.*

Time complexity

As before, we define a tree as the collection of links that start from one and the same point. The links of a tree extend in both the t- and q-directions, which appears troublesome from a computational complexity point of view. Should all trees require the same angular span in the two directions, the total number of links would be

$$\sum_{i=2}^{\lg N_\theta - 1} \underbrace{N_t N_q N_\theta 2^{1-i}}_{\text{Number of trees}} \cdot \underbrace{c2^{i-1}2^{i-1}}_{\text{Links per tree}} = cN_tN_qN_\theta \sum_{i=2}^{\lg N_\theta - 1} 2^{i-1} = cN_tN_qN_\theta(2^{\lg N_\theta - 1} - 2) \in \mathcal{O}(N^4) \tag{3.45}$$

with, alas, no gain in the asymptotic behaviour compared to traditional backprojection. Fortunately, as this section will show, the growth of trees is essentially a one-dimensional process. The actual trees resemble bent fans, extending over an angular span in the t-direction, while being curved in the q-direction.

To estimate the shape of a tree exactly, we have to take the discreteness of the reconstruction volume into consideration, but we disregard this initially. Figure 3.17 shows how we geometrically can derive the shape of an ideal $\frac{\pi}{2}$-tree with root in $(\theta_{n_1}, t_{k_1}, q_{m_1})$. The image points that need the links of this tree lie in the FOV along the ray $(\theta_{n_1}, t_{k_1}, q_{m_1})$. After a projection angle of $\frac{\pi}{2}$, these image points have

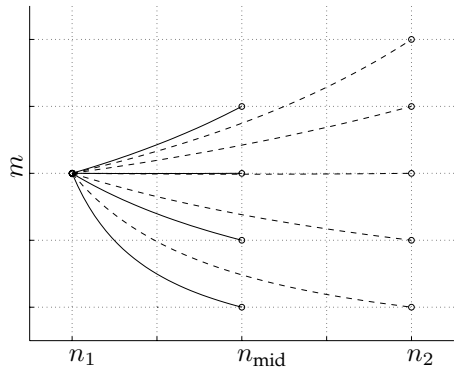


Figure 3.18 *Due to the interpolation margin necessary in the basic step, the thickness of a tree (drawn dashed) is approximately preserved for a tree of half the length (drawn solid).*

branched out and are projected onto the detector along the curve

$$q(t) = q_{m_1} \frac{\sqrt{R^2 - t_{k_1}^2} + t}{\sqrt{R^2 - t^2} + t_{k_1}}, \quad -\sqrt{t_{\text{max}}^2 - t_{k_1}^2} \leqslant t \leqslant \sqrt{t_{\text{max}}^2 - t_{k_1}^2} \tag{3.46}$$

It is exactly the links from $(\theta_{n_1}, t_{k_1}, q_{m_1})$ to the set of end-points described by this equation that constitute the ideal tree. Since there is only one q-value for each t-value in (3.46), we have now showed that the trees form fans rather than cones, and their growth is one-dimensional rather than two-dimensional. The ideal shape of shorter trees may be derived in a similar manner, changing the projection angle difference, $\frac{\pi}{2}$ above, to the desired tree length.

It might also be illuminating to observe the following. The image points are in $\mathbb{R}^3$ while the set of possible rays through a 3D-volume is in $\mathbb{R}^4$. The set of rays measured from the circular trajectory is only an $\mathbb{R}^3$-subset of these possible rays. Hence, the number of curves (image points) passing through a (t, q)-plane is also in $\mathbb{R}^3$. Therefore, the curves emanating from a certain (t, q)-value in such a plane must form a fan rather than a cone.

To find out the computational complexity we have to examine the tree structure further. In the discrete reconstruction volume the ray $(\theta_{n_1}, t_{k_1}, q_{m_1})$ in Figure 3.17 traverses voxels in a 2×2 thick tube of voxels surrounding the ray. When projected onto the detector, the tube thickness is scaled by a factor of approximately $R/(R + t_{k_1})$ due to the distance to the detector and a factor of approximately $1/\cos\kappa$ due to the cone-angle. Hence, the resulting projected thickness,

Type	Unique	Duplicates	Quantity [10^6]	ADD	MULT	MFLOPs
Voxels	825088	1	0.8	63	64	105
64-links	3052781	4	12	7	4	134
32-links	2477849	8	20	7	4	218
16-links	1462633	16	23	7	4	257
8-links	845834	32	27	7	4	298
4-links	544317	64	35	7	4	383
2-links	380836	128	49	7	4	536
Total	9589338		167			1932

Table 3.3 *Calculation of FLOPs when $N = 128$.*

measured in detector rows, is

$$\frac{2R\Delta z}{\Delta q(R + t_{k_1})\cos\kappa} \lessapprox 5 \tag{3.47}$$

for typical parameter values ($\Delta z = \Delta q$, $\kappa_{\max} = 10^\circ$, $t_{\max} = R/2$). To summarise, a $\frac{\pi}{2}$-tree looks like a non-planar fan, typically extending over a large interval in the t-direction but having a thickness of only a few grid points in the q-direction.

An approximation of the thickness of shorter trees is made in Figure 3.18. Each step, i, in the top-down construction process halves the thickness of the long tree, d_{i-1}. However, it adds a margin needed in the basic step interpolation, and thus giving the short tree the approximate thickness

$$d_i = \lfloor d_{i-1}/2 \rfloor + 2 \tag{3.48}$$

The thickness therefore converges to around 3 or 4 detector rows, regardless of the resolutions chosen.

There are half as many 2-trees as 1-trees since there are half as many θ-positions for them to start from. Assume that a 1-tree contains c links. The total number of links created in step 2 to step $\lg N_\theta - 1$ is then

$$\sum_{i=2}^{\lg N_\theta - 1} \underbrace{N_\theta N_t N_q 2^{1-i}}_{\text{Number of trees}} \cdot \underbrace{c2^{i-1}}_{\text{Links per tree}} = cN_\theta N_t N_q(\lg N_\theta - 2) \in \mathcal{O}(N^3 \log N) \tag{3.49}$$

The number of links per step in a specific case is shown in Table 3.3. The total number of Floating Point Operations (FLOP) needed for computation of these links is now possible to estimate. The basic step (3.41) requires 4 multiplications

N	64	128	256	512	1024
Trad.	$1 \cdot 10^8$	$2 \cdot 10^9$	$3 \cdot 10^{10}$	$5 \cdot 10^{11}$	$9 \cdot 10^{12}$
Fast	$2 \cdot 10^8$	$2 \cdot 10^9$	$2 \cdot 10^{10}$	$2 \cdot 10^{11}$	$1 \cdot 10^{12}$

Table 3.4 *Number of FLOPs on projection data in traditional and fast backprojection as a functrion of $N = N_x = N_y = 2N_z = N_t = N_q = N_\theta/2$. The estimate for fast backprojection when $N = 1024$ is extrapolated.*

and 7 additions if all four interpolation weights are tabulated. The final link assembly uses 16 multiplications and 15 additions in each quadrant. Table 3.3 shows how the total number of FLOPs is estimated when $N = 128$.

Traditional backprojection requires a bi-linear interpolation for each voxel and projection angle. A bi-linear interpolation consists of three linear interpolations, each of which requires two multiplications and one addition. The result should be accumulated, and the total number of FLOPS is therefore

$$(3 \cdot 3 + 1) \underbrace{N_x N_y N_z \frac{\pi}{4}}_{\text{Voxels in FOV}} N_\theta = 2.5\pi N^4, \quad N = N_x = N_y = 2N_z = N_\theta/2 \tag{3.50}$$

A comparison between fast and traditional backprojection based on the estimates above is shown in Table 3.4. This comparison does not include indexing or geometrical calculations, which, if included, would make our algorithm seem even more efficient, due to the pre-calculated tables.

Space complexity

There are two major memory areas needed by the algorithm: the link value area and the link construction table area containing interpolation coefficients and address information to access link value data. The former needs a random access memory (RAM), whereas the latter may be stored once and for all in a read-only memory (ROM).

The link values produced in each of the $\lg N_\theta - 2$ steps of the algorithm are only needed as input data for the consecutive step and may be discarded thereafter. Therefore, the RAM storing these values can be partitioned into two segments, one large enough to store the values produced in the previous step and the other one large enough to store the values produced in the present. After one step is completed, the contents of the older segment is obsolete and can be used for the upcoming step in a Ping-Pong buffer style. Table 3.3 shows that the number of links decreases from step to step. Hence it is the number of 2-links that determines the exact memory size which is always $\mathcal{O}(N^3)$.

For any long link in the top-down link construction approach of Section 3.4.1, the information on all necessary links of shorter length must be placed in another table. The basic idea of the fast backprojection algorithm is that each link value will be used in several basic step calculations. Hence, as a child a link will serve many fathers and its value will be accessed several times. In the top-down process we are therefore forced to check if the child is already tabulated in order to avoid duplicates.

Since we need $\mathcal{O}(N^3 \log N)$ links, a naive guess would be that the number of ROM entries should also be of this magnitude, but thanks to a symmetry explained in the next paragraph, the required number of entries can be reduced to $\mathcal{O}(N^3)$.

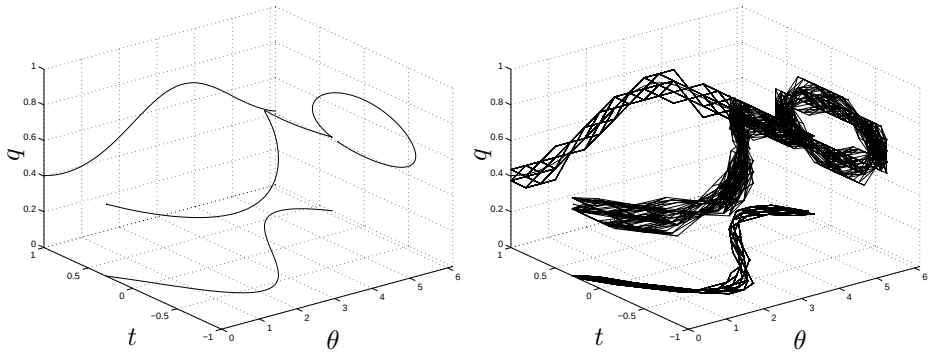
The discussions on the ideal trees shapes in Section 3.4.2 only considered the length of the trees, $\theta_{n_2} - \theta_{n_1}$, not the actual values of θ_{n_1} and θ_{n_2}. The shape of an ideal tree is therefore independent on the θ-coordinate of its root. Taking the discreteisation effects into account, this nice property is not completely true; a few links at the border of the tree may differ between trees along the θ-axis. Despite these small differences, we can make the link construction tables θ-independent. In order for this to work, the union of all links of these slightly different trees is tabulated, so that all necessary links are available. This results in that a few link values will be calculated in vain, but reduces the total table size to

$$\sum_{i=2}^{\lg N_\theta - 1} \underbrace{N_t N_q}_{\text{Number of trees}} \cdot \underbrace{c2^{i-1}}_{\text{Links per tree}} = cN_tN_q(2^{\lg N_\theta - 2} - 2) \in \mathcal{O}(N^3) \tag{3.51}$$

entries, where each entry contains eight table indexes pointing to the children of the link, and four interpolation weights. The table index pointers point into the $\mathcal{O}(N^3)$-sized RAM and therefore require $\mathcal{O}(\lg N^3) = \mathcal{O}(\log N)$ bits each which makes the actual ROM size $\mathcal{O}(N^3 \log N)$.

Another advantage of making the table θ-independent is that we may assume all trees to start at $\theta_{n_1} = 0$, and thereby significantly simplifying the geometry calculations in (3.34) – (3.36). To save some memory on expense of computation, only w_t and w_q of (3.42) are necessary to tabulate since the four interpolation weights then are easily calculated using two subtractions and four multiplications.

A ROM table for traditional backprojection would require $N_x N_y N_z N_\beta \in \mathcal{O}(N^4)$ entries, but as pointed out by Buck, Maisl, and Reiter (1996), it may be reduced to $N_\phi N_r N_z \in \mathcal{O}(N^3)$ entries if the voxels to be reconstructed are placed on a cylindrical grid, (ϕ, r, z), centred on the axis of rotation. The projection geometry is then independent on the projection angle, provided that $\Delta\beta$ is a multiple of $\Delta\phi$. Buck et al. call their implementation of a traditional backprojection utilising such a pre-calculated table The Cylinder Algorithm. It is only mentioned here to point out that it is possible to perform traditional backprojection with a table of $\mathcal{O}(N^3)$



(a) (b)

Figure 3.19 *(a) The projection curve of a voxel and its projections onto the (θ, t)-, (θ, q)-, and (t, q)-planes. (b) The $\frac{\pi}{8}$-links needed for this curve. Note how a tq-slice of the lattice structure of links surrounding the ideal curve typically is 3×3 links wide but narrows down to the ideal 2×2 links for $\theta = 0, \frac{\pi}{2}, \ldots, 2\pi$.*

size (but still with $\mathcal{O}(N^4)$ calculations) and we will not use the cylindrical grid in our work.

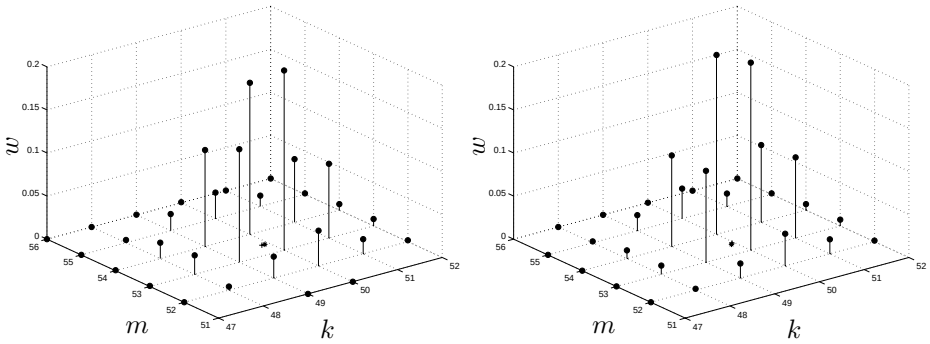
3.4.3 Iterative Interpolation

Due to the iterative interpolation, the smoothing effects of the actual interpolation kernel become shift variant and wider than the kernel used in every step. As an example, Figure 3.19 shows the $\frac{\pi}{8}$-links needed for a certain voxel, all contributing to the final voxel value.

We have studied the actual kernel in terms of how much a filtered measurement $\tilde{p}^P[n, k, m]$ contributes to the voxel $(x_{i_1}, y_{i_2}, z_{i_3})$. This contribution is denoted $w[i_1, i_2, i_3, n, k, m]$. For a specific voxel, the actual kernel has its support on an intestine shaped volume around the projection curve of the voxel, being ideally thin at $\theta = 0, \frac{\pi}{2}, \ldots, 2\pi$. We searched for the voxel and the projection angle where the actual kernel was at its widest, i.e. where it had the largest variance in the (t, q)-plane, expressed as

$$\max_{i_1,i_2,i_3,n} \sum_{k'} \sum_{m'} w[i_1, i_2, i_3, n, k', m'] \big((k' - k(x_{i_1}, y_{i_2}, \theta_n))^2 + (m' - m(x_{i_1}, y_{i_2}, z_{i_3}, \theta_n))^2 \big) \tag{3.52}$$

The ideal kernel centre $(k(x, y, \theta), m(x, y, \theta))$ is given by combining (2.10) and (3.40) with (2.17) and (3.24). This actual kernel is shown in Figure 3.20 (a). The



(a) Bi-linear interpolation (b) Bi-cubic interpolation

Figure 3.20 *The widest actual kernel for $N_\theta = 128$. The plus indicates the centre of gravity and the asterisk the ideal kernel centre. The geometry parameters are $(x, y, z) = (0.4, 0.5, 0.15)$, $R = 2.0$, and $n = 107$.*

centre of gravity of the actual kernel is very close to the ideal kernel centre, with a deviation never exceeding a quarter of a grid unit.

By replacing the triangular interpolation kernel of the bi-linear interpolations in (3.42) with a sharper 2-point cubic kernel as in

$$\begin{aligned} w_t &= 2(k_{\text{mid}} - \lfloor k_{\text{mid}} \rfloor)^3 - 3(k_{\text{mid}} - \lfloor k_{\text{mid}} \rfloor)^2 + 1, \quad w'_t = 1 - w_t \\ w_q &= 2(m_{\text{mid}} - \lfloor m_{\text{mid}} \rfloor)^3 - 3(m_{\text{mid}} - \lfloor m_{\text{mid}} \rfloor)^2 + 1, \quad w'_q = 1 - w_q \end{aligned} \tag{3.53}$$

the resulting actual kernel becomes somewhat sharper as shown in Figure 3.20 (b). If the interpolation weights are pre-calculated and tabulated, there is no performance loss in using this seemingly more complicated bi-cubic interpolation.

3.4.4 Hardware Implementation

This section presents a sketch of two possible hardware architectures for implementation of the algorithm. These sketches should not be taken as serious hardware designs, but are included to illuminate how the calculations may be organised and optimised. We concentrate on the basic step of the algorithm and assume that the filtering, rebinning and final assembly of voxel values from $\frac{\pi}{2}$-links is performed elsewhere.

For simplicity and clarity, we neglect the optimisations of the size of the ROM discussed in Section 3.4.2 and assume that the children and interpolation weights of *all* links are tabulated. The architectures presented can easily be expanded to incorporate the important optimisations.

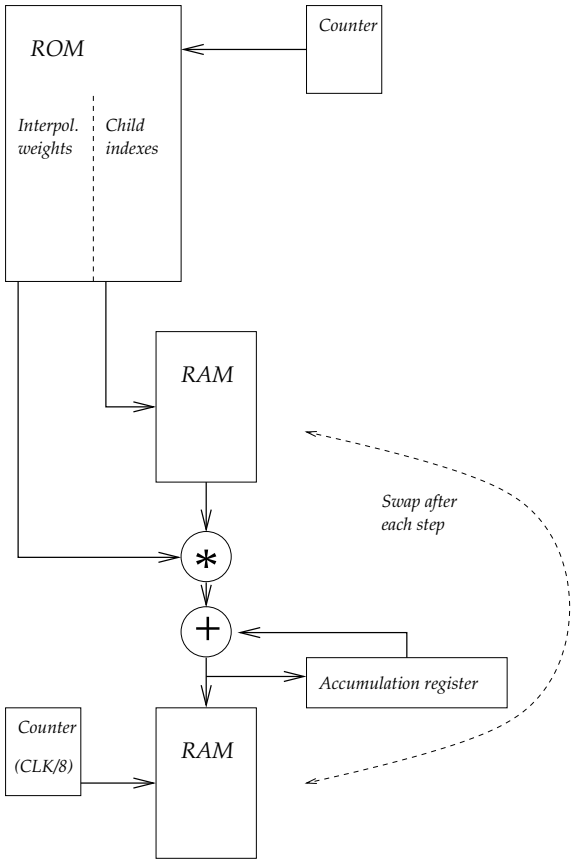


Figure 3.21 *A simple but inefficient hardware architecture performing the back-projection core of the algorithm.*

Calculation of one link value every eighth clock-cycle

A first approach to a hardware architecture is shown in Figure 3.21. The pre-computed interpolation weights and pointers to the children links are stored in a large ROM with one child pointer and the corresponding interpolation weight per address. Each basic step therefore uses eight words of the ROM. These eight-word-blocks of a link only have to be sorted according to the length of the link, the internal order of all links of the same length is arbitrary. The link values are stored in a RAM that is partitioned into two swappable halves as discussed earlier. Initially, the top half of the RAM is filled with the filtered projection data. For each clock-cycle then on, the link value of a child is weighted and accumulated. Every eighth clock-cycle a new complete link value is ready, and then the accu-

mulation register is cleared and the counter addressing the lower half of the RAM is incremented by one. After one step of the algorithm, i.e. when the values of all links of a certain length have been calculated, the upper and lower halves of the RAM are swapped. This swap is preferably realised by some addressing logic of the RAM, instead of an actual data transfer. When the execution is completed, the link values of all $\frac{\pi}{2}$-links are stored in the bottom half of the RAM, ready to be assembled into voxel values by some other piece of hardware or software.

The content of the ROM is read in sequential order and therefore lends itself to be stored on a secondary storage device, e.g. a hard drive. Two shortcomings of this simple approach are that it needs eight clock-cycles for each link value to be calculated and that all interpolation weights are stored twice in the ROM.

Calculation of one link value per clock-cycle

In 2-point one-dimensional interpolation, we always use one grid point to the left and one to the right of the desired point. In other words, we use one grid point with an even index and one with an odd index. If we generalize this observation for our basic step, we notice that the eight short links of length l, always can be divided into the following eight disjoint classes:

- θ_{n_1}/l is even, k_2 is even, m_2 is even
- θ_{n_1}/l is even, k_2 is even, m_2 is odd
- θ_{n_1}/l is even, k_2 is odd, m_2 is even

  $\vdots$

- θ_{n_1}/l is odd, k_1 is odd, m_1 is odd

with exactly one link in each class. It is therefore possible to partition each of the two halves of the link value RAM into the same eight classes and thus perform the complete interpolation step in one clock-cycle as shown in Figure 3.22.

The new link values are stored in the correct section of the lower half of the RAM. The decision on which section is correct does not have to be performed at run time. It can be achieved by pre-sorting the links in the ROM in a suitable order such that the three least significant bits of the counter addressing the lower half of the ROM can simply cycle through the eight sections as indicated in the figure.

An architecture incorporating the θ-independence discussed in Section 3.4.2 would only require an $\mathcal{O}(N^3)$-sized ROM. For step $i = 2, \ldots, \lg N_\theta - 1$, the counter addressing the ROM would then increment every $N_\theta 2^{1-i}$:th clock-cycle and a multiple of a pre-calculated offset would be added to all RAM addresses.

The required size of each RAM segment is decided by the number of 2-links and would according to Table 3.3 be $49 \cdot 10^6 < 2^{23}$ for $N = 128$. The pointers in

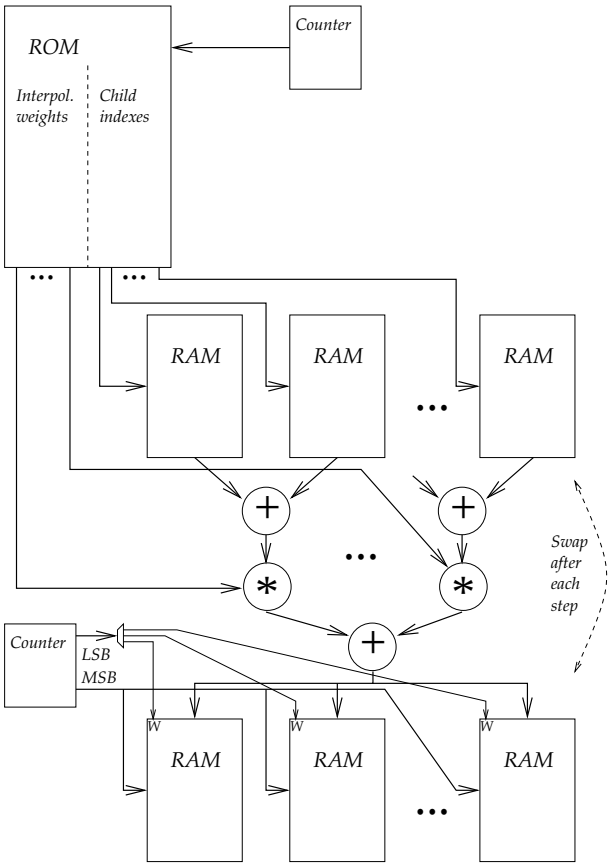


Figure 3.22 *A possible hardware architecture calculating one link value per clock cycle.*

the ROM would therefore require 23 bits each. Consider the link values of the first four RAM segments and assume that the pre-sorting of the ROM is performed so that they are sorted according to the descending priority $\theta_{n_1}, t_{k_1}, q_{m_1}, t_{k_2}, q_{m_2}$. The four first child value pointers would then have approximately the same value for a given parent. It is therefore very efficient to store the difference, the *offset*, to the first pointer for pointer 2, 3, and 4 instead of the actual pointer values. The same scheme can be used for the last four RAM segments if they sorted in the descending priority $\theta_{n_2}, t_{k_2}, q_{m_2}, t_{k_1}, q_{m_1}$.

There are several possibilities of further parallelisation. The reconstruction volume can be divided at the mid-plane into two halves, which may be reconstructed separately, using projection data from the upper half and the lower half of

the projection rows respectively. It can further be observed that no data is shared in the computation of $\frac{\pi}{2}$-links with roots in different θ-coordinates. Hence, the projection data may be segmented into the following eight disjoint groups:

- $0 \leqslant \theta < \frac{\pi}{2}, \quad q < 0$
- $0 \leqslant \theta < \frac{\pi}{2}, \quad q \geqslant 0$
- $\frac{\pi}{2} \leqslant \theta < \pi, \quad q < 0$

$\vdots$

- $\frac{3\pi}{2} \leqslant \theta < 2\pi, \quad q \geqslant 0$

with hardware performing the backprojection core of the algorithm on each one of the groups in parallel.

If both acquisition and reconstruction is seen as one system, the total data throughput is delayed due to the rebinning to oblique parallel beams. The first oblique parallel projection is available after a source rotation of $2\gamma_{\text{max}}$, but from then on the data rate is constant. The fast backprojection may start once a quadrant of oblique parallel data is measured.

An architecture requiring less memory

The main objective to pre-calculate the ROM was to simplify and optimise the pruning of unnecessary links. This pruning information is the only part of the ROM that really requires the top-down construction approach. We will now sketch an architecture that decreases the ROM size by only storing the pruning information and calculating the interpolation weights and link value addresses on the fly.

We imagine a pruning table with only one bit per possible link that indicates whether the link value should be evaluated or not. The entropy of such a pruning table is low and a simple run-length coding would reduce its size considerably. The unpacking of the run-length coding is essentially instant and the table could therefore be used to create a stream of $(t_{k_1}, q_{m_1}, 0; t_{k_2}, q_{m_2}, \theta_{n_2})$ link identifiers. Equations (3.35)-(3.42) could then be used to determine the identifiers of the eight children and the corresponding interpolation weights. As discussed earlier, these equations may be simplified by setting $\theta_{n_1} = 0$ and then use the same interpolation weights for the $(\frac{N_\theta}{n_2} - 1)$ duplicate links of the same length.

The remaining problem is the addressing of the link value RAM. A straightforward addressing scheme would require a $\mathcal{O}(N^4)$-sized RAM, with mainly unused addresses, since there only exist $\mathcal{O}(N^3)$ unpruned links in each step. However, since the pruning is known in advance, we can design a hashing scheme with pre-calculated look-up tables so that a $\mathcal{O}(N^3)$-sized RAM would suffice.

Source trajectory radius	$R = 2.0$ length units (l.u.)
Projections per turn	$N_\beta = N_\theta = 256$
Detector rows	$N_q = 125$
Detector row height	$\Delta q = \frac{2}{125}$
Detector elements per row	$N_\gamma = N_t = 125$
Maximum fan-angle	$\gamma_{\max} = 30^\circ$
Maximum cone-angle at $\gamma = 0$	$\kappa_{\max} = 26.6^\circ$
Reconstruction grid	$N_x \times N_y \times N_z = 125 \times 125 \times 63$
Reconstruction grid resolution	$\Delta x = \Delta y = \Delta z = \frac{2}{125}$ l.u.
Simulated rays per detector element	9

Table 3.5 *Experiment parameters for Figures 3.24 and 3.25.*

3.4.5 Experimental Results

We have reconstructed the three-dimensional Shepp-Logan as defined in Appendix B.2 using FDK-C, FDK-P, and FDK-FAST. The results in Figure 3.24 show no apparent visible difference in image quality between the three techniques. As mentioned above, the intensity drop for large cone angles is a well-known artifact of all FDK-like algorithms.

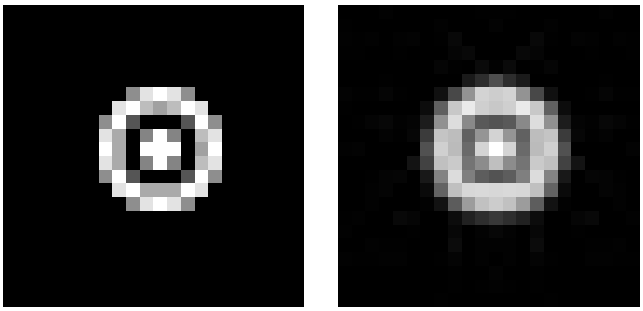
To obtain a better measurement of the effects of the iterative interpolation, we have used the methodology of Grangeat, Le Mason, Melennec, and Sire (1991) and Rizo, Grangeat, Sire, Le Mason, and Melennec (1991) to study the modulation transfer function (MTF) of the reconstruction. This function indicates how well different frequencies are reconstructed which in our case is a space dependent property. A small test object dominated by the wanted frequency is reconstructed and the reconstructed response for this frequency is measured relative the phantom. The procedure is repeated for each frequency to be measured. A set of small concentric spheres centred on $(x, y, z) = (0, 0.4, 0)$ was used as test objects. See Figure 3.23. The results in Figure 3.25 show that the parallel rebinning does not affect the MTF. A clearly visible decrease in response for high frequencies is on the other hand seen for PI-FAST. This is due to the iterative interpolation. Frequency response is dependent on many factors, one being the ramp-filter used. We normally use the frequency weighted ramp-filter

$$g_2^P[k] = \frac{1}{2\pi}\left(\operatorname{sinc}\left(\frac{k+0.5}{2}\right)\cos\left(\pi\frac{k-0.5}{2}\right) + \operatorname{sinc}\left(\frac{k-0.5}{2}\right)\cos\left(\pi\frac{k+0.5}{2}\right)\right) \quad (3.54)$$

but in Figure 3.25(c)–(f), the band-limited pure ramp-filter

$$g_1^P[k] = \frac{1}{2}\operatorname{sinc}(k) - \frac{1}{4}\operatorname{sinc}^2\left(\frac{k}{2}\right) \quad (3.55)$$

is also used for comparison. The latter filter results in somewhat sharper images.



(a) Voxelized ideal phantom

(b) Reconstructed result using fast backprojection

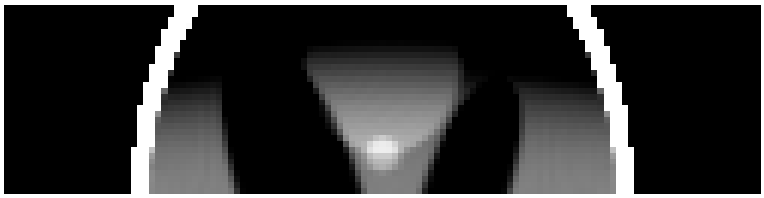
Figure 3.23 *Test object for calculation of the modulation transfer function.*

3.5 Discussion

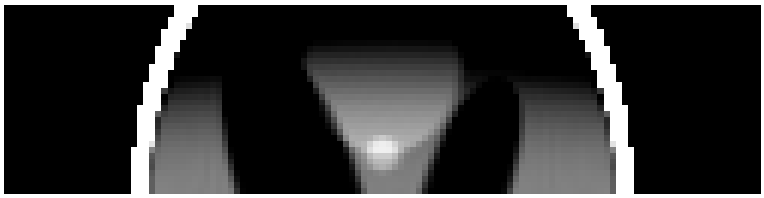
The FDK algorithm has dominated circular cone-beam reconstruction since it was introduced in 1984. Relatively little effort has been put into improving it. The paper of Grass et al. (2000a) which is an outgrowth of the work of Turbell (1999) shows that their algorithm FDK-HT not only produces reconstruction results superior to FDK but also is computationally more efficient and is able to reconstuct a larger volume than FDK, elegantly confined within a cylinder. Their image quality observations are validated by the experiments in this chapter.

We have tried to improve FDK even further with FDK-SLANT. The results are promising in some ways and discouraging in others. If one algorithm is to be recommended it is bound to be FDK-HT.

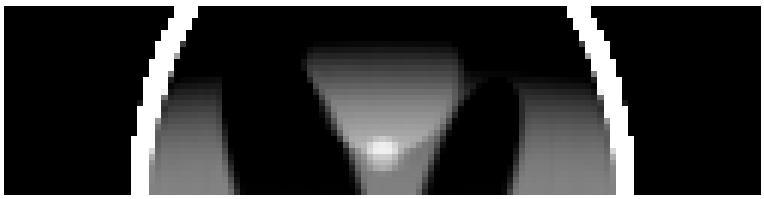
We are somewhat ambigous about the practicality of FDK-FAST. The memory requirements, the implementation complexity, the inherent smoothing effects, and the rather modest decrease in computation time are obvious shortcomings. We have nevertheless included it here as it is the first fast backprojection algorithm for the cone-beam case. It also includes some techniques that will be used in PI-FAST presented in Section 4.3.4.



(a) FDK-C

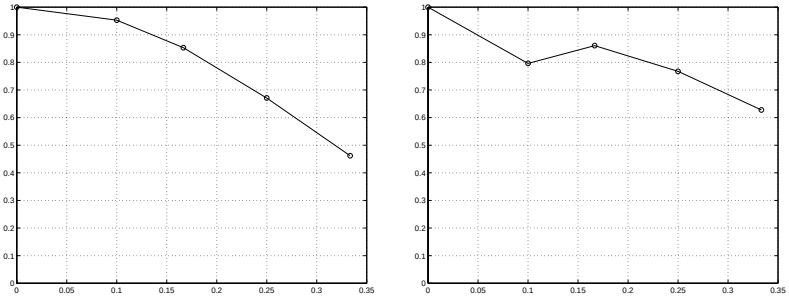


(b) FDK-P

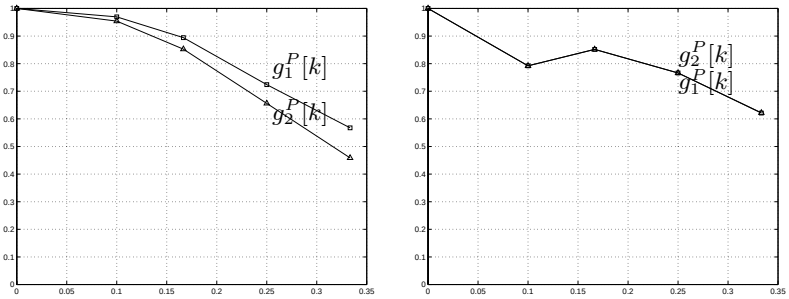


(c) FDK-FAST

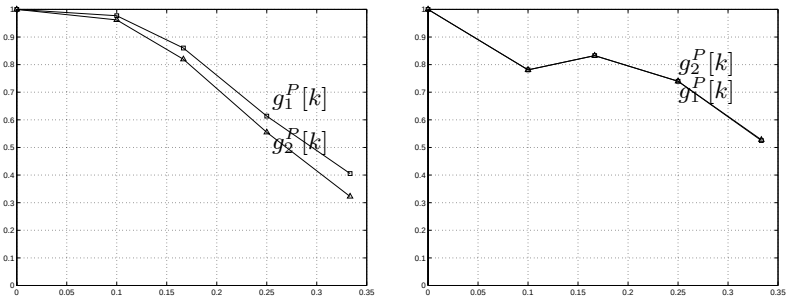
Figure 3.24 *Reconstruction results of the Shepp-Logan phantom.* (x, z)*-slices at* $y = 0.25$ *for* $x \in [-1, 1]$ *and* $z \in [0, 0.5]$. *Grayscale interval* $[1, 1.04]$. *Other parameters as in Table 3.5*



(a) FDK-C, MTF_x (b) FDK-C, MTF_z



(c) FDK-P, MTF_x (d) FDK-P, MTF_z



(e) FDK-FAST, MTF_x (f) FDK-FAST, MTF_z

Figure 3.25 *Modulation transfer functions. In (c)-(f) two different kind of ramp-filters, defined in (3.54) and (3.55), were used. Other parameters as in Table 3.5*

4

Helical Cone-Beam Reconstruction

The helical source trajectory is natural for volume scanning of long objects. A continuously translated object and a rotating source-detector system yield a helical source trajectory around the object. Helical scanning has been used for many years with one-dimensional detectors, and has now been extended for use with multi-row detectors. Some radiologists, e.g. Berland and Smith (1998), claim that the new helical multi-rows systems could be as important as the advent of single-row helical CT was ten years ago. Manufacturers promise simultaneous improvements in scanning speed, z-resolution, and image noise, but there is in fact a trade-off between these factors.

4.1 Exact Methods

Exact reconsruction from helical cone-beam data was an open problem when the work of this thesis began in 1996. Important contributions by Tam (1995) and Kudo, Noo, and Defrise (1998) lead to developments of exact algorithms able to cope with long objects not fully covered by the helix. Recent efforts aim to make these computationally expensive algorithms more efficient.

Table 4.1 lists some important contributions in the area. The table classifies the contributions into the following three cases: reconstruction from untruncated data, already discussed in Section 3.1, reconstruction of short objects from truncated data, discussed in Section 4.1.2, and reconstruction of long objects from truncated data, discussed in Section 4.1.3. Short objects are fully covered by the helical scan whereas long objects extend outside the helix in the z-direction.

	Radon-based	**Filtered backprojection**
Untruncated projections *(arbitrary trajectory)*	Grangeat (1987)	Defrise and Clack (1994)
Truncated projections, short object	Tam (1995)	Kudo et al. (1998)
Truncated projections, long object *(with additional circlular trajectories)*	Tam (1995)	Tam et al. (2000)
Truncated projections, long object *(without additional circlular trajectories)*	The PHI method (Schaller et al. 1999)	The MS-CB-FBP method (Kudo et al. 1999); The ZB method (Defrise et al. 2000); The PHI method (Schaller et al. 1999)

Table 4.1 *Algorithms for exact helical reconstruction.*

4.1.1 The Helix Geometry

The geometry of helical cone-beam scanning is a minor augmentation of the circular cone-beam geometry. See Figure 4.1. The pitch, P, is defined as the distance in the z-direction of two points on the helix that are exactly one turn apart. It is measured in length units. Many variations on the pitch concept exist in the literature. One of these is the relative pitch, defined as $\frac{P}{\Delta q}$ or sometimes as $\frac{P}{N_q \Delta q}$. We use the integer parameter $\lambda = 0, \ldots, N_\lambda - 1$ to distinguish between the N_λ turns of the helix. The source S is moving constantly in the z-direction with the projection angle β as

$$z_S(\beta) = \beta \frac{P}{2\pi} + z_{S_0} \tag{4.1}$$

where z_{S_0} denotes the source starting position. We use N_β to denote the number of projections for one turn as in the two-dimensional case, and hence we have a total of $N_\lambda N_\beta$ projections.

All exact algorithms are derived using a planar detector (a, b) as shown to the right in Figure 4.1. In the figure the detector is placed at a distance of $3R$ from the source but it is customary to define it on the axis of rotation. The coordinate system (a', b'), obtained by rotating the (a, b) system by $\eta = \arctan \frac{P}{2\pi R}$, will be shown to be useful. The rotation is such that the a'-axis is parallel with the source velocity vector.

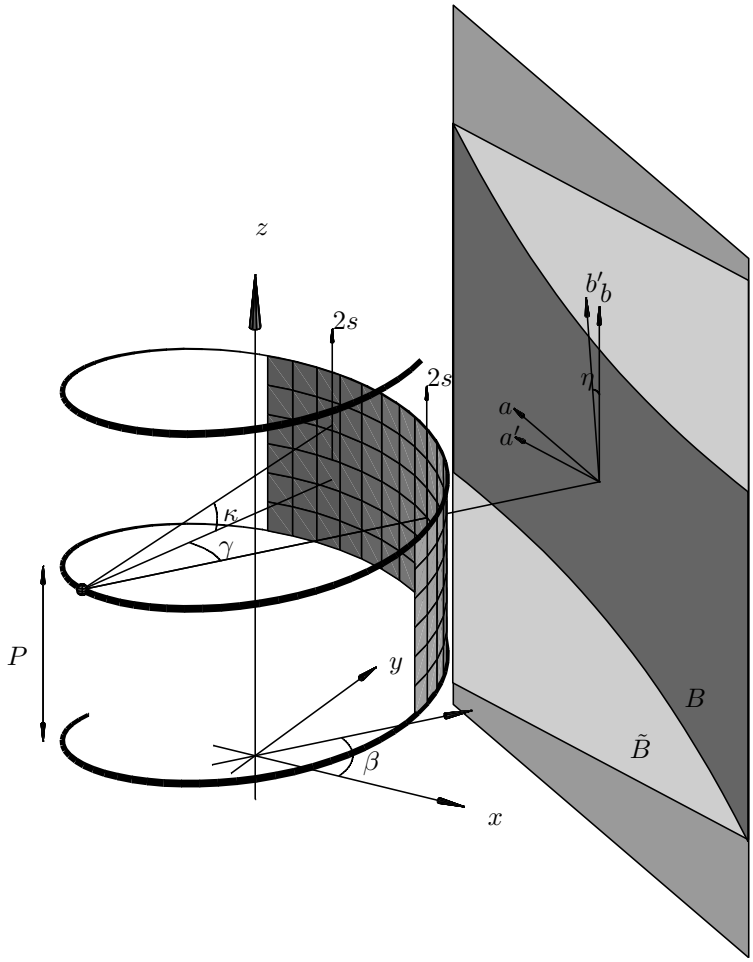


Figure 4.1 *The PI-detector and its projection on a planar detector.*

4.1.2 Reconstruction of Short Objects

Grangeat's method for exact cone-beam reconstruction, briefly mentioned in Section 3.1, is not restricted to a circular source trajectory, but can also be used for the helical case. An implementation using linograms is described by Eriksson (1998). The main limitation of a direct use of the Grangeat method is that the projections need to be untruncated. To overcome this limitation Tam (1995), Eberhard and Tam (1995), and Tam, Samarasekera, and Sauer (1997) suggested the use of a specially shaped detector window. Projected as a detector onto the same cylinder as

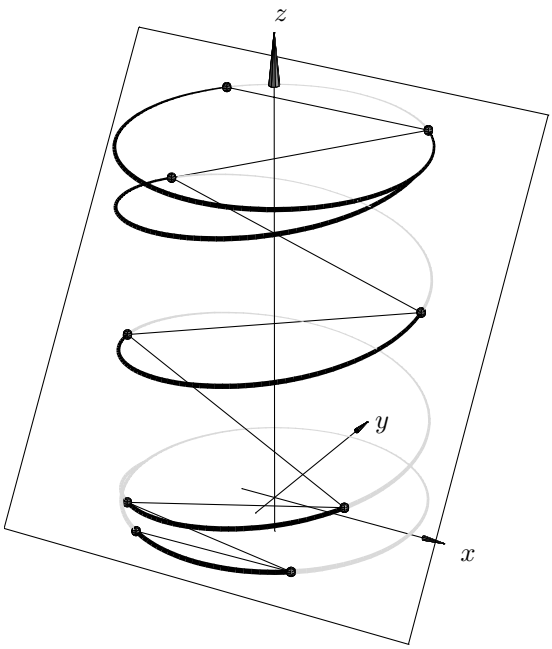


Figure 4.2 *Any plane can be triangulated by the source points intersecting the plane. To avoid truncation problems when scanning long objects, the helical source trajectory is augmented with two extra circles.*

the helical source trajectory, it takes the appearance shown in Figure 4.1. The window is limited by two consecutive helix turns in height and by two vertical lines in width. We refer to this detector window as the Tam window and, for reasons apparent in Section 4.3, to detectors shaped accordingly as PI-detectors. Note that the physical detector still may be planar or cylindrical around the source, as long it is limited by this window. The projection of the Tam window onto the planar detector (a, b) is confined within the curves

$$b_1(a) = P(1 + \frac{a^2}{R^2}) \cdot \left(\frac{\arctan \frac{a}{R}}{2\pi} + \frac{1}{4} \right) \tag{4.2}$$

and

$$b_2(a) = P(1 + \frac{a^2}{R^2}) \cdot \left(\frac{\arctan \frac{a}{R}}{2\pi} - \frac{1}{4} \right) \tag{4.3}$$

The first step of Grangeat's method is to calculate the derivative of all plane integrals through the object. Figure 4.2 shows how an arbitrary plane can be triangulated by the source points at the intersection of the source trajectory and the plane. For the moment, we ask the reader to ignore the two circles at the top and

the bottom of the helix in this figure. The Tam window limits the measured rays perfectly so that all areas of the plane are covered exactly once. Thus, in principle, the complete derivative of the plane integral can be obtained by adding the contributions from all triangular patches. As a corollary we conclude that the PI-detector is sufficient for exact reconstruction.

Kudo, Noo, and Defrise (1998) have showed that a mere addition of the contributions from the triangular patches will not produce an exact result. A small but essential boundary term must be included in the piece-wise computation of the Radon plane derivatives. Intuitively, this can be seen as a compensation for the fact that the triangles change area slightly when the plane is moved along its normal in the differentiation process. Also, Kudo et al. (1998) reformulated Tam's Radon-based algorithm into a filtered backprojection version, called CB-FBP, in a similar fashion to what Defrise and Clack (1994) did to the original algorithm of Grangeat (1987). The filtering is two-dimensional and shift-variant and although the unfiltered data is truncated, the support of the filtered data is unbounded. Hence, the filtered projection data has to be calculated over a detector region that covers the complete object. Conversely, each measured and truncated projection data affects all voxels in the complete object.

4.1.3 Reconstruction of Long Objects

The non-locality of the above algorithms implies that the whole object has to be scanned before reconstruction. This is unwanted in medical applications where only a section of the body is to be imaged.

Two extra circles

To handle long objects Tam (1995) suggested to add two circular paths at the start and the end of the helix, which are already included in Figure 4.2. The reconstruction may then be limited to a region of interest (ROI) restricted by the cylindrical FOV and the planes of these two circles. It is then possible to only consider rays within the ROI for all Radon planes. Without the two circles, complete data capture for a cylindrical ROI would have to employ rays passing through the object both inside and outside the ROI which would corrupt the result. Such rays are referred to as contaminated. Tam, Lauritsch, Sourbelle, Sauer, and Ladendorf (2000) formulated this Radon-based approach as a filtered backprojection. In practice, the two extra circles will become a severe burden, since they require acceleration and deceleration of the patient table.

A result by Kudo, Park, Noo, and Defrise (1999) shows that a ROI of a long object can be exactly reconstructed without the two extra circles needed by Tam (1995). The resulting algorithm uses the filtered backprojection technique of CB-FBP and is called multi-slice cone-beam filtered backprojection (MS-CB-FBP). The

authors show that the measurements from the extra circles are only needed in a boundary term calculation. Such boundary rays intersect the top or bottom of the Tam window. In other words, they intersect the helix and are therefore not only measured from the circles but also, in the opposing direction, from the helix as well. Figure 4.2 contains two examples of such rays. The only data needed from the circles can therefore be replaced by identical data measured from the helix. The circles can be thought of as virtual and actually placed anywhere along the helix. In an alternative algorithm called single-slice cone-beam filtered backprojection (SS-CB-FBP) the reconstruction is split up on a slice by slice basis. The virtual circles are placed on each side of the slice and the slice is then reconstructed. This makes the algorithm faster and more local.

The PHI method

The so-called PHI-method of Schaller, Noo, Sauer, Tam, Lauritsch, and Flohr (1999) avoids the extra circles by defining ROIs with shapes such that no contaminated rays are needed in the reconstruction. The shapes of the ROIs change with the vertical elevation angle ϕ of the normals of the Radon planes. This variation of the ROI shape is allowed since the Radon value backprojection algorithm of Marr, Chen, and Lauterbur (1981) only considers Radon planes of identical ϕ in its first stage. A full detailed description of the algorithm is found in Schaller et al. (2000).

Sourbelle, Lauritsch, Tam, Noo, and Kalender (2000) have compared a Radon-based implementation and a filtered backprojection implementation of the PHI-method. The filtered backprojection was found to be slightly better in all aspects.

The ZB method

The CB-FBP method discussed above only uses data within the Tam window. Defrise, Noo, and Kudo (2000) show that CB-FBP simplifies considerably if the object is such that the projection values tend smoothly to zero along the borders of the Tam window. The boundary term then becomes zero and the shift-variant filtering reduces to shift-invariant ramp-filtering along lines parallel to a'. The values within the region B contribute to ramp-filtered projection values in the parallelogram area $\tilde{B}$. These filtered data are then backprojected as in FDK and the reconstruction is consequently called B-FDK. It is essential that the original data tend smoothly to zero along the border since the ramp-filter is operating over the edge and would introduce artifacts if the transition was sharp. See Figure 2.8(b).

Projections with vanishing values along the detector borders are said to have a *zero boundary*. Note that a smooth windowing of the projection data of an arbitrary object would alter the projections and result in non-exact reconstruction. It is the *object* that has to generate projections with zero boundary. No real object fulfils the

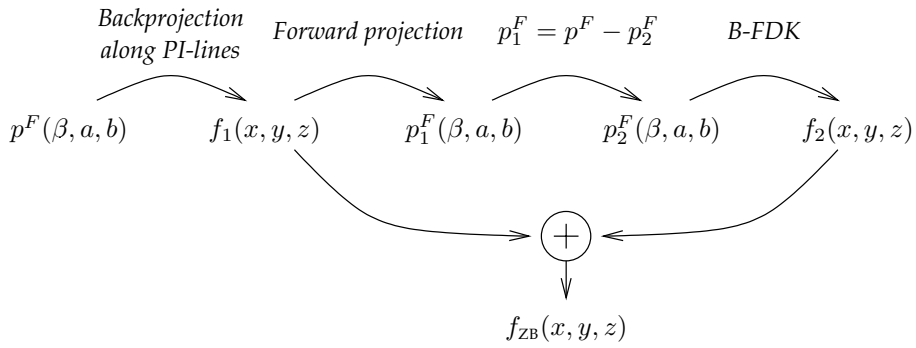


Figure 4.3 *Calculation steps in the ZB method.*

zero boundary property, but the following geometrical property of the helix makes it possible to write any object f as a sum of two hypothetical objects $f(x, y, z) = f_1(x, y, z) + f_2(x, y, z)$ where the projections of $f_2(x, y, z)$ have the zero boundary property.

We define a *PI-line* as a line that connects two points on the helix separated by less than one pitch. The lines separating the triangles in Figure 4.2 are all examples of PI-lines, apart from the four lines connected to the circles. Danielsson, Edholm, Eriksson, and Magnusson-Seger (1997) geometrically showed that each image point within the helix cylinder belongs to one and only one PI-line. An analytic proof was made by Defrise et al. (2000). The one-to-one mapping between image points and PI-lines are essential in the PI-methods discussed in Section 4.3.

The algorithm described by Defrise et al. (2000) is called the zero boundary (ZB) method. Figure 4.3 illustrates the main steps of the algorithm. The one-to-one mapping between PI-lines and image points is used to separate the projection data $p^F(\beta, a, b)$ into projection data $p_1^F(\beta, a, b)$ and $p_2^F(\beta, a, b)$ from the two objects $f_1(x, y, z)$ and $f_2(x, y, z)$ respectively. The construction of $f_1(x, y, z)$ is made by smearing back all detector border values along the corresponding PI-lines. This process can be thought of as backprojection of the border values. The one-to-one mapping ensures that all image points will receive exactly one backprojection contribution. In the actual sampled situation this is not exactly true calling for interpolations with proper normalisations. A border value may be distributed anywhere along the PI-line, as long as the integral over the PI-line is equal to the border value. In the ZB method the backprojection is weighted with a bell-shaped profile in order to obtain a smooth volume $f_1(x, y, z)$. A final smoothing with a three-dimensional Hamming filter is then applied.

The projections $p_1^F(\beta, a, b)$ are produced by numerical line integration through the constructed volume $f_1(x, y, z)$. The line integration is implemented using the method of Joseph (1982), described in Section 5.1.2.

The projections $p_2^F(\beta, a, b)$ are calculated as $p_2^F(\beta, a, b) = p^F(\beta, a, b) - p_1^F(\beta, a, b)$. The values of $p_2^F(\beta, a, b)$ will vanish along the border, allowing for reconstruction of $f_2(x, y, z)$ using B-FDK. The final result is then computed as the sum $f_{\text{ZB}}(x, y, z) = f_1(x, y, z) + f_2(x, y, z)$. However, the two volumes are obtained in different ways and therefore have unmatched spatial resolution. An extra smoothing of $f_1(x, y, z)$ is therefore applied before the summation.

With these new contributions the previously mentioned long object problem has been solved. It is indeed possible to reconstruct a ROI separately and exactly without waiting for the whole long object to be scanned.

4.2 Approximate Methods

Just as in circular tomography, the relatively complex exact reconstruction methods are often rejected in favour of more efficient, albeit approximate, ones. This section surveys several different approaches to approximate helical cone-beam reconstruction. We follow the taxonomy of Table 1.2, starting with methods employing two-dimensional backprojection and continuing with more complicated methods with three-dimensional backprojection. The three ways of handling short-scan data, discussed in Section 2.2.3, will be used for both kinds of backprojection.

4.2.1 Two-Dimensional Backprojection

We distinguish between the reconstruction algorithms for single-row detector systems and multi-row systems. The former have been in used since the introduction of helical scanning, whereas the latter are developed for and used in the new multi-row tomographs.

Single-Row Data

A typical reconstruction algorithm for single-row detector systems using two-dimensional backprojection works as follows when a slice at $z = z_{\text{slice}}$ is to be reconstructed. In general, there is no projection data measured for a ray of a specific projection angle $\beta = \beta_0$ that lies perfectly in the slice. Instead it must be obtained from linear interpolation between data from two rays emitted from the two closest turns of the helix. This is commonly illustrated as in Figure 4.4(a). These rays are 360° apart, and the method is therefore referred to as full-scan or 360°LI, where LI stands for Linear Interpolation. Once this planar projection set $p_{z_{\text{slice}}}(\beta, \gamma)$ is constructed, the two-dimensional fan-beam filtered backprojection algorithm of Section 2.2.2 can be applied.

Kalender (1995) has published a thorough investigation, comparing helical single-slice methods to older methods with circular trajectories and slice-by-slice

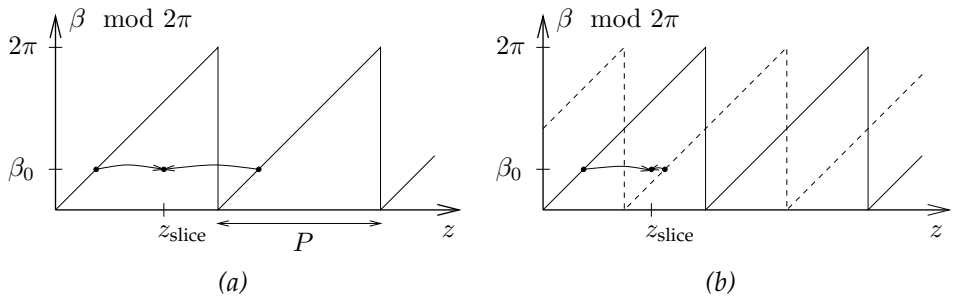


Figure 4.4 *A projection value for the projection angle $\beta = \beta_0$ in the plane $z = z_{slice}$ is obtained by linear interpolation. (a) $360°$LI. (b) $180°$LI, solid: direct ray, dashed: central complementary ray.*

scanning. He concludes that the helical methods give a better or comparable image quality in every aspect. An important difference, which also speaks in favour of helical scanning, is that the exact z-position of each single slice may be selected arbitrarily at reconstruction time without loss of image quality. This is possible since the projections produce a continuous flow of measurements, which are not predetermined to be separated into sets before reconstruction.

The three methods to handle short-scan fan-beam data discussed in Section 2.2.3 can also be applied to helical single-row reconstruction. They are then generally known as $180°$LI-algorithms.

Parallel Rebinning

Parallel rebinning produces data sets originating from parallel rays that are distributed over a helix angular interval of $2\gamma_{\text{max}}$. Unlike the original fan-beam sets, these rays can be used in both directions. Therefore, they produce twice as many parallel projections as original fan-beam projections. The backprojection of a slice utilises such projections over an angular θ-interval of π, which correspond to original fan-beam data captured over a β-interval of $\pi + 2\gamma_{\text{max}}$. On the average, the interpolation distance in the z-direction is then only half of what it is in the $360°$LI-algorithm.

Complementary rebinning

Complementary rebinning produces data sets originating from a fan of rays measured at different projection angles. See Figure 4.5. The complementary virtual fan-beam source will not be a single point, but a line segment parallel to the z-axis. Still, the rays in this beam are parallel to the (x, y)-plane and their end points

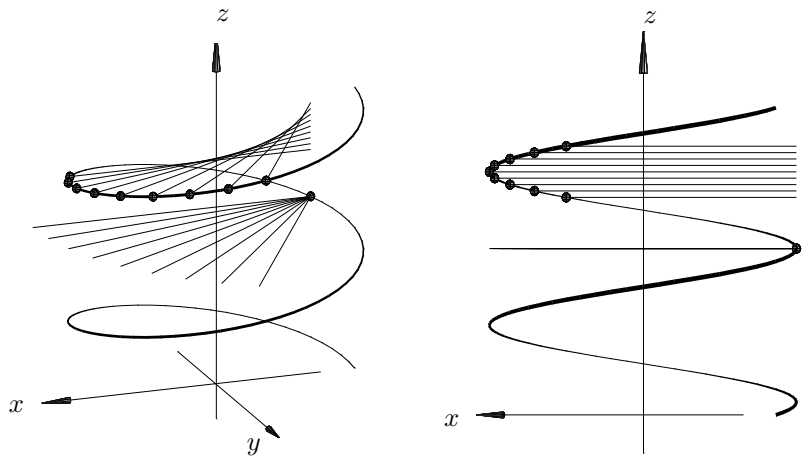


Figure 4.5 *Perspective and side view of a complementary fan above a direct fan. The spheres indicate the direct source positions. The complementary source becomes a line segment. Compare with the two-dimensional case in Figure 2.6.*

.

are positioned on the original helix. The central ray ($\gamma = 0$) in a complementary fan will be positioned exactly halfway in the z-direction between its two nearest direct fans. Other rays are more than halfway or less than halfway from rays of the same direction in the two nearest direct fans. Therefore, on the average, the z-interpolation width is halved by interpolating between one direct and one complementary fan as shown in Figure 4.4. The reconstruction of one slice uses original data from the same short acquisition interval $\pi + 2\gamma_{\max}$ as in the previous case which means rays as close as possible to the desired slice.

Smooth sinogram windowing

The smooth sinogram windowing technique is also possible to employ for helical single-row reconstruction but does not decrease the z-interpolation distance and we refrain from outlining the details here.

Multi-Row Data

All of the above methods for single-row data are naturally extended to multi-row detectors. The z-interpolation is then still performed between the two measured rays that are closest, in some sense, to the desired ray. These may now belong to different rows of the same or adjacent turns of the helix. Figure 4.6(a) shows how this interpolation is performed for a 3-row detector. For the non-central fans

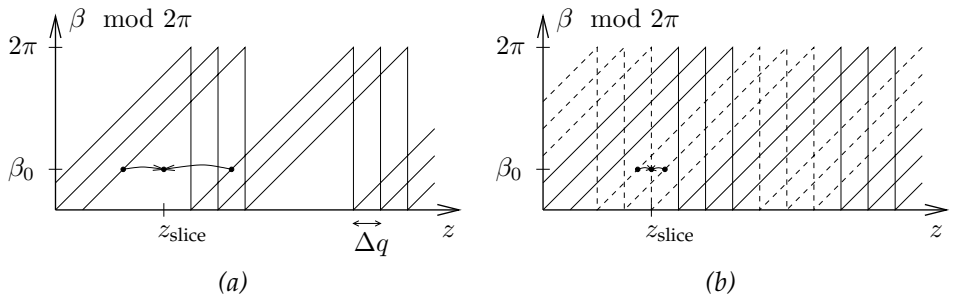


Figure 4.6 *Interpolation in the z-direction for a 3-row detector with pitch $P = 6\Delta q$. The lines indicate the z-coordinate of the intersection of the central ($\gamma = 0$) ray and the z-axis. (a) $360°$LI. (b) $180°$LI, solid: direct ray, dashed: central complementary ray.*

($q \neq 0$), the z-position varies along each ray, so we only indicate the z-coordinate of the intersection of the z-axis and the central ray of each fan in the figure.

Taguchi and Aradate (1998) have investigated different ratios between the pitch and the detector row width, Δq. For smaller pitches the saw-tooth curves in Figure 4.6 overlap and interlace. They claim the sampling to be "optimal" for non-integer pitches such as $2.5\Delta q$ or $3.5\Delta q$ for a 4-row detector. Wang and Vannier (1999) investigate the role of the pitch using the concept of sensitivity of signal reconstruction. Their conclusion is also to avoid pitches of integer multiples of Δq. Both of the above articles consider the situation in a small neighbourhood along the z-axis where only the central rays of the fan are used. The situation is more complex for non-central voxels. This observation together with the fact that one manufacturer (Hu 1999) has chosen to use a integer pitch of $P = 6\Delta q$ as a standard value indicates that the above recommendations of "optimal" pitch should be taken with a pinch of salt. A complete simulation including the full reconstruction algorithm and evaluations of the dose and the resulting image quality is necessary for an optimal choice of pitch.

As mentioned, all three ways of handling short-scan data have been extended to the multi-row case. Here follows a presentation compliant with the order in Table 1.2.

Parallel Rebinning

Rebinning to parallel projections was investigated by Schaller, Flohr, Wolf, and Kalender (1998) and seems to give a flexible and fast algorithm. This algorithm also uses a interpolation kernel with a modifiable width to control the mentioned trade-offs.

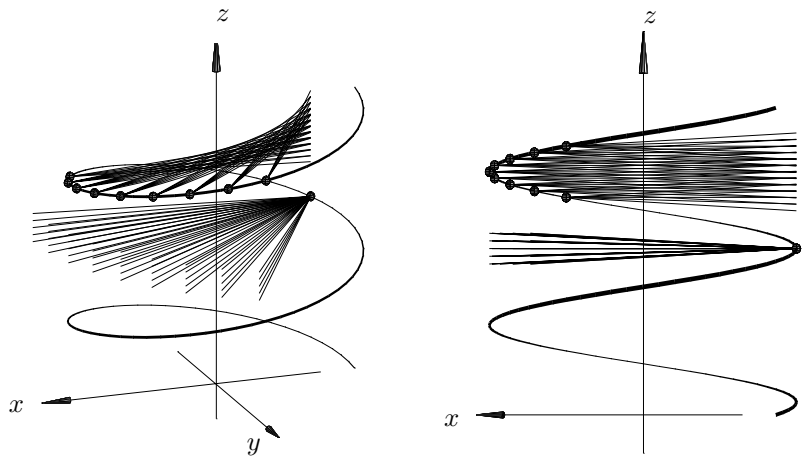


Figure 4.7 *Perspective and side view of a complementary cone above a direct cone. The spheres indicate the direct source positions.*

Complementary rebinning

A complementary projection set $p^C(\beta, \gamma, q)$ is constructed as

$$p^C(\beta, \gamma, q) = p(\beta + \pi + 2\gamma, -\gamma, q) \tag{4.4}$$

Note that the row coordinate q is left unchanged. This results in a complementary cone as shown in Figure 4.7.

An example of the resulting z-interpolation is shown in Figure 4.6(b). We notice that the interpolation is sometimes performed between data from different turns, sometimes between direct and complementary data, which may introduce artifacts. Furthermore, for certain high-speed, low-resolution imaging modes the slice thickness indicate that the interpolation function should extended over more than two rows of data, some of which might be direct, some complementary. This problem has been studied by Taguchi and Aradate (1998), who propose an interpolation function stretching over several detector rows, or rather several measurements in the z-direction. See Figure 4.8. Each interpolation is individually normalised so that the weights of the linear combination of filtered projection values add up to unity. Clearly, many trade-offs between fast scanning, high z-resolution and signal-to-noise ratio can be achieved with appropriate interpolation kernels.

Smooth Sinogram Windowing

A multi-row algorithm called Cone-Beam Short-Scan Rebinning (CB-SSRB) employing smooth sinogram windowing was presented by Noo, Kudo, and Defrise

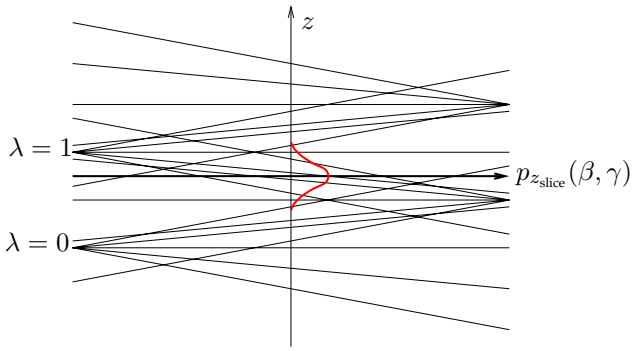
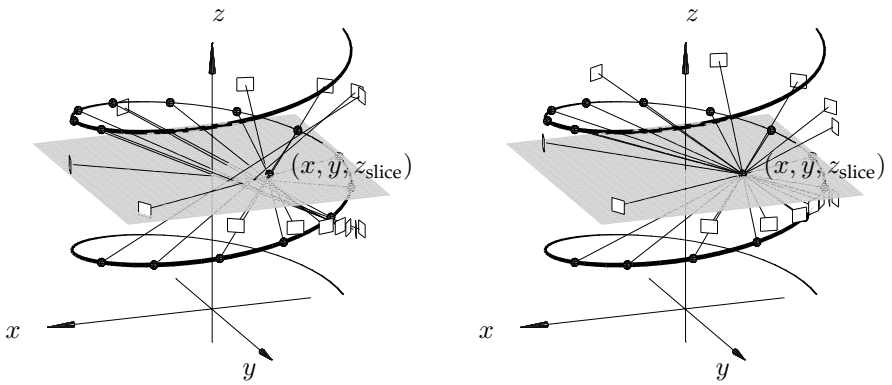


Figure 4.8 *A kernel with a modifiable shape and width is used in the z-interpolation.*

(1998). A two-dimensional short-scan fan-beam projection data set is assembled for each specific slice using only the single closest turn of the helix. For each (β, γ) within this short-scan interval one ray along the detector column is picked. This ray is chosen such that it intersects the slice exactly halfway between its two intersections with the cylindrical field of view (FOV). See Figure 4.9(a). This is in a sense the ray closest to the given slice. Normally, such a ray hits the detector between two rows, which requires interpolation to get the wanted projection value. The reconstruction of the slice then comprises pre-weighting with $\cos\kappa$, smooth sinogram windowing, ramp-filtering, and two-dimensional backprojection. Noo et al. (1998) show that CB-SSRB requires a detector that is slightly larger than the minimal Tam window.



(a) Two-dimensional backprojection.

(b) Three-dimensional backprojection.

Figure 4.9 *Ray positions of filtered data used to reconstruct the voxel* (x, y, z_{slice}). *[Based on a similar plot by Thomas Köhler and Roland Proksa.]*

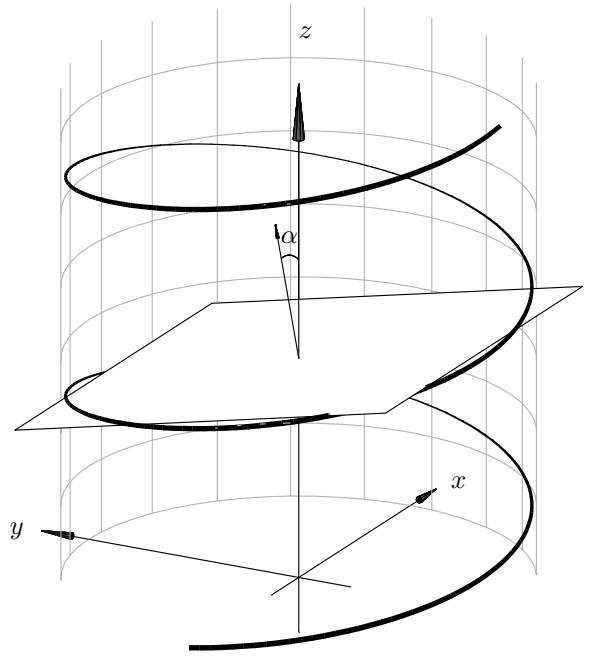


Figure 4.10 *By tiliting the slice to be reconstructed the geometrical mismatch between measuring and reconstucting rays is decreased.*

4.2.2 Nutating Surface Reconstruction

The only severe approximation used in the methods employing two-dimensional backprojection is the assumption that the original rays are fully positioned in the slice, while they in reality intersect the slice at more or less oblique angles. This approximation becomes critical, however, since it introduces a geometrical mismatch between measuring rays and reconstructing rays. Larson, Ruth, and Crawford (1998) describe a way to reduce the mismatch by tilting the slice so that it no longer is parallel to the plane. See Figure 4.10. Regard one projection angle, which will be the central projection angle of the short scan interval used to collect data for the slice. One axis of the slice is identical to the central ray of this projection and hence parallel to the (x, y)-plane. The other axis of the slice is tilted. The tilt angle α between the normal of the slice and the z-axis is constant but the slices are rotated relative to each other along the z-axis, forming a nutating set of planar slices.

The optimal tilt angle α^* is found as follows. Object points in a slice with this tilt are projected by the cone-beam onto a detector (γ, q) on the source-centred

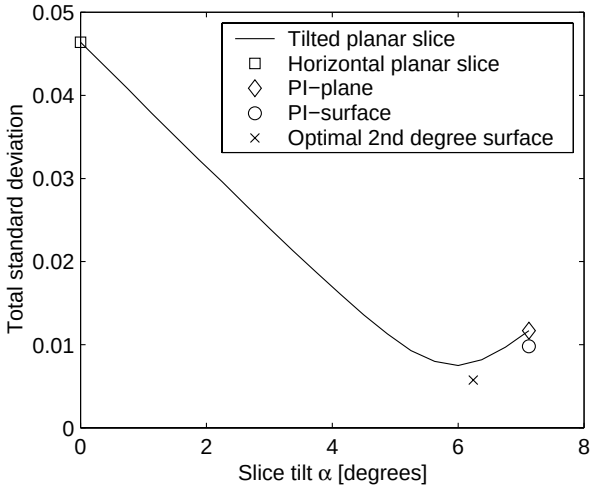


Figure 4.11 *Total standard deviation of the projection of image points indside the FOV from the optimal filtering curves as a function of the slice tilt for* $R = 2$, $R_{FOV} = 1$, $P = 1$. *The PI-plane and PI-surface are discussed in Section 4.3.2.*

cylinder for all projections employed in half-scan reconstruction. For each projection angle β, the projection spread out over a banana-shaped area on the detector. For each column in the detector the variance $\sigma_\alpha^2(\beta, \gamma)$ in the q-value is computed. The sum over all projection angles β and all fan angles γ of the variances is taken as an error measure and minimised with respect to α. See Figure 4.11. Once this optimal tilt angle α^* is found, the point of gravity $q_{\alpha^*}(\beta, \gamma)$ is computed for each projection angle β and each column γ yielding a set of curves $q_{\alpha^*,\beta}(\gamma)$ on the detectors. These curves are pre-computed for the given scanning geometry. The reconstruction algorithm then starts with interpolating data along the curves $q_{\alpha^*,\beta}(\gamma)$ on the detectors. From the resampled detector values a two-dimensional short-scan fan-beam sinogram is assembled for each nutating slice. This is followed by rebinning to parallel data, ramp-filtering and two-dimensional backprojection. The final voxels sampled on a Cartesian grid are obtained from interpolating, in z-direction only, between the reconstructed nutating slices.

It is important to note that the reconstruction grid of the nutating surfaces only is modified in the z-direction. The sampling distances in x, y, t, and are kept constant, so that seen along the z-axis, the situation is identical to the non-nutating methods. Identical ramp-filters are thus employed, despite the fact that filtering is performed along curves of different physical lengths.

The idea of tilting the slices has also been proposed by Heuscher (1999), Kachelriess, Schaller, and Kalender (2000b) and Turbell and Danielsson (1999a). As far

as we know, these proposals were made independently of each other, although Larson et al. (1998) were undoubtedly first. The methods are very similar, with slightly different definitions of the optimal shape and tilt of the nutating surfaces.

Heuscher (1999) describes an iterative approach to construct a non-planar surface that has minimal dispersion in the z-direction when projected onto the detector for a half-scan interval. The iteration starts from the centre of the surface and continues outwards radially. Once the surface is found the algorithm becomes principally identical to the Larson algorithm above. The optimality of the surface found by the above methodology is not proved.

To get a better control of the optimality we here propose to use a polynomial surface

$$\{(x,y,z)\,|\,z=\theta\frac{P}{2\pi}+c_0+c_1t+c_2v+c_3t^2+c_4v^2+c_5tv+\dots\,, \\ t=y\cos\theta-x\sin\theta,\quad v=x\cos\theta+y\sin\theta\} \quad (4.5)$$

where the coefficients $(c_0, c_1, \dots)$ are found by a standard gradient descent minimisation. The total deviation when using such an optimal second order surface is shown in Figure 4.11. The tilt angle was calculated from the coefficient for the first-degree v-term in (4.5) as $\alpha = \arctan c_2$.

Kachelriess, Schaller, and Kalender (2000a) present an algorithm called Advanced Single-Slice Rebinning (ASSR). As in the algorithm of Larson et al. (1998) the reconstruction is made on tilted planes. The only difference is the way to optimise the tilt angle α. Instead of looking at the projection of the plane, Kachelriess et al. (2000b) minimise the distance between the plane and a $180°$-segment of the source trajectory. They show that the optimal angle is such that the trajectory intersects the plane at the centre of the segment and at the two points $60°$ off the centre. The plane in Figure 4.10 has this optimal tilt angle. Using these planes the reconstruction is then made by two-dimensional filtered backprojection as in Larson et al. (1998) or by a standard two-dimensional direct Fourier method in each slice. The potential increase in image quality by using direct Fourier techniques on generalised rays, as will be discussed in Section 4.2.4, is hinted at but never used by the authors. Kachelriess and Kalender (2000) have also showed that is easy to modify ASSR to handle geometrical deviations from the perfect helical path. An example of such a deviation is the sheared helix geometry resulting from tilting the gantry. This is used clinically when trying to avoid radiating sensitive body parts, such as the eyes, close to the area of interest. More details on ASSR can be found in the articles of Kachelriess, Schaller, and Kalender (2000b) and Bruder, Kachelriess, Schaller, and Mertelmeier (2000).

The methods PI-SLANT and PI-2D of Turbell and Danielsson (1999a) also use nutating surfaces. These will be presented in more detail in Sections 4.3.2 and 4.3.3.

The good fit between a helix segment and a plane is the basis for the nutating slice algorithms. It is important to note that the fit is good only for a short-scan segment of the helix. It is impossible to fit a plane to a complete helix turn, not to mention over-scan geometries where the slice is illuminated for several turns. As over-scan methods are used in most medical tomographs to trade image quality for dose and scanning time, it is remarkable that none of the above mentioned articles discuss this obvious limitation of the nutating slice algorithms. We will look at over-scan algorithms more closely in Section 4.3.5.

4.2.3 Three-Dimensional Backprojection

The algorithms in the previous two sections all start with a z-interpolation in order to construct a set of projection data for the slice to be reconstructed. Filtering and backprojection are then performed in a completely two-dimensional fashion. As illustrated in Figure 4.9(b) a backprojected value accumulated to a voxel may not then necessarily belong to a ray that has passed the voxel. This obvious shortcoming should be possible to avoid with three-dimensional backprojection.

Kudo and Saito (1991) and Wang, Lin, Cheng, and Shinozaki (1993) have generalised the FDK-method for a helical trajectory. The derivations of the two proposals are quite different but they both end up with the following algorithm that is almost identical to the original FDK method:

Algorithm 4.1 *Full-scan cone-beam filtered backprojection*

1: pre-weighting with $\cos\gamma\cos\kappa$.
2: one-dimensional ramp-filtering along each detector row.
3: cone-beam backprojection with the L^{-2}-factor of filtered projection data from the closest turn.

Each voxel receives backprojection contributions from a full scan interval of 2π. Yan and Leahy (1992) present a similar algorithm, the difference being that the ramp-filtering in step 2 is performed along tilted lines parallel to the a'-axis defined in conjunction with Figure 4.1. In all three algorithms (Kudo and Saito 1991; Wang et al. 1993; Yan and Leahy 1992), for a given voxel and a given projection angle, only rays of the cone stemming from the closest turn are backprojected. Therefore, all interpolations take place between rays within the same cone-beam projection. As pointed out by Schaller, Flohr, and Steffen (1996), this and other facts give the full-scan methods the following shortcomings:

- Many of the measured and filtered projection data are wasted, since only the ray stemming from the closest turn is used.

- The pitch is severely restricted by the detector height according to

$$P \leq N_q \Delta q \frac{R - R_{\text{FOV}}}{R} \tag{4.6}$$

  Typically, for a given detector height, this pitch is only one fourth of what is required with a PI-detector.

- The average interpolation width in the z-direction is Δq since the possibility of using interlaced rays from nearby turns has been neglected.

Several authors have modified the full-scan algorithms into short-scan algorithms using one of the three previously discussed approaches. For the fourth and final time of this thesis, we here present the approaches one by one.

Parallel Rebinning

By performing parallel rebinning on the helical data in a similar manner as for the circular data in (3.20), we obtain the data set

$$p^P(\theta, t, q) = p(\beta, \gamma, q) \tag{4.7}$$

where the relationships between (θ, t) and (β, γ) are described by (2.31), and the row coordinate q is left unchanged.

Silver (1997) uses these oblique parallel beams in an algorithm called Inconsistent Helical Cone-Beam Reconstruction (IHCB). It is formulated in general terms allowing for a smooth sinogram windowing of the parallel rays to cope with redundancy. Unlike other algorithms, these weights are applied *after* ramp filtering, and can therefore be very sharp or even binary. No reconstructed images are presented in the article, but the two-dimensional experiment presented in Figure 2.8(d) indicates that one should expect severe artifacts due to the weighting.

If binary weights are used, the backprojection interval used for pixel (x, y, z) is chosen as $\theta \in [\theta_z - \pi/2, \theta_z + \pi/2]$, where θ_z is the projection angle where the central source is positioned at height z. Silver (1997) contrasts IHCB with an algorithm where the θ-interval is fixed and therefore requires a much smaller pitch for a given detector height.

Complementary Rebinning

The complementary filtered backprojection algorithm (CFBP) by Schaller, Flohr, and Steffen (1996) uses complementary rebinning together with a true cone-beam backprojection incorporating the wide normalised interpolation discussed earlier in Figure 4.8. The complementary rebinning is identical to the one later used by Taguchi and Aradate (1998) and Hu (1999) and results in the complementary beam geometry already shown in Figure 4.7. The CFBP algorithm consists of the following steps:

Algorithm 4.2 *Complementary filtered backprojection*

1: Complementary rebinning as in (4.4).
2: Preweighting of projection data with $\cos\kappa$.
3: Ramp-filtering along each row of both direct and complementary data.
4: Cone-beam backprojection with an L^{-2}-factor where all rays, both direct and complementary, from nearby rows and turns are weighted together as a function of the z-distance between the voxel and the ray.

The L^{-2}-factor used in the backprojection is dependent on the distance between the voxel and the direct or complementary source.

Smooth Sinogram Windowing

Wang, Liu, Lin, and Cheng (1994), Silver (1998), and Noo, Kudo, and Defrise (1998) have presented short-scan cone-beam algorithms using smooth sinogram weighting. The short-scan FDK algorithm, SS-FDK, presented by Noo et al. (1998) is closely related to the CB-SSRB algorithm by the same authors. For a given slice, the projection is smoothly windowed, pre-weighted, ramp-filtered along each row, and then 3D-backprojected. The smooth sinogram window is applied so that the slice receives projection contributions from a β-interval of length $\pi+2\gamma_{\max}$ centered around the slice. This interval changes for different slices and the smooth sinogram windowing and ramp-filtering therefore has to be performed separately for each slice.

The study by Noo et al. (1998) shows that the short-scan algorithms outperform the full-scan algorithms in image quality, reconstruction time, and the ration between pitch and detector height noise-free data. A comparison between the two short-scan algorithms CB-SSRB and SS-FDK gave no conclusive results on differences in image quality.

Silver (1998) reformulates the fact that rays with small cone-angles are used relatively more frequently in short-scan cone-beam algorithms than in full-scan algorithms into the equivalent observation that data from the central rows of the detector are used more often than data from the outer rows. He suggests that the detector resolution therefore could be reduced for the outer rows.

4.2.4 Multirow Fourier Reconstruction

The Fourier slice theorem is the basis of most two-dimensional reconstruction algorithms. A direct use of the theorem results in the direct Fourier methods, e.g. the gridding method of O'Sullivan (1985), where the projections are Fourier transformed, ramp-filtered, and placed on a polar grid in the two-dimensional Fourier

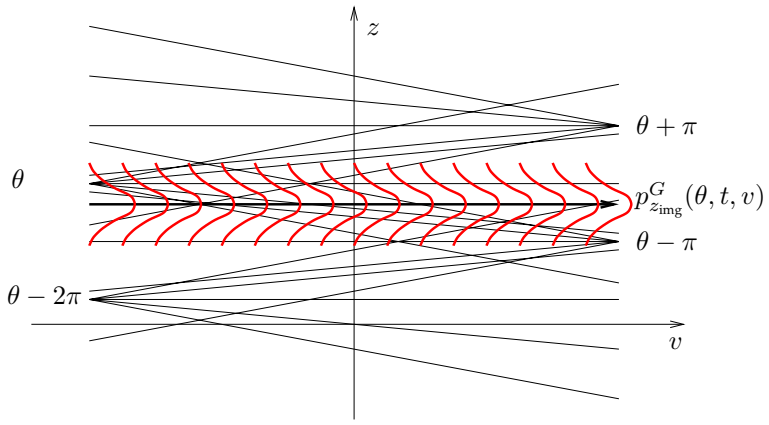
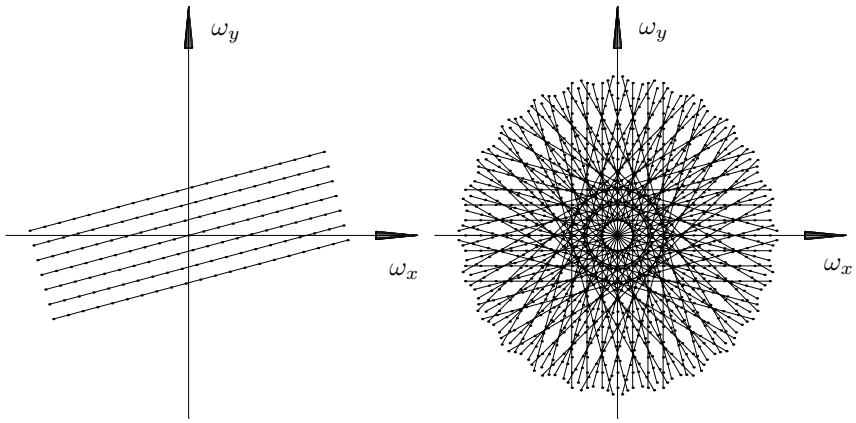


Figure 4.12 *Construction of a generalized projection. At every point along the projection, the value is interpolated from the nearby rays.*

domain. Resampling to a Cartesian grid followed by inverse two-dimensional Fourier transform and a certain post-weighting then yields the reconstructed image. A straightforward use of direct Fourier methods in the helical cone-beam case would be to construct, by interpolation, a two-dimensional parallel projection data set for each slice. These data could then be handled by the standard two-dimensional gridding algorithm. Compared to the filtered backprojection methods discussed in Sections 4.2.1 and 4.2.2, the computational complexity might then be decreased. The main shortcoming would be identical to the one in the filtered backprojection methods, namely that the cone-angle of the rays is neglected.

In an attempt to incorporate the three-dimensionality of the rays in a direct Fourier method, Schaller, Flohr, and Steffen (1997) introduced the concept of *generalised parallel projections*. A generalised parallel projection can be thought of as being generated by a set of horizontal parallel rays confined in the slice to be reconstructed. Unlike conventional projections the value of a generalised projection can change value along the projection direction v. We denote a generalised parallel projection for the slice $z = z_{\text{img}}$ by $p^G_{z_{\text{img}}}(\theta, t, v)$.

To construct generalised projections, the cone-beam data is first rebinned into oblique parallel data as in (4.7). A generalised projection value is constructed by interpolation of the projection values of all rays in the same position when projected onto the slice. These rays belong to oblique parallel projections taken at $\theta + i\pi$, four of which are shown in the vertical slice in Figure 4.12. The generalised projection will have a smoothly changing value along its direction v. The



(a) A generalised projection contributes to points along N_μ parallel lines.

(b) The resulting sampling pattern for all projection angles.

Figure 4.13 *Sampling patterns in the Fourier space of the slice. The drawn lines are truncated and should normally extend further out.*

interpolation is given by

$$p^G_{z_{\text{img}}}(\theta,t,v) = \frac{\sum_\lambda \sum_m \frac{R}{\sqrt{R^2+q_m^2}}(p^P(\theta+2\lambda\pi,t,q_m)w_{1,\lambda,m} + p^P(\theta+2\lambda\pi+\pi,-t,q_m)w_{2,\lambda,m})}{\sum_\lambda \sum_m (w_{1,\lambda,m}+w_{2,\lambda,m})} \tag{4.8}$$

where the weights $w_{1,\lambda,m}$ and $w_{2,\lambda,m}$ depend on the distance between the slice and the ray at the position v.

The reconstruction is performed by applying direct Fourier methods on the generalised projections and the complete algorithm is called multi-row Fourier reconstruction (MFR). Since a generalised projection changes value in both the t- and the v-direction, its Fourier transform will not be a single line in the two-dimensional Fourier domain of the object, but a complete image. However, since the variation along the projection is smooth and relatively slow, it can be approximated by a truncated Fourier series with N_μ terms. The generalised projection will then contribute to N_μ parallel lines in the two-dimensional Fourier space of the object. See Figure 4.13(a). The Fourier series coefficients can be pre-calculated since they only depend on the geometry, not on the actual projection values. The standard $\cos\kappa$-factor, compensating for the different lengths of rays of different

cone-angles, may be incorporated into these weights. This leads to the following algorithm:

Algorithm 4.3 *Multi-row Fourier reconstruction*

1: Rebinning to oblique parallel projections, $p^P(\theta, t, q)$.
2: **for all** slices **do**
3: Construct N_μ vectors of projection data for each projection angle. These correspond to the Fourier series of the generalised projection, and are calculated as a linear combination of projection data from all rows and all turns with pre-calculated weights.
4: Perform an FFT in the t-direction of the vectors
5: Place the result in the two-dimensional Fourier-transform of the image along N_μ parallel lines for each projection angle.
6: Perform an inverse two-dimensional FFT using the gridding technique to obtain the image of the slice.
7: **end for**

Note that the generalised projections are never explicitly calculated. Experiments by Schaller et al. (1997) have shown that $N_\mu = 7$ is sufficient to obtain high quality results for cone-angles up to $\kappa_{\max} = 1.6°$. The computational complexity of the algorithm is $\mathcal{O}(N^3 \log N)$.

It is obvious that MFR is an approximate method. The reconstructed slice will be contaminated by objects outside the slice. The contamination signal is carried by the rays oblique to the slice. The innovative concept of generalised rays combined with direct Fourier techniques makes it difficult to analyse and compare the approximations in MFR with the approximations found in the filtered backprojection algorithms discussed above. The ramp-filtering is inherent in the gridding step which makes the actual filtering direction in MFR unclear.

Gridding is a commonly used reconstruction technique in magnetic resonance imaging. For this imaging modality the measurements are made directly in the Fourier space, also known as the k-space, on a typically non-uniform grid, such as points along a set of interleaving spirals. The gridding is generally performed by the following steps:

Algorithm 4.4 *Gridding*

1: Density compensation of the non-uniform samples
2: Interpolation to a uniform grid
3: 2D IFFT of uniform data to the image domain
4: Post-compensation by the inverse of the inverse transform of the interpolation kernel

The interpolation kernel in step 2 must be shift-invariant in the frequency space. It is this property that enables the post-compensation in step 4. The interpolation is thus preferably performed using a input-data-driven approach.

In two-dimensional parallel beam CT the measurements are given on sampling points placed equidistantly on radial lines. This gives a sampling density inversely proportional to the distance ρ to the origin in frequency space. The density compensation in step 1 above can therefore be performed by ramp-filtering of each projection.

Figure 4.13(b) shows the sampling pattern for MFR when all projection angles are considered. Although the sampling density of this pattern also asymptotically decays as $\frac{1}{\rho}$, it has a more complicated behaviour around the origin. The proposed density compensation in MFR is nevertheless performed by applying the ramp-filter on each line. We expect that a more careful density compensation, using Voronoi polygons as described by Schomberg and Timmer (1995) or singular value decomposition techniques as described by Rosenfeld (1998) and Sedarat and Nishimura (2000), will result in better image quality. Another approach might be the direct Fourier method of Lanzavecchia and Bellon (1996) for electron tomography in a conical tilt geometry that handles sampling pattern similar to the one in Figure 4.13(b).

4.3 The PI Methods

The PI methods are a collection of approximate short-scan methods with three-dimensional backprojection. By only considering projection data within the Tam window and rebinning them to oblique parallel beams, algorithms with fast and simple filtering and backprojection steps are obtained. The first PI method was first proposed by Danielsson, Edholm, Eriksson, Magnusson-Seger, and Turbell (1998a). It is discussed in the following section where we refer to it as PI-ORIGINAL. A number of extensions and alternatives have emerged since and are presented in the subsequent sections.

4.3.1 The PI-ORIGINAL Method

A major concern in many of the algorithms presented so far is how to handle redundancy. Silver (1997) has investigated different combinations of parallel rebinning and smooth sinogram windowing to ensure that each voxel obtain backprojection contributions from an interval of the exact length π. However, to achieve this, Silver applies the weighting after filtering, something which, as mentioned in Section 2.2.3, is undesirable. A solution to this problem is the PI-ORIGINAL method by Danielsson, Edholm, Eriksson, Magnusson-Seger, and Turbell (1998a) which uses the Tam window to ensure non-redundant and complete data capture.

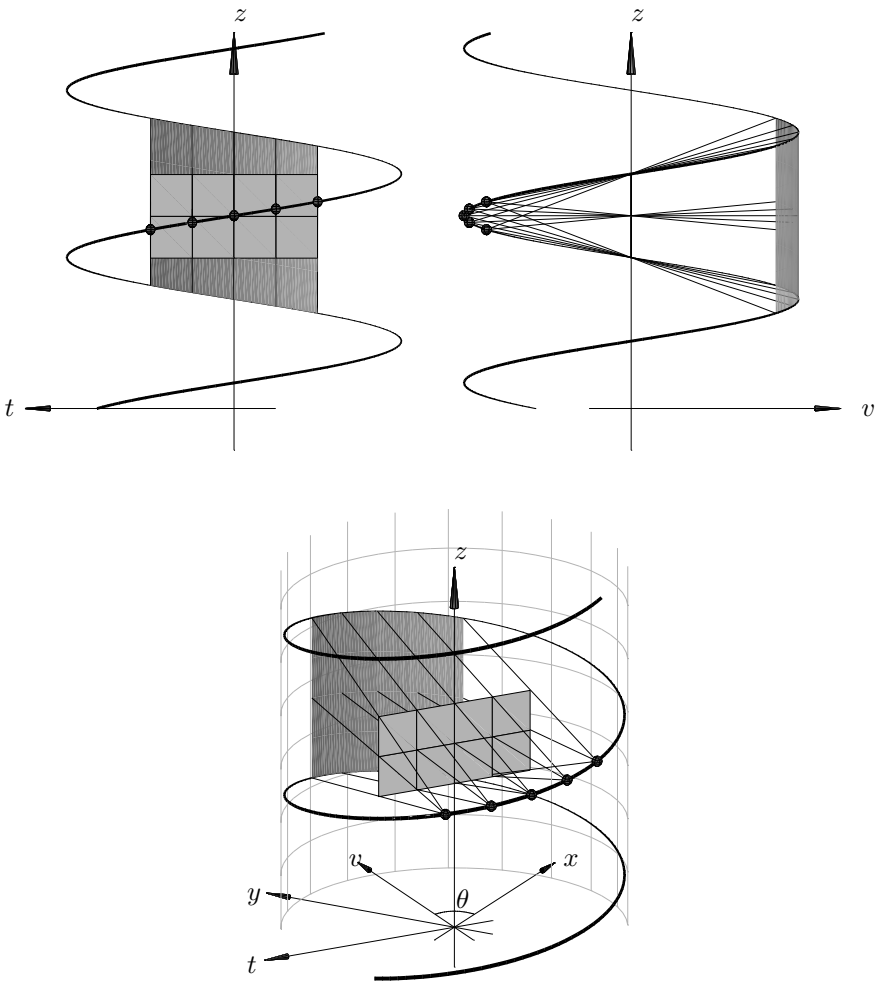


Figure 4.14 *The beam geometry after parallel rebinning. The rays intersect the* (t, z)*-plane in a perfectly rectangular area. Top: two side views. Bottom: a perspective view with a cylindrical grid for visual guidance.*

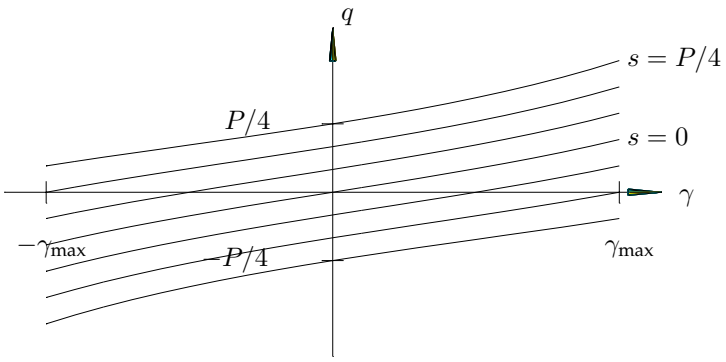


Figure 4.15 *The rows of the PI-detector projected onto the cylindrical detector centered around the source ($\gamma_{max} = \pi/6$).*

Algorithm and properties

Imagine a detector, the PI-detector, fitted to the Tam window and located on the helix cylinder as shown in Figure 4.1. This cylinder is different from the cylinder centred on the source on which the real detector is likely to be placed. We will refer to the two cylinders as the helix and the outer cylinder respectively. The detector rows of the PI-detector on the helix cylinder follow the borders (two neighbouring turns of the helix) with equidistant spacing. These detector rows become slanted but straight if the PI-detector is rolled out onto a plane. Projected onto the outer cylinder, the slanted rows end up along the curves

$$q(s,\gamma) = \frac{s + \gamma\frac{P}{\pi}}{\cos\gamma} \quad s \in [-\frac{P}{4}, \frac{P}{4}] \tag{4.9}$$

of constant s as shown in Figure 4.15. The division with $\cos\gamma$ is due to the magnification when projecting the inner cylinder onto the outer cylinder. For comparison we note that a magnification of $\cos^{-2}\gamma$ appears when the projection is made onto the planar detector as in (4.2) and (4.3). The addition with $\gamma\frac{P}{\pi}$ in (4.9) is due to the tilted rows of the PI-detector. We will use the integer variable c to index the rows as s_c when necessary.

Developing and manufacturing two-dimensional detectors is an enormous engineering feat. A physical detector shaped as the PI-detector is not likely to be designed. Instead, a tomograph using the PI-method would probably use a standard outer cylinder detector collimated to the Tam-window. PI-detector data along rows of constant s would be obtained after interpolation using (4.9).

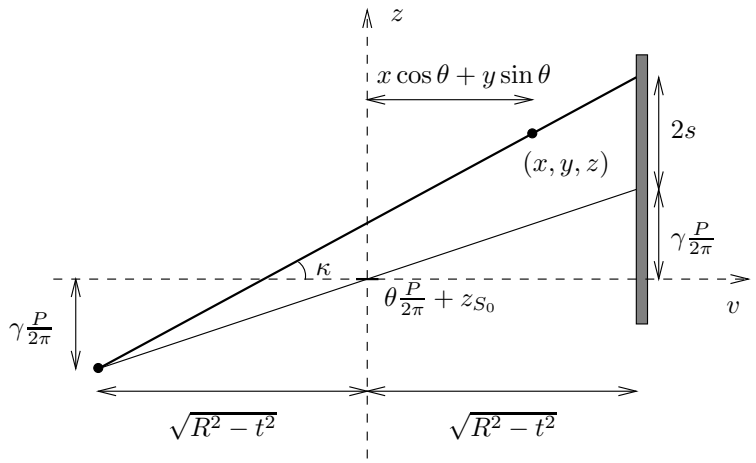


Figure 4.16 *Projection of the object point (x, y, z) onto the detector in the rebinned geometry. The figure is drawn in the plane parallel to the (v, z)-plane which contains the object point.*

With a slight abuse of notation we denote the projection data from the PI-detector in an identical manner to the data from the outer cylinder detector as $p(\beta, \gamma, s)$. PI-ORIGINAL starts with a parallel rebinning row by row performed as

$$p^P(\theta, t, s) = p(\beta, \gamma, s) \tag{4.10}$$

with the standard parallel rebinning relations in (2.31). This leads to the beam geometry in Figure 4.14.

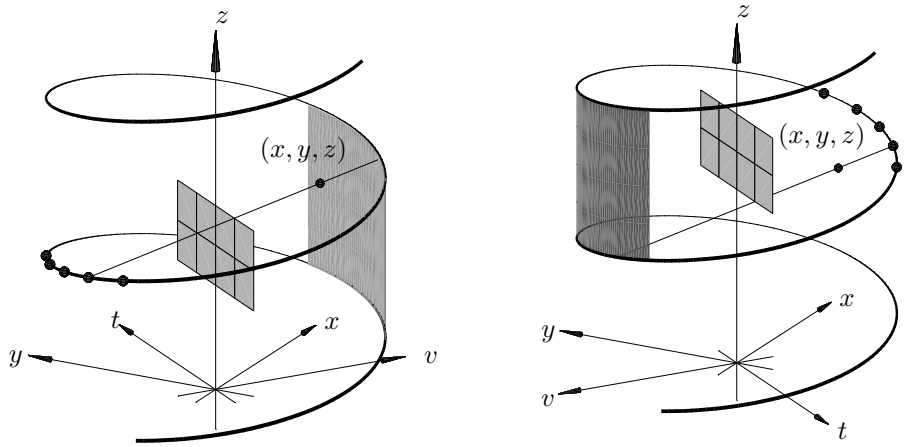
The rebinned data is pre-weighted and ramp-filtered row by row

$$\tilde{p}^P(\theta, t, s) = (p^P(\theta, t, s)\cos\kappa) * g^P(t) \tag{4.11}$$

where the expression of the pre-weight

$$\cos\kappa = \frac{\sqrt{R^2 - t^2}}{\sqrt{R^2 - t^2 + (\gamma\frac{P}{2\pi} + s)^2}}, \quad \gamma = \arcsin\frac{t}{R} \tag{4.12}$$

is derived from the rebinned projection geometry in Figure 4.16. The pre-weighting may be performed before or after the parallel rebinning. Just as in the FDK algorithm, this pre-weighting compensates for the longer path of oblique rays which makes the method collapse into two-dimensional reconstruction if the object is z-homogeneous.



(a) $\theta = \theta_{\text{in}}$ (b) $\theta = \theta_{\text{out}} = \theta_{\text{in}} + \pi$

Figure 4.17 *The PI-detector restricts the illumination interval of a voxel* (x, y, z) *to exactly* $180°$*. Note that the entrance and exit rays are identical except from direction.*

The rebinned geometry has several interesting properties. Imagine a planar virtual detector on the (t, z)-plane. Although the oblique parallel rays start at different z-positions and have different cone-angles, they have a symmetry that results in a perfectly rectangular window of height $P/2$, drawn in Figure 4.14. The sampling points in this window lie on a Cartesian grid so that one horizontal row on the virtual planar detector corresponds to a slanted row on the cylindrical detector. The virtual planar detector is hence the (t, s)-space.

An even more appealing property of the rebinned geometry is that every point (x, y, z) will be illuminated for exactly half a turn. Figure 4.17(a) shows the projection angle $\theta = \theta_{\text{in}}(x, y, z)$ where the projection of the object point (x, y, z) first appears on the detector at the top border ($s = P/4$). The detector and source will then move upwards as the projection of the object point moves downwards on the detector along a sinus-like curve. Finally, when $\theta = \theta_{\text{out}}(x, y, z)$, the projection of the object point disappears from the detector, shown in Figure 4.17(b). The entrance and exit rays are identical apart from direction. Hence, they are parallel and belong to projections exactly $180°$ apart, which we write as

$$\theta_{\text{out}}(x, y, z) = \theta_{\text{in}}(x, y, z) + \pi \tag{4.13}$$

This is the key property of the method.

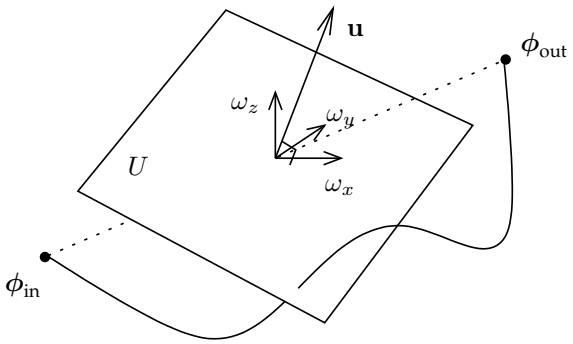


Figure 4.18 *The Fourier component at* $\mathbf{u}$ *is measured when the projection direction vector* ϕ *lies in the plane* U*.*

In the original, non-rebinned, geometry, the length of the illumination interval changes for different voxels. However, seen from the point itself, the angle between the first and final illuminating source positions is always π. This is true both when looking at the situation two-dimensionally as projected along the z-axis as in (4.13) and for the truly three-dimensional geometry. Consider an arbitrary object point. We define the projection direction $\phi \in \mathbb{R}^3$ as a unit vector on the line between the object point and the source. The first projection direction ϕ_{in} is then the negative of the final direction $\phi_{\text{out}} = -\phi_{\text{in}}$. The angle between them is therefore π. Danielsson, Edholm, Eriksson, and Magnusson-Seger (1997) used a theorem of Orlov (1975) to show that this property guarantees that the complete three-dimensional Fourier space of any small neighbourhood in the object is without missing data. We now present an alternative argument for this conclusion.

Consider a small neighbourhood of an arbitrary object point. Each original cone-beam projection can for this small neighbourhood be seen as a truly parallel projection. The Fourier-Slice theorem says that this projection measures all Fourier components on the plane through the origin that is orthogonal to the projection direction ϕ. Consider an arbitrary position $\mathbf{u}$ in Fourier space and the corresponding normal plane U through the origin. See Figure 4.18. Clearly, if the projection direction vector ϕ is positioned in this plane, the Fourier component at $\mathbf{u}$ will be measured. The plane U divides the Fourier space in half. Consider a projection direction trajectory with start and end directions ϕ_{in} and $\phi_{\text{out}} = -\phi_{\text{in}}$ diametrically apart. Inevitably, the trajectory will intersect U at least once. The projection will then measure the Fourier component at $\mathbf{u}$. Therefore, all Fourier components are measured. This fact is not used explicitly in PI-ORIGINAL, but served as a starting point for developing an algorithm that only uses projection data within the Tam window.

Note that the above argument only guarantees completeness of the local Fourier data, but not non-redundancy. With a helical source movement, there exist planes U that have three intersections with the projection direction trajectory. An example of such a plane, although illustrated in the signal space for quite a different context, is shown in Figure 4.10. Some of the local Fourier components are therefore measured three times. They are very rare and closely connected to the concept of redundant segments discussed by Kudo et al. (1998). In the Radon based algorithms it is possible to handle this small redundancy in the input data, whereas PI-ORIGINAL makes no such attempts. The redundancy is not the main reason for PI-ORIGINAL being approximate. The reason for inexactness is rather the simplified one-dimensional filtering.

Both of the above illumination arguments apply to the acquisition of projection data as well as to the backprojection. The PI-detector restricts the illumination of voxels perfectly so that the backprojection can be performed without any redundancy as

$$f_{\text{PI-ORIGINAL}}(x,y,z) = \frac{2}{N_\theta} \sum_{n=n_{\text{in}}(x,y,z)}^{n_{\text{in}}(x,y,z)+\frac{N_\theta}{2}-1} \tilde{p}^P(\theta_n, t(x,y,\theta_n), s(x,y,z,\theta_n)) \qquad (4.14)$$

where the detector coordinates (t,s) are given by

$$t(x,y,\theta) = y\cos\theta - x\sin\theta \qquad (4.15)$$

and

$$s(x,y,z,\theta) = \frac{\sqrt{R^2-t^2}(z - z_{S_0} - (\theta-\gamma)\frac{P}{2\pi})}{\sqrt{R^2-t^2}+v(x,y,\theta)} - \gamma\frac{P}{2\pi}, \quad \gamma = \arcsin\frac{t}{R} \qquad (4.16)$$

indicate the position on the detector where the voxel is projected. The integer n_{in} is the index of the projection angle θ_{in} and the coordinate $v(x,y,\theta)$ is defined in (3.19). Figure 4.16 shows the projection geometry used to derive (4.16).

To summarise, PI-ORIGINAL contains the following computation steps:

Algorithm 4.5 *PI-ORIGINAL*

1: Parallel rebinning of data restricted by the Tam window
2: Pre-weighting with $\cos\kappa$
3: One-dimensional ramp-filtering along each row on the virtual planar detector
4: Three-dimensional backprojection without any L^{-2}-factor

We note that the concept of quarter offset, just as for the oblique parallel beam in the circular trajectory of Section 3.3.1, not can be utilised in the PI-method.

Implementation of the backprojection

The backprojection can be implemented in many ways. A direct implementation of (4.14) is

Algorithm 4.6 *Voxel driven backprojection*

1: **for all** (i_1, i_2, i_3) inside the FOV **do**
2: find n_{in}
3: **for** $n = n_{\text{in}}$ to $n_{\text{in}} + N_\theta/2 - 1$ **do**
4: calculate t and s
5: interpolate between the four detector elements around (t, s)
6: accumulate the interpolated value in $f_{\text{PI-ORIGINAL}}[i_1, i_2, i_3]$
7: **end for**
8: **end for**

At the outer halves of the two outer detector rows, interpolation on the detector is not possible. If extrapolation is used to handle data in this region artifacts will occur. A remedy is to extend the detector with an extra row so that the centre of the outermost measurement lies exactly on the source helix which allows for interpolation everywhere inside the PI-window. This is shown to the left in Figure 4.19.

A difficulty with the voxel driven approach is to calculate n_{in} on line 2. We have found no closed form solution, so an iterative search or look-up table has to be used. To eliminate this problem we consider all voxels inside the FOV at the specific height $z = z_{\text{slice}}$ and the rebinned beam centred at this height. This beam has the projection angle

$$\theta_{\text{slice}} = (z_{\text{slice}} - z_{S_0})\frac{2\pi}{P} \tag{4.17}$$

We now ask ourselves if all of these voxels within the field of view in the slice are illuminated by this beam. A geometrical argument gives that this is the case provided that

$$\frac{\frac{P}{4}\sqrt{R^2 - t^2}}{\frac{P}{4} + \frac{P}{2\pi}\arcsin\frac{t}{R}} > \sqrt{t_{\text{max}}^2 - t^2}, \quad |t| < t_{\text{max}} \tag{4.18}$$

which, independently of P and R, is satisfied for fan-angles up to somewhat more than $\pm 55°$. For such systems we then know one illuminating projection angle, namely $\theta = \theta_{\text{slice}}$, for each voxel. It is straightforward to formulate a backprojection that starts with this projection angle and iterates in the two opposite directions away from it until the voxel is out of illumination as:

Algorithm 4.7 *Voxel driven backprojection in two steps*

for all i_3 **do**
  calculate n_{slice} according to (4.17)
  for all (i_1, i_2) inside the FOV **do**
    set $n = \lfloor n_{\text{slice}} \rfloor$
    calculate t and s
    while $s < P/4$ **do**
      interpolate between the four detector elements around (t, s)
      accumulate interpolated value in $f_{\text{PI-ORIGINAL}}[i_1, i_2, i_3]$
      decrement n
      calculate t and s
    end while
    set $n = \lfloor n_{\text{slice}} \rfloor + 1$
    calculate t and s
    while $s > -P/4$ **do**
      interpolate between the four detector elements around (t, s)
      accumulate interpolated value in $f_{\text{PI-ORIGINAL}}[i_1, i_2, i_3]$
      increment n
      calculate t and s
    end while
  end for
end for

Not utilising the above observation, we may change the order of the loops to obtain:

Algorithm 4.8 *Projection driven backprojection*

for all n **do**
  find upper and lower z-bound for voxels illuminated by the current beam
  for all (i_1, i_2) inside the FOV **do**
    incrementally update t
    for all i_3 limited by the upper and lower z-bound **do**
      incrementally update s
      if $-P/4 < s < P/4$ **then**
        interpolate between the four detector elements around (t, s)
        accumulate interpolated value in $f[i_1, i_2, i_3]$
      end if
    end for
  end for
end for

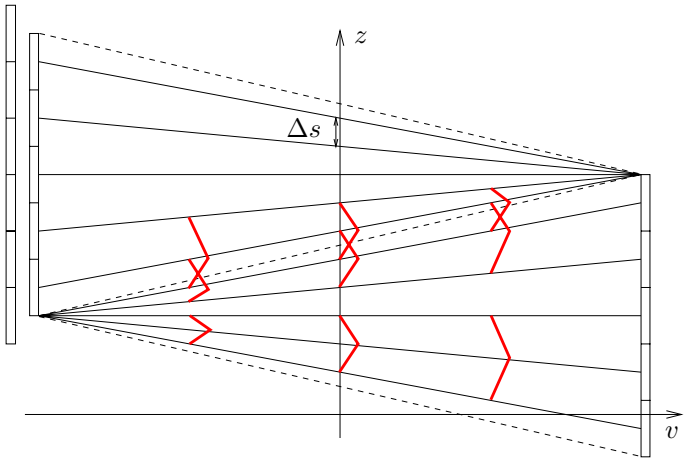


Figure 4.19 *Interpolation in the z-direction. A space-invariant linear interpolation kernel of width $2\Delta s$ is used for the outer halves of the outer detector rows and a space-variant linear kernel is used for the other detector positions. The space-variant kernel is space-invariant on the detector and vice versa. The dashed lines indicate the exact borders of the beams. To the left is shown a detector with one extra row needed to avoid extrapolation if the space-invariant kernel is not used.*

The incremental update of s on line 6 is possible since $\frac{ds}{dz}$ is constant for a given (x, y). The iteration over voxels in the z-direction on line 5 will visit many voxels outside the beam. This could be avoided by a bi-directional traversal as in Algorithm 4.7. A safe starting voxel is then given by (4.17).

A problem with the two above approaches are the while-statements on line 6 and 14 in Algorithm 4.7 and the if-statement on line 7 in Algorithm 4.8 that check whether the voxel is inside the cone or not. The calculation of s has to be performed with high precision to ensure that all voxels get illumination from exactly $N_\theta/2$ projection angles. This can be critical in an implementation using fixed-point arithmetic. Since this number of backprojection contributions should be constant for all voxels, no normalisation with the actual number is made. A single extra contribution could thus change the reconstructed value with several parts per thousand for typical values of N_θ, which results in clearly visible artifacts. Note that this problem does not appear in the voxel driven algorithm 4.6 since the for-statement on line 3 loops over a fixed number of projection angles.

The interpolation in Algorithms 4.6, 4.7, and 4.8 is performed with a fixed sized kernel on the detector, which corresponds to a space-variant kernel in the object. They can be reformulated to use a space-invariant kernel by adjusting the width of the kernel on the detector according to the magnification due to the divergence.

The extra row necessary to allow for interpolation at the detector borders can be discarded if the interpolation is performed between the top row of one projection angle the bottom row (mirrored in the t-direction) of a projection with an angular difference of π. As seen in Figure 4.19, the distance between these two rays is constant throughout the object which calls for a space-invariant interpolation in this region. This corresponds to a space-variant kernel on the detector. The smooth edges at the top and bottom of the cone do not violate π-illumination but eliminates the problem of ensuring exactly $N_\theta/2$ contributions.

All of the above backprojection algorithms have complexity $\mathcal{O}(N_x N_y N_z N_\theta)$. The actual performance depends heavily on the hardware architecture of the system the implementation is executed on. The voxel and projection memories may for example have different bandwidths or caching schemes if special hardware is used. Such factors then have to be taken into account when evaluating the algorithm performance.

4.3.2 The PI-SLANT Method

For large pitches with detectors of many rows PI-ORIGINAL is known to produce artifacts in the reconstructed result. The aim of this section is to improve the performance of PI-ORIGINAL while still keeping the algorithmic simplicity of the one-dimensional ramp filtering together with the redundancy handling of the PI-window. The new reconstruction method presented here, called PI-SLANT, is still approximate but better capable of handling large pitches.

In order to track the inexactness of PI-ORIGINAL, we study the set of image points entering the rectangular window of the planar detector at the same time. These points lie on a surface, which we will call a PI-surface, defined as

$$\{(x,y,z)\,|\,x^2+y^2<R^2,\quad z=\theta\frac{P}{2\pi}+z_{S_0}+\frac{P}{4}+v\frac{\frac{\gamma P}{2\pi}+\frac{P}{4}}{\sqrt{R^2-t^2}},$$
$$t=y\cos\theta-x\sin\theta,\quad v=x\cos\theta+y\sin\theta,\quad \gamma=\arcsin\frac{t}{R}\} \quad (4.19)$$

This definition can be derived from (4.16) or Figure 4.16 by setting $s=P/4$. The top rays in a rebinned parallel beam all lie in the same PI-surface. These rays intersect the helix twice and are therefore PI-lines as defined on page 83. From the fact that each image point lies on one and only one PI-line, it follows directly that each image point belongs to one and only one PI-surface. The complete set of PI-surfaces has a nutation around the rotation axis and fills up the complete volume to be reconstructed. See Figure 4.20.

We assume that the projection system is rotating upwards in which case the points of a PI-surface enter the rectangular detector window on a perfect line on the upper border simultaneously. As the rotation of the projection system contin-

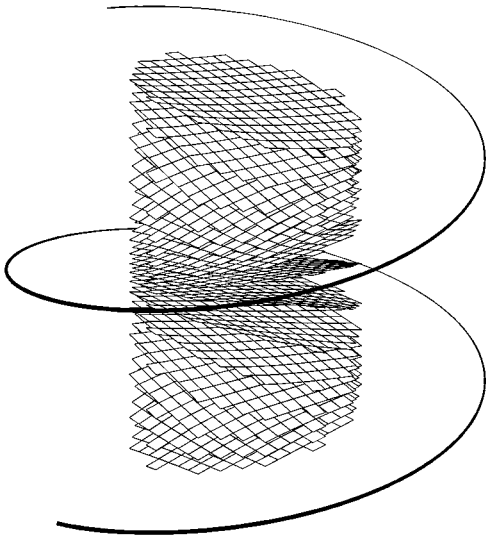
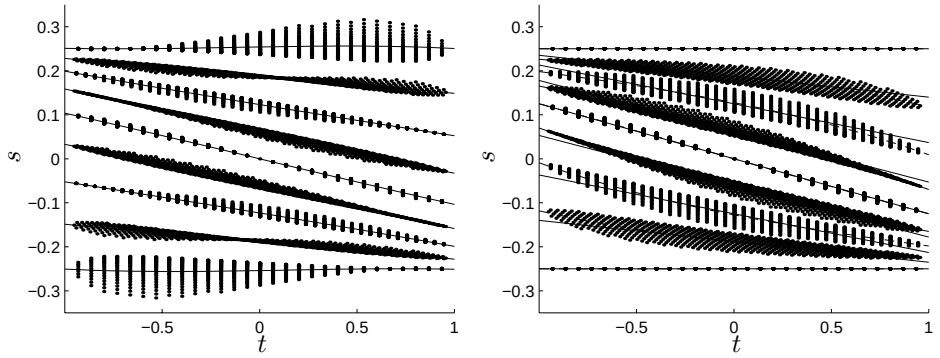


Figure 4.20 *The nutating PI-surfaces fill up the reconstuction space.*

ues, the projection of the PI-surface moves downwards on the detector but it immediately starts to deviate from the line shape. Instead, the PI-surface is projected within an elongated area with a non-horizontal mid-line. See Figure 4.21(b). However, after a rotation of exactly $180°$, all points on the PI-surface are again lined up horizontally, now along the lower border of the window, and exit the rectangular window simultaneously.

We note that if the PI-surface would be projected along lines, or any other one-dimensional curve, during the complete $180°$-illumination interval, the projection values along these curves would be constructed from line-integrals fully inside the PI-surface. It would then be straightforward to extract the projection values along the curves and put them in a two-dimensional sinogram. A mathematically exact reconstruction of the density values of the PI-surface would then be obtained by performing two-dimensional filtered backprojection on this sinogram data. The exactness is obvious since all points on the PI-surface are involved in all filtering events without any contamination from neighbouring PI-surfaces.

The procedure in the previous paragraph is not feasible since the projection of a PI-surface extends in both the t- and s-directions. A detector value therefore consists of line integral contributions from several PI-surfaces. Ideally, these contributions should be separated before the reconstruction of a PI-surface can be performed as a two-dimensional filtered backprojection. An attempt to such a separation is presented in Section 4.3.4.



(a) Nutating planes

(b) Nutating PI-surfaces

Figure 4.21 *Projection of points on a nutating surface onto the virtual planar detector at different projection angles over an interval of length π with $R = 2$, $R_{FOV} = 1$, $P = 1$.*

As an approximation of a PI-surface we define the corresponding *PI-plane* as the plane spanned by the uppermost central ray ($t = 0$) of an oblique parallel projection and the uppermost boundary ($s = P/4$) of the virtual planar detector. These two lines both lie in the PI-surface.

Both the PI-surfaces and PI-planes are nutating around the z-axis, just like the planes described by Larson, Ruth, and Crawford (1998). We have indicated their performance in Figure 4.11 using the same deviation measure as in Section 4.2.2. The PI-plane is identical to a nutating plane with the tilt angle $\alpha = \arctan \frac{P}{4R}$. We have positioned the deviation value of both the PI-plane and the PI-surface at this tilt angle in the figure. The PI-surface has lower deviation than the PI-plane but higher deviation than the optimally tilted plane.

The intersection of the PI-plane and the virtual detector is a slanted line on the detector given by

$$s(l,t) = l + t\frac{ds}{dt}(l) = l - t\frac{P}{4R\sqrt{1 + \left(1 + \left(\frac{P}{4R}\right)^2\right)\tan^2\left(l\frac{2\pi}{P}\right)}} \tag{4.20}$$

for constant values of $l \in [-P/4, P/4]$. For typical fan-angles ($\gamma_{\max} \approx \pi/6$), the intersection lines given by (4.20) are virtually identical to the lines obtained by performing least square line-fitting on the projection of the points on the PI-surface inside the FOV.

In the modified PI-method, called PI-SLANT, we replace the horizontal filtering with filtering along these slanted lines resulting in the following algorithm:

Algorithm 4.9 *PI-SLANT*

1: Parallel rebinning from (β, γ, s) to (θ, t, s) of data within the Tam window
2: Pre-weighting with $\cos\kappa$ as in (4.12)
3: Column-wise resampling from (θ, t, s) to (θ, t, l) using (4.20)
4: One-dimensional ramp-filtering along the slanted lines
5: Three-dimensional backprojection without any L^{-2}-factor

The backprojection requires the calculation of the value of l given the values of t and s. We have found no way to derive this relation from (4.20). Instead we use the following iterative approach in our implementation. Given (s, t) calculate l iteratively as

$$l^{(i+1)} = s - t\frac{ds}{dt}(l^{(i)}) \tag{4.21}$$

Start with $l^{(0)} = s$. A small number of iterations (≈ 5) then give a sufficiently accurate estimate of l. For faster performance a look-up table could be used. A third alternative is to resample the filtered projection values back to the original (t, s)-system with extra fine sampling in the s-direction before backprojection. The original backprojection formula (4.16) can then be used.

The slanted filtering lines are nicely confined to the same rectangular window on the planar detector as in the original PI-method, which means that the $180°$ illumination property is unaffected and that no filtering over truncated data is necessary. This is however not true for large fan-angles ($\gamma_{\max} > 39.5°$) where the lines in (4.20) sprawl outside the PI-window. For such systems, the PI-plane used to derive (4.20) is no good approximation of the PI-surface. Curves optimally fitted to the projection of the PI-surface can then be used instead.

4.3.3 The PI-2D Method

Figure 4.21(a) shows the points on the nutating planes in the algorithm of Larson et al. (1998). When we compare to the respective plot of PI-SLANT in Figure 4.21(b) we make the observation, not mentioned elsewhere, that the all rays needed in the Larsson algorithm almost perfectly fit the Tam window. The filtering directions in this algorithm are also very similar to the ones in PI-SLANT. The major difference between the Larsson algorithm and PI-SLANT is that PI-SLANT performs the backprojection three-dimensionally whereas the Larsson algorithm backprojects in two-dimensions onto each nutating slice.

Based on the comparison above, it is straight forward to propose a variation of the PI-method where step 5 in PI-SLANT is replaced by a two-dimensional backprojection to the nutating PI-surfaces, followed by a z-interpolation of the reconstructed PI-surfaces to a Cartesian grid. This algorithm, which we call PI-2D, then becomes almost identical to the algorithm of Larson et al. (1998).

Algorithm 4.10 *PI-2D*

1: Parallel rebinning from (β, γ, s) to (θ, t, s) of data within the Tam window
2: Pre-weighting with $\cos\kappa$ as in (4.12)
3: Column-wise resampling from (θ, t, s) to (θ, t, l) using (4.20)
4: One-dimensional ramp-filtering along the slanted lines
5: Two-dimensional backprojection onto each PI-surface
6: Resampling in the z-direction to a Cartesian voxel grid

4.3.4 The PI-FAST Method

The PI-SLANT and PI-2D algorithms utilize the observation that the image points on the PI-surface are concentrated along slanted lines in-between entrance and exit. The fact that some of the image points on the PI-surface are projected above such a slanted line and some below is not handled in the filtering step of these algorithms. Consider a neighbourhood of points on a PI-surface. They are projected onto a neighbourhood on the detector. They furthermore have similar velocities in the t-direction on the detector. The projection values of this detector neighbourhood are however contaminated by image points belonging to other PI-surfaces. The projections of these points have velocities different from the considered point projections. This observation lead to the development of an algorithm that tries to use the velocity in the t-direction as a further way of discriminating between projection values of different PI-surfaces. We initially tried to use the frequency-distance relation of Edholm, Lewitt, and Lindholm (1986) for such discrimination. This resulted in algorithms with many computation steps, and poor discrimination. We will now instead present our preferred algorithm, PI-FAST, which uses some of the fast backprojection concepts as a way to discriminate between contributions from different PI-surfaces. Defrise and Noo (1997) describe a different way of utilising the frequency-distance for helical cone-beam reconstruction.

Switching the order of backprojection and filtering

Before tackling the cone-beam problem we return to fast backprojection in two dimensions described in Section 2.3. We simplify this algorithm by not dividing

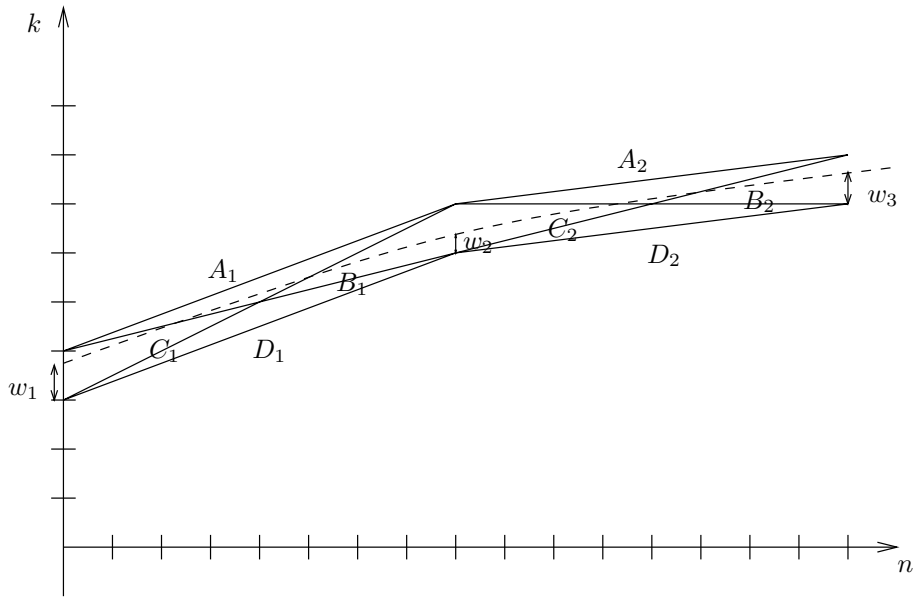


Figure 4.22 *A pixel value is the sum of interpolations between groups of four links.*

the computation into $\lg N_\theta$-steps but only into 2 steps. The first step calculates the values of links of a certain length N_l, and the second step sums these values along sinusoids in the sinogram into pixel values. A further simplification is to use links short enough to be validly approximated by line segments. The link value can then be calculated as

$$\tilde{I}[n_1, k_1; n_2, k_2] = \sum_{n=n_1}^{n_2-1} \tilde{p}^P[n, k_1 + \frac{n - n_1}{n_2 - n_1}(k_2 - k_1)] \tag{4.22}$$

Although t_{k_1} and t_{k_2} are sample points in the sinogram, the intermediate values along the link require an implicit one-dimensional each in (4.22). The link is of length $N_l = n_2 - n_1$, which gives $\frac{N_\theta}{N_l}$ link starting positions in the θ-direction.

The second step sums the link values into pixel values. Using the notation in Figure 4.22, we write this as

$$f(x, y) = \sum_{i=1}^{\frac{N_\theta}{N_l}} \left(w_i \left(w_{i+1} A_i + w'_{i+1} B_i \right) + w'_i \left(w_{i+1} C_i + w'_{i+1} D_i \right) \right) \tag{4.23}$$

where A_i, B_i, C_i, and D_i are the link values and w_i and $w'_i = 1 - w_i$ are interpolation coefficients.

If we insert the ramp filter convolution (2.13) into (4.22), the link value can be written as

$$\begin{aligned}
\tilde{I}[n_1,k_1;n_2,k_2] &= \sum_{n=n_1}^{n_2-1}\left(p^P[n,k_1+\frac{n-n_1}{n_2-n_1}(k_2-k_1)]*g^P[k]\right)\\
&= \sum_{n=n_1}^{n_2-1}\sum_{k'}p^P[n,k'-k_1-\frac{n-n_1}{n_2-n_1}(k_2-k_1)]g^P[k']\\
&= \sum_{k'}\sum_{n=n_1}^{n_2-1}p^P[n,k'-k_1-\frac{n-n_1}{n_2-n_1}(k_2-k_1)]g^P[k']\\
&= \left(\sum_{n=n_1}^{n_2-1}p^P[n,k_1+\frac{n-n_1}{n_2-n_1}(k_2-k_1)]\right)*g^P[k]\\
&= I[n_1,k_1;n_2,k_2]*g^P[k]
\end{aligned} \tag{4.24}$$

where $I[n_1,k_1;n_2,k_2]$ naturally denotes the link value calculated from unfiltered projection data. The change of order between summation and convolution is possible since the links are linear and equidistantly spaced along the t-axis. Equation (4.24) tells us that we may wait with the filtering of projection data until the link values have been calculated and then perform the filtering on the link values instead. The right hand side convolution should be seen as a one-dimensional convolution where all links with the same starting projection angle θ_{n_1} and the same slope $\frac{dt}{d\theta}$ will participate in one filtering event. Although the end result evidently is independent of the order of filtering and link construction in the two-dimensional case, it is exactly this possibility to switch order that is the key for handling points projected off the filtering line in PI-SLANT.

Algorithm

We now return to the cone-beam geometry. For each PI-surface let us construct a complete set of link values $I[n_1,k_1;n_2,k_2]$ from the pre-weighted data. These will later be ramp filtered and finally summed together into pixel values. The links are positioned in the three-dimensional (θ,t,s) projection space. See Figure 4.23. The (θ,t)-coordinates of the links are identical to the two-dimensional case and we may regard the reconstruction as two-dimensional when seen from along the rotation axis. The s-coordinates require some further analysis.

Given a PI-surface and the values of θ_{n_1}, θ_{n_2}, t_{k_1}, and t_{k_2} we use (2.38) to compute the (x,y)-coordinate of the point on the PI-surface corresponding to the link. Knowing x and y, the z-coordinate of the point is uniquely given by the equation of the PI-surface (4.19). The link endpoint coordinates s_{c_1} and s_{c_2} can then be calculated by projecting the point (x,y,z) onto the virtual planar detector

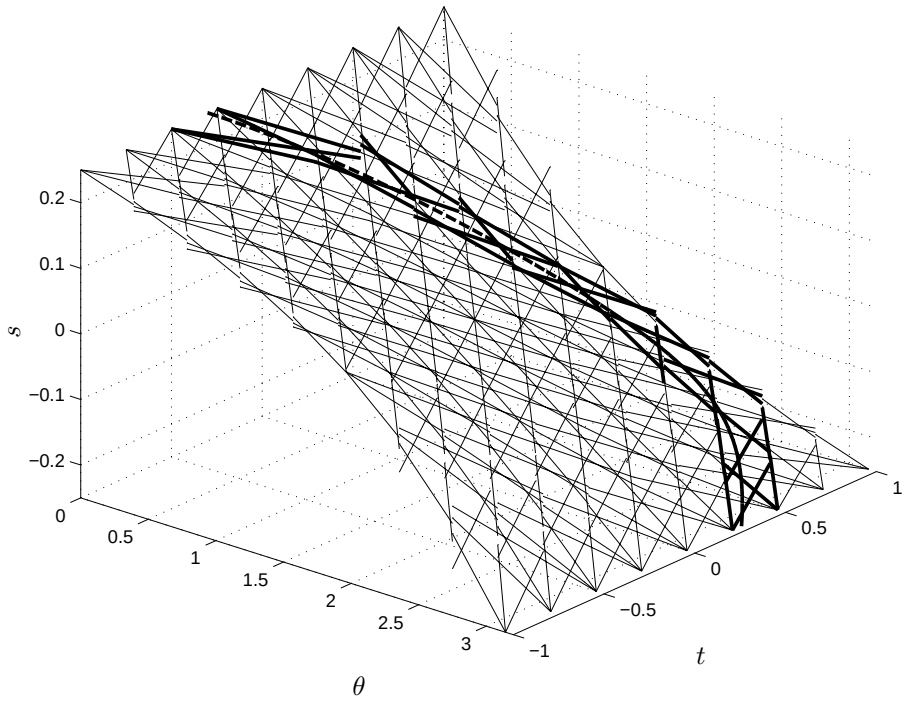


Figure 4.23 *The links needed for a PI-surface of a helix of pitch $P = 1$. To simplify the illustration the number of links in both t- and θ-rirection is small compared to a typical case. This makes the links sprawl out in the s-direction more than in a typical case.*

from projection angles θ_{n_1} and θ_{n_1} respectively. These endpoints will typically not end up on integer values of c_1 and c_2.

The complete set of links for a PI-surface will approximately follow the projection tracks of the points in the PI-surface from their movements in the projection space. At the entrance projection angle $\theta_{n_{\text{in}}}$ of the PI-surface, the links will all start at $s = P/4$ and at the final projection angle $\theta_{n_{\text{out}}}$ the links will all end at $s = -P/4$. In-between these two angles, the links will spread out in the s-direction.

When the end-points of the links have been determined it is possible to calculate each link value along the line segment between the link end point as

$$I[n_1, k_1; n_2, k_2] = \sum_{n=n_1}^{n_2-1} p^P[n, k_1 + \frac{n-n_1}{n_2-n_1}(k_2-k_1), c_1 + \frac{n-n_1}{n_2-n_1}(c_2-c_1)] \quad (4.25)$$

The implicit one-dimensional interpolation in the t-direction for the two-dimensional case in (4.22) has here become an implicit two-dimensional interpolation on each (t, s)-plane.

If we insert the parallel rebinning step (4.10) into (4.25), we may construct the link values directly from the cone-beam data $p(\beta, \gamma, s)$ without the intermediate re-sampling step in (4.10). There is furthermore no need to initially re-sample the original cone-beam data to rows of constant s since the mapping (4.26) between the actual physical detector sampling points and the (β, γ, s)-system also can be inserted into (4.25). The sampling points of the terms of the summation in (4.25) will then not coincide with the projection data sampling points even in the projection angle direction. An implicit three-dimensional interpolation is thus necessary for each term in (4.25). To avoid aliasing, it may be necessary to have a smaller step size in the summation along the θ-direction than the sampling distance step used in (4.25).

Equation (4.25) says that each link can be seen as a linear combination of the projection data. Since the coefficients in this linear combination only depend on the acquisition geometry and not on the actual projection measurements, they may be computed in advance. A substantially smaller step size to avoid aliasing would then not lead to more computations in the actual reconstruction, but would only result in more correct coefficients in the linear combination.

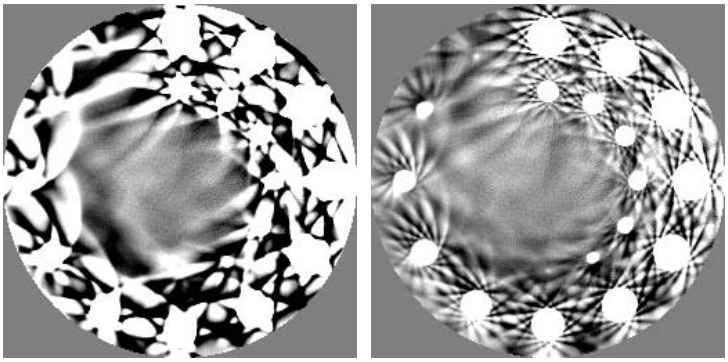
Once the link values of a PI-surface have been computed they may be ramp-filtered as described in the right hand side of (4.24). The filtered links are then combined into pixel values as in (4.23).

This backprojection step is identical to the two-dimensional case, but will nevertheless perform a three-dimensional backprojection. It does not have to consider the link positions in the s-direction, since this information is already taken care of in the link construction step in (4.25). The pixels on the PI-surfaces are finally interpolated in z-direction onto a Cartesian grid, for presentation and analysis.

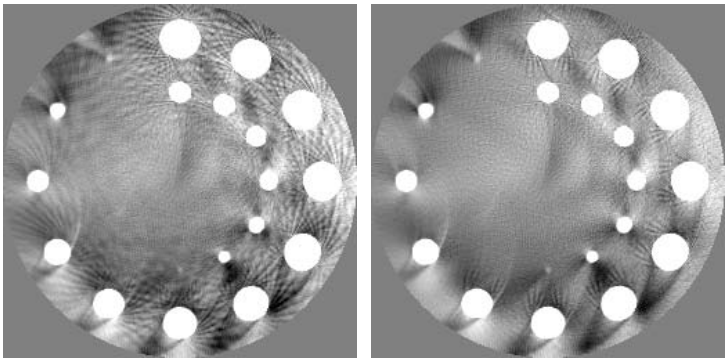
We summarise the computation steps in the PI-FAST algorithm as

Algorithm 4.11 *PI-FAST*

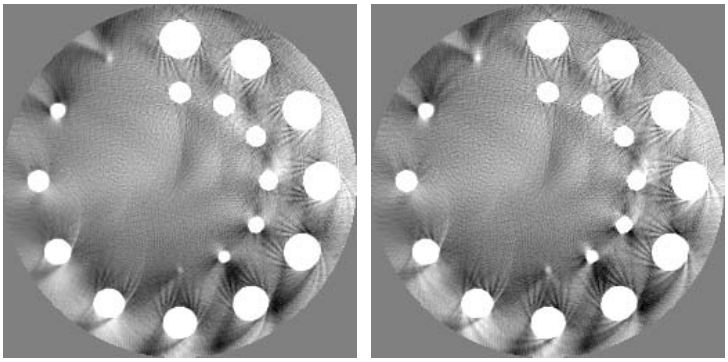
1: Pre-weight the projection data with $\cos\kappa$, where κ is the angle between each ray and a plane orthogonal to the rotation axis.
2: **for all** PI-surfaces **do**
3: Compute the link values according to (4.25) using the appropriate mappings to the acquisition geometry of the original data.
4: Ramp-filter the link values as in (4.24).
5: Compute the pixel values of the PI-surface using (4.23).
6: **end for**
7: Resample the pixels of the PI-surfaces in the z-direction onto a Cartesian grid.



(a) 4 sets of 128-links, $\sigma = 0.189$ (b) 8 sets of 64-links, $\sigma = 0.073$



(c) 16 sets of 32-links, $\sigma = 0.038$ (d) 32 sets of 16-links, $\sigma = 0.034$



(e) 64 sets of 8-links, $\sigma = 0.034$ (f) 128 sets of 4-links, $\sigma = 0.035$

Figure 4.24 *Reconstruction of the sphere clock phantom using PI-FAST with links of different lengths. Parameters as in Table 4.2. Greyscale interval* $[-0.05, 0.05]$.

Computational complexity

The computational complexity for reconstructing one PI-surface is derived as follows. With a link length of $\sqrt{N_\theta}$ we have $N_t\sqrt{N_\theta}$ trees. Each tree consists of $\mathcal{O}(\sqrt{N_\theta})$ links. The first step of the algorithm calculates the value of each link from $\sqrt{N_\theta}$ interpolation points along the link. The total number of interpolations and accumulations therefore becomes $\mathcal{O}(N_t\sqrt{N_\theta}\sqrt{N_\theta}\sqrt{N_\theta})$. The link values are then filtered using FFT. Switching the order of filtering and backprojection does not alter the total filtering effort. A total of $\mathcal{O}\sqrt{N_\theta}\sqrt{N_\theta}$ ramp-filterings with $\mathcal{O}(N_t \log N_t)$ operations each totals $\mathcal{O}(N_\theta N_t \log N_t)$ operations.

The second step of the algorithm sums the filtered links values along the sinusoids for each voxel. This requires $N_x N_y \sqrt{N_\theta}$ interpolations and accumulations. The total complexity of the algorithm is thus $\mathcal{O}(N_t N_\theta^{1.5} + N_\theta N_t \log N_t + N_x N_y \sqrt{N_\theta}) = \mathcal{O}(N^{2.5})$ per PI-surface and $\mathcal{O}(N^{3.5})$ in total for normal setups.

Discussion

The length of the links is a critical parameter in PI-FAST. If the links are too long they can not be approximated by line segments, which will result in artifacts. But short links do not discriminate between the contributions of different PI-surfaces as good as long links. Figure 4.24 shows how the reconstruction of the sphere clock phantom performs best when the link length is around $8-16$. The number of projection angles was $N_\theta = 1024$ in this experiment, which means that the optimal links length is slightly shorter than the previously mentioned rule of thumb that linear links can be used for links shorter than $\sqrt{N_\theta/2}$. Since the recursive nature of the algorithm is replaced by a two-step procedure the length of the links is not limited to a power of 2.

We may obtain a further insight into the actual filtering in PI-FAST by reformulating the fast backprojection as an ordinary backprojection. The frequency-distance relation of Edholm et al. (1986) says that sinusoids with the same slope $\frac{dt}{d\theta}$ for a specific projection angle θ in the sinogram correspond to image points on the same distance v from the virtual planar detector. The links are restricted to a few discrete slopes. The links of the same slope and starting angle therefore correspond to a swath of pixels extending all along the t-axis but restricted to a shorter segment along the v-axis. As the discretisation of the link slope becomes finer, the swaths become narrower in the v-direction. PI-FAST filters links with the same slope together. This implies that the shape of the projection of points belonging to the same PI-surface and with the same v-value determine the filtering curve to be used for these points. See Figure 4.25.

With this observation we can formulate PI-FAST as a filtered backprojection where each projection is filtered several times.

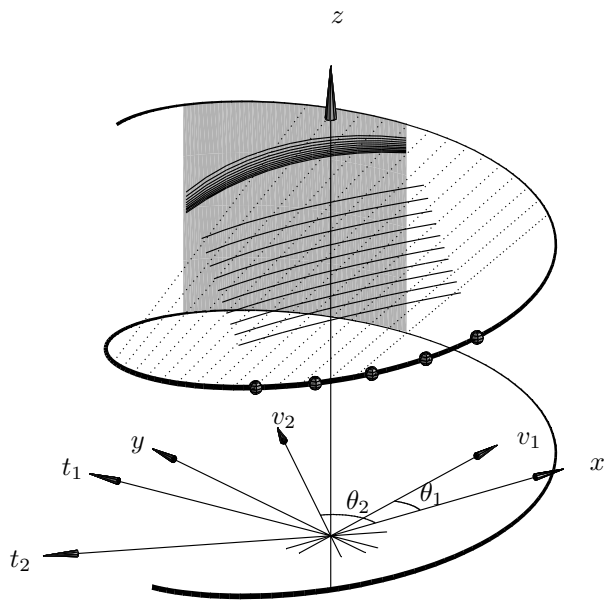


Figure 4.25 *Reconstruction of the PI-surface at $\theta = \theta_1$. The curves of constant v_2 for projection angle $\theta = \theta_2$ on this PI-surface are projected onto the PI-detector. The projected curves determine the filtering direction in the reformulation of PI-FAST.*

Algorithm 4.12 *Reformulation of PI-FAST*

1: Paralell rebinning of data restricted by the Tam window
2: Pre-weighting with $\cos\kappa$
3: **for all** projection angles θ **do**
4: **for all** PI-surfaces within the parallel cone **do**
5: **for all** swaths of equal v **do**
6: Projection of the swath on the PI-surface onto the planar virtual detector to determine the filtering curve
7: One-dimensional ramp-filtering along this curve
8: Three-dimensional backprojection of the filtered projection data onto the pixels of the swath
9: **end for**
10: **end for**
11: **end for**
12: Resampling in z-direction to a Cartesian voxel grid

The final interpolation on line 12 may be dispensed with if the backprojection on line 8 is made to the discrete voxel positions. Care must then be taken so that each voxel only gets one contribution from each projection angle. The order of the loops on line 3 and 4 may be switched.

The filtration curves are in essence the same as the ones in PI-SLANT. The difference is that in PI-SLANT the projection of the single line at $v = 0$ determines the filtration for the complete PI-surface. Image points projected outside this filtration line are filtered by filtration lines derived from neighbouring PI-surfaces. The filtration curves of one projection in PI-FAST may cross, which is why the projection data has to be filtered and backprojected swath by swath.

This reformulation shows that PI-FAST is approximate because the reconstruction of one PI-surface is made from projection data contaminated by nearby surfaces. The filtering is different from other proposed methods, although rather similar to PI-SLANT. Unfortunately, we have no better analysis tool than the experimental results in Section 4.3.7 to evaluate the actual performance of the algorithm.

4.3.5 The $\mathfrak{n}$-PI Methods

Proksa, Köhler, Grass, and Timmer (2000) have introduced a new important family of PI methods named the $\mathfrak{n}$-PI methods. By using properly shaped detector windows, some of the problems with the Tam-window are reduced while many of the nice properties of PI-ORIGINAL are retained.

Geometry and interrupted illumination

The PI-window is restricted in height by two consecutive turns of the helix. An $\mathfrak{n}$-PI-window is restricted by segments of the helix that are $\mathfrak{n}$ turns apart. See Figure 4.26. Valid values of $\mathfrak{n}$ are all odd positive integers, but only $\mathfrak{n} = 1, 3, 5, 7$ are used in practice. The beam is thus restricted by lines from the source to points on the helix approximately $\mathfrak{n}/2$ turns up or down. These lines are called $\mathfrak{n}$-PI-lines. Note that the PI-detector and PI-lines discussed in the sections above become a special case of the $\mathfrak{n}$-PI-detector and $\mathfrak{n}$-PI-lines when $\mathfrak{n} = 1$. We will therefore sometimes refer to the PI-detector as the 1-PI-detector.

Proksa et al. (2000) show how the Radon planes are triangulated in a similar fashion to the 1-PI case. The triangles are overlapping such that each point in each Radon plane is measured exactly $\mathfrak{n}$ times. An adaptation of the exact Radon-based algorithm of Kudo, Noo, and Defrise (1998) is formulated for the $\mathfrak{n}$-PI case. Furthermore, PI-ORIGINAL is reformulated for the $\mathfrak{n}$-PI geometry. We will call this algorithm $\mathfrak{n}$-PI-ORIGINAL.

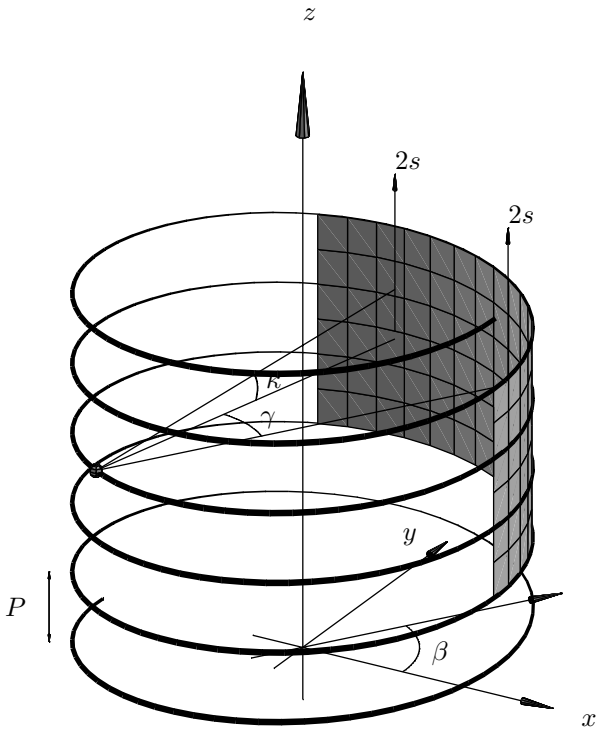


Figure 4.26 *The 3-PI-detector.*

The sampling of the $\mathfrak{n}$-PI-detector is performed along curved detector rows, projected onto the outer cylinder as

$$q(s,\gamma,\mathfrak{n}) = \frac{s+\gamma\frac{P}{\mathfrak{n}\pi}}{\cos\gamma}, \quad s \in [-\frac{\mathfrak{n}P}{4},\frac{\mathfrak{n}P}{4}] \tag{4.26}$$

These rows become less tilted as $\mathfrak{n}$ increases. A larger portion of the actual physical (γ, q)-detector on the outer cylinder can therefore be used.

Proksa et al. (2000) observe that some image points enter illumination more than once on the $\mathfrak{n}$-PI-detector when $\mathfrak{n} \geq 3$. This does not happen for the 1-PI geometry since every object point belongs to one and only one PI-line. But in the $\mathfrak{n}$-PI case, some points belong to more than one $\mathfrak{n}$-PI-line.

Figure 4.27 shows the points on a 3-PI-surface as projected onto the virtual planar detector during a projection interval of 3π. For $\theta = 0$ and $\theta = 3\pi$ they line up on the top and bottom rows of the detector respectively. But some points enter the detector window before and after the 3π interval and also enter and exit during the 3π interval.

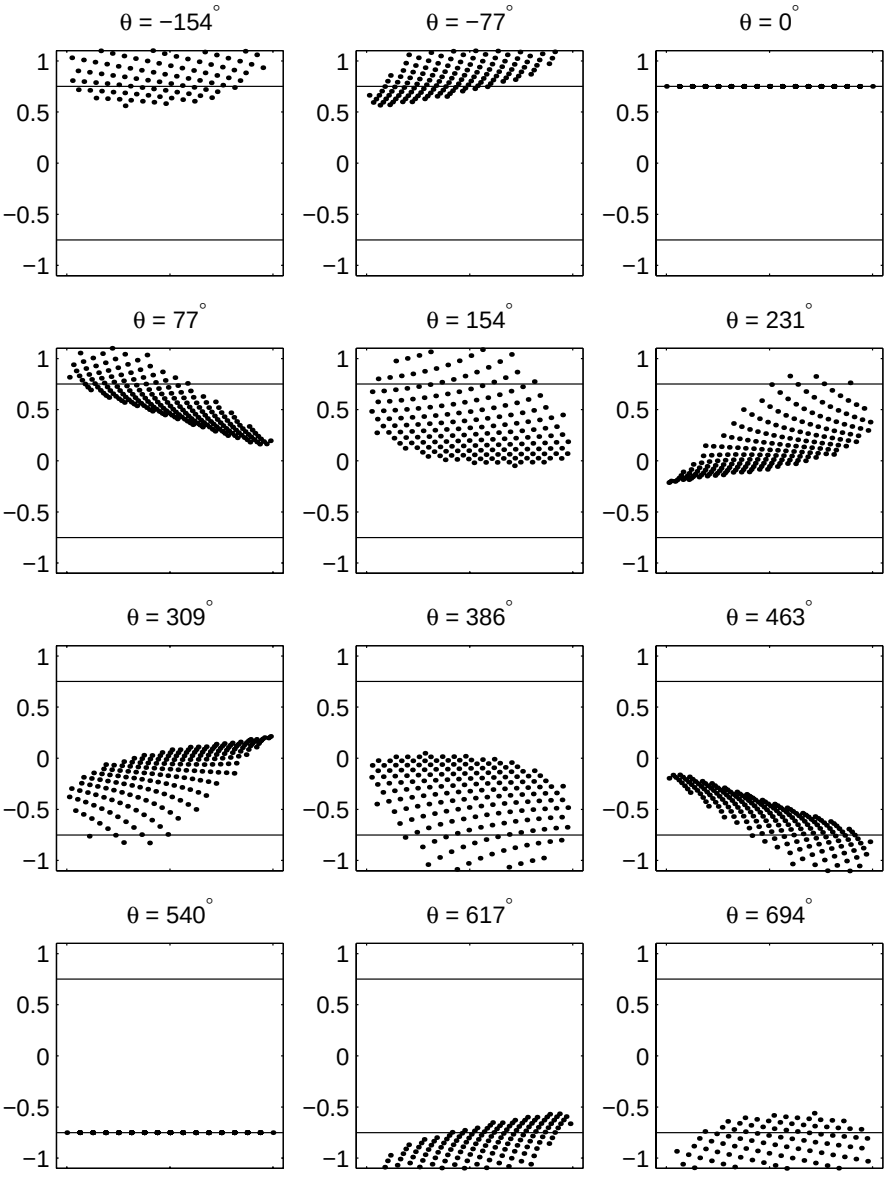


Figure 4.27 *Points on a 3-PI-surface as projected on the (t, s)-detector space. The detector window is limited to $s \in [-\frac{3P}{4}, \frac{3P}{4}] = [-0.75, 0.75]$, indicated with two lines.*

Algorithm

The $\mathfrak{n}$-PI-ORIGINAL algorithm is very similar to PI-ORIGINAL. The slightly altered geometry is implemented by exchanging the occurrences of the factor $\frac{P}{2\pi}$ with $\frac{P}{2\mathfrak{n}\pi}$ in the pre-weighting equation (4.12) and in the calculation of s in (4.16).

Algorithm 4.13 $\mathfrak{n}$*-PI-ORIGINAL*

1: Paralell rebinning of data restricted by the $\mathfrak{n}$-PI window
2: Pre-weighting with $\cos\kappa$
3: One-dimensional ramp-filtering along each row on the virtual planar detector
4: Three-dimensional backprojection without any L^{-2}-factor

The interrupted illumination makes the two voxel driven backprojections in Algorithm 4.6 and 4.7 unusable. However, the projection driven approach in Algorithm 4.8 will still work if the detector limits $[-\frac{P}{4}, \frac{P}{4}]$ are replaced by $[-\frac{\mathfrak{n}P}{4}, \frac{\mathfrak{n}P}{4}]$.

Possible extensions

As we have noted before, the use of nutating planes only makes sense for short-scan geometries. The projections of the 3-PI-surface in Figure 4.27 show that the methodology used in PI-SLANT and PI-2D does not work for $\mathfrak{n} > 1$. Filtering curves derived by curve fitting would intersect each other and extend outside the window.

The $\mathfrak{n}$-PI-detector can be seen as consisting of $\mathfrak{n}$ segments separated by the source trajectory. The mid-segment is always a 1-PI-detector. For this segment of the $\mathfrak{n}$-PI-detector, the filtering directions of PI-SLANT could be applied. For the other segments it is fruitless to find any nutating surfaces with projections that stay within the segment. Nothing but the original horizontal filtering can be recommended there. This combination of different filterings for different parts of the $\mathfrak{n}$-PI-detector has not been implemented.

The exact ZB and PHI methods are still to be reformulated for the $\mathfrak{n}$-PI geometry. An obvious problem with the ZB method is that it relies on the one-to-one correspondence between image points and PI-lines, something which does not hold for $\mathfrak{n}$-PI-lines.

4.3.6 Noise Homogeneity

Although the Tam-window ensures 180° illumination of all points after rebinning, the β-interval of real X-ray illumination before rebinning will be different for different points. This is an important observation that is likely to affect the noise properties of the PI methods. The images in Figure 4.28 show an (x, y)-slice such that the source trajectory intersects the slice to the left at $(-R, 0)$. The greyscale

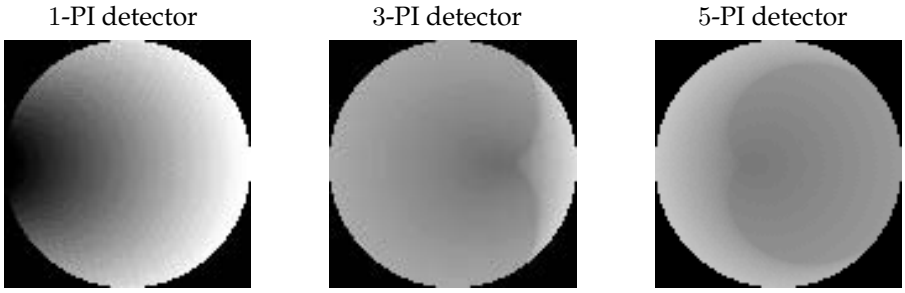


Figure 4.28 *Illumination maps of an (x, y)-slice with a source trajectory intersection in $(-R, 0)$. $R = 2.0$, $R_{FOV} = 1.0$, $P = 0.1$. The images are normalized such that the central pixel value is unity. Greyscale interval [0.7 1.3].*

of the voxels within the circular field of view is set to the number of X-ray projections each voxel receives before rebinning. This number is proportional to the real exposure time of the voxel. Slices above and below the one illustrated have identical uneven exposure maps apart form a rotation. Voxels close to the source are exposed for a shorter time than voxels further away.

The X-ray source-detector system is adequately modelled as a Poisson process. Each detector cell generates a stochastic intensity value I. For a Poisson distribution the variance σ_I^2 is equal to the expected number of photons μ_I. In this sense the noise in the projections is signal dependent. Line integral values are obtained by taking the logarithm according to (2.2). This results in a new stochastic variable $P = -\log(\frac{I}{I_0})$. The non-attenuated intensity I_0 can be seen as a constant. We write the conversion using the non-linear function $h(\cdot)$ as $P = h(I)$. The variance of P is given by the approximation rule of Gauss (Blom 1989) as

$$\sigma_P^2 = \sigma_I^2 \cdot (h'(\mu_I))^2 = \mu_I \frac{1}{\mu_I^2} = \frac{1}{\mu_I} \tag{4.27}$$

In contrast to the intensities, the line-integrals therefore have smaller variance as the intensity increases.

The line-integration values P can be placed in a long one-dimensional vector of length $N_\beta \cdot N_\gamma \cdot N_q$. Each element P_i is then an independent stochastic variable. In a similar way, we place all reconstructed values in a vector of length $N_x \cdot N_y \cdot N_z$. All steps in the PI methods are linear. The reconstructed value F_j is therefore a new stochastic variable that can be written as a linear combination of the projection values P_i:

$$F_j = w_{1,j}P_1 + w_{2,j}P_2 + \ldots + w_{N_\beta \cdot N_\gamma \cdot N_q, j}P_{N_\beta \cdot N_\gamma \cdot N_q, j} \tag{4.28}$$

The variance $\sigma^2_{F_j}$ of the voxel value F_j becomes

$$\sigma^2_{F_j} = w^2_{1,j}\sigma^2_{P_1} + w^2_{2,j}\sigma^2_{P_2} + \ldots + w^2_{N_\beta \cdot N_\gamma \cdot N_q, j}\sigma^2_{P_{N_\beta \cdot N_\gamma \cdot N_q, j}} \tag{4.29}$$

The values of $w_{i,j}$ are independent of the projection data but depend on the illumination geometry, the interpolation method used in the different rebinning steps, the ramp-filter convolution, and the backprojection implementation. It is the net effect of all these steps that determines the actual noise behaviour. We will not present any analytical expression of $\sigma^2_{F_j}$. Instead, we will use our reconstruction algorithm implementation to get an estimation of the noise in the slice as follows.

We add signal-dependent noise to synthetically generated noise-free projections of a homogeneous cylinder with a radius slightly smaller than R_{FOV}. The simple noise generation method described by Fuchs (2000) is used. The line-integral measurements $p(\beta, \gamma, q)$ are first turned into photon intensity values as

$$I(\beta, \gamma, q) = e^{-p(\beta,\gamma,q)} \tag{4.30}$$

Signal dependent noise is then added as

$$I^N(\beta, \gamma, q) = I(\beta, \gamma, q) + \sqrt{I(\beta, \gamma, q)}X, \quad X \in N(0, \sigma_X) \tag{4.31}$$

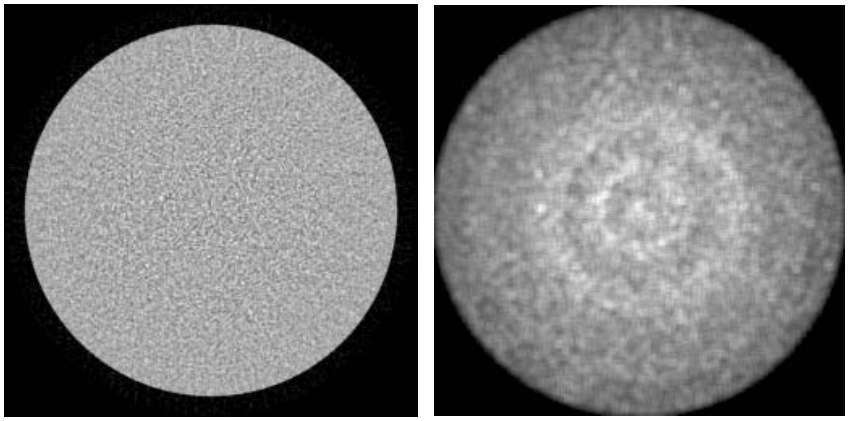
where we have set the variance of the normal distribution $N(0, \sigma_X)$ to 10% of the intensity of the central ray, $\sigma^2_X = 0.1I(0, 0, 0)$. The noisy intensity values are converted back into line-integral values as $p^N(\beta, \gamma, q) = -\log I^N(\beta, \gamma, q)$.

A slice of the volume reconstructed form the noisy data using PI-ORIGINAL is shown in Figure 4.29(a). The reconstruction used the parameters listed in Table 4.2. The local noise level σ_f is calculated using neighbouring slices from a noisy reconstruction f^N and a noise-free reconstruction f as

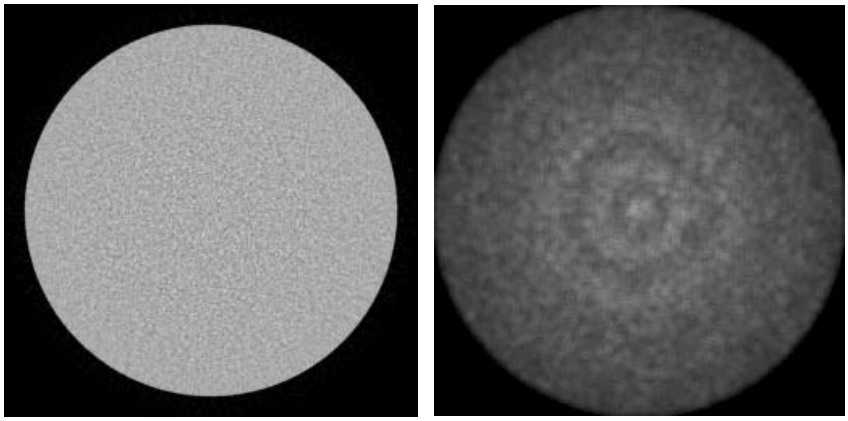
$$\sigma_f[i_1, i_2, i_3] = \sqrt{\frac{1}{125} \sum_{i'_1=i_1-2}^{i_1+2} \sum_{i'_2=i_2-2}^{i_2+2} \sum_{i'_3=i_3-2}^{i_3+2} \left(f^N[i'_1, i'_2, i'_3] - f[i'_1, i'_2, i'_3]\right)^2} \tag{4.32}$$

and plotted in Figure 4.29(b). The slices are very thin compared to the pitch, so that the averaging between the slices will not eliminate any possible non-rotationally symmetric component due to the helix. There is a clear increase in noise towards the centre of rotation. This is the case for most reconstruction algorithms. It can be explained by that fact that the reconstruction of the central parts mainly relies on projection values of rays intersecting the object at its widest. According to (4.27) such rays correspond to line-integral values with large variances.

No non-rotationally symmetric component due to the short-scan is apparent. We have no explanation for the centred circular wave pattern. The Figures 4.29(c) and (d) show similar plots for the 3-PI geometry. Naturally, the noise level is smaller due to the longer illumination interval. No non-rotationally symmetric component is visible here either.



(a) PI-ORIGINAL (b) σ_f



(c) 3-PI-ORIGINAL (d) σ_f

Figure 4.29 *Reconstruction from noisy data of a homogeneous cylinder of density 1.0. Parameters as in Table 4.2. Greyscale interval:* $[0.0, 1.5]$ *in (a) and (c);* $[0.0, 0.15]$ *in (b) and (d).*

Source trajectory radius	$R = 2.0$ length units (l.u.)
Projections per turn	$N_\beta = N_\theta = 512$
Pitch	$P = 1.0$ l.u. ($P = \frac{1}{\mathfrak{n}}$ l.u. for $\mathfrak{n} = 3, 5$)
Detector rows	$N_s = 64$
Detector row height	$\Delta s = \frac{P}{2N_s} = \frac{1}{128}$ l.u.
Detector elements per row	$N_\gamma = N_t = 255$
Maximum fan-angle	$\gamma_{\text{max}} = 30°$
Maximum cone-angle at $\gamma = 0$	$\kappa_{\text{max}} = \arctan \frac{P}{4R} \approx 7.13°$
Reconstruction grid	$N_x \times N_y \times N_z = 256 \times 256 \times 5$
Reconstruction grid resolution	$\Delta x = \Delta y = \Delta z = \frac{1}{128}$ l.u.
Simulated rays per detector element	18

Table 4.2 *Experiment parameters for the sphere clock phantom and for the noise simulations.*

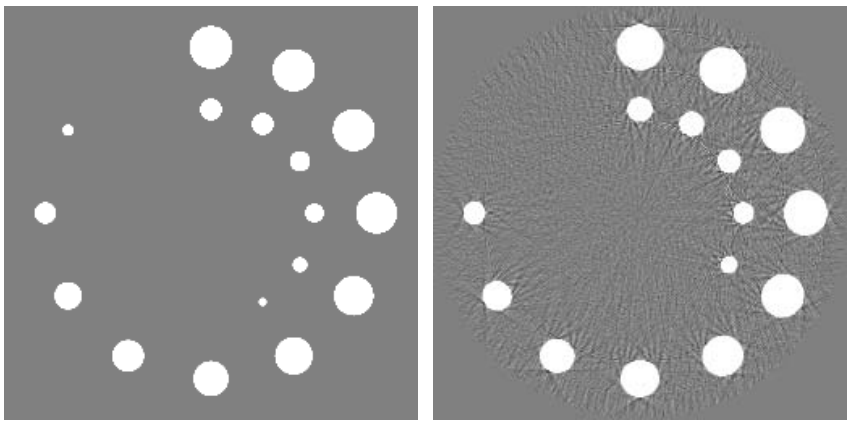
The variations of the exposure time for different voxels is obviously an insufficient model for the noise behaviour. Our experiment shows that noise inhomogeneity is not a major problem in 1-PI-ORIGINAL. The motivation for the $\mathfrak{n}$-PI methods should not be stated in terms of noise homogeneity but rather in their ability to obtain better SNR in general and reconstruct images with less artifacts.

4.3.7 Experimental Results

We will use two different experimental setups when comparing the algorithms presented above. The first phantom is designed to disclose artifacts stemming from the one-dimensional filtering employed by the different approximate algorithms. The second setup is an attempt to simulate the behaviour of the different algorithms in a possible future medical tomograph.

The sphere clock phantom

All algorithms employing one-dimensional ramp filtering presented in this chapter become exact if the object is homogeneous in the z-direction. We have therefore used a high contrast phantom named the sphere clock phantom presented in detail in Appendix B.1. This phantom has many sharp density variations in the z-direction to illustrate the image quality degradation due to the non-exactness approximations in eight of the algorithms presented above. The (x, y)-slice at $z = 0$ of the phantom is shown in Figure 4.30(a). The spheres at 12 o'clock are centred in this slice and the large sphere at 10 o'clock and the small sphere at 5 o'clock touch the slice. Because of the slice width, these touching spheres should be visible, but



(a) Phantom (b) Two-dimenisional reconstruction

Figure 4.30 *The sphere clock phantom. Greyscale interval* $[-0.05, 0.05]$.

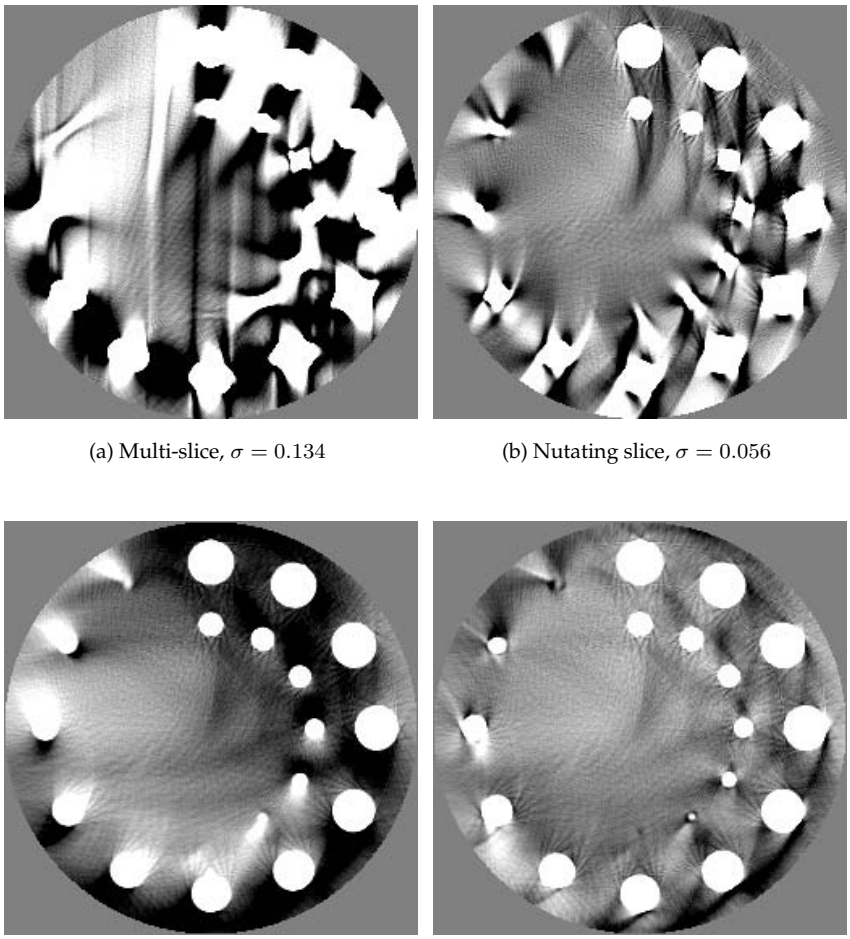
the spheres at 11 o'clock and 6 o'clock outside the slice should not appear in the reconstructed result. For comparison, the result using a circular trajectory and a two-dimensional filtered backprojection with corresponding parameters is shown in Figure 4.30(b).

Figures 4.31 and 4.32 shows reconstruction results using the parameters in Table 4.2. To emphasise the differences we have used a tight greyscale window and a detector with relatively many rows where none of the methods give perfect results. However, the experiment clearly shows that both the introduction of nutating slices and three-dimensional backprojection improve image quality. This conclusion is supported by the values of the root mean square error reported in the subfigure captions.

The artifacts in the nutating slice reconstruction resemble the ones in PI-2D. This is not surprising due to the similarities between the algorithms. PI-2D shows somewhat larger artifacts. We believe that this difference should be explained by the fact that the PI-surfaces are less optimal than the nutated planes as seen in Figure 4.11. The artifacts are similar and PI-2D will be used in the next experiment to indicate the expected performance of the other nutating slice algorithms.

The voxelised head phantom

Figures 4.33 and 4.35 show reconstruction results from simulated projections through the voxelised head phantom defined in Appendix B.3. The forward projection method of Köhler, Turbell, and Grass (2000), described in Section 5.1.4, has been



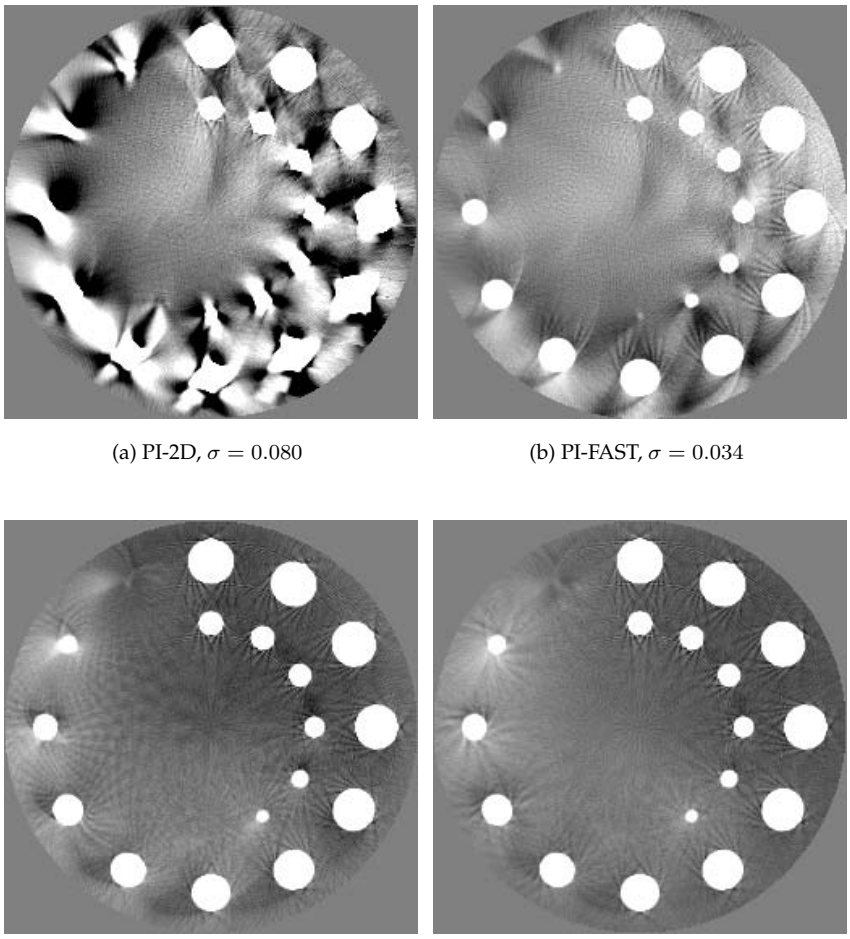
(a) Multi-slice, $\sigma = 0.134$

(b) Nutating slice, $\sigma = 0.056$

(c) PI-ORIGINAL, $\sigma = 0.050$

(d) PI-SLANT, $\sigma = 0.042$

Figure 4.31 *Reconstruction of the sphere clock phantom. Parameters as in Table 4.2. Greyscale interval* $[-0.05, 0.05]$.



(a) PI-2D, $\sigma = 0.080$

(b) PI-FAST, $\sigma = 0.034$

(c) 3-PI-ORIGINAL, $\sigma = 0.029$

(d) 5-PI-ORIGINAL, $\sigma = 0.028$

Figure 4.32 *Reconstruction of the sphere clock phantom. Parameters as in Table 4.2. Greyscale interval* $[-0.05, 0.05]$.

	64-row detector		16-row detector	
	σ	σ_{soft}	σ	σ_{soft}
PI-ORIGINAL	70.2	25.1	68.1	22.2
PI-SLANT	65.5	21.6	64.1	20.9
PI-2D	77.1	26.7	67.6	21.9
PI-FAST	73.5	24.1	71.8	23.3
3-PI-ORIGINAL	67.2	21.7	-	-

Table 4.3 *Total reconstruction error for all slices of the voxelised head phantom.*

used. Severe artifacts are shown for PI-ORIGINAL in Figures 4.33(b) and 4.3.7(b). The parameters for this experiment are found in Table 4.4. The detector consists of 64 rows which is more than expected in the next generation of tomographs. An alternative, less challenging, geometry with a 16-row detector is given in Table 4.5. The artifacts for PI-ORIGINAL are reduced greatly for this smaller detector as shown in Figure 4.35(b).

The reconstruction has been made on the same grid as the voxelised phantom was given on. It is thereby simple to calculate

$$\sigma = \sqrt{\frac{1}{N_x N_y N_z} \sum_{i_1=0}^{Nx-1} \sum_{i_2=0}^{Ny-1} \sum_{i_3=0}^{Nz-1} \left(f_{\text{RECONSTRUCTED}}[i_1, i_2, i_3] - f_{\text{PHANTOM}}[i_1, i_2, i_3]\right)^2} \tag{4.33}$$

as an error measure. Voxels close to bone contribute greatly to σ. As an alternative we picked out the voxels

$$S = \{(i_1, i_2, i_3) \,|\, 1000 < f_{\text{PHANTOM}}[i_1, i_2, i_3] < 1060\} \tag{4.34}$$

that contain soft tissue. The set S was used to create a binary volume. To avoid contributions from soft tissue voxels close to bone this binary volume was morphologically eroded. The set of remaining voxel positions, S', were used for the error measurement

$$\sigma_{\text{soft}} = \sqrt{\frac{1}{|S'|} \sum_{(i_1,i_2,i_3)\in S'} \left(f_{\text{RECONSTRUCTED}}[i_1, i_2, i_3] - f_{\text{PHANTOM}}[i_1, i_2, i_3]\right)^2} \tag{4.35}$$

We observe that the worst artifacts in PI-ORIGINAL disappear with the introduction of nutating PI-surfaces. The result from PI-SLANT is clearly superior to that of PI-2D. The major differences are found in the soft tissue close to the bone, where PI-2D produces "bleeding" effects.

PI-FAST produces a smoother result than the other algorithms. We expect this to be a result of the extra interpolation step due to the intermediate link value calculations.

Source trajectory radius	$R = 570$ mm
Projections per turn	$N_\beta = N_\theta = 1000$
Pitch	$P = 88.4$ mm (29.47 mm for $\mathfrak{n} = 3$)
Detector rows	$N_s = 64$
Detector row height	$\Delta s = \frac{P}{2N_s} \approx 0.69$ mm
Detector elements per row	$N_\gamma = N_t = 320$
Maximum fan-angle	$\gamma_{\text{max}} = 12°$
Maximum cone-angle at $\gamma = 0$	$\kappa_{\text{max}} = \arctan \frac{P}{4R} \approx 2.2°$
Reconstruction grid	$N_x \times N_y \times N_z = 320 \times 320 \times 64$
Reconstruction grid resolution	$\Delta x = \Delta y = \Delta z = \frac{221}{320}$ mm ≈ 0.69 mm
Simulated rays per detector element	18

Table 4.4 *Experiment parameters for the voxelised head phantom using a 64-row detector.*

Pitch	$P = 22.1$ mm
Detector rows	$N_s = 16$
Maximum cone-angle at $\gamma = 0$	$\kappa_{\text{max}} = \arctan \frac{P}{4R} \approx 0.56°$

Table 4.5 *Experiment parameters for the voxelised head phantom using a 16-row detector, other parameters as in Table 4.4.*

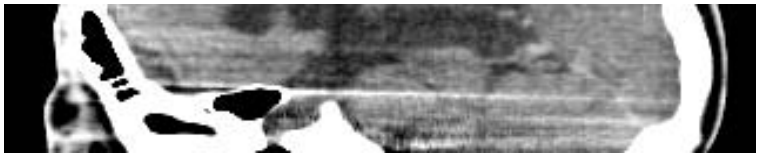
The main artifact of PI-ORIGINAL in 4.33(b) is substantially reduced but still visible in the result of 3-PI-ORIGINAL in 4.33(f). The comparison may be considered unfair since 3-PI-ORIGINAL required a three times larger projection data set for the reconstruction.

From the experiments we draw the conclusions:

- PI-SLANT produces the best results of the four 1-PI methods.
- in particular, PI-SLANT outperforms PI-2D, indicating that all nutating slice algorithms would benefit substantially if the two-dimensional backprojection was replaced by three-dimensional backprojection.
- 3-PI-ORIGINAL clearly outperforms 1-PI-ORIGINAL.

The results give no definite answer when comparing PI-ORIGINAL with PI-2D, or PI-SLANT with 3-PI-ORIGINAL.

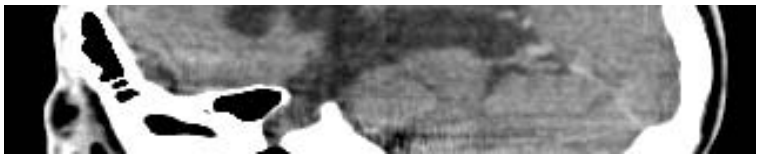
(a) Phantom



(b) PI-ORIGINAL, $\sigma_{\text{soft}} = 19.95$

(c) PI-SLANT, $\sigma_{\text{soft}} = 16.01$

(d) PI-2D, $\sigma_{\text{soft}} = 21.46$

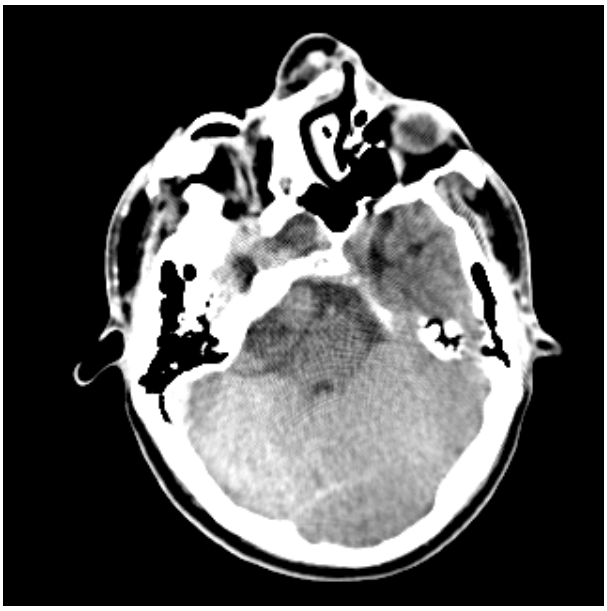


(e) PI-FAST, $\sigma_{\text{soft}} = 17.77$

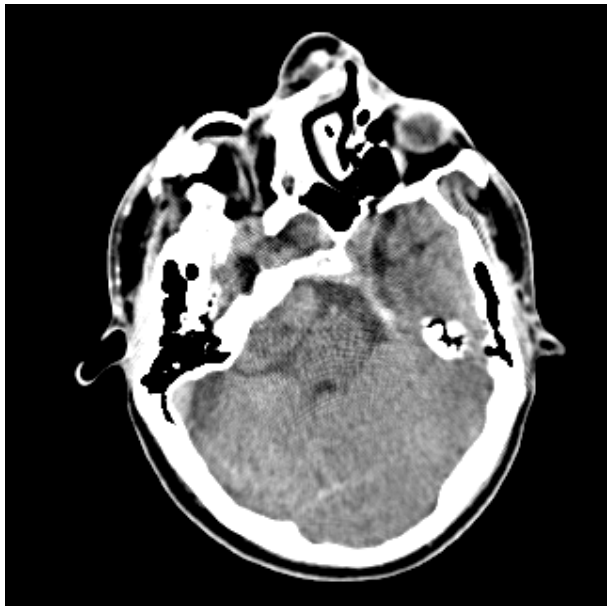
(f) 3-PI-ORIGINAL, $\sigma_{\text{soft}} = 15.98$

Figure 4.33 *Reconstructed (y, z)-slice through the voxelised head phantom with a 64-row detector. Parameters as in Table 4.4. Greyscale interval* $[980, 1080]$.

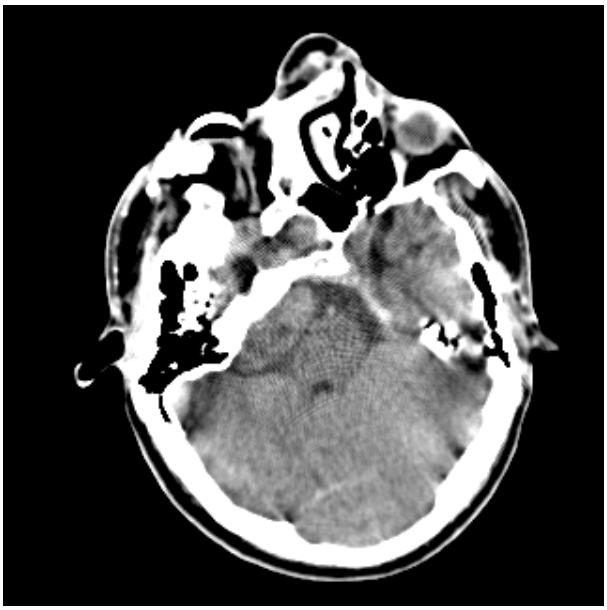
(a) Phantom



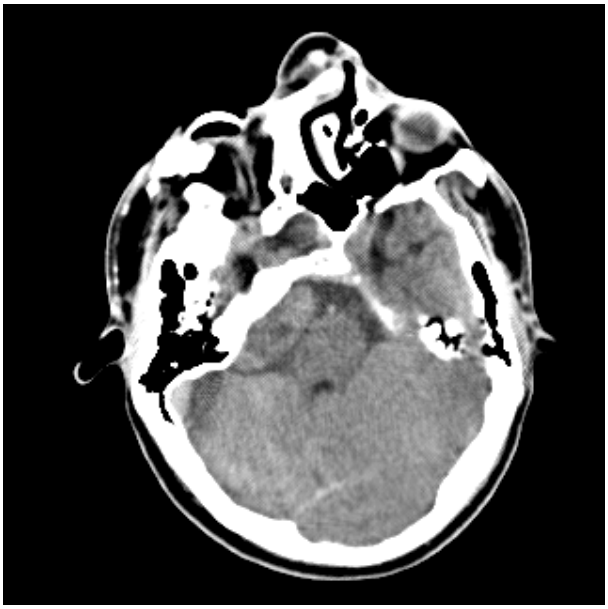
(b) PI-ORIGINAL, $\sigma_{\text{soft}} = 29.30$



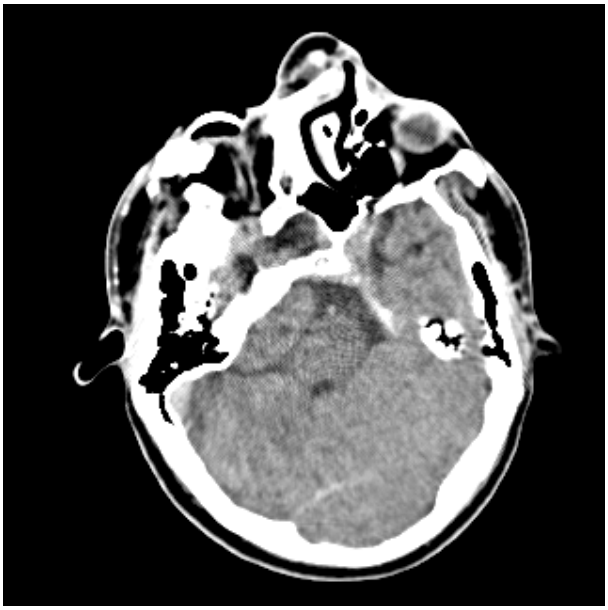
(c) PI-SLANT, $\sigma_{\text{soft}} = 24.00$



(d) PI-2D, $\sigma_{\text{soft}} = 29.67$



(e) PI-FAST, $\sigma_{\text{soft}} = 26.97$



(f) 3-PI-ORIGINAL, $\sigma_{\text{soft}} = 24.14$

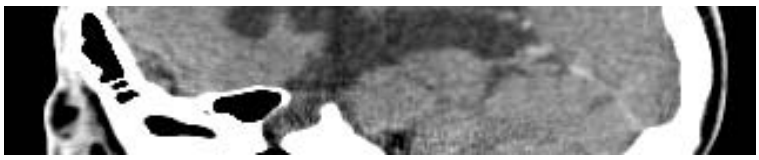
Figure 4.34 *Reconstructed* (x, y)*-slice through the voxelised head phantom with a 64-row detector. Parameters as in Table 4.4. Greyscale interval* $[980, 1080]$.

(a) Phantom

(b) PI-ORIGINAL, $\sigma_{\text{soft}} = 16.50$

(c) PI-SLANT, $\sigma_{\text{soft}} = 15.38$

(d) PI-2D, $\sigma_{\text{soft}} = 17.04$



(e) PI-FAST, $\sigma_{\text{soft}} = 16.42$

Figure 4.35 *Reconstructed (y, z)-slice through the voxelised head phantom with a 16-row detector. Parameters as in Table 4.5. Greyscale interval* $[980, 1080]$.

4.4 Discussion

In this chapter we have reviewed the existing methods for helical cone-beam reconstruction and presented a number of new methods. Different algorithms are suitable for different detector setups. An approximate general consensus is that multi-slice algorithms work for 4-row detectors and that the approximate algorithms using nutating slices or three-dimensional backprojection work for detectors with up to approximately 32 rows. Exact algorithms are required for larger detectors.

Our experiments indicate that the nutating slice algorithms presently investigated by several manufacturers can be improved upon by using three-dimensional backprojection as in PI-SLANT. Given these results, statements such as "ASSR, compared with other approximate cone-beam reconstruction algorithms, is the most promising method available today" (Kalender 2000) seem somewhat premature.

The experiments furthermore show that the n-PI-methods have a remarkable ability to reduce many of the artifacts appearing in the short-scan methods. They also provide control over the trade-off between scanning time and signal to noise ratio.

Most of the compared algorithms require $\mathcal{O}(N^4)$ operations to reconstruct a complete volume. PI-FAST requires $\mathcal{O}(N^{3.5})$ and the algorithms with two-dimensional backprojection can be accelerated to $\mathcal{O}(N^3 \log N)$ using direct Fourier techniques or fast backprojection on each slice. The actual resulting performance depends on many intricate implementation details. Our experience says that three-dimensional backprojection, as in PI-ORIGINAL, PI-SLANT, and n-PI-ORIGINAL, gives a computationally more expensive algorithm than the two-dimensional backprojection with pre- and post-interpolation found in the nutating surface algorithms.

5

Forward Projection through Voxel Volumes

Forward projection through voxel volumes is a major step in all iterative reconstruction methods. In the work of this thesis we have used forward projection when generating synthetic projections of the voxelized head phantom defined in Appendix B.3.

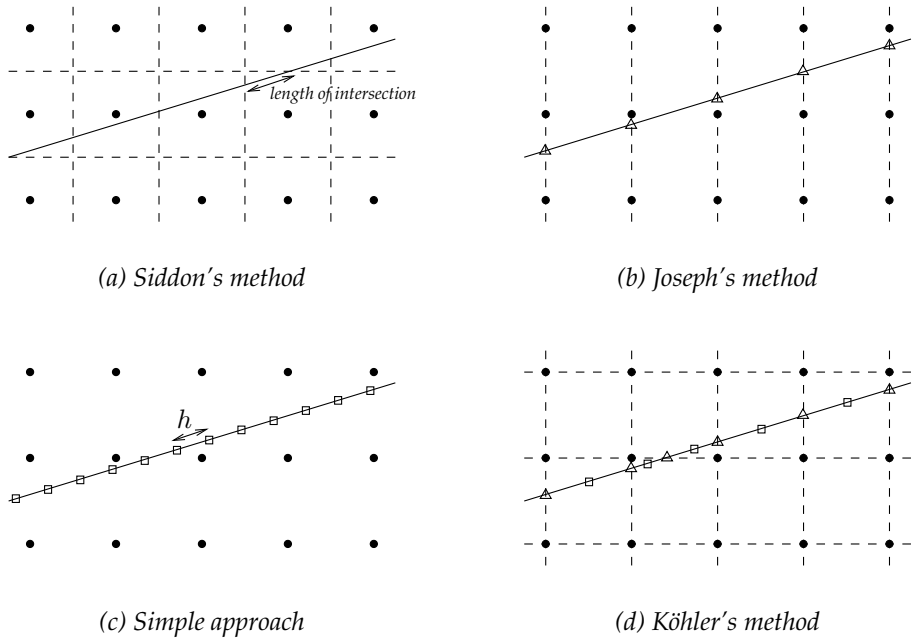
Most of the material in this chapter, apart from Section 5.3, is based on the presentation by Köhler, Turbell, and Grass (2000). Additional experiments have also been designed in collaboration with Thomas Köhler.

5.1 Methods

We present four different approaches to forward projection. All of these methods can be classified as ray-driven, in the sense that they traverse each ray through the voxel volume while accumulating the line-integration value. Alternative approaches are discussed in Section 5.3.

5.1.1 Siddon's Method

Figure 5.1(a) shows a three-dimensional voxel grid along one axis. The sample points are centred in the voxels. If nearest-neighbour interpolation is used, the line-integral can be easily calculated as a weighted sum of the values of the intersected voxels. The weights in the sum are naturally the length of intersection between each voxel and the ray. Siddon (1985) formulated an efficient way of



(a) Siddon's method *(b) Joseph's method*

(c) Simple approach *(d) Köhler's method*

Figure 5.1 *Four methods of line-integration as seen along one axis. Triangles indicate bi-linear interpolation, squares indicate tri-linear interpolation.*

calculating this length while traversing the ray. Other efficient ways of incrementally stepping through voxel volumes have been developed for use in computer graphics, see Cohen (1994) for further references.

The nearest neighbour interpolation is a too simplified model for most applications. Goertzen, Beekman, and Cherry (2000) present CT reconstruction experiments from data obtained using Siddon's method. They draw the conclusion that the input voxel volume has to be of at least the double resolution in each dimension compared to the desired resolution of the reconstruction.

5.1.2 Joseph's Method

Joseph (1982) describes a somewhat more elaborate forward projection method for the two-dimensional case. It is easily extended to three dimensions. Assume that the integral of the line segment between (x_1, y_1, z_1) and (x_2, y_2, z_2) is to be computed. By comparing $|x_1 - x_2|$, $|y_1 - y_2|$, and $|z_1 - z_2|$, the principal direction of the ray is first determined. We will now discuss the case where the principal direction is along the x-axis, i.e. when $|x_1 - x_2|$ is the largest of the three absolute differences. The other cases are handled in similar ways.

Figure 5.1(b) shows how a bi-linear interpolation is performed on the intersection points of each (y, z)-plane of the sampling grid. The sum of these interpolations is finally weighted with the factor

$$\frac{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}}{|x_1 - x_2|} \tag{5.1}$$

to compensate for the longer traversal of rays deviating from the principal direction.

5.1.3 A Simple Method

Köhler, Turbell, and Grass (2000) noted that the problem of calculating line-integrals can be separated into the two steps:

1. Construction of a continuous volume by interpolation from the discrete data
2. Line-integration in the continuous volume

A simple algorithm is obtained if the interpolation is chosen to be tri-linear and the integration through the continuous volume is approximated by summing values at equidistant points along the ray. See Figure 5.1(c).

The step width h is a free parameter that can be used to trade off between image quality and computation time. We introduce the quantity

$$N = \frac{\Delta x}{h} \tag{5.2}$$

A value of $N = 1$ corresponds to a sampling distance equal to the voxel sampling distance. The quantity N is therefore reffered to as an oversampling factor.

5.1.4 Köhler's Method

The Simpson rule of integration (Råde and Westergren 1990)

$$\int_a^b f(l)\, dl \approx \frac{b-a}{6}(f(a) + 4f(\frac{a+b}{2}) + f(b)) \tag{5.3}$$

can be used to numerically approximate the integral of a function given the function values at the integration borders and at the centre of the integration interval. For functions which are polynomials of third order the Simpson rule can be shown to be exact.

Let us define a *cell* as a box of sides Δx, Δy, and Δz with sampling points at its corners. The cells can be seen as voxels shifted half a sampling distance. Köhler et al. (2000) showed that tri-linear interpolation in step 1 above results in a volume where the density along a ray varies as a polynomial of third order within each

cell. It is therefore possible to calculate the line-integral in step 2 above analytically within each cell using the Simpson rule of integration.

The resulting method is illustrated in Figure 5.1(d) and consists of the following steps:

Algorithm 5.1 *Köhler's method of line-integration*

1: **for** each cell intersected by the ray **do**
2: calculate the function value at the two intersections of the cell wall using bi-linear interpolation
3: calculate the function value at the point halfway between the two intersection points above using tri-linear interpolation
4: combine the three values using Simpson's rule (5.3)
5: accumulate the result
6: **end for**

The values obtained by bi-linear interpolation at the wall intersections can be reused for the neighbouring cells. The free parameter h appearing in the simple approach is not used in the Köhler method.

5.2 Experimental results

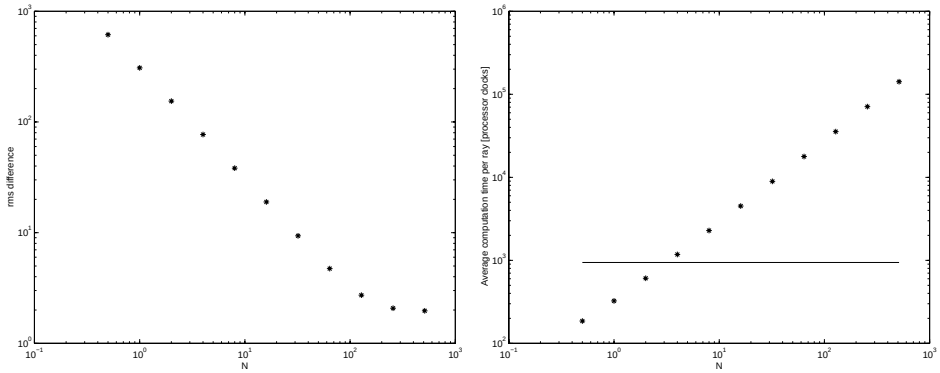
We have calculated forward projections through the voxelized head phantom using the simple approach and Köhler's method. Figure 5.2(a) shows the root mean square difference between the two methods for different oversampling factors. A high oversampling factor is clearly needed in the simple approach to obtain the accuracy of Köhler's method. Figure 5.2(b) shows that the simple approach requires substantially more computation time for such a high oversampling factor.

In a second experiment we used an analytically described phantom with known projection values. The phantom was discretized as input to the algorithms. The results could then be compared with the ideal analytical values.

In order to avoid most of the aliasing a phantom such as a solid box or ellipsoid would result in we have chosen a spherically symmetric Kaiser-Bessel window as phantom. This window has only small frequency components above the Nyquist frequency related to our sampling distance Δx. It is defined as

$$f(x,y,z) = \frac{I_0(\alpha\sqrt{1-\frac{x^2+y^2+z^2}{a^2}})}{I_0(\alpha)} \tag{5.4}$$

where I_0 is the modified Bessel function of order 0. The free parameters a and α have been chosen as $a = N_x/2 = 4$ and $\alpha = \pi a$. The phantom was sampled at $(N_x \cdot M)^3 = 512M^3$ voxels of side $1/M$, where M is a variable oversampling factor.



(a) Root mean square difference in arbitrary units between the simple approach and Köler's method. as a function of the oversampling N used in the simple approach.

(b) Average time needed by the simple approach as a function of the oversampling N. The line indicates the time needed using Köhler's approach.

Figure 5.2 *Comparisons between the simple approach and Köhler's method.*

Lewitt (1990) showed that the projection of a Kaiser-Bessel window only depends on the distance d between the ray and the window centre as

$$p(d) = \frac{2a}{\alpha I_0(\alpha)} \sinh\left(\alpha\sqrt{1-\frac{d^2}{a^2}}\right) \tag{5.5}$$

The root mean square of the difference between these ideal projection values and the calculated ones are plotted in Figure 5.3 as a function of the oversampling factor M. The Siddon method is clearly inferior to the methods of Joseph and Köhler that perform similarly.

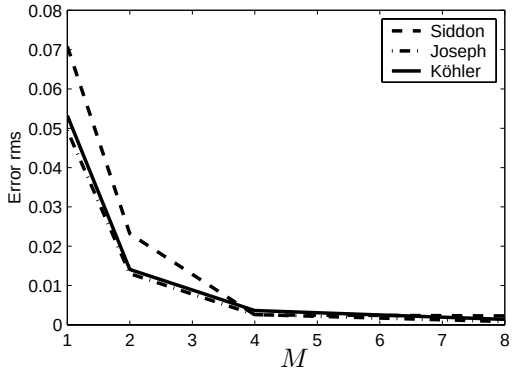


Figure 5.3 *Root mean square error of the projection values for three different forward projection methods as a function of the oversampling rate M.*

5.3 Discussion

At first sight, it might be surprising to see that the method of Joseph performs as good as the method of Köhler. We will now present an non-formal explanation.

The Siddon method can easily be expressed in the two steps mentioned above. The first step is an interpolation with the box function shown in Figure 5.4(a). The resulting volume is constant within each voxel. The weighting with the length of intersection clearly evaluates the line-integral through this volume analytically correct.

The Joseph method can also be expressed in the two steps. The interpolation should then be seen as being performed with a *sheared* tri-linear kernel as shown in Figure 5.4(b). The kernel axis parallel to the principal direction of the ray is sheared to be parallel with the ray. The other two kernel axes are kept unchanged. When the sheared axis of the kernel is parallel to the ray it is clear that an analytical integral through the kernel is identical to the bi-linear interpolation and compensation factor (5.1) in Joseph's method. Any smoothing in the direction of the ray is eliminated by the integration.

The only difference between the Joseph and Köhler methods is hence that the interpolation kernel in the Joseph method is a skewed version of the Köhler kernel in Figure 5.4(c). This should not have dramatic effects on the obtained quality. It is not obvious which kernel is optimal.

Figure 5.4(d) shows a rotated tri-linear kernel which could be considered as an alternative. However, the sum of the kernel values at the sample points not will be constant over the volume. It is therefore hard to formulate a practical algorithm for this kernel.

The four presented methods are all ray-driven. Alternative methods also exist. Lewitt (1992) has proposed the use of spherically symmetric interpolation kernels, also known as blobs. Mueller, Yagel, and Wheller (1999) investigate voxel driven approaches where a projection, a *footprint*, of each voxel is accumulated, *splatted*, onto the detector.

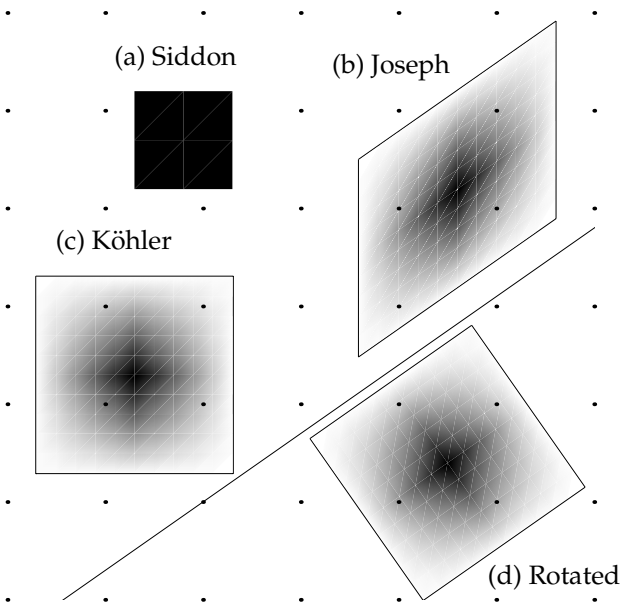


Figure 5.4 *Four interpolation kernels as seen along one axis. The line indicates the ray along which the integration is performed.*

6

Summary

We have in this thesis reviewed existing algorithms and presented new methods for cone-beam reconstruction. The main focus has been on approximate algorithms employing filtered backprojection with one-dimensional ramp-filtering.

In Chapter 3 we introduced the concept of parallel rebinning for the circular cone-beam geometry. The resulting algorithm P-FDK was the basis for the T-FDK method of Grass et al. (2000a). Our experiments confirmed that T-FDK gives reconstruction results of higher quality than the original FDK method. We also proposed the new FDK-SLANT method as an attempt of transferring the successful nutating slice algorithm from the helical to the circular case. The experimental results show both promising and discouraging features in the reconstructed images. A more thorough image quality investigation is under way.

We formulated the FDK-FAST method, which is the first cone-beam algorithm employing fast backprojection, reducing the asymptotic computational complexity from $\mathcal{O}(N^4)$ to $\mathcal{O}(N^3 \log N)$. We indicated that the practicality of FDK-FAST could be limited, despite several schemes of reducing the memory requirements of the algorithm.

In Chapter 4 we presented both exact and approximate methods for helical cone-beam reconstruction. Central in the presentation was the detector window of Tam (1995), which has been used by other authors for exact reconstruction. We named this window the PI-detector since it restricts the illuminatation interval length of all object points to π as seen from the points.

Although parallel rebinning has been known for long and suggested for use in helical cone-beam scanning by other authors, nowhere is this technique more

valuable than when applied to PI-detector data as done in the PI methods. We presented and evaluated several alternative implementations of the backprojection. We also gave a new proof of the fact that the Tam window ensures that the local Fourier domains of all object points are fully measured.

The original PI method performs well for moderate cone-angles, but starts to produce artifacts as the cone-angle increases. The predecessor of this thesis, the licentiate thesis of Turbell (1999), ended with the statement "It is quite likely that a deeper theoretical understanding of the approximations involved may result in a more exact algorithm and become an important part of future work on helical cone-beam reconstruction." We are pleased to see that this statement has turned out to be true. Evidence of this fact are the three new PI methods: PI-SLANT, PI-2D, and PI-FAST, together with the n-PI methods of Proksa et al. (2000).

We have investigated the noise homogeneity of the PI methods. Despite the uneven exposure time for different voxels, the noise in the reconstructed images was shown to be rotationally symmetric.

In addition to presenting new algorithms, large parts of the thesis have reviewed existing algorithms of other authors. Some of the new contributions of these review sections are:

- the novel categorisation of approximate cone-beam algorithms in Table 1.2.
- the investigation on which properties hold for the different variations on FDK reconstruction in Section 3.2.3.
- the discussion on the limitations of texture mapping hardware for the acceleration of three-dimensional backprojection in Section 3.2.1.
- the categorisation of exact helical cone-beam algorithms in Table 4.1.
- the discussion on the similarities and limitations of the nutating slice methods in Section 4.2.2.
- the proposal of further optimisation of the nutating slice algorithms in Section 4.2.2 by using polynomial surfaces and gradient descent minimisation.
- the suggestions on improvement of the gridding step of the MFR method in Section 4.2.4.
- the suggestions on extensions of the n-PI methods in Section 4.3.5.
- the use of a sheared tri-linear kernel in the comparison between the forward projection methods of Joseph and Köhler in Section 5.3.

It is not unlikely that some of the newer algorithms reviewed or presented in the thesis will be used in future CT systems. The final choices made by the manufacturers depend on a multitude of factors beyond the scope of this thesis.

A

Preservation of Line Integrals

In Section 3.3 we mentioned that the FDK-property of preserved line integrals in the z-direction is not valid when the filtering is performed along non-parallel lines. In this appendix we have a more formal look at the problem and the remedy.

A.1 Projection of a Point

For the discussions in the following sections we need to derive the analytic expression of the projection of a point in a cone-beam system. We start with the two-dimensional fan-beam case.

Consider a two-dimensional point density at $\mathbf{x}_0 = (x_0, y_0)^T$. See Figure A.1. Using the dirac delta function we write the the point density as ${}^2\delta(\mathbf{x}-\mathbf{x}_0)$. We will compute its fan-beam projection in the projection direction $\boldsymbol{\beta} = (\cos\beta, \sin\beta)^T$. A virtual linear detector detects rays intersecting the detector at $a\boldsymbol{\beta}^\perp$ where $\boldsymbol{\beta}^\perp = (-\sin\beta, \cos\beta)^T$ is orthogonal to $\boldsymbol{\beta}$. An arbitrary ray starts at $-R\boldsymbol{\beta}$ and continues in the direction $(a\boldsymbol{\beta}^\perp - (-R)\boldsymbol{\beta})$. The projection of the point density ${}^2\delta(\mathbf{x} - \mathbf{x}_0)$ becomes a scaled dirac pulse on the detector.

$$p^F(\beta, a) = \frac{\sqrt{R^2 + a^2}}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}} \delta(a - R\frac{\mathbf{x}_0 \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}) \tag{A.1}$$

To get a better intuitive understanding of the scaling factor we may factorise it as

$$\frac{\sqrt{R^2 + a^2}}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}} = \frac{R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}} \cdot \frac{\sqrt{R^2 + a^2}}{R} = \frac{R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}} \cdot \frac{1}{\cos\gamma} \tag{A.2}$$

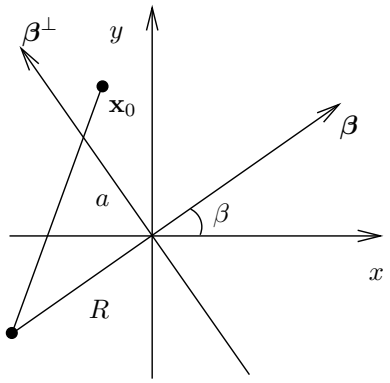


Figure A.1 *Fan-beam projection of a point.*

where the first factor can be interpreted as the geometrical magnification due to the beam divergence. The second factor has the same geometrical explanation as the fact that the sun casts longer shadows when it moves from its zenith.

We now consider the three-dimensional cone-beam case. We are interested in the projection $p^F(\beta, a, b)$ of the point density at $\mathbf{x}_0 = (x_0, y_0, z_0)^T$ in the projection direction $\boldsymbol{\beta} = (\cos\beta, \sin\beta, 0)^T$. Similar to the fan-beam case, the scaling of the dirac pulse in the z-direction consists of a magnification and a cos-factor, written as

$$\frac{dz}{db} \cdot \frac{1}{\cos\kappa} = \frac{\sqrt{R^2+a^2}}{\sqrt{a^2+(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}} \cdot \frac{\sqrt{R^2+a^2+b^2}}{\sqrt{R^2+a^2}} = \frac{\sqrt{R^2+a^2+b^2}}{\sqrt{a^2+(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}} \quad \text{(A.3)}$$

which multiplied with the fan-beam scaling gives us the total cone-beam scaling

$$\frac{\sqrt{R^2+a^2}}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}} \cdot \frac{\sqrt{R^2+a^2+b^2}}{\sqrt{a^2+(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}} = \frac{R\sqrt{R^2+a^2+b^2}}{(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2} \quad \text{(A.4)}$$

The point density ${}^3\delta(\mathbf{x}-\mathbf{x}_0)$ is thus projected as

$$p^F(\beta,a,b) = \frac{R\sqrt{R^2+a^2+b^2}}{(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}\delta(a - R\frac{\mathbf{x}_0\cdot\boldsymbol{\beta}^\perp}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}})\delta(b - \frac{z_0 R}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}}) \quad \text{(A.5)}$$

The projection of a complete object $f(\mathbf{x})$ may be written as the superposition of all point densities

$$p^F(\beta,a,b) = \iiint f(\mathbf{x}_0)\frac{R\sqrt{R^2+a^2+b^2}}{(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}\delta(a - R\frac{\mathbf{x}_0\cdot\boldsymbol{\beta}^\perp}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}})\delta(b - \frac{z_0 R}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}})\, dx_0 dy_0 dz_0 \quad \text{(A.6)}$$

A.2 FDK Reconstruction

The FDK-method reconstructs the image value $f_{\text{FDK}}(\mathbf{x})$ from the two-dimensional projections $p^F(\beta, a, b)$ as

$$f_{\text{FDK}}(\mathbf{x}) = \int_0^{2\pi} \underbrace{\frac{R^2}{(R+\mathbf{x}\cdot\boldsymbol{\beta})^2}}_{U^{-2}\text{-factor}} \int \underbrace{g^P(a-a')}_{\text{Ramp-filter}} p^F(\beta, a', b) \underbrace{\frac{R}{\sqrt{R^2+a'^2+b^2}}}_{\text{Pre-weighting}} da' d\beta \tag{A.7}$$

where

$$a = R\frac{\mathbf{x}\cdot\boldsymbol{\beta}^\perp}{R+\mathbf{x}\cdot\boldsymbol{\beta}} \tag{A.8}$$

and

$$b = z\frac{R}{R+\mathbf{x}\cdot\boldsymbol{\beta}} \tag{A.9}$$

The result $f_{\text{FDK}}(\mathbf{x})$ is only approximate and differs from the true $f(\mathbf{x})$. We will now show that line integrals in the z-direction are reconstructed exactly, so that

$$\int f_{\text{FDK}}(x,y,z)\,dz = \int f(x,y,z)\,dz \tag{A.10}$$

We follow the proof in the appendix of the original article of Feldkamp, Davis, and Kress (1984).

If we insert the projections of $f(\mathbf{x})$ as written in (A.6) into (A.7) and take an integral over z, we find that

$$\begin{aligned}\int f_{\text{FDK}}(\mathbf{x})\,dz = \int\int_0^{2\pi} \frac{R^2}{(R+\mathbf{x}\cdot\boldsymbol{\beta})^2} \int g^P(a-a') \\ \times \iiint f(\mathbf{x}_0)\frac{R\sqrt{R^2+a'^2+b^2}}{(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}\delta(a' - R\frac{\mathbf{x}_0\cdot\boldsymbol{\beta}^\perp}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}})\delta(b-\frac{z_0 R}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}})\,dx_0dy_0dz_0 \\ \times \frac{R}{\sqrt{R^2+a'^2+b^2}}\,da'd\beta dz\end{aligned} \tag{A.11}$$

The dirac pulse fixates the integration over a' which together with further simplification yields

$$\begin{aligned}\int f_{\text{FDK}}(\mathbf{x})\,dz = \int_0^{2\pi} \frac{R^2}{(R+\mathbf{x}\cdot\boldsymbol{\beta})^2} \iiint g^F(a - R\frac{\mathbf{x}_0\cdot\boldsymbol{\beta}^\perp}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}}) \\ \times f(\mathbf{x}_0)\frac{R^2}{(R+\mathbf{x}_0\cdot\boldsymbol{\beta})^2}\int \delta(b-\frac{z_0 R}{R+\mathbf{x}_0\cdot\boldsymbol{\beta}})\,dz dx_0 dy_0 dz_0 d\beta\end{aligned} \tag{A.12}$$

The final integral over z can be evaluated by inserting (A.9) as

$$\int \delta(b - \frac{z_0 R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}})\, dz = \int \delta(z \frac{R}{R + \mathbf{x} \cdot \boldsymbol{\beta}} - \frac{z_0 R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}})\, dz = \frac{R + \mathbf{x} \cdot \boldsymbol{\beta}}{R} \quad \text{(A.13)}$$

Equation (A.12) then simplifies to

$$\int f_{\text{FDK}}(\mathbf{x})\, dz = \iiint f(\mathbf{x}_0) \int_0^{2\pi} \frac{R}{(R + \mathbf{x} \cdot \boldsymbol{\beta})} g^P(a - R \frac{\mathbf{x}_0 \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}) \times \frac{R^2}{(R + \mathbf{x}_0 \cdot \boldsymbol{\beta})^2}\, d\beta dx_0 dy_0 dz_0 \quad \text{(A.14)}$$

The integral over β describes the two-dimensional fan-beam filtered backprojection reconstruction of the point density ${}^2\delta(\mathbf{x} - \mathbf{x}_0)$. For sufficient detector and angular resolution, this reconstruction is exact. Equation A.14 thus reduces to

$$\int f_{\text{FDK}}(\mathbf{x})\, dz \;=\; \iiint f(\mathbf{x}_0) \delta(x - x_0) \delta(y - y_0)\, dx_0 dy_0 dz_0 \;=\; \int f(\mathbf{x})\, dz \quad \text{(A.15)}$$

which completes the proof.

A.3 Slanted Filtering

Now assume that the filtering is performed along curves $b = h(a, l)$. We then parametrise our projection data over the curves, yielding

$$p^h(\beta, a, l) = p^F(\beta, a, b) \quad b = h(a, l) \quad l = h^{-1}(a, b) \quad \text{(A.16)}$$

The function $h^{-1}(a, b)$ is defined such that $h(a, h^{-1}(a, b)) = b$ and $h^{-1}(a, h(a, l)) = l$. The projection of the point density ${}^3\delta(\mathbf{x} - \mathbf{x}_0)$ can then written as

$$\begin{aligned} p^h(\beta, a, l) &= p^F(\beta, a, h(a, l)) \\ &= \frac{R\sqrt{R^2 + a^2 + h(a,l)^2}}{(R + \mathbf{x}_0 \cdot \boldsymbol{\beta})^2} \delta(a - R \frac{\mathbf{x}_0 \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}) \delta(h(a, l) - \frac{z_0 R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}) \\ &= \frac{1}{\left|\frac{\partial h}{\partial l}(a, l)\right|} \frac{R\sqrt{R^2 + a^2 + h(a,l)^2}}{(R + \mathbf{x}_0 \cdot \boldsymbol{\beta})^2} \delta(a - R \frac{\mathbf{x}_0 \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}) \delta(l - h^{-1}(a, \frac{z_0 R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}})) \end{aligned} \quad \text{(A.17)}$$

The modified FDK algorithm performing ramp-filtering along the curves described by $h(a, l)$ is written as

$$f_{\text{FDK-}h}(\mathbf{x}) = \int_0^{2\pi} \frac{R^2}{(R + \mathbf{x} \cdot \boldsymbol{\beta})^2} \times \int g^P(a - a') p^h(\beta, a', l) \frac{R}{\sqrt{R^2 + a'^2 + h(a, l)^2}}\, da' d\beta \quad \text{(A.18)}$$

where

$$l = h^{-1}(a', b) = h^{-1}(a', z\frac{R}{R + \mathbf{x} \cdot \boldsymbol{\beta}}) \tag{A.19}$$

By inserting the projections (A.17) into (A.18) and simplifying as in (A.14) while noting that

$$\int \delta(l - \dots)\, dz = \frac{1}{\left|\frac{\partial l}{\partial z}\right|} = \frac{1}{\left|\frac{\partial l}{\partial b}\frac{\partial b}{\partial z}\right|} = \left|\frac{\partial h}{\partial l}\right| \frac{R + \mathbf{x} \cdot \boldsymbol{\beta}}{R} \tag{A.20}$$

we obtain

$$\begin{aligned}\int f_{\text{FDK-}h}(\mathbf{x})\, dz = \iiint & \frac{1}{\left|\frac{\partial h}{\partial l}(R\frac{\mathbf{x}_0 \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}, \frac{z_0 R}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}})\right|} f(\mathbf{x}_0) \\ \times \int_0^{2\pi} & \left|\frac{\partial h}{\partial l}(R\frac{\mathbf{x} \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x} \cdot \boldsymbol{\beta}}, \frac{zR}{R + \mathbf{x} \cdot \boldsymbol{\beta}})\right| \frac{R}{(R + \mathbf{x} \cdot \boldsymbol{\beta})} g^P(a - R\frac{\mathbf{x}_0 \cdot \boldsymbol{\beta}^\perp}{R + \mathbf{x}_0 \cdot \boldsymbol{\beta}}) \\ & \times \frac{R^2}{(R + \mathbf{x}_0 \cdot \boldsymbol{\beta})^2}\, d\beta dx_0 dy_0 dz_0 \end{aligned} \tag{A.21}$$

The factors making (A.21) different from (A.14) will disappear if each projection value is multiplied with $\frac{\partial h}{\partial l}(a, l)$ *before* ramp-filtering and divided with $\frac{\partial h}{\partial l}(a, l)$ *after* ramp-filtering. The reconstruction will then preserve the line integrals in the z-direction. The partial derivative $\frac{\partial h}{\partial l}(a, l)$ can be seen as the inverse density of the filtering curves at detector position (a, l).

B

Phantom Definitions

B.1 The Sphere Clock Phantom

The sphere clock phantom is shown in Figure B.1. It consists of 12 spheres placed on a helix of radius 0.8 and pitch 0.12 together with 12 smaller spheres placed on a helix in the opposite direction of radius 0.5 and pitch 0.12. All spheres are of density 1.0. The phantom parameters are given in Table B.1.

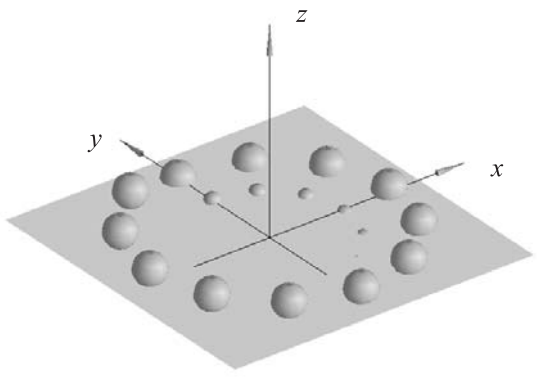


Figure B.1 *The sphere clock phantom.*

c_x	c_y	c_z	r	ρ
0.00	0.80	0.00	0.10	1.0
0.40	0.69	0.01	0.10	1.0
0.69	0.40	0.02	0.10	1.0
0.80	0.00	0.03	0.10	1.0
0.69	−0.40	0.04	0.10	1.0
0.40	−0.69	0.05	0.10	1.0
0.00	−0.80	0.06	0.10	1.0
−0.40	−0.69	0.07	0.10	1.0
−0.69	−0.40	0.08	0.10	1.0
−0.80	0.00	0.09	0.10	1.0
−0.69	0.40	0.10	0.10	1.0
−0.40	0.69	0.11	0.10	1.0
0.00	0.50	0.00	0.05	1.0
0.25	0.43	−0.01	0.05	1.0
0.43	0.25	−0.02	0.05	1.0
0.50	0.00	−0.03	0.05	1.0
0.43	−0.25	−0.04	0.05	1.0
0.25	−0.43	−0.05	0.05	1.0
0.00	−0.50	−0.06	0.05	1.0
−0.25	−0.43	−0.07	0.05	1.0
−0.43	−0.25	−0.08	0.05	1.0
−0.50	0.00	−0.09	0.05	1.0
−0.43	0.25	−0.10	0.05	1.0
−0.25	0.43	−0.11	0.05	1.0

Table B.1 *The centres (c_x, c_y, c_z), radii r, and densities ρ of the spheres of the sphere clock phantom.*

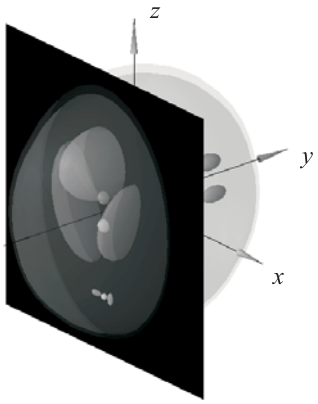


Figure B.2 *The three-dimensional Shepp-Logan phantom. The plane $y = -0.25$ is also drawn.*

B.2 The Three-Dimensional Shepp-Logan Phantom

The three-dimensional Shepp-Logan phantom is defined by twelve of solid ellipsoids shown in Figure B.2. The ellipsoids are constructed by transforming the points within the unit sphere $\{(x, y, z)^T | x^2 + y^2 + z^2 \leq 1\}$ by scaling, rotation around the y-axis, and translation as:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{pmatrix} \begin{pmatrix} r_x & 0 & 0 \\ 0 & r_y & 0 \\ 0 & 0 & r_z \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} c_x \\ c_y \\ c_z \end{pmatrix} \tag{B.1}$$

Table B.2 contains the parameter values. The densities ρ are additive.

B.3 The Voxelised Head Phantom

The voxelised head phantom is defined using a medical head data set of a severley deformed human child. The data set has been made available by Philips Research Laboratory, Hamburg. It consists of $320 \times 320 \times 230$ voxels of size $\Delta x = \Delta y = \frac{221}{320}$ mm, and $\Delta z = 1.0$ mm. In order to work with isotropic data, we have squeezed the phantom in the z-direction in the experiments of this thesis so that $\Delta x = \Delta y = \Delta z = \frac{221}{320}$ mm.

The voxel values are given in offset Hounsfield units such that air has the value 0 and water has the value 1000. We always place the greyscale window at the soft tissue interval $[980, 1080]$.

r_x	r_y	r_z	c_x	c_y	c_z	α	ρ
0.69	0.9	0.92	0.0	0.0	0.0	$0.0°$	2.0
0.6624	0.88	0.874	0.0	0.0	-0.0184	$0.0°$	-0.98
0.41	0.21	0.16	-0.22	-0.25	0.0	$72.0°$	-0.02
0.31	0.22	0.11	0.22	-0.25	0.0	$-72.0°$	-0.02
0.21	0.35	0.25	0.0	-0.25	0.35	$0.0°$	0.01
0.046	0.046	0.046	0.0	-0.25	0.1	$0.0°$	0.01
0.046	0.02	0.023	-0.08	-0.25	-0.605	$0.0°$	0.01
0.046	0.02	0.023	0.06	-0.25	-0.605	$90.0°$	0.01
0.056	0.1	0.04	0.06	0.625	-0.105	$90.0°$	0.02
0.056	0.1	0.056	0.0	0.625	0.1	$0.0°$	-0.02
0.046	0.046	0.046	0.0	-0.25	-0.1	$0.0°$	0.01
0.023	0.023	0.023	0.0	-0.25	-0.605	$0.0°$	0.01

Table B.2 *The centres (c_x, c_y, c_z), radii (r_x, r_y, r_z), rotations α, and densities ρ of the ellipsoids of the three-dimensional Shepp-Logan phantom.*

C

Notation

Symbols

	Continuous	*Discrete*	*Definition*
Fan-beam projection angle	β	j	*p. 17*
Fan-angle	γ	o	*p. 17*
Parallel beam projection angle	θ	n	*p. 12*
Parallel beam detector position	t	k	*p. 12*
Position along a parallel ray	v		*p. 16*
Cone-angle	κ		*p. 33*
Vertical cylindrical detector position	q	m	*p. 33*
Vertical PI-detector position	s	c	*p. 45*
Slanted filtering lines	l		*p. 49*
Horizontal planar detector position	a		*p. 32*
Vertical planar detector position	b		*p. 32*
Source-pixel distance	L		*p. 18*
Projected source-pixel distance	U		*p. 19*
Radius of source trajectory	R		*p. 17*
Radius of reconstructable region	R_{FOV}		*p. 15*

Helical pitch	P		*p. 78*
Helical turn index		λ	*p. 78*
PI-detector extension		$\mathfrak{n}$	*p. 121*
Image space coordinates	x, y, z	i_1, i_2, i_3	*p. 15*
Nutating plane tilt angle	α		*p. 90*
Projection direction vector	$\boldsymbol{\phi}$		*p. 104*

Functions

Equiangular fan-beam projection values	$p(\beta, \gamma)$	*p. 17*
Equidistant fan-beam projection values	$p^F(\beta, a)$	*p. 19*
Parallel beam projection values	$p^P(\theta, t)$	*p. 12*
Projection values for the slice $z = z_{\text{img}}$	$p_{z_{\text{img}}}(\beta, \gamma)$	*p. 84*
Cone-beam projection on cylindrical detector	$p(\beta, \gamma, q)$	*p. 32*
Cone-beam projection on planar detector	$p^F(\beta, a, b)$	*p. 32*
Oblique parallel beam from planar detector	$p^{CP}(\theta, t, q)$	*p. 43*
Oblique parallel beam from cylindrical detector	$p^{FP}(\theta, t, b)$	*p. 41*
Oblique parallel beam projection values	$p^P(\theta, t, s)$	*p. 102*
Complementary projection values	$p^C(\beta, \gamma, q)$	*p. 88*
Generalized projection data for the slice $z = z_{\text{img}}$	$p^G_{z_{\text{img}}}(\theta, t, v)$	*p. 97*
Fan-beam ramp-filter	$g(\gamma)$	*p. 18*
Parallel beam ramp-filter	$g^P(t)$	*p. 15*
Link values from unfiltered data	$I[n_1, k_1; n_2, k_2]$	*p. 115*
Link values from ramp-filtered data	$\tilde{I}[n_1, k_1; n_2, k_2]$	*p. 25*
z-position of source	$z_S(\beta)$	*p. 78*
The base 2 logarithm	$\lg n = {}^2\log n$	
Dirac's delta function	$\delta(\cdot)$	
Dirac's two-dimensional delta function	${}^2\delta(\cdot)$	
Dirac's three-dimensional delta function	${}^3\delta(\cdot)$	

Special notation

$\tilde{}$	The tilde indicates pre-weighted and filtered projection data.	e.g. $\tilde{p}(\beta, \gamma)$	*p. 15*
[ ]	The brackets indicate a sampled signal.	e.g. $p[j, l]$	*p. 14*

Acronyms

Advanced single-slice rebinning	ASSR	*p. 92*
FDK reconstruction from data within B	B-FDK	*p. 82*
Backprojection	BP	*p. 13*
Cone-beam short-scan rebinning	CB-SSRB	*p. 88*
Complementary filtered backprojection	CFBP	*p. 94*
FDK reconstruction on cylindical detector	C-FDK	*p. 37*
Computed tomography	CT	*p. 1*
Filtered backprojection	FBP	*p. 13*
Feldkamp, David and Kress reconstruction	FDK	*p. 35*
Field of view	FOV	*p. 15*
Hybrid tent FDK reconstruction	HT-FDK	*p. 46*
Incinsitent helical cone-beam algorithm	IHCB	*p. 94*
Multi-row Fourier reconstruction	MFR	*p. 95*
Multi-slice cone-beam filtered backprojection	MS-CB-FBP	*p. 81*
Modulation transfer function	MTF	*p. 72*
Pelles ingeneous method	PI method	*p. 99*
Region of interest	ROI	*p. 81*
Sequential FDK reconstruction	S-FDK	*p. 46*
Single-slice cone-beam filtered backprojection	SS-CB-FBP	*p. 82*
Short-scan FDK reconstruction	SS-FDK	*p. 95*
Tent FDK reconstruction	T-FDK	*p. 45*

Author Index

Bibliography

Basu, S. and Y. Bressler (2000). $O(N^2 \log_2 N)$ Filtered Backprojection Reconstruction Algorithm for Tomography. *IEEE Transactions on Image Processing 9*(10), 1760–1773.

Berland, L. L. and J. K. Smith (1998). Multidetector-Array CT: Once Again, Technology Creates New Opportunities. *Radiology 209*(2), 327–329. Editorial.

Besson, G. (1996). CT Fan-Beam Parameterizations Leading to Shift-Invariant Filtering. *Inverse Problems 12*(6), 815–833.

Besson, G. (1998). CT Reconstruction from Fan-Parallel Data. In *IEEE Medical Imaging, Nov 8–14, 1998, Toronto, Canada*.

Blom, G. (1989). *Sannolikhetsteori och statistikteori med tillämpningar* (4 ed.). Studentlitteratur, Lund, Sweden.

Brady, M. L. (1998). A Fast Discrete Approximation Algorithm for the Radon Transform. *SIAM Journal of Computing 27*(1), 107–119.

Brandt, A., J. Mann, M. Brodski, and M. Galun (1999). A fast and accuarte multilevel inversion of the Radon transform. *SIAM Journal of Applied Mathematics 60*(2), 437–462.

Bruder, H., M. Kachelriess, S. Schaller, and T. Mertelmeier (2000). Performance of approximate cone-beam reconstruction in multi-slice computed tomography. In *SPIE Medical Imaging, Feb. 12–17, 2000, San Diego, California, USA*, pp. 541–554.

Buck, J., M. Maisl, and H. Reiter (1996). The Cylinder Algorithm – An Efficient Reconstruction Algorithm for the 3D X-Ray Computed Tomography (3D-CT) in NDT. In *ASNT's Industriual Computed Tomography Topical Conference, May 13–15, Huntsville, Alabama, USA*, pp. 88–93.

Cabral, B., N. Cam, and J. Foran (1994). Accelerated Volume Rendering and Tomographic Reconstruction Using Texture Mapping Hardware. In *Proc. 1994 Symposium on Volume Visualization, Oct 17–18, Washington, D.C., USA*, pp. 91–98, 131.

Clasén, B. (1999). Evaluation of texture mapping hardware for acceleration of 3D reconstruction algorithms in computed tomography. Master's thesis, Linköping University. LiTH-ISY-EX-3036.

Cohen, D. (1994). *Graphic Gems IV*, Chapter V.3. Voxel traversal along a 3D line, pp. 366–369. Academic Press. Editor P. S. Heckbert.

Danielsson, P.-E. (1997). Iterative Techiques for Projection and Back-Projection. Technical Report LiTH-ISY-R-1960, ISSN 1400-3902, Department of Electrical Engineering, Linköping University.

Danielsson, P.-E., P. Edholm, J. Eriksson, and M. Magnusson-Seger (1997). Towards Exact 3D-reconstruction for Helical Cone-Beam Scanning of Long Objects: A New Arrangement and a New Completeness Condition. In *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 25-28, 1997, Nemacolin, Pennsylvania, USA*, pp. 141–144.

Danielsson, P.-E., P. Edholm, J. Eriksson, M. Magnusson-Seger, and H. Turbell (1998a). Helical Cone-Beam Scanning and Reconstruction of Long Objects Using 180 Degrees Exposure. Patent Application PCT/SE 98/00029, Filed 13 January 1998.

Danielsson, P.-E., P. Edholm, J. Eriksson, M. Magnusson-Seger, and H. Turbell (1998b). Helical Cone-Beam Scanning and Reconstruction of Long Objects Using 180 Degrees Exposure. In *Problems and Experiments in Cone-Beam Tomography*. Jan Eriksson. Licentiate Thesis. Appendix A. Department of Electrical Engineering, Linkoping University. ISBN 91-7219-255-0, ISSN 0280-7971.

Danielsson, P.-E. and M. Ingerhed (1997). Backprojection in $O(N^2 \log N)$ time. In *IEEE Medical Imaging, Nov 13–15, 1997, Alberquerque, New Mexico, USA*.

Defrise, M. and R. Clack (1994). A Cone-Beam Reconstruction Algorithm Using Shift-Variant Filtering and Cone-Beam Backprojection. *IEEE Transactions on Medical Imaging 13*(1), 186–195.

Defrise, M. and F. Noo (1997). Reconstruction of truncated cone-beam projections using the frequency-distance relation. In *International Meeting on Fully*

Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 25-28, 1997, Nemacolin, Pennsylvania, USA, pp. 32–35.

Defrise, M., F. Noo, and H. Kudo (1999). Quasi-Exact Region-of-Interest Reconstruction from Helical Cone-Beam Data. In *IEEE Medical Imaging, 1999, Seattle, Washington, USA*, Volume 2, pp. 1101–1103.

Defrise, M., F. Noo, and H. Kudo (2000). A solution to the long object problem in helical CB tomography. *Physics in Medicine and Biology 45*, 623–643.

Eberhard, J. and K. C. Tam (1995). Helical and Circle Scan Region of Interest Computerized Tomography. US Patent 5,463,666, Oct. 31.

Edholm, P. and B. Jacobsson (1975). PD5-Tomogram Construction by Photographic Techniques. In *Image Processing for 2D and 3D Reconstruction from Projections, Stanford, CA, USA. Collection of post-deadline papers.*

Edholm, P., R. M. Lewitt, and B. Lindholm (1986). Novel Properties of the Fourier Decomposition of the Sinogram. In *Int. Workshop on Physics and Engineeing of Computerized Multidimensional Imaging and Processing, Proc. of the SPIE*, Volume 671, pp. 8–18.

Eriksson, J. (1998). *Problems and Experiments in Cone-Beam Tomography*. Licentiate Thesis. Department of Electrical Engineering, Linköping University. ISBN 91-7219-255-0, ISSN 0280-7971.

Feldkamp, L. A., L. Davis, and J. Kress (1984). Practical Cone-beam Algorithm. *Journal of the Optical Society of America 1*, 612–619.

Foley, J., A. van Dam, S. Feiner, and J. Hughes (1990). *Computer Graphics: principles and practice* (2 ed.). Addison Wesley.

Fuchs, T. (2000). Simulation of noise. As available electronically in December 2000 at `http://www.imp.uni-erlangen.de/forbild/`.

Goertzen, A. L., F. J. Beekman, and S. R. Cherry (2000). Effect of voxel size in CT simulations. In *IEEE Medical Imaging, Oct 16–20, 2000, Lyon, France*.

Grangeat, P. (1987). *Analyse d'un Système d'Imagerie 3D par Reconstrucion à partir de Radiographies X en Géométrie Conique*. Ph. D. thesis, Ecole Nationale Supérieure des Télécommunications.

Grangeat, P. (1991). Mathematical framework of cone beam 3D reconstruction via the first derivative of the Radon transform. In G. T. Herman, A. K. Louis, and F. Natterer (Eds.), *Mathematical Methods in Tomography*, Number 1497 in Lecture Notes in Mathematics, pp. 66–97. Springer Verlag.

Grangeat, P., P. Le Mason, P. Melennec, and P. Sire (1991). Evaluation of the 3D Radon Transform Algorithm for Cone Beam Reconstruction. In *SPIE Medical Imaging V: Image Processing, Technical Conference 1445, Feb. 23 – March 1, 1991, San Jose, California, USA*.

Grass, M., T. Köhler, and R. Proksa (2000a). 3D cone-beam CT reconstruction for circular trajectories. *Physics in Medicine and Biology 45*(2), 329–347.

Grass, M., T. Köhler, and R. Proksa (2000b). Weighted hybrid cone beam reconstruction for circular trajectories. In *IEEE Medical Imaging, Oct 16–20, 2000, Lyon, France*.

Herman, G. T. (1980). *Image reconstruction from projections – the fundamentals of computerized tomography*. Academic Press, New York. ISBN 0-12-342050-4.

Herman, G. T. and A. Naparstek (1977). Fast image reconstruction based on a radon inversion formula appropiate for rapidly collected data. *SIAM Journal of Applied Mathematics 33*, 511–533.

Heuscher, D. (1999). Helical cone beam scans using oblique 2D surface reconstruction. In *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 23–26, 1999, Egmond aan Zee, The Netherlands*, pp. 204–207.

Hu, H. (1999). Multi-Slice Helical CT: Scan and Reconstruction. *Medical Physics 26*(1), 5–18.

Ingerhed, M. (1999). *Fast Backprojection in Computed Tomography: Implementation and Evaluation*. Licentiate Thesis No 760. Department of Electrical Engineering, Linkoping University. ISBN 91-7219-462-6, ISSN 0280-7971.

Ingerhed, M. and P.-E. Danielsson (1998). Implementation of Backprojection in $\mathcal{O}(N^2 \log N)$ Time. In *SSAB Symposium on Image Analysis, March 16–17, 1998, Uppsala*, pp. 25–28. Swedish Society for Automated Image Analysis. ISSN 1100-6641.

Jacobson, C. (1996). *Fourier Methods in 3D-Reconstruction from Cone-Beam Data*. Ph. D. thesis, Department of Electrical Engineering, Linköping University.

Joseph, P. M. (1982). An Improved Algorithm for Reprojecting Rays Through Pixel Images. *IEEE Transactions on Medical Imaging MI-1*, 192–196.

Kachelriess, M. and W. Kalender (2000). Advanced Single-Slice Rebinning (ASSR) and Misalignment Correction for Large Cone-Beam Spiral CT. In *RSNA'00, Nov. 26 – Dec 1, Chicago, USA, supplement to Radiology, volume 217 (P)*, pp. 405.

Kachelriess, M., S. Schaller, and W. Kalender (2000a). Advanced Single-Slice Rebinning in Cone-Beam Spiral CT. *Medical Physics 27*(4), 754–772.

Kachelriess, M., S. Schaller, and W. Kalender (2000b). Advanced Single-Slice Rebinning in Cone-Beam Spiral CT: Theoretical considerations and medical applications. In *SPIE Medical Imaging, Feb. 12–17, 2000, San Diego, California, USA*, pp. 494–505.

Kak, A. C. and M. Slaney (1987). *Principles of Computerized Tomographic Imaging*. IEEE Press. ISBN 0-87942-198-3.

Kalender, W. (1995). Principles and performance of spiral ct. In L. W. Goldman and J. B. Fowlkes (Eds.), *Medical CT and Ultrasound: Current Technology and Applications*, pp. 379–410. Advanced Medical Publishing. ISBN 1-883526-03-5.

Kalender, W. (2000). *Computed Tomography: Fundamentals, System Technology, Image Quality, Applications*. Publics MCD Verlag. ISBN 3-89578-081-2.

Köhler, T., R. Proksa, and M. Grass (2000). A Fast and Efficient Method for Sequential Cone-Beam Tomography. In *IEEE Medical Imaging, Oct 16–20, 2000, Lyon, France*.

Köhler, T., H. Turbell, and M. Grass (2000). Efficient Forward Projection Through Discrete Data Sets Using Tri-Linear Interpolation. In *IEEE Medical Imaging, Oct 16–20, 2000, Lyon, France*.

Kopecky, K. K., K. A. Buckwalter, and R. Sokiranski (1999). Multi-slice CT spirals past single-slice CT in diagnostic efficacy.

Kreuder, F., M. Grass, V. Rasche, H. Braunisch, and O. Dössel (1999). Fast calculation of the X-ray transform from the radial derivative of the 3D Radon transform using gridding and fast backprojection techniques. In *European Medical and Biological Engineering Conference, Vienna, Nov. 4–7*.

Kudo, H., F. Noo, and M. Defrise (1998). Cone-Beam Filtered-Backprojection Algorithm for Truncated Helical Data. *Physics in Medicine and Biology 43*, 2885–2909.

Kudo, H., S. Park, F. Noo, and M. Defrise (1999). Performance of quasi-exact cone-beam filtered backprojection algorithm for axially truncated data. *IEEE Transactions on Nuclear Science 46*(3), 608–617.

Kudo, H. and T. Saito (1991). Helical-Scan Computed Tomography Using Cone-Beam Projections. In Baldwin (Ed.), *IEEE Medical Imaging, 1991*, pp. 1958–1962.

Lakshminarayanan, A. V. (1975). Reconstruction from divergent ray data. Technical Report 92, Deptartment of Computer Science, State University of New York at Buffalo.

Lanzavecchia, S. and P. L. Bellon (1996). Electron tomography in conical tilt geometry. The accuracy of a direct Fourier method (DFM) and the suppression of non-tomographic noise. *Ultramiscoscopy 63*, 247–261.

Larson, G. L., C. C. Ruth, and C. R. Crawford (1998). Nutating slice CT image reconstruction. Patent Application WO 98/44847.

Lewitt, R. M. (1990). Multidimensional digital image representations using generalized Kaiser-Bessel window functions. *Journal of the Optical Society of America 7*(10), 1834–1846.

Lewitt, R. M. (1992). Alternatives to voxels for image representation in iterative reconstruction algorithms. *Physics in Medicine and Biology 37*, 705–716.

Magnusson-Seger, M. (1993). *Linogram and Other Direct Fourier Methods for Tomographic Reconstruction*. Ph. D. thesis, Department of Electrical Engineering, Linköping University.

Marr, R. B., C. Chen, and P. C. Lauterbur (1981). On two approaches to 3D reconstruction in NMR zeugmatography. In G. T. Herman and F. Natterer (Eds.), *Mathematical Aspects of Computerized Tomography, Oberwolfach (FRG)*, Lecturer Notes in Mathematics, pp. 225–240. Springer Verlag.

Müller-Merbach, J. (1996). Simulation of X-ray Projections for Experimental 3D Tomography. Technical Report LiTH-ISY-R-1866, ISSN 1400-3902, Image Processing Lab, Department of Electrical Engineering, Linköping University.

Mueller, K. (1998). *Fast and accurate three-dimensional reconstruction from cone-beam projection data using algebraic methods*. Ph. D. thesis, The Ohio State University, USA.

Mueller, K., R. Yagel, and J. Wheller (1999). Fast Implementation of Algebraic Methods for Three-Dimensional Reconstruction from Cone-Beam Data. *IEEE Transactions on Medical Imaging 18*(6), 538–548.

Natterer, F. (1986). *The Mathematics of Computerized Tomography*. John Wiley and Sons. ISBN 0-471-90959-9.

Nilsson, S. (1996). Fast Backprojection. Technical Report LiTH-ISY-R-1865, ISSN 1400-3902, Department of Electrical Engineering, Linköping University.

Nilsson, S. (1997). *Application of Fast Backprojection Techniques for Some Inverse Problems of Integral Geometry*. Ph. D. thesis, Department of Mathematics, Linköping University. No. 499.

Noo, F., H. Kudo, and M. Defrise (1998). Approximate Short-Scan Filtered-Backprojection for Helical CB Reconstruction. In *IEEE Medical Imaging, Nov 8–14, 1998, Toronto, Canada*.

Oppenheim, A. V. and R. Schafer (1975). *Digital Signal Processing*. Prentice-Hall.

Oppenheim, A. V. and A. S. Willsky (1997). *Signals and Systems*. Prentice Hall.

Orlov, S. S. (1975). Theory of three dimensional reconstruction – Conditions for a complete set of projections. *Sov. Phys. Crystallogr. 20*(3), 312–314.

O'Sullivan, J. (1985). A Fast Sinc Function Gridding Algorithm for Fourier Inversion in Computer Tomography. *IEEE Transactions on Medical Imaging 4*(4), 200–207.

Pan, T. and Y. Shen (2000). New Multi-sector Reconstruction for Cardiac CT. In *IEEE Medical Imaging, Oct 16–20, 2000, Lyon, France*.

Parker, D. L. (1982). Optimal Short Scan Convolution Reconstrucion for Fanbeam CT. *Medical Physics 9*(2), 254–257.

Proksa, R., T. Köhler, M. Grass, and J. Timmer (2000). The n-PI-Method for Helical Cone-Beam CT. *IEEE Transactions on Medical Imaging 19*(9), 848–863.

Råde, L. and B. Westergren (1990). *BETA: Mathematics Handbook* (2 ed.). Studentlitteratur, Lund, Sweden.

Reimann, D. A. and M. J. Flynn (1992). Automated Distortion Correction of X-ray Image Intensifier Images. In *IEEE Medical Imaging, 1992*, Volume 2, pp. 1339–41.

Rizo, P., P. Grangeat, P. Sire, P. Le Mason, and P. Melennec (1991). Comparison of two three-dimensional X-ray cone-beam-reconstruction algorithms with circular source trajectories. *Journal of the Optical Society of America 8*(10), 1639–1648.

Rosenfeld, D. (1998). An optimal and efficient new gridding algorithm using singular value decomposition. *Magnetic Resononance Medicine 40*, 14–23.

Schaller, S. (1998). *Practical Image Reconstruction for Cone Beam Computed Tomography*. Ph. D. thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany.

Schaller, S., T. Flohr, and P. Steffen (1996). A New Approximate Algorithm for Image Reconstruction in Cone-Beam Spiral CT at Small Cone-Angles. In *IEEE Medical Imaging, Anaheim, California, USA*.

Schaller, S., T. Flohr, and P. Steffen (1997). New, Efficient Fourier-reconstruction Method for Approximate Image Reconstruction in Spiral Cone-Beam CT at Small Cone Angles. In *SPIE Medical Imaging, Newport Beach, California, USA*.

Schaller, S., T. Flohr, H. Wolf, and W. Kalender (1998). Evaluation of a Spiral Reconstruction Algorithm for Multirow-CT. In *RSNA'98, Nov. 29 – Dec 4, Chicago, USA, supplement to Radiology, volume 209 (P)*.

Schaller, S., M. Karolczak, K. Engelke, K. Wiesent, and W. Kalender (1998). Implementation of a Fast Cone-Beam Backprojection Algorithm for Microcomputed Tomography (HCT) Using Homogeneous Coordinates. In *RSNA'98, Nov. 29 – Dec 4, Chicago, USA, supplement to Radiology, volume 209 (P)*.

Schaller, S., F. Noo, F. Sauer, K. C. Tam, G. Lauritsch, and T. Flohr (1999). Exact Radon rebinning algorithms using local region-of-interest for helical cone-beam CT. In *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 23–26, 1999, Egmond aan Zee, The Netherlands*, pp. 11–14.

Schaller, S., F. Noo, F. Sauer, K. C. Tam, G. Lauritsch, and T. Flohr (2000). Exact Radon Rebinning Algorithm for the Long Object Problem in Helical Cone-Beam CT. *IEEE Transactions on Medical Imaging 19*(5), 361–375.

Schomberg, H. and J. Timmer (1995). The Gridding Method for Image Reconstruction by Fourier Transformation. *IEEE Transactions on Medical Imaging 14* (3), 596–607.

Sedarat, H. and D. G. Nishimura (2000). On the optimality of the gridding reconstruction algorithm. *IEEE Transactions on Medical Imaging 19*(4), 406–317.

Siddon, R. L. (1985). Fast calculation of the exact radiological path length for a three-dimensional CT array. *Medical Physics 12*, 252–255.

Silver, M. D. (1997). Practical Limits to High Helical Pitch, Cone-Beam Computed Tomography. In *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 25-28, 1997, Nemacolin, Pennsylvania, USA*, pp. 44–47.

Silver, M. D. (1998). High-Helical-Pitch, Cone-Beam Computed Tomography. *Physics in Medicine and Biology 43*(4), 847–855.

Sourbelle, K., G. Lauritsch, K. C. Tam, F. Noo, and W. Kalender (2000). Performance evaluation of local ROI algorithms for exact ROI reconstruction in spriral cone-beam computed tomography. In *IEEE Medical Imaging, Oct 16–20, 2000, Lyon, France*.

Taguchi, K. and H. Aradate (1998). Algorithm for image reconstruction in multi-slice helical CT. *Medical Physics 25*(4), 550–561. ISSN 0094-2405.

Tam, K. C. (1995). Three-Dimensional Computerized Tomography Scanning Method and System for Large Objects with Smaller Area Detectors. US Patent 5,390,112, Feb. 14.

Tam, K. C., G. Lauritsch, K. Sourbelle, F. Sauer, and B. Ladendorf (2000). Exact (Spiral + Circles) Scan Region-of-Interest Cone Beam Reconstruction via Backprojection. *IEEE Transactions on Medical Imaging 19*(5), 376–383.

Tam, K. C., S. Samarasekera, and F. Sauer (1997). Exact Cone-Beam CT with a Spiral Scan. In *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 25-28, 1997, Nemacolin, Pennsylvania, USA*.

Turbell, H. (1997). New Functionality in `take` Version 2.0. Technical report, Image Processing Lab, Department of Electrical Engineering, Linköping University.

Turbell, H. (1999). *Three-Dimensional Image Reconstruction in Circular and Helical Computed Tomography*. Licentiate Thesis. Department of Electrical Engineering, Linköping University. ISBN 91-7219-463-4, ISSN 0345-7524.

Turbell, H. and P.-E. Danielsson (1998). The PI Method – Non-Redundant Data Capture and Efficient Reconstruction for Helical Cone-Beam CT. In *IEEE Medical Imaging, Nov 8–14, 1998, Toronto, Canada*.

Turbell, H. and P.-E. Danielsson (1999a). An improved PI-method for reconstruction from helical cone-beam projections. In *IEEE Medical Imaging, 1999, Seattle, Washington, USA*.

Turbell, H. and P.-E. Danielsson (1999b). Fast Feldkamp Reconstruction. In *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine, June 23–26, 1999, Egmond aan Zee, The Netherlands*, pp. 311–314.

Turbell, H. and P.-E. Danielsson (2000). Helical Cone-Beam Tomography. *International Journal of Imaging Systems and Technology 11*(1), 91–100.

Tuy, H. (1983). An inversion formula for cone-beam reconstruction. *SIAM Journal of Applied Mathematics 43*, 546–552.

Wang, G., T.-H. Lin, P.-C. Cheng, and D. M. Shinozaki (1993). A General Cone-Beam Reconstruction Algorithm. In *IEEE Transactions on Medical Imaging*, Volume 12.

Wang, G., Y. Liu, T.-H. Lin, and P.-C. Cheng (1994). Half-Scan Cone-Beam X-Ray Microtomography Formula. *Journal of Scanning Microscopies 16*, 216–220.

Wang, G. and M. Vannier (1999). The effect of pitch in multislice spiral/helical CT. *Medical Physics 26*(12), 2648–2653.

Webb, S. (1990). *From the watching of shadows*. Adam Hilger, New York.

Yan, X.-H. and R. M. Leahy (1992). Cone Beam Tomography with Circular, Elliptical and Spiral Orbits. *Physics in Medicine and Biology 37*(3), 493–506.

Zeng, G. L. and G. T. Gullberg (1990). Short-Scan Cone Beam Algorithm for Circular and Non-Circular Detector Orbits. In *SPIE Medical Imaging IV: Image Processing*, Volume 1233, pp. 453–463.

Zeng, G. L. and G. T. Gullberg (1991). Short-Scan Fan Beam Algorithm for Non-Circular Detector Orbits. In *SPIE Medical Imaging V: Image Processing, Technical Conference 1445, Feb. 23 – March 1, 1991, San Jose, California, USA*, Volume 1445, pp. 332–340.