...en Faculty of Mathematics and Physical Sciences

Department of Computing Science

Parallelisation of Tomographic Reconstruction Methods

Michel A. Westenberg

Advisor:
dr. J.B.T.M. Roerdink

November, 1996

RuG

Parallelisation of Tomographic Reconstruction Methods

Michel A. Westenberg

Master's Thesis
Supervisor: Jos B.T.M. Roerdink
Department of Computing Science
Rijksuniversiteit Groningen
November 1996

Samenvatting

Er is in de geneeskunde een nog steeds toenemende belangstelling voor onderzoekstechnieken die anatomische structuren kunnen afbeelden zonder daarbij het lichaam te beschadigen. Onder dit soort methoden vallen Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET) en Computerised Tomography (CT). Al deze methoden zijn gebaseerd op hetzelfde principe: onder een aantal hoeken wordt een stel lijnintegralen in een vlak gemeten en dit levert een verzameling profielen op. Deze verzameling profielen wordt de Radon-transformatie van het object genoemd. De moeilijkheid is nu een tweedimensionaal beeld te reconstrueren uit zijn Radon-transformatie.

Het is mogelijk om reconstructie-algoritmen af te leiden uit de zogenaamde Fourier slice stelling. Deze stelling legt een relatie tussen de ééndimensionale Fourier-transformatie van een profiel en de tweedimensionale Fourier-transformatie van het te reconstrueren object. Er worden drie verschillende op deze stelling gebaseerde reconstructiemethoden bekeken in deze scriptie: de eerste is het filtered backprojection algoritme, de tweede is directe Fourier-reconstructie en de laatste methode is een multiresolutie-algoritme, dat gebaseerd is op de wavelet-transformatie en filtered backprojection. Dit multiresolutie-algoritme heeft de tweedimensionale wavelet-transformatie van het object als resultaat.

Deze scriptie behandelt de parallellisatie van bovengenoemde reconstructiemethoden en de wavelet-transformatie op een Connection Machine CM-5. Het is bekend dat filtered backprojection veel betere reconstructies geeft dan de directe Fourier-methode, maar de laatste is sneller te berekenen. Het filtered backprojection algoritme blijkt het moeilijkst te parallelliseren. Omdat de CM-5 een gedistribueerd geheugen heeft, kunnen de vergelijkingen voor reconstructie niet zonder meer worden vertaald in een programma. Er moeten verscheidene optimalisaties worden gedaan, die afhankelijk zijn van de architectuur van de CM-5 om tot een bevredigend resultaat te komen. Dit geldt ook voor het multiresolutie-algoritme, omdat dit het backprojection deel van het filtered backprojection algoritme gebruikt. Directe Fourier-reconstructie blijkt daarentegen zeer eenvoudig te parallelliseren op de CM-5. De implementatie van de wavelet-transformatie heeft een snel algoritme opgeleverd.

Abstract

In medicine, there is a still growing interest in non-invasive examination techniques which can depict anatomical structures. Amongst these methods are Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET) and Computerised Tomography (CT). These are all based on the same principle: under a number of angles, a set of line integrals in a plane is measured resulting in a set of profiles. This set of profiles is called the Radon transform of the object. The problem now is to reconstruct a two-dimensional image from its Radon transform.

It is possible to derive reconstruction algorithms from the so-called Fourier slice theorem. This theorem links the one-dimensional Fourier transform of a profile to the two-dimensional Fourier transform of the object which is to be reconstructed. Three different reconstruction methods, based on this theorem, are considered in this thesis: the first one is the filtered backprojection algorithm, the second one is direct Fourier reconstruction, and the last method is a multiresolution reconstruction algorithm based on the wavelet transform and filtered backprojection. The multiresolution algorithm results in the two-dimensional wavelet transform of the object.

This thesis deals with the parallelisation of the above mentioned reconstruction methods and of the wavelet transform on a Connection Machine CM-5. It is known that filtered backprojection gives far better reconstructions than the direct Fourier method, whereas the latter is the fastest to compute. Of the algorithms considered, the filtered backprojection algorithm is the most difficult to parallelise. Because the CM-5 has a distributed memory, the reconstruction equations cannot be translated into a program in a straightforward manner. Several optimisations depending on the architecture of the CM-5 have to be made in order to get a satisfactory performance. This also holds for the multiresolution algorithm, as it uses the backprojection part of the filtered backprojection algorithm. Direct Fourier reconstruction on the contrary is very easy parallelised on the CM-5. The implementation of the wavelet transform has led to an efficient algorithm.

Preface

This thesis is the result of six months of hard labour in order to obtain the Master's Degree in Computing Science. The work was carried out at the Department of Computing Science at the University of Groningen.

At this point, I wish to thank my supervisor, Jos Roerdink, for his support and critical comments on my work. This not only made me review my own work more critically, but also improved my apprehension of the topic.

I also want to thank some of my fellow students who—either wittingly or unwittingly—have been of support to me throughout the past five years.

Michel Westenberg,
Groningen, November 1996.

Contents

1

Introduction

Tomographic reconstruction is the art of image reconstruction from projections, i.e. line integrals, and has arisen independently in a number of fields. The problem is to determine the internal structure of an object without having to damage it. Generally, various probes such as X-rays, gamma rays, visible light, electrons and other kinds of radiation are used to produce a so-called *profile*. A set of such profiles is called the Radon transform of the object. The mathematical framework of such reconstruction problems has been developed by Johann Radon in 1917. There have been numerous 're-discoveries' of his results throughout the literature. There is also evidence that the physicist Lorentz knew the inversion formula for the three-dimensional case around the turn of the century (Deans 1983). According to Uhlenbeck (1925) he once gave a proof of the formula in one of his lectures. The number of different fields in which reconstruction problems arise is quite large: astronomy, molecular biology, geophysics, optics and medicine, to name a few. The field which has contributed most to the knowledge, however, has been computerised tomography in medicine. This is reflected in the awarding of the Nobel Prize in physiology or medicine to G. N. Hounsfield and A. M. Cormack in 1979.

In diagnostic medicine, a number of different tomographic methods can be distinguished. The oldest one, computerised tomography (CT) uses one or more X-ray sources and an array of detectors to measure the attenuation of X-rays along lines, resulting in a profile for some angle ϕ. These profiles are measured for incremental values of ϕ and constitute a sampling of the Radon transform. Chapter 2 gives a mathematical description of this process. By using an appropriate reconstruction algorithm, a cross-sectional slice of some part of the human body is obtained. Tissues which have different attenuation coefficients can be distinguished from each other and this property can be used to find the presence of, for instance, tumours. An example of a CT image of the head is given in Fig. 1.1.

Unlike CT, where radiation is generated outside the patient, positron emission tomography (PET) uses radioactive isotopes inside the human body. The goal is to reconstruct the distribution of the isotopes by detecting the emitted photons in a ring which is placed around the patient. This method is, for instance, used in functional brain imaging, where the radioactive isotopes are injected in the bloodstream. The patient is given a certain task and as a result more blood will flow to the parts in the brain that are used for that particular task. By studying the bloodflow the physician can determine whether or not there are anomalies.

A third method which is of interest is magnetic resonance imaging (MRI). This technique is based on the observation that certain nuclei have an intrinsic spin. The patient

Figure 1.1: An example of a CT image of a cross section of the head. Courtesy of Imaging Center Utrecht.

is placed in a strong magnetic field and the nuclei are excited by sending an RF pulse. In the decay phase, the nuclei emit radiation which can be detected. Tissues consisting for a great deal of water give a strong response, whereas bones give a low response.

All the above mentioned data acquisition techniques can be related in one or the other way to the Radon transform.

This report focuses on parallel ray computed tomography and reconstruction methods for this technique. Chapter 2 gives the mathematical background of the Radon transform and its inversion. The next two chapters describe two different reconstruction algorithms, viz. filtered backprojection and direct Fourier reconstruction. The concept of the wavelet transform is introduced in Chapter 5 and this is used to construct a multiresolution reconstruction algorithm which is described in Chapter 6. For all algorithms it is described how they can be parallelised on a Connection Machine CM-5.

2

The Radon transform and its inverse

In this chapter, a mathematical description of the tomographic reconstruction problem will be given. First, the Radon transform will be explained which gives a mathematical notion of the concept of projections. Then the Fourier slice theorem will give a relation between the one-dimensional Fourier transform of the projections and the two-dimensional Fourier transform of the object function. Lastly, a way of analytically computing the Radon transform for simple functions will be given resulting in a way to generate projections of an object function.

2.1 The Radon transform in two dimensions

Let $f(x_1, x_2)$ be a continuous function that is decreasing sufficiently fast at infinity. Let $\theta = (\theta_1, \theta_2)$ be a unit vector in $\mathbb{R}^2$, t a real number, and Γ the line defined by

$$\Gamma = \{x \in \mathbb{R}^2 : \langle x, \theta \rangle = t\}, \tag{2.1}$$

where $\langle\ ,\ \rangle$ denotes the Euclidean inner product (see Fig. 2.1). The integral of f over the line Γ,

$$\mathcal{R}f(\theta, t) := \int_\Gamma f(x)\,\mathrm{d}x, \tag{2.2}$$

is called the *Radon transform* of the function f. The integral $\mathcal{R}f(\theta, t)$, with θ and t fixed, is called a *projection* and the function $\mathcal{R}_\theta : t \mapsto \mathcal{R}f(\theta, t)$ a *profile*, cf. Fig. 2.2(a).

To the Radon operator is associated a *backprojection* operator $\mathcal{R}^\#$ by

$$\mathcal{R}^\# g(x) = \int g(\theta, \langle x, \theta \rangle)\,\mathrm{d}\theta. \tag{2.3}$$

Whereas $\mathcal{R}$ integrates over all points on a line, $\mathcal{R}^\#$ integrates over all lines through a point.

There are two different modes used for sampling the line integrals. One is *parallel beam scanning* where parallel line integrals are determined for a fixed direction. This process is then repeated for a number of different directions. The other is *fan beam scanning* where line integrals emanating from a source point are determined. This is repeated for a number of different source points, see Fig. 2.2.

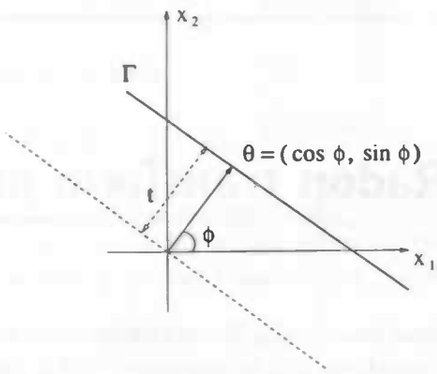
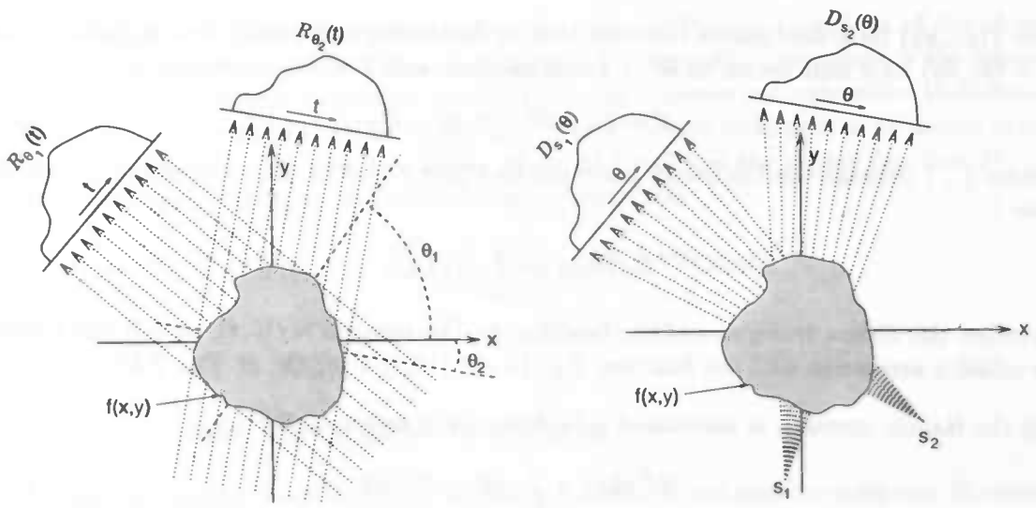


Figure 2.1: Parameters θ, t of the 2D Radon transform for the line Γ.



(a) Parallel beam scanning

(b) Fan beam scanning

Figure 2.2: Scanning modes used in sampling the Radon transform.

2.2 Fourier slice theorem

The relationship between the one-dimensional Fourier transform of a profile and a slice of the two-dimensional Fourier transform of the original object is stated in the *Fourier slice theorem* (Kak & Slaney 1988):

> The Fourier transform of a parallel projection of an image $f(x, y)$ taken at angle θ gives a slice of the two-dimensional Fourier transform of f, $F(u, v)$, subtending an angle θ with the u-axis. In other words, the Fourier transform of $\mathcal{R}_\theta$ gives the values of $F(u, v)$ along line BB in Fig. 2.3.

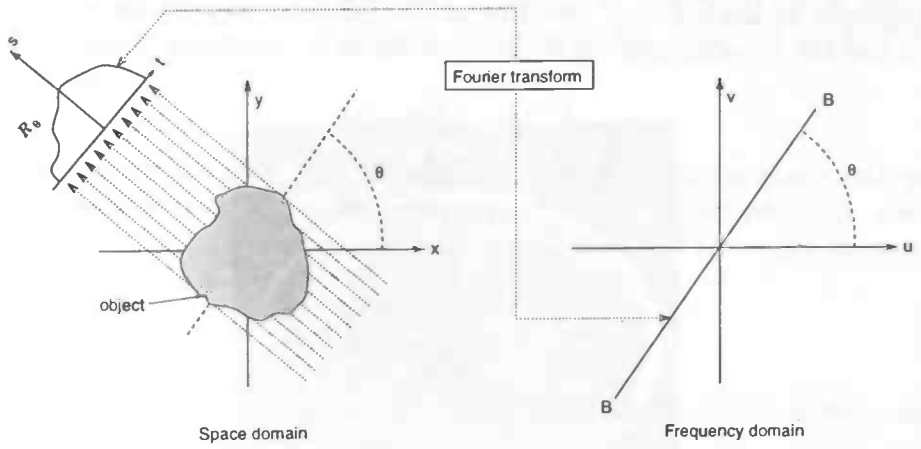


Figure 2.3: Relationship between the Fourier transform of a projection and the Fourier transform of the object along a radial line.

The following simple derivation of the Fourier slice theorem is taken from Kak & Slaney (1988).

Define the two-dimensional Fourier transform of the object function as

$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-i2\pi(ux+vy)} \, \mathrm{d}x \, \mathrm{d}y \tag{2.4}$$

and the Fourier transform of a projection at an angle θ by

$$\hat{\mathcal{R}}_\theta(w) = \int_{-\infty}^{\infty} \mathcal{R}_\theta(t) e^{-i2\pi wt} \, \mathrm{d}t. \tag{2.5}$$

The simplest example of the theorem is given for a projection at $\theta = 0$. Consider the Fourier transform of the object along the line in the frequency domain given by $v = 0$. The integral (2.4) now simplifies to

$$F(u, 0) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-i2\pi ux} \, \mathrm{d}x \, \mathrm{d}y. \tag{2.6}$$

The phase factor is no longer dependent on y, so the integral can be split in two parts,

$$F(u, 0) = \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(x, y) \, \mathrm{d}y \right] e^{-i2\pi ux} \, \mathrm{d}x. \tag{2.7}$$

The term in brackets corresponds to the equation for parallel projection along lines of constant x, c.f. (2.2),

$$\mathcal{R}_{\theta=0}(x) = \int_{-\infty}^{\infty} f(x,y)\,\mathrm{d}y. \tag{2.8}$$

Substituting this in (2.7) leaves us with

$$F(u,0) = \int_{-\infty}^{\infty} \mathcal{R}_{\theta=0}(x) e^{-i2\pi ux}\,\mathrm{d}x. \tag{2.9}$$

Now, the right-hand side of this equation represents the one-dimensional Fourier transform of the projection at angle $\theta = 0$, resulting in the following relation between the vertical projection and the two-dimensional Fourier transform of the object function

$$F(u,0) = \hat{\mathcal{R}}_{\theta=0}(u). \tag{2.10}$$

Obviously, this result is independent of the orientation between the object and the coordinate system. Consider the (t,s) coordinate system to be a rotated version of the original (x,y) system as expressed by the following relation

$$\begin{pmatrix} t \\ s \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{2.11}$$

Using this relation, it can can be shown that

$$\hat{\mathcal{R}}_{\theta}(w) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) e^{-i2\pi w(x\cos\theta + y\sin\theta)}\,\mathrm{d}x\,\mathrm{d}y, \tag{2.12}$$

where the right-hand side represents the two-dimensional Fourier transform of the object function at a spatial frequency vector $(u = w\cos\theta, v = w\sin\theta)$. This result is the essence of parallel ray tomography and forms the basis of many different reconstruction techniques.

If infinitely many angles and rays are taken, $F(u,v)$ would be known at all points in the uv-plane (and not only on a finite number of radial lines) and the object function $f(x,y)$ could be reconstructed using the inverse Fourier transform

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v) e^{i2\pi(ux+vy)}\,\mathrm{d}u\,\mathrm{d}v. \tag{2.13}$$

2.3 Computing the Radon transform

In general, the Radon transform of an arbitrary function is not easy to compute. However, if one uses simple functions, like ellipses, then it is possible to give analytical expressions for the projection along a certain line t at an angle θ. For a particular angle θ, these expressions can be evaluated for any number of lines resulting in a profile for that angle. Taking a number of different angles, the set of profiles results in a complete set of projection data for a given object function.

The standard parallel scanning procedure therefore is to sample $\mathcal{R}f(\theta,t)$ on a polar grid, i.e. taking p directions (number of angles) uniformly distributed over the half-circle and

$2q+1$ equally spaced values of t (number of rays), where $p \geq b$ and $q \geq b/\pi$ (Natterer 1986). Here b is the essential bandwidth of the function f, meaning that the Fourier transform $\hat{f}(\xi)$ of f is negligible for $|\xi| > b$. Insufficiency of the data, either by undersampling a projection or by taking the number of projections too small, causes *aliasing* artifacts such as Gibbs phenomena, streaks (q too small) and Moiré patterns (display resolution too small) (Kak & Slaney 1988). The ideal relation between p and q is $p = \pi q$ (Natterer 1986).

In order to be able to test the accuracy of the reconstruction algorithms which will be described in the following chapters, one needs a reference image. This image should consist of simple objects in order to be able to compute the Radon transform of it. The so-called Shepp-Logan "head phantom" as described in Shepp & Logan (1974) is used for this purpose. This image consists of 10 ellipses as shown in Fig. 2.4. The parameters of the ellipses are given in Table 2.1.

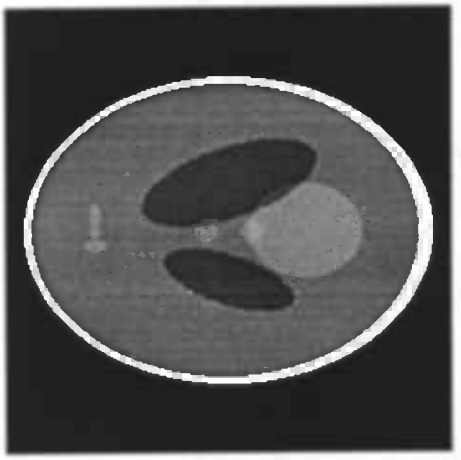


Figure 2.4: The Shepp and Logan "head phantom".

Centre	Major axis	Minor axis	Rotation angle	Refractive index
$(0, 0)$	0.92	0.69	90	2.0
$(0, -0.0184)$	0.874	0.6624	90	-0.98
$(0.22, 0)$	0.31	0.11	72	-0.02
$(-0.22, 0)$	0.41	0.16	108	-0.02
$(0, 0.35)$	0.25	0.21	90	0.01
$(0, 0.1)$	0.046	0.046	0	0.01
$(0, -0.1)$	0.046	0.046	0	0.01
$(-0.08, -0.605)$	0.046	0.023	0	0.01
$(0, -0.605)$	0.023	0.023	0	0.01
$(0.06, -0.605)$	0.046	0.023	90	0.01

Table 2.1: Parameters of the Shepp and Logan phantom.

An advantage of using a phantom as described above is that one can give analytical expressions for the projections. The projections of an image consisting of several ellipses

are simply the sum of the projections for each of the ellipses because of the linearity of the Radon transform. Let $f(x, y)$ be a filled ellipse centred at the origin (see Fig. 2.5),

$$f(x,y) = \begin{cases} \rho & \text{for } \frac{x^2}{A^2} + \frac{y^2}{B^2} \leq 1 \text{ (inside the ellipse)}, \\ 0 & \text{otherwise (outside the ellipse)}, \end{cases} \tag{2.14}$$

where A and B denote the half-lengths of the major and minor axis respectively, and ρ denotes the refractive index.

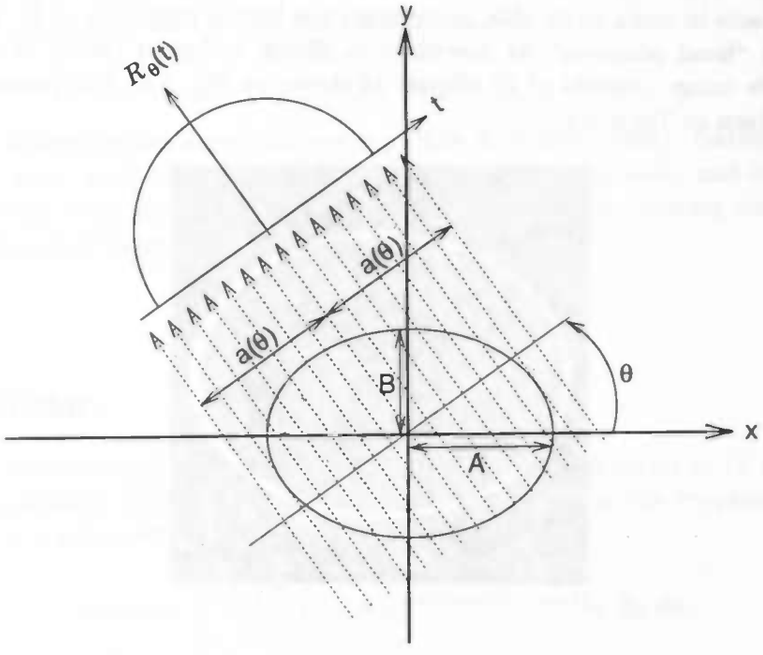


Figure 2.5: Projection of an ellipse at an angle θ.

The projections of such an ellipse at an angle θ are given by

$$\mathcal{R}_\theta(t) = \begin{cases} \frac{2\rho AB}{a^2(\theta)} \sqrt{a^2(\theta) - t^2} & \text{for } |t| \leq a(\theta), \\ 0 & \text{for } |t| > a(\theta), \end{cases} \tag{2.15}$$

where $a^2(\theta) = A^2 \cos^2\theta + B^2 \sin^2\theta$.

The projections $\tilde{\mathcal{R}}_\theta(t)$ for an ellipse centred at (x_1, y_1) and rotated over an angle α are related to $\mathcal{R}_\theta(t)$ in equation (2.15) by

$$\tilde{\mathcal{R}}_\theta(t) = \mathcal{R}_{\theta-\alpha}(t - s\cos(\gamma - \theta)) \tag{2.16}$$

where $s = \sqrt{x_1^2 + y_1^2}$ and $\gamma = \tan^{-1}(y_1/x_1)$, see Fig. 2.6.

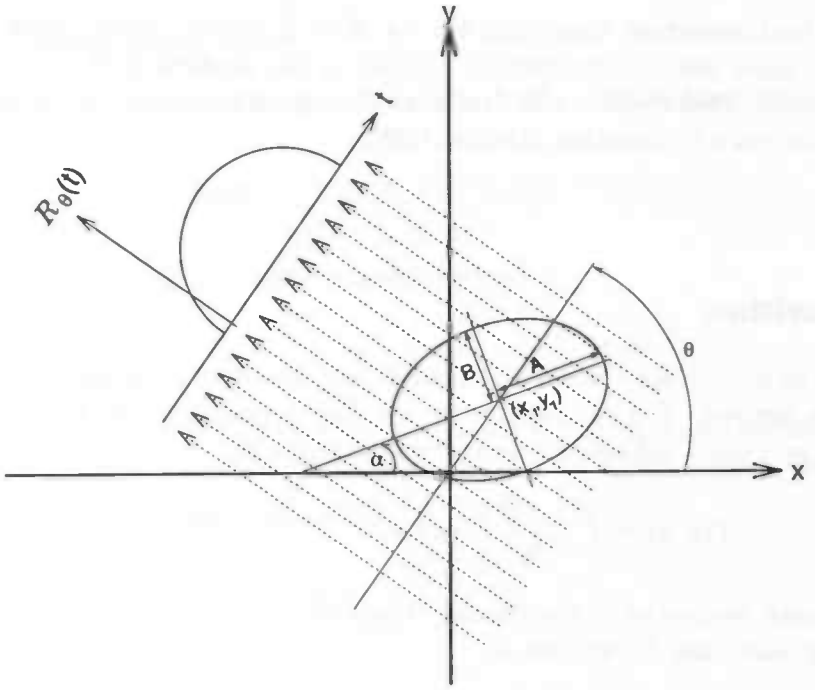


Figure 2.6: The projection of an ellipse centred at (x_1, y_1) and rotated over an angle α.

3

Filtered backprojection

The filtered backprojection algorithm, see e.g. Kak & Slaney (1988), Natterer (1986), is currently the most used reconstruction method in the medical field, and has proved to be very accurate. This chapter will deal with the algorithm, its accuracy and its parallel implementation on a Connection Machine CM-5.

3.1 Algorithm

By a change of coordinate system and rearranging the integral terms in (2.13), the algorithm can be derived. The rectangular uv-coordinate system in the frequency domain is transformed to a polar $w\theta$-coordinate system, resulting in

$$f(x,y) = \int_0^{2\pi} \int_0^{\infty} F(w,\theta) e^{i2\pi w(x\cos\theta + y\sin\theta)} w \,\mathrm{d}w \,\mathrm{d}\theta. \tag{3.1}$$

Using symmetry properties of the Fourier transform and substituting $\hat{\mathcal{R}}_\theta(w)$ for $F(w,\theta)$, the above equation may be written as

$$f(x,y) = \int_0^{\pi} \int_{-\infty}^{\infty} \hat{\mathcal{R}}_\theta(w)|w| e^{i2\pi wt} \,\mathrm{d}w \,\mathrm{d}\theta, \tag{3.2}$$

with $t = x\cos\theta + y\sin\theta$. The multiplication with $|w|$ in the Fourier domain represents a filtering operation, where $|w|$ is called the *ramp* filter. The integral over θ corresponds to the backprojection operator.

The following identity holds,

$$W_b * f = \mathcal{R}^{\#}(w_b * \mathcal{R}f), \qquad W_b = \mathcal{R}^{\#} w_b, \tag{3.3}$$

where $*$ denotes convolution on $\mathbb{R}^2$ and $\mathbb{R}$, respectively (the second convolution is with respect to the last argument of $\mathcal{R}f$). In practice, one chooses w_b in such a way that W_b is a low-pass filter approximating a δ-distribution. This formula forms the basis of the *filtered backprojection* algorithm of inverting the Radon transform.

In the discrete case the Radon transform $\mathcal{R}f(\theta, t)$ is assumed to be available for (θ_j, s_l), $j = 1, \ldots, p$, $l = -q, \ldots, q$, with

$$\theta_j = \begin{pmatrix} \cos\phi_j \\ \sin\phi_j \end{pmatrix}, \qquad \phi_j = \pi(j-1)/p, \quad s_l = hl, \quad h = 1/q. \tag{3.4}$$

Here it is assumed that the support of f is the unit disk, i.e. $|f(x)| = 0$ for $|x| > 1$. The filtered backprojection algorithm in this case goes as follows.

Step 1 For $j = 1, \ldots, p$ carry out the convolutions

$$v_{j,k} = h \sum_{l=-q}^{q} w_b(s_k - s_l)\mathcal{R}_{\theta_j}(s_l), \qquad k = -q, \ldots, q. \tag{3.5}$$

For the function w_b, see below.

Step 2 For each reconstruction point x, compute the discrete backprojection

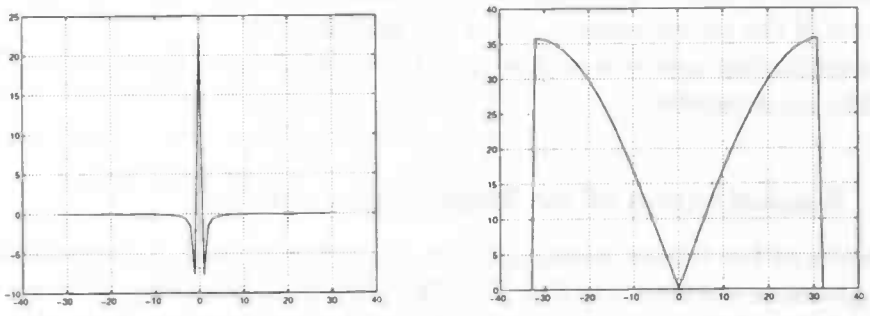
$$f_{FBP}(x) = \frac{2\pi}{p} \sum_{j=1}^{p} \left((1-u)v_{j,k} + uv_{j,k+1}\right), \tag{3.6}$$

where, for each x and j, k and u are determined by

$$s = \langle \theta_j, x \rangle, \qquad k \le \frac{s}{h} \le k+1, \qquad u = \frac{s}{h} - k.$$

Reconstruction may be performed on a $(2q+1) \times (2q+1)$ grid, which corresponds to the sampling of the profiles. One possible choice for w_b is the Shepp-Logan filter. This is a filter with cut-off frequency $b = \pi/h$, see also Fig. 3.1

$$w_b(s_l) = \frac{b^2}{\pi^4} \frac{1}{1-4l^2} \tag{3.7}$$



(a) Time domain (b) Frequency domain

Figure 3.1: Time and frequency domain representation of the Shepp-Logan ramp filter.

3.2 Complexity

The number of operations needed for the convolutions is $O(pq \log q)$ if a FFT is used. The backprojection operation takes $O(p)$ operations for each x. If the reconstruction is performed on a $(2q+1) \times (2q+1)$ grid, this leads to $O(pq^2)$ operations. Using the optimal relation $p = \pi q$, the total complexity is $O(q^3)$.

3.3 Accuracy

The relative error norm is computed by

$$\frac{\sum_{x,y} |O(x,y) - R(x,y)|^2}{\sum_{x,y} |O(x,y)|^2}, \tag{3.8}$$

where $O(x,y)$ represents the original phantom and $R(x,y)$ the reconstructed phantom. The relative error should always be considered together with a visual evaluation, because it is not a very good measure on itself (Herman 1980).

3.3.1 Reconstruction of a disk

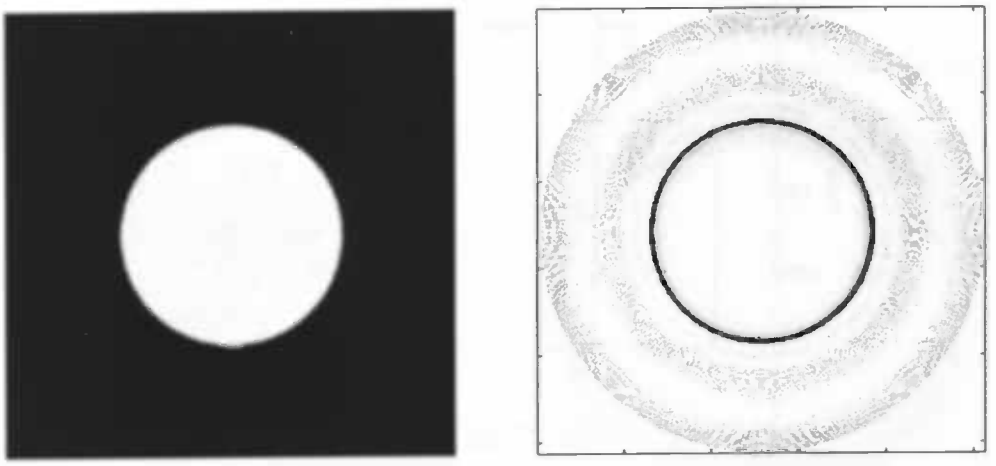
The accuracy of the filtered backprojection algorithm was first tested on a single disk of radius 0.5 and refractive index 0.01. Projection data were generated for $p = 256$ and $q = 127$, and the reconstruction was computed on a 255×255 grid. The results are shown in Fig. 3.2. On the left-hand side the reconstructed image is shown and on the right-hand side the absolute value of the difference with the original disk. The contrast of the latter image has been stretched and inverted in order to make reconstruction artefacts visible. This causes values above a certain threshold to be cropped to the maximum value, but it makes the artefacts in the image visible. The error computed over the whole image is 0.048, but this is mostly due to error near the edge of the disk. The interior of the disk is reconstructed almost perfectly, which is also clear from Fig. 3.3. This plot shows the $y = 0$ line of the reconstruction and of the original disk.

Reconstruction artefacts appear near the edges of the disk. These artefacts are due to the Gibbs phenomenon.

3.3.2 Reconstruction of the Shepp-Logan phantom

The results of the filtered backprojection algorithm applied on projections of the Shepp-Logan phantom are shown in Fig. 3.4. The projection data were generated using $p = 256$ and $q = 127$ and the reconstruction grid was of size 255×255. This plot shows a part of the $y = 0$ line of the Shepp-Logan phantom (the part inside the largest two ellipses).

Applying (3.8) to the part of the reconstruction shown in the plot results in a relative error of 0.002. The relative error computed over the whole image is 0.073. This large difference is due to the large error near the edges of the phantom as this part suffers from the Gibbs-phenomenon. This can be observed in Fig. 3.5. The image on the left shows



(a) (b)

Figure 3.2: Reconstruction of a disk of radius 0.5 and refractive index 0.01 (a) and the difference with the original (b). The values from 0–5% were scaled to the full range of grey values to make the differences visible. Projection data were generated for $p = 256$ and $q = 127$ and the reconstruction grid was of size 255×255.

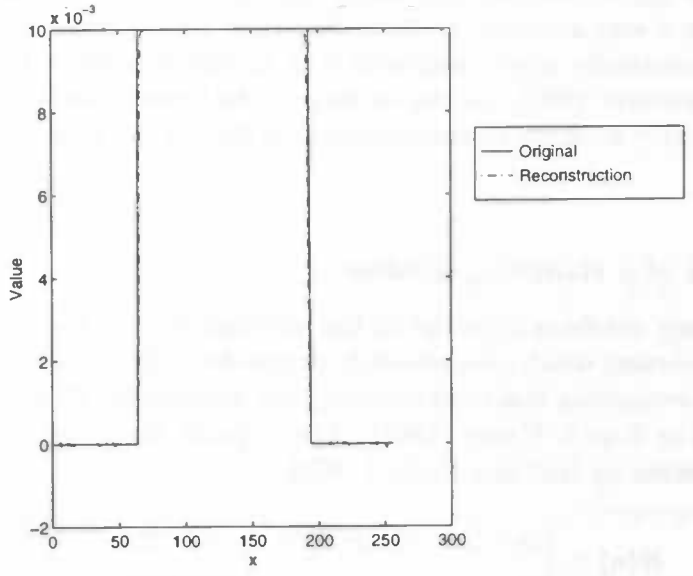


Figure 3.3: A numerical comparison of the $y = 0$ line of the reconstruction with the true values of a disk having radius 0.5 and refractive index 0.01. Here, projection data were generated using $p = 256$ and $q = 127$. Reconstruction was performed on a 255×255 grid.

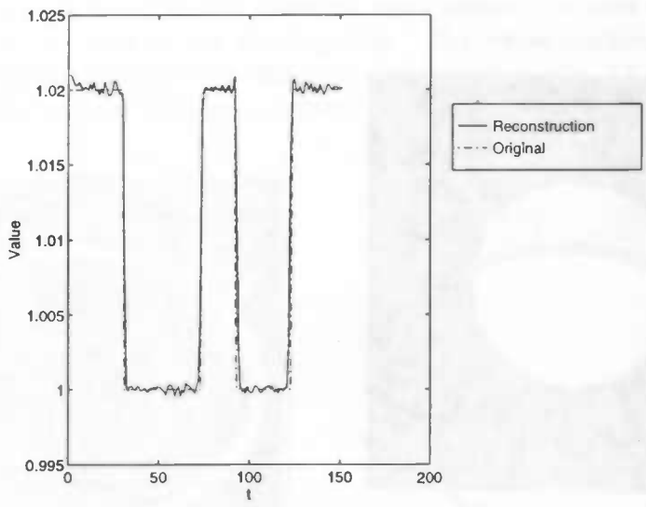


Figure 3.4: A numerical comparison of a part of the $y = 0$ line of the reconstruction with the true values of the Shepp-Logan phantom. Here, projection data were generated using $p = 256$ and $q = 127$. Reconstruction was performed on a 255×255 grid.

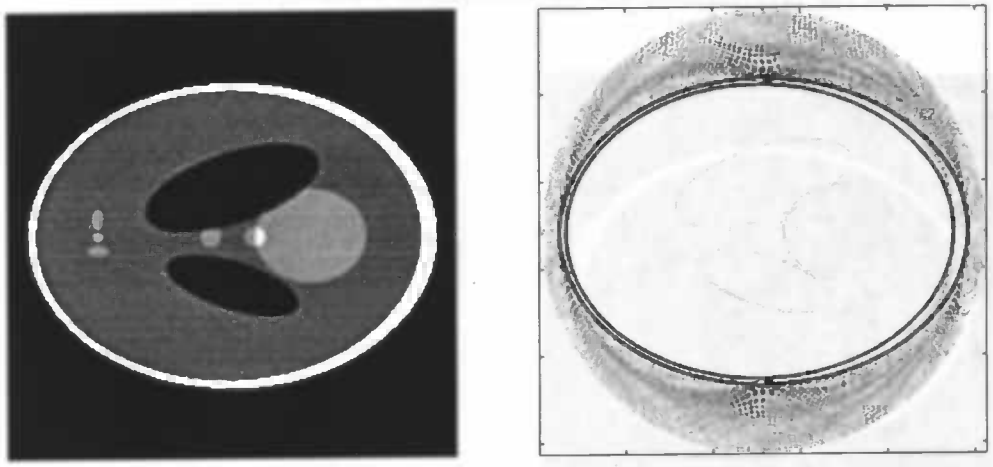
the reconstructed phantom and the image on the right the absolute value of the difference between the original phantom and the reconstruction. The values from 0–10% of the range of the difference image have been inverted and scaled to the full range of gray-values. As can be seen in the figure, there are only small edge effects inside the "head" and elsewhere the reconstruction is very accurate. Outside the largest ellipse Gibbs phenomenon artefacts appear which theoretically would disappear if an infinite number of rays were used (Kak & Slaney 1988, Natterer 1986). As can be seen in the figures, the filtered backprojection algorithm gives a very accurate reconstruction even though the ideal relation $p = \pi q$ is not exactly satisfied.

3.3.3 The use of a Hamming window

The high frequency artefacts reported in the previous section can be smoothed out by using a window function which goes smoothly to zero for high frequencies. Rowland (1979) describes several windowing functions including the 'generalised' Hamming window, which is also suggested by Kak & Slaney (1988). The N-point Hamming window is defined in the frequency domain by (Rabiner & Gold 1975)

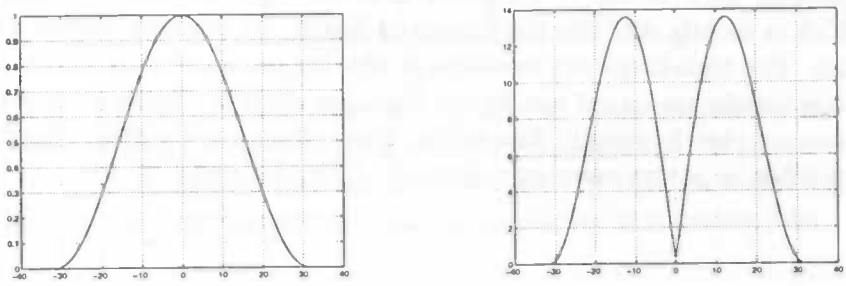
$$H(n) = \begin{cases} \alpha + (1-\alpha)\cos(\frac{2\pi n}{N}) & -\frac{N-1}{2} \leq n \leq \frac{N-1}{2}, \\ 0 & \text{otherwise}, \end{cases} \tag{3.9}$$

where N is assumed to be odd and α is chosen in the range $0 \leq \alpha \leq 1$. If $\alpha = 0.54$ the window is called a *Hamming* window, if $\alpha = 0.5$ it is called a *Hanning* window. The frequency response of the Hanning window is plotted in Fig. 3.6 on the left. On the right the frequency response of the ramp filter convolved with the Hanning window is plotted.



(a) (b)

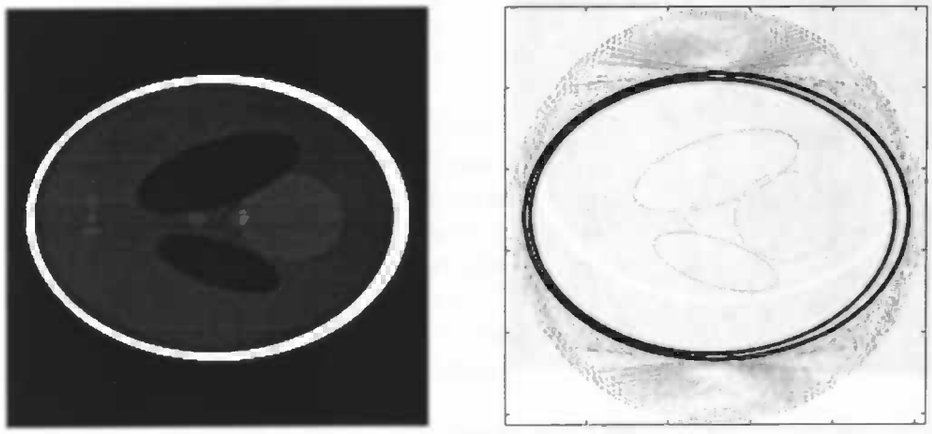
Figure 3.5: Reconstruction of the phantom (a) and the difference with the original (b). The values from 0–10% were scaled to the full range of grey values to make the differences visible. Projection data were generated for $p = 256$ and $q = 127$ and the reconstruction grid was of size 255×255. Inside the "head" the reconstruction is very accurate and only small edge effects are visible. Outside the largest ellipse, the Gibbs phenomenon is visible.



(a) (b)

Figure 3.6: Frequency responses of a Hanning window (a) and the windowed ramp filter (b).

The experiment of reconstructing the phantom was repeated, but now the Hanning windowed ramp filter was used in the filtering step. The reconstruction of the phantom together with a difference image with the original is shown in Fig. 3.7. A comparison of the $y = 0$ line of the original with the reconstruction is shown in Fig. 3.8.



(a) (b)

Figure 3.7: Reconstruction of the phantom (a) and the difference with the original (b). The values from 0–10% were scaled to the full range of grey values to make the differences visible. Projection data were generated for $p = 256$ and $q = 127$ and the reconstruction grid was of size 255×255. Inside the "head" the reconstruction is very accurate and only small edge effects are visible. Outside the largest ellipse, the Gibbs phenomenon is visible although it is much less prominent because of the use of a Hanning window.

The relative error computed over the whole image for the reconstruction of the phantom was 0.034 and 0.002 for the part of the $y = 0$ line. The relative error over the whole image is almost reduced by a factor of 2 compared to the reconstruction without a Hanning window, which is mostly due the the improved quality of the reconstruction outside the largest ellipse. The high-frequency artefacts in this region, due to the Gibbs phenomenon, are almost completely smoothed out by the Hanning window. These improvements in the reconstruction are clearly visible. The edges of the ellipses are a little smoothed out, but this is only visible by a very close examination of Fig. 3.4 and Fig. 3.8.

3.4 Implementation

Timings were carried out for different inputs on a Connection Machine CM-5 with 16 processors. In order to keep matters simple, only one input variable N is used for the number of rays, angles and the reconstruction grid. So, projections were generated using

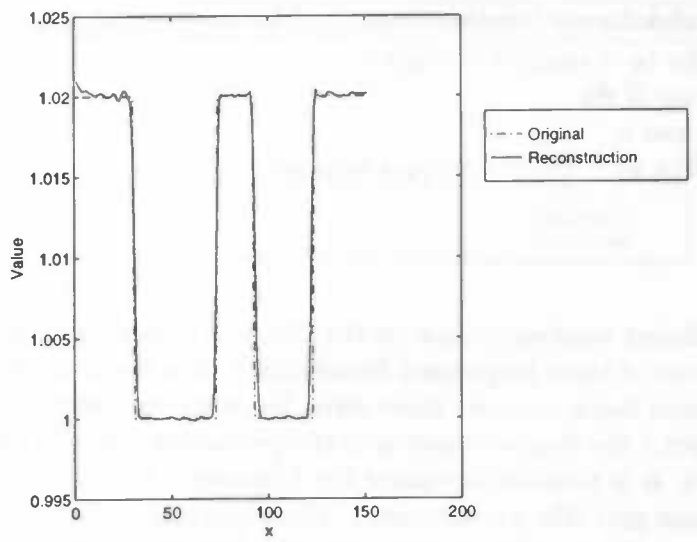


Figure 3.8: A numerical comparison of a part of the $y = 0$ line of the reconstruction with the true values of the Shepp-Logan phantom. Here, projection data were generated using $p = 256$ and $q = 127$. Reconstruction was performed on a 255×255 grid. The ramp filter was modified by a Hanning window.

N rays over N angles, and the reconstruction grid was of size $N \times N$. Table 3.1 shows the run-time of the algorithm for different values of N.

There are three different implementations of the filtered backprojection algorithm. These are described in the sections below.

	Straightforward		**Lookup tables**	**Detailed FFT**	**One lookup table**
N	*Filtering*	*Backproject*	*Backproject*	*Filtering*	*Backproject*
32	0.472	0.148	0.023	0.012	0.022
64	1.598	0.647	0.074	0.020	0.054
128	2.953	3.820	0.224	0.059	0.207
256	7.342	22.876	1.310	0.100	1.129
512	15.349	120.581	8.541	0.213	7.652
1024	33.340	1004.963	57.435	0.754	???

Table 3.1: Timing results in seconds on the CM-5 for different implementations of the filtered backprojection algorithm. The first method is a straightforward implementation, the second one uses lookup tables in the backprojection step, the third one uses a different FFT from the scientific library CMSSL. The last method uses one large lookup table containing all the filtered projection data.

3.4.1 Straightforward method

In the first attempt, formula (3.6) for the backprojection step was translated directly into CM-Fortran. Let D be the reconstruction grid, then the algorithm (in pseudo-code) goes as shown in Algorithm 1.

In contrast to what (3.6) suggests, the loops over j and x, y are interchanged. This

Algorithm 1 Straightforward implementation of backprojection step.

```
for j = 1 to p do {p = number of angles}
  for all (x, y) on D do
    compute k and u
    R(x, y) = R(x, y) + (2π/p)((1 − u)v_{j,k} + u v_{j,k+1})
  end for
end for
```

leads to a more efficient implementation on the CM-5. If a number of loops is used on this machine, at least one of these loops must be serialised. It is faster to make one loop serial instead of two nested loops, because these serial loops are executed by the front-end. So in this case, for each j the front-end instructs the processing nodes to execute the loop on x and y in parallel. It is possible to remove the loop over j by adding a third component to the reconstruction grid (the j component). Computations could then be carried out on all dimensions in parallel and afterwards be summed in the j-direction. A drawback of this method is that it consumes a lot of memory.

Timing results are shown in Table 3.1. As can be seen, the backprojection step consumes a lot of time and grows roughly by a factor of 6 when N is doubled. Detailed investigation showed that most time was consumed by communications between different processors (in the assignment to R in Algorithm 1). The last step in the algorithm, i.e. the actual backprojection, requires for each pixel in the reconstruction different values of the filtered projections for fixed angle. As these projections are distributed over different processors, this step requires a lot of communication, i.e. a maximum of $O(N^2)$ communications for each angle.

3.4.2 Use of a lookup table

The communicational demand can be decreased by making use of a so-called lookup table. This is a construct provided by the Fortran Library on the CM-5. It creates a lookup table for a given array of values and assigns a copy of this table to each processor. Addressing can then be done locally on each processor and thus requires no communication. This reduces the number of communications for a given angle from $O(N^2)$ to $O(N)$. The pseudo code is given in Algorithm 2.

Algorithm 2 Backprojection step using lookup tables.

```
for j = 1 to p do {p = numes of angles}
  for all (x, y) on D do
    compute k and u
    create a lookup table l for v_{j,*}
    R(x, y) = R(x, y) + (2π/p)((1 − u)l_k + u l_{k+1})
  end for
end for
```

Fig. 3.9 shows the dramatic reduction of run-time (see also Table 3.1) when using the lookup tables. A mean speedup by a factor of about 16 is achieved for values of $N \geq 128$.

Note the reduction in run-time for $N = 1024$, from almost 17 minutes down to 57 seconds.

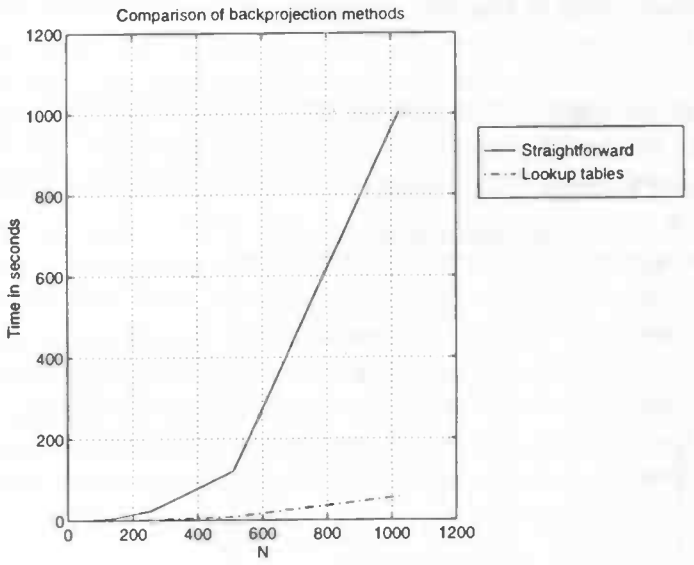


Figure 3.9: Comparison of run-time of the backprojection step between the straightforward method and the one using lookup tables.

3.4.3 Alternative Fourier transform

The filtering step up to now used the so-called 'simple' FFT from the Connection Machine Scientific Software Library (CMSSL) (Thi 1994). This routine is optimised for the architecture of the CM-5 and achieves a high performance. However, it can only be used to transform a data set along all dimensions (rows and columns in the two-dimensional case) in the same direction (forward or reverse). To transform along a single dimension, one has to use a `DO` loop and transform each data vector separately. This loop is serialised and gives thus a very poor performance.

Fortunately, CMSSL also provides the so-called 'detailed' FFT. This routine has a greater flexibility, at the cost of a more complex interface, than the simple FFT. For each dimension separately, one can specify which operation is to be carried out, i.e. forward or reverse transform or no transform at all. Furthermore, it can be specified which bit-ordering is to be used for the addresses of the results. The Cooley-Tukey FFT bit-reverses the addresses in which it stores the results. This means that, after a transform, the elements would have to be sorted in order to put them in the right order. However, in a filtering step one needs to carry out an element-wise multiplication of two Fourier transformed vectors. The ordering of the elements can be arbitrary in this case, but it has to be the same for both vectors. Thus, to perform a filtering step in the frequency domain, it is not necessary to sort the elements after the transform. So the filtering step is carried out as follows:

1. Take the Fourier transform of the filter and the data vector and use bit-reversed ordering.
2. Carry out an element-wise multiplication.

3. Take the inverse Fourier transform and use normal ordering.

Computing the filtering step in this way saves two sorting operations and achieves a high performance.

The filtering step in the algorithm is now replaced by the one described above. As can be seen from Table 3.1 the performance increases immensely, see also Fig. 3.10. Even for the largest value of N used, 1024, the filtering step requires less than a second.

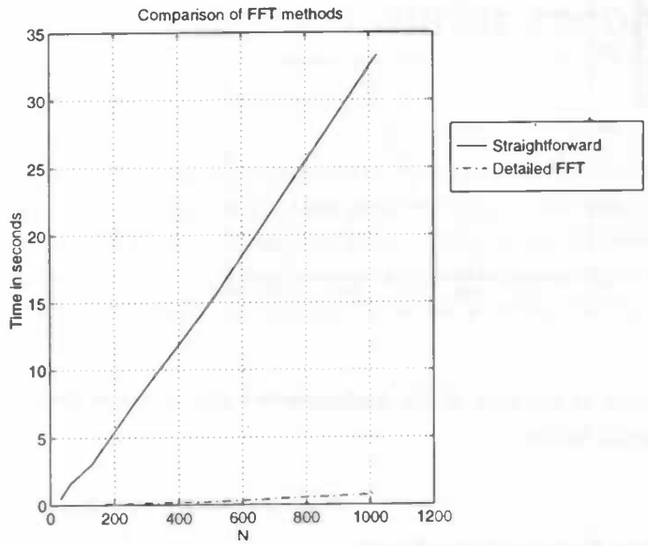


Figure 3.10: Comparison of run-time of the filtering step between the straightforward method and the one using the detailed FFT.

3.4.4 Use of one lookup table

In Section 3.4.2 the usage of lookup tables was explained and it became clear that they provide a significant reduction of run-time. The method used there iterates over the angles at which the profiles were available and creates in each iteration a lookup table of that profile. However, this causes some overhead in each iteration for the creation and destruction of such a lookup table. Another method would therefore be to create one large lookup table of all profiles before the iteration is started. It is then quite plausible that this would cause even more reduction of run-time. The timing results are shown in Table 3.1 in the rightmost column. As can be seen, the results are a little disappointing. For $N = 512$ a reduction of almost a second is achieved which is only about 10%. A problem with this method is the memory usage, since a copy of all projection data is assigned to each processor. For $N = 1024$ the memory requirements are too large and the program cannot reconstruct images of this size.

The results show that using one large lookup table hardly gives improvements in run-time. Together with the increase of memory usage, this method seems less useful than the one described in Section 3.4.2.

3.4.5 Final algorithm

The final algorithm consists of both the improved filtering step as the improved backprojection step from Section 3.4.2. In Fig. 3.11 the time used for filtering and backprojecting is plotted. As can be seen, the backprojection step still uses most of the run-time. Improvement could possibly be gained by using a shared memory computer instead of a distributed memory computer. For each point in the reconstruction grid namely, information of all angles is needed, and this information is distributed over different processors. It is not possible to choose the layout of the data in such a way that there will be (almost) no communication. For a part, this problem is solved by using the lookup tables as described in Algorithm 2 above, but there remains a lot of communication in the creation of the table. But the use of the lookup tables shows that having the filtered projection data available locally speeds up the backprojection process significantly, indicating that more speedup could be gained from a shared memory computer. However, communication with the memory of such a computer would go via a switching network and it is not a priori clear if the algorithm would gain in speed. This leaves an open question which would be interesting to investigate.

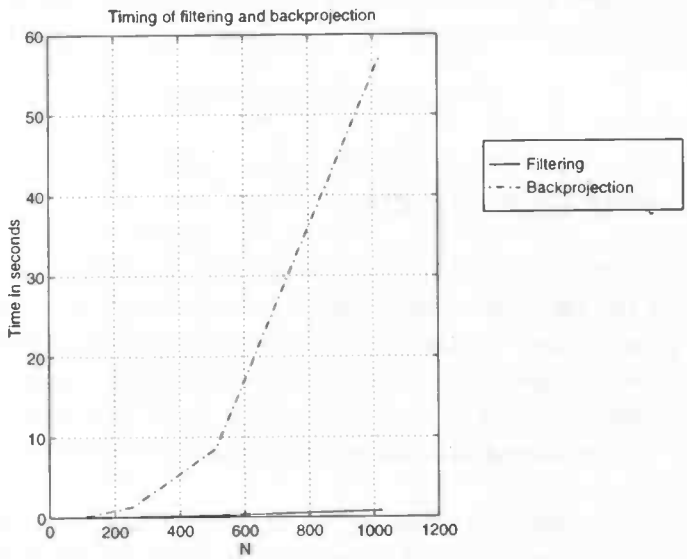


Figure 3.11: Graph of the time needed for filtering and backprojecting in the final algorithm. As can be seen from the plot, most time is needed for backprojecting.

4

Fourier reconstruction

In this chapter the direct Fourier reconstruction method will be described. This has a lower complexity than filtered backprojection making it an interesting method to investigate. First the idea behind Fourier reconstruction will be given followed by the standard Fourier reconstruction algorithm. Then a new and better algorithm is proposed. The implementation of this new algorithm together with its accuracy will be considered at the end of this chapter.

4.1 Basic Fourier reconstruction

The *Fourier slice theorem* from Section 2.2 links the one-dimensional Fourier transform of a profile to a slice of the two-dimensional Fourier transform of the object function. This theorem not only gives a basis for the filtered backprojection algorithm but also for the so-called *direct Fourier* methods. These methods consist of implementations of (2.5) (the one dimensional Fourier transform of the profiles) and (2.13) (the two-dimensional inverse Fourier transform). The problems with these methods are due to the discretisation of these equations.

In the discrete case the Radon transform $\mathcal{R}f(\theta,t)$ is assumed to be sampled at (θ_j, s_l), $j = 1,\ldots,p$, $l = -q,\ldots,q$, with θ_j and s_l as given in (3.4). In the following, the largest value for l is chosen to be $q-1$ in order to have an even number of values for l. In the ideal case l has a number of values equal to a power of 2, so that FFT algorithms can be used to arrive at fast implementations. The polar coordinate grid $\mathcal{G}_{p,q}$ is given by

$$\mathcal{G}_{p,q} = \{\pi r\theta_j \quad : \quad r = -q,\ldots,q-1, \quad j = 1,\ldots,p\}. \tag{4.1}$$

The standard Fourier reconstruction algorithm goes as follows (Natterer 1986):

Step 1 For $j = 1,\ldots,p$, compute approximations $\hat{g}_{jr}$ to $\widehat{\mathcal{R}f}(\theta_j, r\pi)$ by

$$\hat{g}_{jr} = \frac{1}{\sqrt{2\pi}} h \sum_{l=-q}^{q-1} e^{-i\pi lr/q} \mathcal{R}f(\theta_j, s_l), \qquad r = -q,\ldots,q-1, \tag{4.2}$$

This first step provides an approximation to the two-dimensional Fourier transform $\hat{f}$ of the object function f on radial lines on $\mathcal{G}_{p,q}$ as expressed by

$$\hat{g}_{jr} \approx \frac{1}{\sqrt{2\pi}} \hat{f}(r\pi\theta_j). \tag{4.3}$$

These radial lines are also shown in Fig. 4.1.

Step 2 The inverse two-dimensional FFT can not be used on the polar coordinate grid, so $\hat{g}_{jr}$ needs to be interpolated onto a Cartesian grid. For each $k \in \mathbb{Z}^2$, $||k|| < q$, find a point $\xi_k = \pi r\theta_j \in \mathcal{G}_{p,q}$ closest to πk and set

$$\hat{f}_k = \frac{1}{\sqrt{2\pi}} \hat{g}_{jr}. \tag{4.4}$$

In this step, nearest neighbour interpolation is used to switch from the polar coordinate grid to a Cartesian grid.

Step 3 Compute an approximation f_m to $f(hm)$, $m \in \mathbb{Z}^2$ by

$$f_m = \frac{\pi}{2} \sum_{||k||<q} e^{i\pi m \cdot k/q} \hat{f}_k, \qquad ||m|| < q. \tag{4.5}$$

This is the discrete inverse two-dimensional Fourier transform.

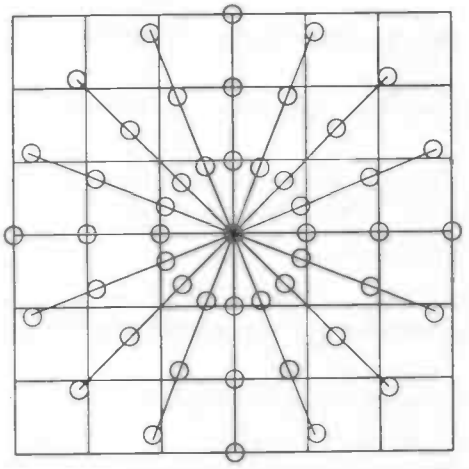


Figure 4.1: The representation of $\hat{g}_{jr}$ on the polar grid for $p = 8$, $q = 3$. The origin of the coordinate axes lies in the middle of the figure. The circles on the polar lines indicate the points on which $\hat{g}$ is known. The Cartesian grid is also shown for reference.

The crucial step in this algorithm is the interpolation from the polar grid to the Cartesian grid. The simple nearest neighbour interpolation described above turns out to be too simple. It produces severe artefacts and is very inaccurate (Natterer 1986), so several other interpolation techniques have been proposed. Stark, Woods, Paul & Hingorani (1981) suggest a complicated sinc-interpolation step and they claim to reach reconstructions of a quality comparable to that of filtered backprojection. Natterer (1986) gives two alternative methods. The first one combines oversampling of $\hat{g}$ with sinc-interpolation and the other is the so-called Pasciak algorithm (Pasciak 1973). This is the one also used by Schulte (1994) and will be described in the following sections.

4.2 Pasciak algorithm

The idea behind this method of Pasciak (1973) is to alter the first step (4.2) in such a way that the values of $\hat{g}(\theta_j, \sigma)$ lie on the vertical lines of the Cartesian grid for $|\cos\phi_j| \geq |\sin\phi_j|$, $\phi_j = \pi(j-1)/p$, and on the horizontal lines otherwise. The so computed values of $\hat{g}$ can then be assigned to the grid points of the Cartesian grid by either using nearest neighbour interpolation or linear interpolation. As the algorithm using linear interpolation gives better results this is the one described below.

Step 1 For $j = 1, \ldots, p$ compute approximations $\hat{g}_{jr}$ to $\hat{g}(\theta_j, \pi r c(j))$ by

$$\hat{g}_{jr} = \frac{1}{\sqrt{2\pi}} h \sum_{l=-q}^{q-1} e^{-i\pi l c(j) r/q} g(\theta_j, s_l), \qquad r = -q, \ldots, q-1, \tag{4.6}$$

where $c(j) = 1/\max(|\sin\phi_j|, |\cos\phi_j|)$. This corresponds to computing the discrete Fourier transform for arbitrary step-size in the frequency domain, which can be done by the chirp z-transform which will be described in Section 4.3. The result of this transform is shown in Fig. 4.2. As can be seen, the points of $\hat{g}$ are much closer to the grid points of $\mathcal{G}_{p,q}$. Note the difference between Fig. 4.1 and Fig. 4.2 for the points on the diagonals, these are now exactly on the grid points and no interpolation is required for them.

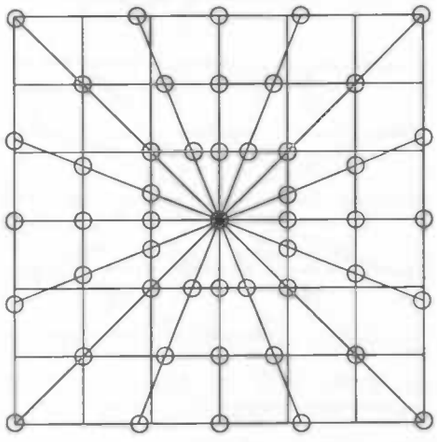


Figure 4.2: Result after applying the chirp z-transform to $\hat{g}$. The origin of the coordinate axes lies in the middle of the figure. In this case $p = 8$ and $q = 3$.

Step 2 The values of $\hat{g}_{jr}$ are now to be interpolated to the Cartesian grid. Let $p \geq \pi q$ and divisible by 4 so that symmetries can be used in the implementation. Compute the approximation $\hat{f}_{k_1 k_2}$ at the point πk, $k = (k_1, k_2)$ with $||k|| \leq q$ in the following way.

- For the coordinate axes, where $k_1 k_2 = 0$, choose

$$\begin{aligned} r &= k_1 + k_2 \\ j_1 &= \begin{cases} 0 & k_2 = 0 \\ p/2 & \text{otherwise} \end{cases} \end{aligned}$$

Set

$$\hat{f}_{k_1k_2} = \frac{1}{\sqrt{2\pi}}\hat{g}_{j_1r}$$

- On the diagonals, $|k_1| = |k_2| > 0$, choose

$$j_1 = \begin{cases} p/4 & k_1k_2 > 0 \\ 3p/4 & k_1k_2 < 0 \end{cases}$$

Set

$$\hat{f}_{k_1k_2} = \frac{1}{\sqrt{2\pi}}\hat{g}_{j_1k_2}$$

- For $|k_1| > |k_2| > 0$, choose

$$r = \begin{cases} k_1 & k_1k_2 > 0 \\ -k_1 & k_1k_2 < 0 \end{cases}$$

Choose $\alpha_1, \alpha_2 \in \{0, \ldots, p/4\}$, $\alpha_1 \neq \alpha_2$, so that $x_i = ||k_2| - |k_1|c(\alpha_i)\sin(\alpha_i\pi/p)|$, $i = 1, 2$, as small as possible.

$$j_i = \begin{cases} \alpha_i & k_1k_2 > 0 \\ p - \alpha_i & k_1k_2 < 0 \end{cases}$$

Set

$$\hat{f}_{k_1k_2} = \frac{1}{\sqrt{2\pi}} \frac{x_2\hat{g}_{j_1r} + x_1\hat{g}_{j_2r}}{x_2 + x_1}$$

- For $|k_2| > |k_1| > 0$, choose $\alpha_1, \alpha_2 \in \{p/4, \ldots, p/2\}$, $\alpha_1 \neq \alpha_2$, so that $x_i = ||k_1| - |k_2|c(\alpha_i)|\cos(\alpha_i\pi/p)||$, $i = 1, 2$ as small as possible.

$$j_i = \begin{cases} \alpha_i & k_1k_2 > 0 \\ p - \alpha_i & k_1k_2 < 0 \end{cases}$$

Set

$$\hat{f}_{k_1k_2} = \frac{1}{\sqrt{2\pi}} \frac{x_2\hat{g}_{j_1k_2} + x_1\hat{g}_{j_2k_2}}{x_2 + x_1}$$

Step 3 To suppress the high frequencies, filter $\hat{f}$ with the cos-filter. For $k \in \mathbb{Z}^2$,

$$\hat{F}_k = \begin{cases} \cos(\frac{\pi||k||}{2q})\hat{f}_k & ||k|| < q \\ 0 & ||k|| \geq q \end{cases}$$

Step 4 Compute the discrete inverse two-dimensional Fourier transform f_m, $m \in \mathbb{Z}^2$, of $\hat{F}$ by

$$f_m = \frac{\pi}{2} \sum_{||k|| < q} e^{i\pi m \cdot k/q} \hat{F}_k, \qquad ||m|| < q. \tag{4.7}$$

4.3 Chirp z-algorithm

The discrete Fourier transform with step-size $u > 0$ of a sequence x of length $2q$ is given by

$$\hat{X}_r = \frac{1}{\sqrt{2\pi}} h \sum_{l=-q}^{q-1} e^{-irlu/q} x_l, \qquad r = -q, \ldots, q-1, \tag{4.8}$$

For $u = \pi$ the above equation is simply a discrete Fourier transform of length $2q$ and one could use the standard Fast Fourier Transform algorithm as an implementation. However, if an arbitrary step-size u is wanted, one needs to rewrite (4.8) in order to arrive at a fast algorithm, the so-called chirp z-algorithm (Nussbaumer 1982, Rabiner & Gold 1975).

Write the exponent in (4.8) as

$$\frac{-ilur}{q} = \frac{iu}{2q}(r-l)^2 - \frac{iu}{2q}(r^2 + l^2). \tag{4.9}$$

Substitution of this in (4.8) results in

$$\hat{X}_r = e^{-i(u/2q)r^2} \left(\frac{1}{\sqrt{2\pi}} h \sum_{l=-q}^{q-1} e^{i(u/2q)(r-l)^2} \left(e^{i(u/2q)l^2} x_l \right) \right). \tag{4.10}$$

From (4.10) the following algorithm can be obtained.

$$\hat{X}_r = e^{-i(u/2q)r^2} \hat{X}'_r, \tag{4.11}$$

$$\hat{X}'_r = \frac{1}{\sqrt{2\pi}} h \sum_{l=-q}^{q-1} e^{i(u/2q)(r-l)^2} x'_l, \qquad r = -q, \ldots, q-1, \tag{4.12}$$

$$x'_l = e^{-i(u/2q)l^2} x_l. \tag{4.13}$$

The second formula is a convolution of length $2q$ which can be implemented by using three discrete Fourier transforms of length $4q$. This can be implemented by using a FFT algorithm. The complexity of this algorithm is then $O(q)$ for the pre-multiplication of the sequence x_l, followed by a convolution of $O(q \log q)$ and post-multiplication of the result which takes $O(q)$ multiplications. The total complexity is therefore of $O(q \log q)$.

4.4 Complexity

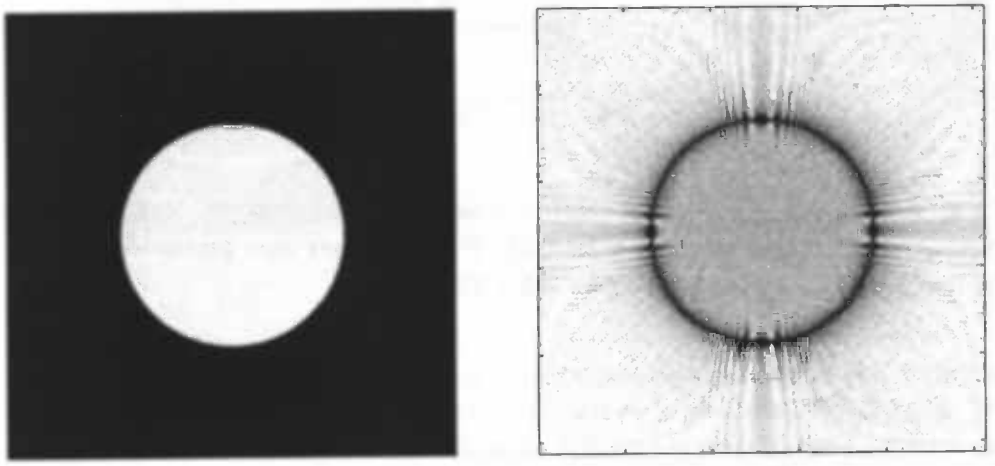
The complexity of the Pasciak algorithm can be estimated as follows. The first step takes $O(pq \log q)$ operations. The second step, the interpolation, uses $O(q^2)$ operations. The filtering with the cos-filter needs $O(q^2)$ multiplications and finally the two-dimensional inverse Fourier transform can be done in $O(q^2 \log q)$ steps. Using the relation $p = \pi q$ the total complexity is $O(q^2 \log q)$.

4.5 Accuracy

The accuracy of the Pasciak algorithm was investigated by taking the same test images as used before, i.e. a disk and the Shepp-Logan phantom.

4.5.1 Reconstruction of a disk

This image consists of a circle of radius 0.5 and has refractive index 0.01 (this image was also used in Section 3.3.1). The error computed over the whole image is 0.051. However, the errors are not only at the edge of the disk but also at the interior of the disk. The reconstruction is shown in Fig. 4.3 on the left and the difference with the original on the right. The right image is scaled as described previously to make the artefacts visible.



(a) (b)

Figure 4.3: Reconstruction of a disk of radius 0.5 and refractive index 0.01 (a) and the difference with the original (b). Projection data were generated for $p = 400$ and $q = 127$. Reconstruction was performed on a 255×255 grid. The lower 5% of the gray values in the image were scaled to the full range in order to make artefacts visible.

Figure 4.4 shows a plot of the $y = 0$ line. Although not as accurate as filtered backprojection, the reconstruction is quite acceptable. The reconstruction artefacts appear mainly at the edges of the disk where the high frequencies of the image are.

4.5.2 Reconstruction of the Shepp-Logan phantom

The accuracy of the Pasciak method was then tested on the standard Shepp-Logan phantom. Projection data were generated for $p = 400$ and $q = 127$, so that p and q satisfy the relation $p = \pi q$. The reconstructed image is shown in Fig. 4.6 on the left. Computing the error norm (3.8) over the whole image results in an error of 0.081. The section of the line $y = 0$ shown in Fig. 4.5 has an error of 0.024. From the figure, it is clear this reconstruction method has a far lower accuracy than filtered backprojection.

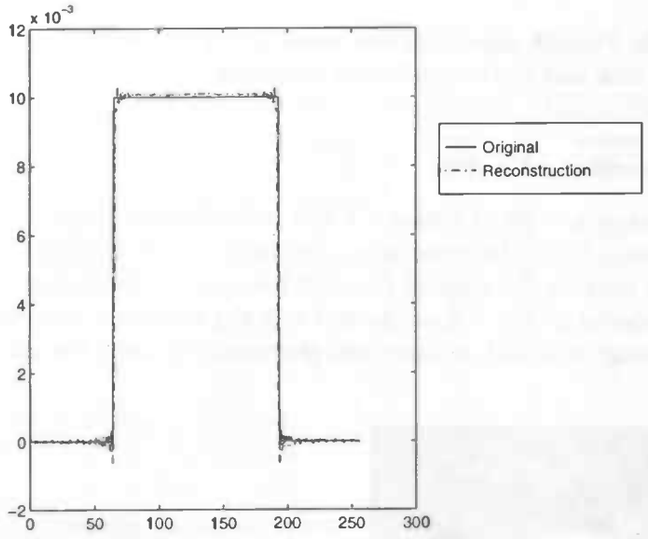


Figure 4.4: A numerical comparison of the $y = 0$ line of the reconstruction with the true values of a disk having radius 0.5 and refractive index 0.01. Projection data were generated for $p = 400$ and $q = 127$. Reconstruction was performed on a 255×255 grid.

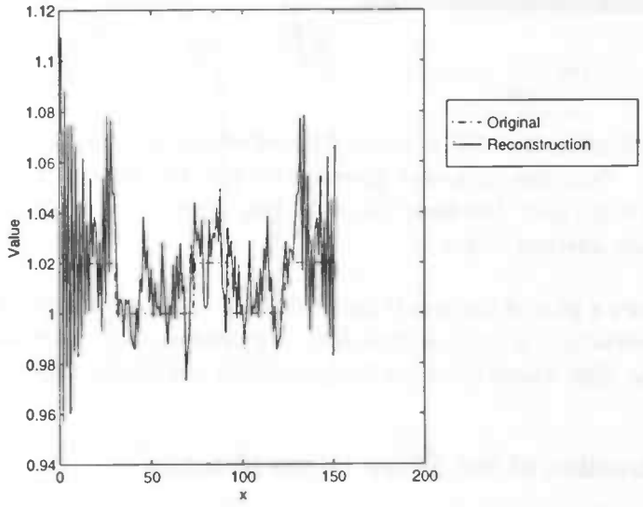
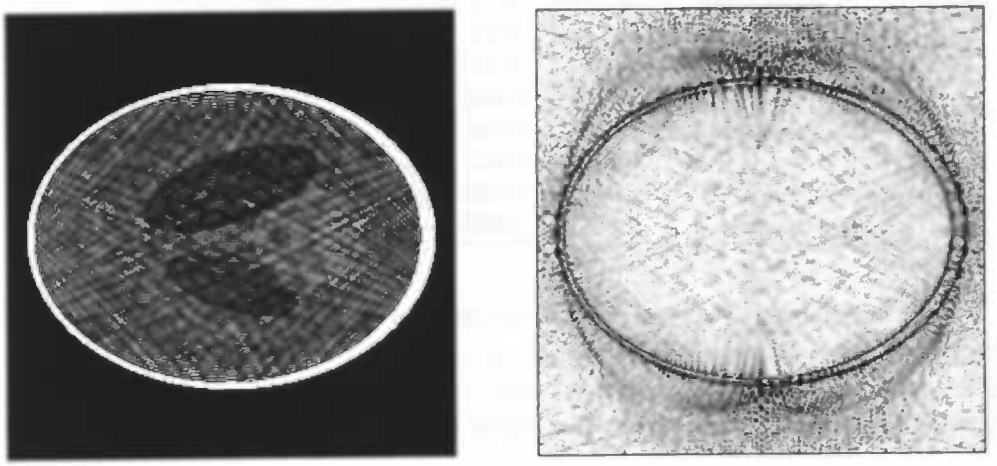


Figure 4.5: A numerical comparison of a part of the $y = 0$ line of the reconstruction with the true values of the Shepp-Logan phantom. Projection data were generated for $p = 400$ and $q = 127$. Reconstruction was performed on a 255×255 grid.

This method is too inaccurate to distinguish between the subtle differences in the refractive index of the ellipses. This is clear from Fig. 4.6. The reconstructed phantom is shown on the left and the absolute value of the difference with the original phantom is shown on the right. The difference image is scaled in such a way that reconstruction artefacts become visible. There are a lot of high frequency artefacts visible in the form of lines over the image.



(a) (b)

Figure 4.6: Reconstruction of phantom (a) and the difference with the original (b). Projection data were generated for $p = 400$ and $q = 127$. Reconstruction was performed on a 255×255 grid. The lower 10% of the gray values in the image were scaled to the full range in order to make artefacts visible.

4.5.3 A simpler phantom

Since the results for the standard phantom are disappointing, but are relatively good for the simple disk, a simpler version of the standard Shepp-Logan phantom was tried to see if this would give better results. This simpler phantom has greater differences between the refractive coefficients of the ellipses. Table 4.1 shows the ellipses used in this phantom.

The results for the simpler version of the phantom are visually much more attractive. The error computed over the whole image gives 0.078 and over the part of the $y = 0$ line 0.059. This reconstruction of this line is compared with the original in Fig. 4.7 and it is clear that the reconstruction of this phantom is much better than for the standard phantom.

On the reconstruction there are virtually no artefacts visible. On the difference image however, the same high frequency effects as described before can be seen (see Fig. 4.8).

Centre	Major axis	Minor axis	Rotation angle	Refractive index
$(0,0)$	0.92	0.69	90	90
$(0,-0.0184)$	0.874	0.6624	90	-40
$(0.22,0)$	0.31	0.11	72	-20
$(-0.22,0)$	0.41	0.16	108	-20
$(0,0.35)$	0.25	0.21	90	10
$(0,0.1)$	0.046	0.046	0	10
$(0,-0.1)$	0.046	0.046	0	10
$(-0.08,-0.605)$	0.046	0.023	0	10
$(0,-0.605)$	0.023	0.023	0	10
$(0.06,-0.605)$	0.046	0.023	90	10

Table 4.1: Parameters of the simpler Shepp and Logan phantom.

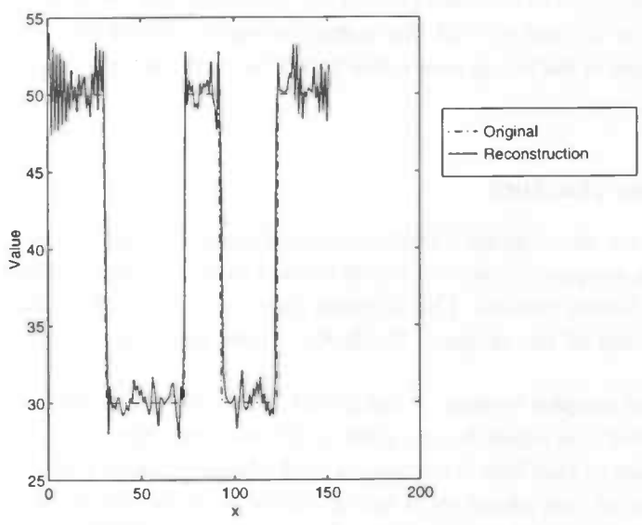
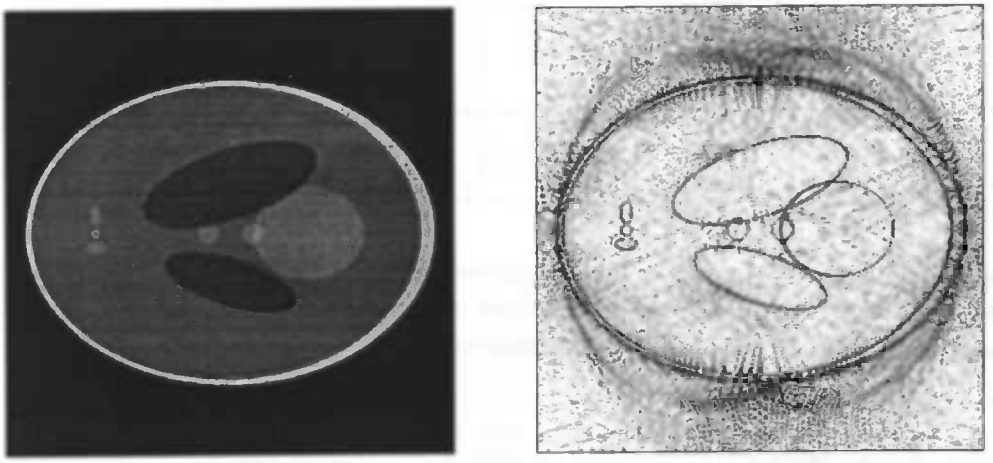


Figure 4.7: Comparison of a part of the $y = 0$ line of the reconstruction with the original simpler Shepp-Logan phantom. In this case $p = 400$ and $q = 127$.



(a) (b)

Figure 4.8: Reconstruction of the simpler phantom (a) and the difference with the original (b). Projection data were generated for $p = 400$ and $q = 127$. Reconstruction was performed on a 255×255 grid. The lower 10% of the gray values in the image were scaled to the full range in order to make artefacts visible.

4.6 Implementation

The Pasciak algorithm is very suitable for parallelisation. The chirp z-transform which can be implemented as a convolution can be transformed into CM-Fortran very easily. The Fourier transform of the CM-5 is very efficient as already could be concluded from Section 3.4.3 so there are not many problems to expect for the chirp z-transform, cf. (4.6), and the two-dimensional inverse Fourier transform, cf. (4.7). The interpolation step is also easy to parallelise using the available constructs of CM-Fortran like `WHERE` and `FORALL`. The `WHERE` construct corresponds more or less to a parallel `IF` statement, and `FORALL` to a parallel `DO`. And as for each (x, y) in the reconstruction grid the same operations are to be carried out, it is also suitable for parallelisation. So, this algorithm can be implemented on the CM-5 in a straightforward manner. Using lookup tables for the different values of θ and the sine and cosine of these angles, it is possible to arrive at fast implementations.

The strategy from Section 3.4 is also used here to carry out timings on the CM-5 for different inputs. The parameter N has the same meaning, i.e. the number of rays and angles. The size of the reconstruction grid is coupled to the number of rays used. The results are shown in Table 4.2 for different values of N.

The chirp z-transform is computed in double precision to prevent round off errors in an early stage. The interpolation step en inverse Fourier transform are computed in single precision. This is the reason why the chirp z-transform and the inverse two-dimensional Fourier transform consume almost equal time.

In the interpolation step there is some communication between processors when at a certain point the actual interpolation is computed. The values needed in this step are distributed across different processors and for this method it is also not possible to

N	*Chirp z*	*Interpolation*	*Filtering and IFFT*	*Total*
32	0.015	0.011	0.014	0.048
64	0.041	0.016	0.015	0.079
128	0.088	0.038	0.033	0.163
256	0.183	0.120	0.112	0.422
512	0.390	0.486	0.407	1.287
1024	1.583	2.046	1.577	5.197

Table 4.2: Timing results in seconds on the CM-5 for different implementations of the Pasciak algorithm. The three major steps in the algorithm are shown, were the filtering step is combined with that of the two-dimensional inverse Fourier transform. The last column shows the total time needed for reconstructing.

select the layout of the arrays in such a way that no communication is involved. The communication burden, however, is not as strong as for filtered backprojection. The latter algorithm iterates over the different angles (and involves communication in each iteration) and the Pasciak algorithm does not iterate, so the communication step is performed only once.

When the timing results in Table 4.2 are compared to the timings for filtered backprojection (Table 3.1), it can be concluded that the Pasciak method is indeed faster than filtered backprojection.

5

The wavelet transform

The wavelet transform is a tool which is used in many different areas such as signal and image processing, numerical analysis and tomography. This chapter deals with the algorithm for computing the fast wavelet transform. Then it is explained how this algorithm can be parallelised on a CM-5.

5.1 Multiresolution analysis

Wavelets are functions derived by shifts in position and scale from a single function ψ called the *basic wavelet*. It is possible to construct orthonormal wavelet bases from a so-called *scaling function* ϕ, to which the basic wavelet ψ can be associated in a unique way. The function ϕ generates a so-called *multiresolution analysis*, represented by the basis functions $\{\phi_{j,k} : j,k \in \mathbb{Z}\}$ and $\{\psi_{j,k} : j,k \in \mathbb{Z}\}$, where $\phi_{j,k}(x) = 2^{-j/2}\phi(2^{-j}x - k)$ and $\psi_{j,k}(x) = 2^{-j/2}\psi(2^{-j}x - k)$. In the implementation of the wavelet transform, the scaling function ϕ is represented by the finite sequence $\{h_k\}$. Likewise, the function ψ is represented by the finite sequence $\{g_k\}$. For an orthonormal basis, the coefficients $\{g_k\}$ satisfy

$$g_k = (-1)^k h_{1-k}. \tag{5.1}$$

In the next sections Mallat's pyramid algorithm (Mallat 1989) will be outlined. The implementation of this algorithm is called the Fast Wavelet Transform (FWT).

5.1.1 Decomposition

Consider a function f represented by a data sequence $c^0 \in \ell^2(\mathbb{R})$ in the following way:

$$f = \sum_n c_n^0 \phi_{0,n}. \tag{5.2}$$

The sequence c^0 can be decomposed into a sequence of approximation coefficients c^j and detail coefficients d^j, $j = 1, \ldots, L$, with L the number of decomposition levels. The

sequences c^j and d^j are constructed by

$$c^j = \mathbf{H}c^{j-1}, \tag{5.3}$$
$$d^j = \mathbf{G}c^{j-1}, \tag{5.4}$$

where

$$(\mathbf{H}a)_k = \sum_n h_{n-2k}a_n, \tag{5.5}$$
$$(\mathbf{G}a)_k = \sum_n g_{n-2k}a_n. \tag{5.6}$$

An L-level decomposition can be graphically represented as in Fig. 5.1.

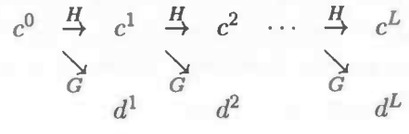


Figure 5.1: L-level wavelet decomposition.

Each arrow consists of a filtering operation after which only the elements with even index are retained (the downsampling step). The operators $\mathbf{H}$ and $\mathbf{G}$ involve to a low-pass and a high-pass filter, respectively. The result of the decomposition is the sequence $\{c^L, d^L, d^{L-1}, \ldots, d^2, d^1\}$, which has the same number of elements as the original sequence c^0.

If the length M of the sequence c^0 is a power of 2, the maximum number of levels in the decomposition is $^2\log M$. The algorithm for the L-level decomposition is given in Algorithm 3.

Algorithm 3 The wavelet decomposition algorithm for L-levels.

for $j = 1$ **to** $L - 1$ **do**
  for $k = 0$ **to** $2^j - 1$ **do**
    $c_k^j = \sum_n h_n c^{j-1}_{(n+2k) \mod 2^{L-j+1}}$
    $d_k^j = \sum_n g_n c^{j-1}_{(n+2k) \mod 2^{L-j+1}}$
  end for
end for

5.1.2 Reconstruction

The function f can be recovered from its L-level decomposition by

$$f = \sum_k c_k^L \phi_{L,k} + \sum_{j=1}^{L} \sum_k d_k^j \psi_{j,k}. \tag{5.7}$$

The sequences c_k^j and d_k^j can be computed recursively by

$$c^{j-1} = \tilde{\mathbf{H}} c^j + \tilde{\mathbf{G}} d^j, \tag{5.8}$$

where

$$(\tilde{\mathbf{H}} a)_k = \sum_n \tilde{h}_{k-2n} a_n, \tag{5.9}$$

$$(\tilde{\mathbf{G}} a)_k = \sum_n \tilde{g}_{k-2n} a_n. \tag{5.10}$$

The sequences $\{\tilde{h}_k\}$ and $\{\tilde{g}_k\}$ are the *dual* filters of $\{h_k\}$ and $\{g_k\}$, respectively, see e.g. Daubechies (1992). The L-level reconstruction can be graphically represented as in Fig. 5.2.

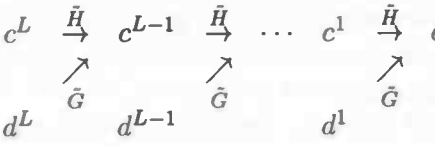


Figure 5.2: L-level wavelet reconstruction.

For the reconstruction, each arrow consists of an upsampling step by a factor of 2 (inserting zeros between the elements of the sequence) followed by a filtering operation; this combined operation is represented by the *dual* operators $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{G}}$. For an orthonormal basis, the filters for the forward and the backward transform are the same ($h = \tilde{h}$ and $g = \tilde{g}$). The algorithm for reconstructing a function from its L-level decomposition is given in Algorithm 4.

Algorithm 4 The wavelet reconstruction algorithm for L-levels.

for $j = L$ **downto** 1 **do**
  for $k = 0$ **to** $2^j - 1$ **do**
    if k even **then**
      $c_k^{j-1} = \sum_n h_{2n} c^j_{(k/2-n) \bmod 2^{L-j+1}} + \sum_n g_{2n} d^j_{(k/2-n) \bmod 2^{L-j+1}}$
    else {k odd}
      $c_k^{j-1} = \sum_n h_{2n+1} c^j_{((k-1)/2-n) \bmod 2^{L-j+1}} + \sum_n g_{2n+1} d^j_{((k-1)/2-n) \bmod 2^{L-j+1}}$
    end if
  end for
end for

5.1.3 Two-dimensional wavelet transform

Given a one-dimensional ϕ and ψ it is possible to construct a separable two-dimensional wavelet basis. The two-dimensional wavelet transform of a matrix can then be computed in a straightforward manner: apply the one-dimensional wavelet transform to each row and,

after that, to each column. Starting from a two-dimensional image c^0, the two-dimensional analogon of (5.3)–(5.4) are

$$c^j = \mathbf{H}_x\mathbf{H}_y c^{j-1}, \tag{5.11}$$
$$d^{(x)j} = \mathbf{G}_x\mathbf{H}_y c^{j-1}, \tag{5.12}$$
$$d^{(y)j} = \mathbf{H}_x\mathbf{G}_y c^{j-1}, \tag{5.13}$$
$$d^{(xy)j} = \mathbf{G}_x\mathbf{G}_y c^{j-1}. \tag{5.14}$$

The reconstruction is computed by

$$c^{j-1} = \tilde{\mathbf{H}}_x\tilde{\mathbf{H}}_y c^j + \tilde{\mathbf{G}}_x\tilde{\mathbf{H}}_y d^{(x)j} + \tilde{\mathbf{H}}_x\tilde{\mathbf{G}}_y d^{(y)j} + \tilde{\mathbf{G}}_x\tilde{\mathbf{G}}_y d^{(xy)j}. \tag{5.15}$$

The operator $\mathbf{H}_x$, for instance, only acts on the index x. Likewise, the other operators only act on the specified indexes. The decomposition which is obtained in this way is composed of a so-called *approximation* image and several so-called *detail* images. The approximation image c^j contains coarse scale information. The detail images $d^{(x)j}$ and $d^{(y)j}$ contain variations in the x-direction and y-direction, respectively (vertical and horizontal edges). The detail image $d^{(xy)j}$ contains variations in both x- and y-direction (corners).

5.2 Parallelisation

Since the algorithms for the forward and inverse wavelet transform are similar, only the forward transform will be considered in the following. It is clear from (5.5)–(5.6) that the one-dimensional wavelet transform would involve a lot of communication on a distributed memory parallel computer, like, for instance, the CM-5. This communication is due to the need to access elements in a with index $2k$ for computation of $(\mathbf{H}a)_k$. Holström (1995) suggests an algorithm that tries to minimise global communication.

5.2.1 One-dimensional transform

In the following discussion, only one step of the forward wavelet transform, i.e. the computation of c^j from c^{j-1}, is considered. An L-level decomposition can be obtained by repeating this step L times.

The idea is to localise computations using two auxiliary arrays, $\bar{c}$ and $\bar{d}$, where c^j and d^j are stored in the elements of $\bar{c}$ and $\bar{d}$ with even index. So, define

$$\bar{c}_{2k} = c^j_k, \qquad \bar{d}_{2k} = d^j_k, \qquad \text{for } k = 0, \ldots, M-1, \quad M = |c^j|, \tag{5.16}$$

where $|x|$ is the number of elements of the sequence x. The elements of $\bar{c}$ and $\bar{d}$ with odd index have an arbitrary value. Using this notation, one step of the forward transform (5.3)–(5.4) can be restated as

$$\bar{c}_k = \sum_n h_n c^{j-1}_{(n+k) \mod 2M}, \tag{5.17}$$
$$\bar{d}_k = \sum_n g_n c^{j-1}_{(n+k) \mod 2M}, \qquad n = 0, \ldots, 2M-1. \tag{5.18}$$

The above equations compute the convolution of the filters h and g with c^{j-1}. After convolving, the elements of the arrays $\bar{c}$ and $\bar{d}$ with even index contain the sequences c^j and d^j, respectively, according to (5.16). The next step is to downsample $\bar{c}$ and $\bar{d}$ in order to obtain c^j and d^j. This downsampling causes communication, but only once for each step of the transform. All computations can now be done locally, in contrast to the original algorithm where the computation of the convolutions causes communication for each element. The algorithm can be summarised as follows: compute the convolution, $\bar{c}$, of h and c^{j-1}. Likewise, compute the convolution, $\bar{d}$, of g and c^{j-1}. Next, downsample $\bar{c}$ and $\bar{d}$ in order to obtain c^j and d^j, respectively. For an L-level decomposition, repeat this convolution followed by downsampling L times. As the algorithm can be translated into CM-Fortran in a straightforward way, the code for computing c^j from c^{j-1} is shown in Algorithm 5.

Algorithm 5 CM-Fortran pseudo-code to compute c^j from c^{j-1}.

```
    ▷  N = length of filters, assumed to be even
    ▷  M = size of data vector c
i = N/2 - 1
c = cshift(c, dim = 1, shift = i)
c^f = c
c^b = c
c̄ = h_i c
do while (i < (N - 2))
  i = i + 1
  c^f = cshift(c^f, dim = 1, shift = 1)
  c^b = cshift(c^b, dim = 1, shift = -1)
  c̄ = c̄ + h_i c^f + h_{N-i-2} c^b
end do
c^f = cshift(c^f, dim = 1, shift = 1)
c^b = c^b + h_{i+1} c^f
c_(0:M/2-1) = c̄_(0:M-1:2)        ▷  Downsample
```

It is assumed that the size of the filter is even, which is the case for the Daubechies wavelets, otherwise the length can be made even by zero-padding the filter with one element.

The convolution algorithm described above can be classified as a *semi-systolic* convolution algorithm (Petkov 1993). It is called semi-systolic because the filter coefficients are broadcasted to all cells (in this case the cells correspond to the elements of $\bar{c}$). A filter coefficient which is input into the leftmost cell of the array is propagated to all other cells in the same clock period. The result $\bar{c}$ is computed in $N/2+1$ clock periods. The corresponding semi-systolic array for computing $\bar{c}$ is given in Fig. 5.3.

5.2.2 Two-dimensional transform

In the case of separable wavelet bases, the two-dimensional wavelet transform can be computed by applying Algorithm 5 first to the rows and then to the columns of a two-dimensional input array. This means that communication during computation in one

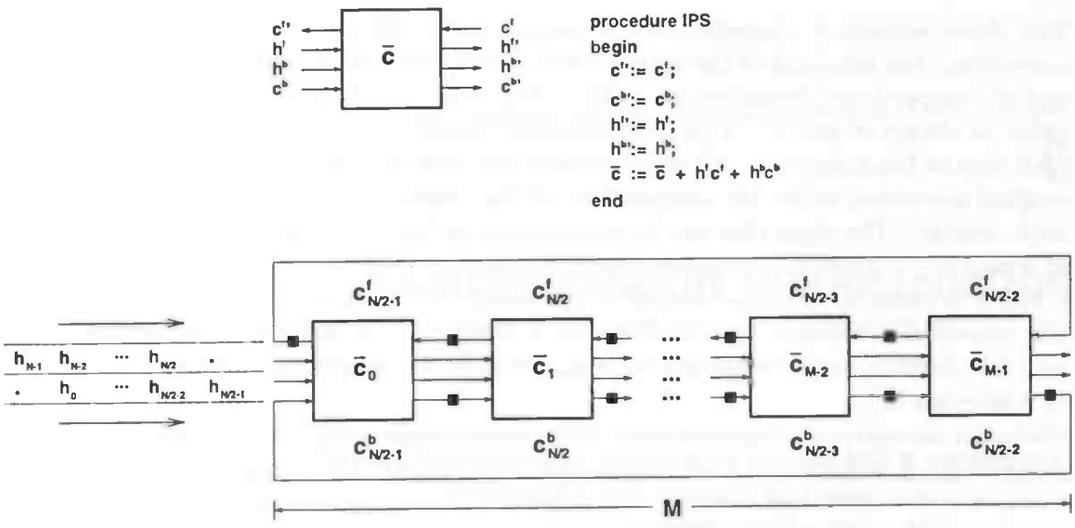


Figure 5.3: The semi-systolic array to compute $\bar{c}$ according to Algorithm 5. In each step, two of the filter coefficients of $\{h_k\}$, $k = 0, \ldots, N-1$, are broadcasted to all cells. The arrays c^f and c^b are shifted over $N/2 - 1$ positions before computation starts. This explains the odd numbering of the coefficients of c^f and c^b. Computation finishes after $N/2 + 1$ clock periods.

dimension can be avoided. If the dimension along which the transform is computed has all its values within a single processor, then the downsampling step causes no communication. To accomplish this on the CM-5, one would select a `:serial` layout in this dimension and a `:news` layout for the other dimension. The `:serial` specification assigns all elements in the chosen dimension to one processor, whereas the `:news` specification assigns elements to different processors. Thus, if the rows are given a serial layout and the columns a news layout, there will be no communication at all when the rows are transformed. The columns can be easily transformed by first transposing the array, then using the same algorithm as for the rows and transposing again afterwards. In this approach the communication appears when the arrays are transposed. For that purpose the transpose routine from the scientific library (Thi 1994) can be used, which is highly optimised for the CM-5.

5.3 Timing results

Timings of the above described algorithm were carried out for increasing values for the size, $M \times M$, of the input array. The size of the filters, N, was varied also. The results are reported in Table 5.1 and Fig. 5.4.

It is clear from these results that the fast wavelet transform is indeed fast. There is also a linear relation between the size of the data and the execution time, which was expected. This relation is independent of the filter size. For instance, the ratio between the run-time for $M = 256$ and $M = 512$ is about 1.6, which is equal for all filter sizes. This ratio is

	Filter size N		
M	2	10	20
64	0.021	0.054	0.091
128	0.037	0.084	0.135
256	0.067	0.138	0.214
512	0.121	0.226	0.351
1024	0.362	0.677	1.065

Table 5.1: Timing results in seconds for a full two-dimensional wavelet transform of an $M \times M$ array, i.e. $^2\log M$ levels.

about 3.0 for $M = 512$ and $M = 1024$.

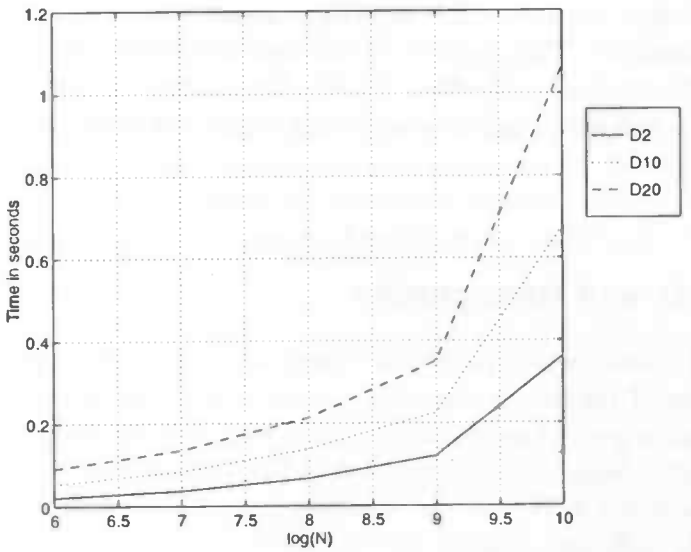


Figure 5.4: Comparison of execution time for different choices of the filter length and the matrix size.

Holström (1995) has done timings for a three-level two-dimensional wavelet transform. The filter which has been used by him was a Daubechies wavelet with 20 coefficients. For a 1024×1024 matrix his algorithm has an execution time of about 1.1 seconds, whereas the algorithm described above takes 0.86 seconds. It has to be noted however, that a straightforward comparison of execution times cannot really be done, because Holmström has used a 32-node CM-5 and in this report a 16-node CM-5 was used. Holmström also has suggested another algorithm in which communication during computation can be avoided altogether. However, after computing the transform, the elements of the array have to be reordered. It is not clear if the reordering step has been accounted for in his timings, but it seems that this is not the case. Therefore it is hard to draw any conclusions from a comparison of his and our results.

6

Multiresolution reconstruction

One of the applications of the wavelet transform is in tomography. This chapter deals with a multiresolution reconstruction algorithm which gives a wavelet decomposition of the object to reconstruct. The accuracy of this multiresolution method is compared with that of standard filtered backprojection. Finally, the implementation of the multiresolution algorithm on the CM-5 is considered and timing results are presented.

6.1 Wavelets and tomography

There are several reasons which justify the use of wavelets in the tomographic reconstruction problem. One of the topics currently studied is local tomography, where the aim is to reconstruct only a part of an image in such a way that the radiation exposure can be reduced significantly. Walnut (1992) was one of the first authors reporting on localisation of the Radon transform in the wavelet domain. These ideas were used in an implementation by Olson & DeStefano (1994), where radiation exposure was significantly reduced. Extensions and modifications to these methods have been made by Delaney & Bresler (1995) and Rashid-Farrokhi, Liu, Berenstein & Walnut (1995) and have led to successful local reconstruction algorithms. All authors just mentioned concentrate on using separable two-dimensional wavelet bases. Peyrin & Zaim (1996) propose non-separable wavelets and they claim that such wavelets give a more accurate reconstruction than separable wavelets. Another possible application of wavelets is denoising (Kolaczyk 1996) or a combination of multiscale reconstruction and denoising (Bhatia, Karl & Willsky 1993). Their methods are based on the idea that noise affects mostly the high frequency parts of an image and they have devised methods to remove noise from these parts of the reconstruction. In this, the wavelet decomposition provides a natural framework for dividing up an image in low and high resolution parts. Yet another application is for the storage of CT data. A multiresolution reconstruction can be used in data compressing methods (Antonini, Barlaud, Mathieu & Daubechies 1992) as most coefficients tend to be very small and therefore negligible.

In this report the focus is on obtaining a multiresolution reconstruction. The algorithm suggested by Delaney & Bresler (1995) is used for that purpose. They also show how their algorithm can be used in local tomography, but this is outside the scope of this

report. The multiresolution algorithm is an adaptation of the standard filtered backprojection algorithm. The differences arise in the use of a ramp filter which is now dependent on the projection angle. If projections are filtered with these modified filters and then backprojected, a wavelet decomposition of the object is obtained.

6.2 Multiresolution reconstruction algorithm

The Fourier transforms of the filters h and g at decomposition level j are defined by

$$H^j(\lambda) = \begin{cases} \prod_{q=0}^{j-1} H(2^q\lambda) & j = 1, \ldots, L-1, \\ 1 & j = 0 \end{cases} \tag{6.1}$$

$$G^j(\lambda) = \begin{cases} G(2^{j-1}\lambda)H^{j-1}(\lambda) & j = 1, \ldots, L-1, \\ 1 & j = 0 \end{cases} \tag{6.2}$$

where H and G represent the Fourier transforms of h and g, respectively, and L denotes the highest decomposition level. Using this representation of the filters h and g, the wavelet transform can be expressed in the frequency domain. Starting from an original two-dimensional image c^0 and its frequency domain representation C^0, the approximation image c^j is given by

$$c^j(m,n) = \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} C^0(\lambda_x, \lambda_y) \overline{H^j}(\lambda_x) \overline{H^j}(\lambda_y) e^{i(2^j m \lambda_x + 2^j n \lambda_y)} \, \mathrm{d}\lambda_x \, \mathrm{d}\lambda_y. \tag{6.3}$$

Now denote the Radon transform of the image c^0 by $\mathcal{R}c^0$ and a profile at angle θ by $\mathcal{R}_\theta$ as defined in Section 2.1. Then, using the Fourier slice theorem (Section 2.2), the approximation image c^j can be expressed by

$$c^j(m,n) = \frac{1}{4\pi^2} \int_0^{\pi} \int_{-\infty}^{\infty} \hat{\mathcal{R}}_\theta(\lambda) \overline{H^j}(\lambda \cos\theta) \overline{H^j}(\lambda \sin\theta) |\lambda| e^{i\lambda(2^j m \cos\theta + 2^j n \sin\theta)} \, \mathrm{d}\lambda \, \mathrm{d}\theta. \tag{6.4}$$

In operator form, after discretisation, this can be written as

$$c^j(m,n) \approx \mathcal{R}_d^{\#} \mathcal{K}_{H^j} \mathcal{R}c^0(2^j m, 2^j n). \tag{6.5}$$

The operator $\mathcal{R}_d^{\#}$ is the discrete backprojection operator, see (3.6), and $\mathcal{K}_{H^j}$ a discrete filter operator defined by

$$\mathcal{K}_{H^j} \mathcal{R}c^0(m,\theta) = k_{H^j,\theta} * \mathcal{R}_\theta(m). \tag{6.6}$$

The filter $k_{H^j,\theta}$, called the *modified ramp filter*, is a combination of the ramp filter and the filters used in the wavelet transform, and its frequency response is

$$K_{H^j,\theta}(\lambda) = W_b(\lambda) \overline{H^j}(\lambda \cos\theta) \overline{H^j}(\lambda \sin\theta), \tag{6.7}$$

where $W_b(\lambda)$ is the Fourier transform of the ramp filter. The only difference between (6.6) and the filtering (3.5) of standard filtered backprojection is the use of a different ramp filter,

which is dependent on θ. Summarising, the approximation image c^j can be computed by filtering with the modified ramp filter $k_{Hj,\theta}$ and backprojecting only those points on the reconstruction grid with coordinates that are a multiple of 2^j.

The detail images can be found by taking different filters in the operator $\mathcal{K}$. Define the filter operators $\mathcal{K}^x_{Gj}$, $\mathcal{K}^y_{Gj}$ and $\mathcal{K}^{xy}_{Gj}$ analogous to (6.5) and (6.6) by

$$d^{(p)j}(m,n) \approx \mathcal{R}_d^{\#}\mathcal{K}^p_{Gj}\mathcal{R}c^0(2^j m, 2^j n), \tag{6.8}$$

$$\mathcal{K}^p_{Gj}\mathcal{R}c^0(m,\theta) = k^p_{Gj,\theta} * \mathcal{R}_\theta(m), \qquad p \in \{x,y,xy\}. \tag{6.9}$$

The frequency responses of the modified ramp filters $k^p_{Gj,\theta}$ are given by

$$K^x_{Gj,\theta}(\lambda) = W_b(\lambda)\overline{G^j}(\lambda\cos\theta)\overline{H^j}(\lambda\sin\theta), \tag{6.10}$$

$$K^y_{Gj,\theta}(\lambda) = W_b(\lambda)\overline{H^j}(\lambda\cos\theta)\overline{G^j}(\lambda\sin\theta), \tag{6.11}$$

$$K^{xy}_{Gj,\theta}(\lambda) = W_b(\lambda)\overline{G^j}(\lambda\cos\theta)\overline{G^j}(\lambda\sin\theta). \tag{6.12}$$

Having the approximation image c^j and the detail images $d^{(x)j}$, $d^{(y)j}$ and $d^{(xy)j}$ at level j, the approximation image c^{j-1} can be found by computing one step of the pyramid algorithm of Mallat (see Chapter 5) for two dimensions. Then by computing the detail images at level $j-1$, the approximation image at level $j-2$ can be found, etcetera, until c^0 is found.

The algorithm can be summarised as follows:

1. Compute four sets of filtered projections at resolution level j by filtering $\mathcal{R}_\theta$ with the modified ramp filters $k_{Hj,\theta}$, $k^x_{Gj,\theta}$, $k^y_{Gj,\theta}$ and $k^{xy}_{Gj,\theta}$.

2. For each set of filtered projections, evaluate the backprojection at every coordinate that is a multiple of 2^j. This results in the approximation image c^j and the detail images $d^{(x)j}$, $d^{(y)j}$ and $d^{(xy)j}$.

3. Compute the approximation image c^{j-1} by combining the approximation image and detail images computed in the previous step according to the pyramid algorithm of Mallat.

4. Repeat step 1 to 3 until the final image c^0 is found.

6.3 Complexity

Assume the Radon transform $\mathcal{R}f(\theta,s)$ is available for (θ_j, s_l), $j = 1,\ldots,p$, $l = -q,\ldots,q$ and reconstruction is performed on a reconstruction grid of size $(2q+1)\times(2q+1)$. Then the number of computations for backprojecting using the algorithm above is the same as for the standard backprojection algorithm. This is due to the fact that the total number of points on the reconstruction grid does not increase, only the set of filtered projections is not the same for each part of the reconstruction. So the order of complexity for backprojecting is $O(pq^2)$ and this is independent of the number of decomposition levels. The latter

argument, independence of the number of levels, does not hold for the filtering step. For the standard algorithm, the filtering step takes about $O(pq \log q)$ operations. Therefore, in the multiresolution algorithm, a number of $O(pq \log q)$ operations is needed for each decomposition level. The total number of levels can be ${}^2\log(2q+1)$, leading to a total number of operations of $O(pq \log^2 q)$. Thus the algorithm described above will spend more time filtering, but the total complexity is still $O(q^3)$—due to the complexity of backprojection—if the optimal relation $p = \pi q$ is satisfied.

6.4 Accuracy

The accuracy of the multiresolution reconstruction algorithm was tested on the standard Shepp-Logan phantom. Projection data were generated for $p = 256$ and $q = 127$, and reconstruction was computed on a 255×255 grid. The wavelet used in this experiment was a Daubechies least asymmetric wavelet with 10 coefficients (Daubechies 1992). One level of the pyramidal decomposition is shown in Fig. 6.1. This image consists of the approximation image c^1 and the detail images $d^{(x)1}$, $d^{(y)1}$ and $d^{(xy)1}$ from which the original image c^0 can be reconstructed by computing one step of the inverse two-dimensional wavelet transform.

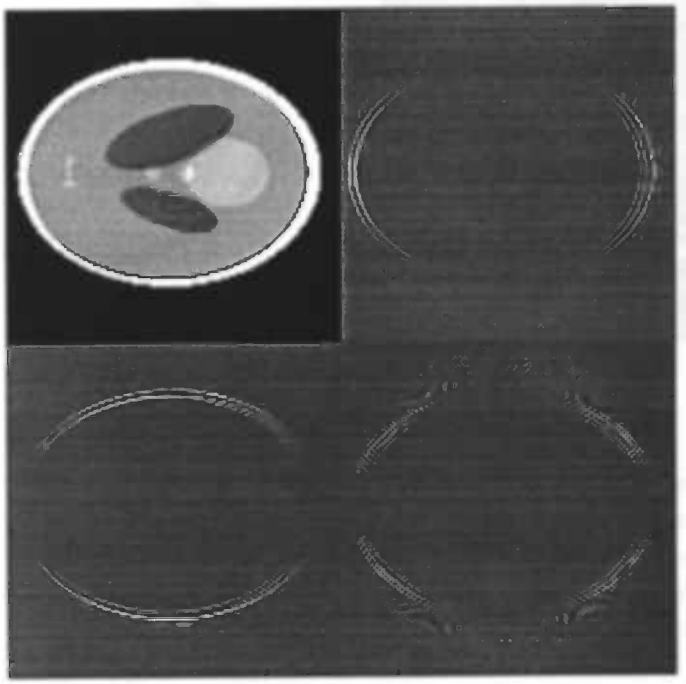
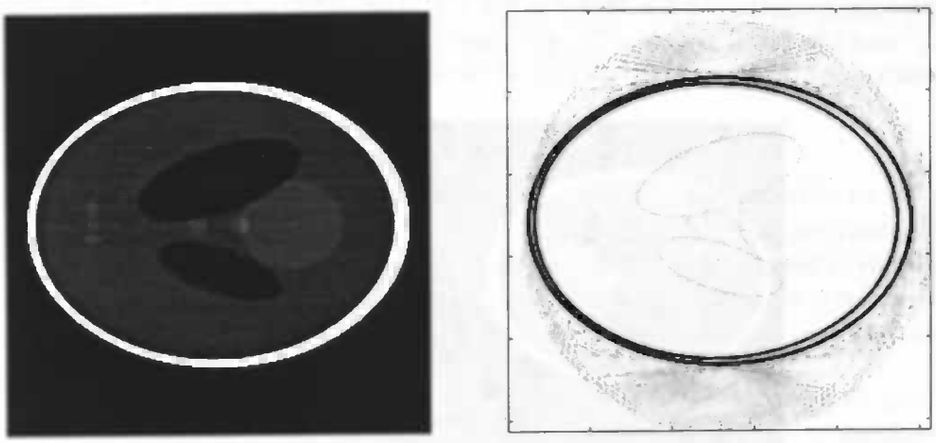


Figure 6.1: Multiresolution decomposition of the Shepp-Logan phantom. The image shows the first level of the decomposition, i.e. c^1 in the upper left corner, $d^{(y)1}$ in the upper right, $d^{(x)1}$ in the lower left and $d^{(xy)1}$ in the lower right corner. The gray values of each of the four parts in this image were scaled in order to make details visible. By applying one step of the two-dimensional inverse wavelet transform, the reconstruction at level 0, c^0, is obtained (see Fig. 6.2).

The reconstructed image is shown in Fig. 6.2 on the left, and the difference with the original phantom on the right. The same reconstruction artefacts as in standard filtered backprojection are visible. However, in the multiresolution reconstruction there are also artefacts visible which are caused by the choice of the wavelet. The circular convolutions used in the wavelet transform will cause edge effects near the sides of the image. This can also be observed in Fig. 6.2, where a 'wrap-around' effect appears at sides of the image. This problem can be solved by zero-padding the projection data with twice the support of the wavelet. These edge effects only appear at the sides of the image, therefore they do not influence the reconstruction quality around the centre of the reconstruction, so we choose not to zero-pad the projection data. The part of the phantom inside the outer two ellipses is reconstructed with a quality comparable to that of standard filtered backprojection. This follows from looking at a part of the $y = 0$ line of the phantom and of the reconstruction, as plotted in Fig. 6.3. The reconstruction is as accurate as the reconstruction obtained by standard filtered backprojection, which can be observed in Fig. 6.4. The two plots shown in the figure coincide, meaning that multiresolution reconstruction has the same quality as the standard method.



(a) (b)

Figure 6.2: Multiresolution reconstruction of the phantom (a) and the difference with the original (b). The projections were generated for $p = 256$ and $q = 127$ and reconstruction was performed on a 255×255 grid. The wavelet used was a Daubechies least asymmetric wavelet with 10 coefficients. The grey values in the right image were scaled in order to make reconstruction artefacts visible. The effects visible at the edge of the disk are due to the size of the wavelet.

The reconstruction algorithm was also tested using a different wavelet. In this case a Daubechies biorthogonal symmetric wavelet (Daubechies 1992) was used, also with 10 coefficients. The difference image with the original phantom is shown in Fig. 6.5.

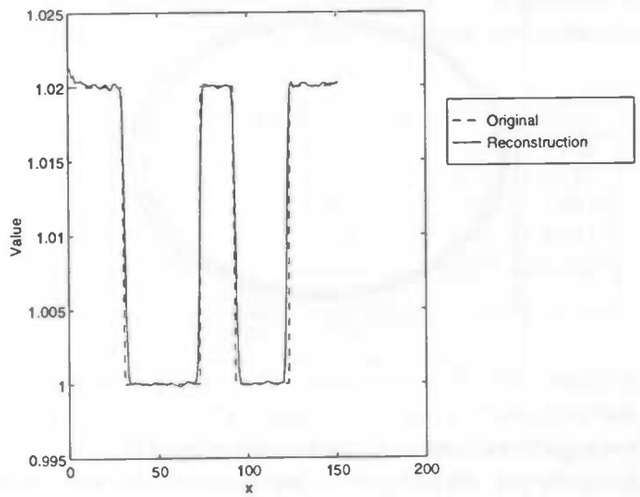


Figure 6.3: A numerical comparison of part of the $y = 0$ line of the multiresolution reconstruction with the true values of the phantom. Projection data were generated for $p = 256$ and $q = 127$. Reconstruction was performed on a 255×255 grid. The reconstruction is comparable to that of the standard filtered backprojection algorithm as can be seen in Fig. 6.4.

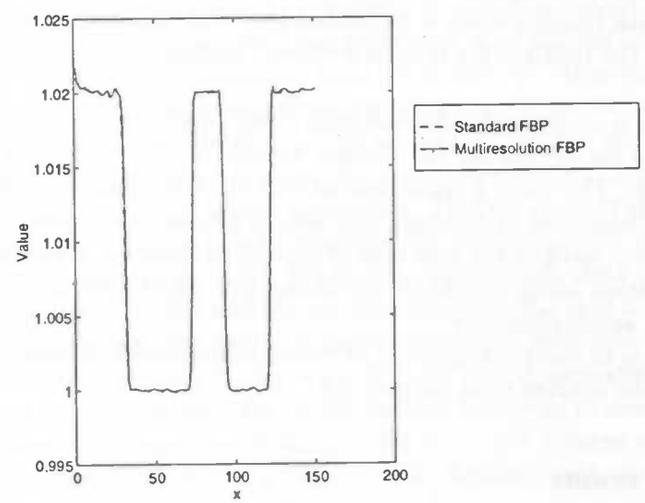


Figure 6.4: Numerical comparison of the multiresolution reconstruction and the reconstruction obtained from standard filtered backprojection for a part of the $y = 0$ line. As can be seen, the reconstructions coincide showing the multiresolution algorithm to be as accurate as standard filtered backprojection.

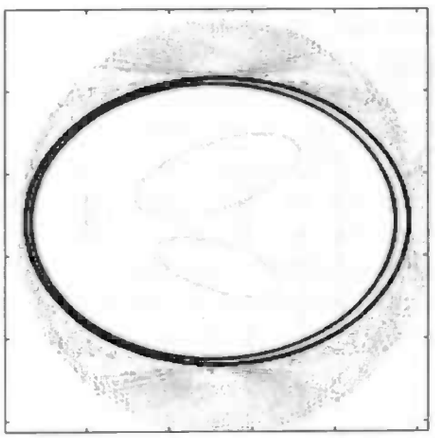


Figure 6.5: Difference image of the multiresolution reconstruction of the phantom with the original. The projections were generated for $p = 256$ and $q = 127$ and reconstruction was performed on a 255×255 grid. The wavelet used was a Daubechies biorthogonal symmetric wavelet with 10 coefficients. The grey values in the right image were scaled in order to make reconstruction artefacts visible. The effects visible at the edge of the disk are due to the size of the wavelet.

6.5 Implementation

The multiresolution reconstruction algorithm was implemented on a CM-5. In the current implementation only one decomposition level is reconstructed. This does not affect the general case however, because all levels require the same operations, but it simplifies the interpretation of the timings which were carried out. The implementation is rather straightforward, because the routines for standard filtered backprojection and wavelet transforms already exist.

The first step is to generate the modified ramp filters according to (6.7) and (6.10)–(6.12). This calls for evaluating the Fourier transform of h and g for a step-size which is dependent on θ. The chirp z-transform from Section 4.3 can be used for this purpose. These filters are then used to compute four sets of filtered projections.

In the next step, each of the four sets of filtered projections is backprojected resulting in one approximation image and three detail images. The backprojection algorithm from Section 3.4.5 can accomplish this.

The last step is to compute one step of the two-dimensional inverse wavelet transform. This is done by the routine from Section 5.2.

6.5.1 Timing results

The algorithm described above was timed on a CM-5 for different sizes of the input. The wavelet which was used is the Daubechies least asymmetric wavelet of length 10. Furthermore, the number of rays and angles were chosen to be the same, which is expressed by the parameter N. The size of the reconstruction grid was $N \times N$ in each case. The results are reported in Table 6.1. There are no timings for $N = 1024$, because the algorithm consumes to much memory for generating the modified ramp filters. However, for a given

geometry and wavelet these filters can be computed in advance and stored, so this problem does not affect the multiresolution method too much. It is possible to make a different implementation for generating the filters which consumes less memory at the expense of run-time.

N	*Filtering*	*Backprojection*	*IWT*	*Total*
32	0.167	0.095	0.022	0.284
64	0.317	0.219	0.039	0.575
128	0.628	0.625	0.075	1.328
256	1.284	2.397	0.153	3.834
512	2.628	12.041	0.350	15.019
1024	–	–	–	–

Table 6.1: Timing results in seconds on the CM-5 for different sizes of the input, i.e. $p = N$, $q = N/2-1$ and reconstruction on an $N \times N$ grid. The wavelet used was a Daubechies least asymmetric wavelet with 10 coefficients. The second column shows the time needed for filtering with the modified ramp filters. This includes the time needed for generation of the filters. The third and fourth column show the time for backprojection and two-dimension inverse wavelet transform. The last column shows the total reconstruction time.

As expected the time needed for filtering increases in comparison with the standard filtered backprojection algorithm. This is due to the fact that four sets of filtered projections are computed. The chirp z-transform has to be evaluated four times — twice for h and twice for g. The number of points for which it is evaluated is doubled with respect to the chirp z-transform in the Pasciak algorithm (Section 4.2). This means that a factor of about 8 increase in run-time can be expected for the results reported for evaluating the chirp z-transform in the Pasciak algorithm (Table 4.2), which is indeed the case.

Looking at the complexity of the backprojection step, as explained in Section 6.3, there should not be any increase in computation time for a given N. However, due to the implementation of the backprojection algorithm, the time needed for reconstruction is more dependent on the number of angles then on the number of rays. In Section 3.4.5 it is explained how the use of lookup tables reduces the communication, but it is also pointed out that for each angle a lookup table has to be created. This means that the multiresolution reconstruction algorithm will need more run-time for backprojection than the standard filtered backprojection algorithm. Take, for instance, projection data consisting of measurements of N rays over N angles and choose the reconstruction grid to be $N \times N$. Then, for a one-level decomposition, the multiresolution algorithm reconstructs four images of size $N/2 \times N/2$, but with a number of angles equal to N. So the reconstruction time should be about four times the time needed for reconstructing an image of size $N/2 \times N/2$ from projection data taken at N rays over N angles. For $N = 256$ it turns out that there is a factor of 5.1 increase and for $N = 512$ a factor of 4.8. Although not exactly a factor of 4, these results confirm the general observations made above.

6.5.2 Conclusion

The multiresolution reconstruction algorithm as implemented on the CM-5 takes more time to compute than standard filtered backprojection. Although both methods have

equal complexity, the multiresolution algorithm will need more time for filtering, due to the modifying of the ramp filters. On the CM-5 there is the added problem of having a distributed memory, which complicates the backprojection step. The results suggest that if one is only interested in obtaining a multiresolution reconstruction (and not, for instance, in local tomography), it is wiser to use the standard filtered backprojection algorithm and compute a wavelet transform afterwards. Take, for instance, $N = 512$. The standard filtered backprojection algorithm takes about 8.5 seconds to compute (Table 3.1). A full wavelet transform for filter size 10 takes about 0.2 seconds to compute (Table 5.1). So, the total time for obtaining a full wavelet decomposition takes about 8.7 seconds, whereas the multiresolution algorithm takes about 12 seconds for just one decomposition level.

7

Discussion

In the previous chapters, it was shown how several different reconstruction algorithms could be parallelised on a CM-5. The wavelet transform was described and also implemented on the CM-5. A combination of filtered backprojection and the wavelet transform was used to derive and implement a multiresolution reconstruction algorithm. The conclusions of the previous chapters are briefly summarised in this chapter. Finally, a few ideas for further research are given.

7.1 Conclusions

Of the reconstruction algorithms considered, the filtered backprojection algorithm turns out to be the most difficult to parallelise. This is mainly due to the distributed memory of the CM-5, which is the source of a lot of communication between the processors. This communication problem is partly solved by using lookup tables for the filtered projections, but still most of the time is spent on communication. Using these lookup tables, the implementation on the CM-5 has led to a fast reconstruction algorithm: it is possible to reconstruct a 512×512 image in about 8.7 seconds. The reconstruction quality of filtered backprojection is excellent, which was already known from the literature.

Direct Fourier reconstruction using the Pasciak algorithm is easier to parallelise on the CM-5. The problem of communication is still present, but the burden is not as strong. The time needed for reconstruction is much less than for filtered backprojection, not only expressed in time complexity, but also in actual run-time. The quality of the reconstruction, however, is not as good. Because there is quite a gap between the run-time of filtered backprojection and the Pasciak algorithm (a 512×512 reconstruction takes about 1.3 seconds), some extra time can be spent for improvement of the interpolation method.

The implementation of the fast wavelet transform on the CM-5 has led to an efficient algorithm. The aim was to keep communication low and perform computations locally on each processor. For the two-dimensional transform this was done by choosing an appropriate layout of the input array across processors so that communication only occurred once, namely by transposing the input array. The fast wavelet transform has a linear time complexity which is reflected in the timing results.

An analysis of the time complexity of the multiresolution reconstruction algorithm showed that the time complexity is the same as for filtered backprojection. The actual

run-time however is much longer. For example, reconstruction of a 512×512 image takes about 15.0 seconds. Partly this is due to the expected increase in run-time for filtering, but backprojection also showed an increase. This can be explained from the distributed memory of the CM-5 and the particular implementation of the discrete backprojection. The results show that, if one is interested in a multiresolution reconstruction, it is wiser to do standard filtered backprojection followed by a wavelet decomposition, instead of using the multiresolution reconstruction algorithm.

7.2 Ideas for further research

As explained above and in the previous chapters, the distributed memory of the CM-5 poses some problems. It was suggested that the use of a shared memory computer could improve the actual run-time of the reconstruction algorithms. It would be interesting to implement the algorithms which were described here on the Cray J932.

Another very interesting field is local tomography, which was already glanced at in Chapter 6. The multiresolution algorithm can be adapted for that purpose. Others have suggested that a one-level decomposition suffices to perform strict local tomography. This means that only data in the region of interest is needed to reconstruct a high quality image, and therefore the radiation exposure is reduced. With the standard filtered backprojection algorithm this is not possible.

This report deals with two-dimensional reconstructions. If a three-dimensional reconstruction is wanted, the standard procedure is to use a stack of two-dimensional reconstructions. However, it is also possible to reconstruct a three-dimensional image by using cone-beam scanning methods. This method calls for different reconstruction algorithms. The combination with three-dimensional imaging methods, such as volume visualisation or even virtual environments opens a large field of interesting problems to investigate.

Bibliography

Antonini, M., Barlaud, M., Mathieu, P. & Daubechies, I. (1992), 'Image coding using wavelet transform', *IEEE Transactions on Image Processing* **1**(2), 205–220.

Bhatia, M., Karl, W. C. & Willsky, A. S. (1993), A wavelet-based method for multiscale tomographic reconstruction, Technical Report LIDS-P-2182, Stochastic Systems Group, Lab. for Information and Dec. Systems, MIT, Cambridge, MA 02139.

Daubechies, I. (1992), *Ten Lectures on Wavelets*, number 61 *in* 'CBMS-NSF regional conference series in applied mathematics', Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

Deans, S. R. (1983), *The Radon Transform and Some of its Applications*, John Wiley and Sons inc.

Delaney, A. H. & Bresler, Y. (1995), 'Multiresolution tomographic reconstruction using wavelets', *IEEE Transactions on Image Processing* **4**(4), 1–16.

Herman, G. T. (1980), *Image Reconstruction from Projections: the Fundamentals of Computerized Tomography*, Academic Press.

Holström, M. (1995), 'Parallelizing the fast wavelet transform', *Parallel Computing* **21**, 1837–1848.

Kak, A. C. & Slaney, M. (1988), *Principles of Computerized Tomographic Imaging*, IEEE Press, New York.

Kolaczyk, E. D. (1996), An application of wavelet shrinkage to tomography, *in* A. Aldroubi & M. Unser, eds, 'Wavelets in Medicine and Biology', CRC Press, Inc., chapter 3, pp. 77–92.

Mallat, S. G. (1989), 'A theory for multiresolution signal decomposition: the wavelet representation', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**, 674–693.

Natterer, F. (1986), *The Mathematics of Computerized Tomography*, B.G. Teubler and J. Wiley.

Nussbaumer, H. J. (1982), *Fast Fourier Transform and Convolution Algorithms*, Springer-Verlag.

Olson, T. & DeStefano, J. (1994), 'Wavelet localization of the Radon transform', *IEEE Transactions on Signal Processing* **42**(8), 2055–2067.

Pasciak, J. E. (1973), 'A note on the Fourier algorithm for image reconstruction', preprint, Applied Mathematics Department, Brookhaven National Laboratory, Upton, New York.

Petkov, N. (1993), *Systolic Parallel Processing*, Elsevier Science Pusblishers B.V.

Peyrin, F. & Zaim, M. (1996), Wavelet transform and tomography: Continuous and discrete approaches, *in* A. Aldroubi & M. Unser, eds, 'Wavelets in Medicine and Biology', CRC Press, Inc., chapter 9, pp. 209–230.

Rabiner, L. R. & Gold, B. (1975), *Theory and Application of Digital Signal Processing*, Prentice-Hall Inc.

Rashid-Farrokhi, F., Liu, K. J. R., Berenstein, C. A. & Walnut, D. (1995), Wavelet-based multiresolution local tomography, Technical Report ISR TR 95-73, Institute for Systems Research, University of Maryland.

Rowland, S. W. (1979), Computer implementation of image reconstruction formulas, *in* G. T. Herman, ed., 'Image Reconstruction from Projections: Implementation and Applications', Vol. 32 of *Topics in Applied Physics*, Springer-Verlag, pp. 9–79.

Schulte, J. (1994), Fourierrekonstruktion in der computer-tomographie, Master's thesis, Westfälischen Wilhelms-Universität Münster.

Shepp, L. A. & Logan, B. F. (1974), 'The Fourier reconstruction of a head section', *IEEE Transactions on Nuclear Science* **NS–21**, 21–43.

Stark, H. H., Woods, J. W., Paul, I. & Hingorani, R. (1981), 'Direct Fourier reconstruction in computer tomography', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **ASSP-29**(2), 237–245.

Thi (1994), *CMSSL for CM Fortran*, 3.2 edn.

Uhlenbeck, G. E. (1925), 'Over een stelling van Lorentz en haar uitbreiding voor meerdimensionale ruimten', *Physica, Nederlands Tijdschrift voor Natuurkunde* (5), 423–428.

Walnut, D. (1992), Applications of Gabor and wavelet expansions to the Radon transform, *in* J. B. et al, ed., 'Probabilistic and Stochastic Methods in Analysis, with Applications', Kluwer Academic Publishers, pp. 187–205.