SIAM J. APPL. MATH.
Vol. 60, No. 2, pp. 437–462

A FAST AND ACCURATE MULTILEVEL INVERSION OF THE RADON TRANSFORM*

ACHI BRANDT†, JORDAN MANN†, MATVEI BRODSKI†, AND MEIRAV GALUN†

Abstract. A number of imaging technologies reconstruct an image function from its Radon projection using the convolution backprojection method. The convolution is an $O(N^2 \log N)$ algorithm, where the image consists of $N \times N$ pixels, while the backprojection is an $O(N^3)$ algorithm, thus constituting the major computational burden of the convolution backprojection method. An $O(N^2 \log N)$ multilevel backprojection method is presented here. When implemented with a Fourier-domain postprocessing technique, also presented here, the resulting image quality is similar or superior to the image quality of the classical backprojection technique.

Key words. Radon transform, inversion of the Radon transform, computed tomography, convolution backprojection, multilevel, Fourier-domain postprocessing

AMS subject classifications. 92C55, 44A12, 65R10, 68U10

PII. S003613999732425X

1. Background.

Reconstruction of a function of two or three variables from its Radon transform has proven vital in computed tomography (CT), nuclear magnetic resonance imaging, astronomy, geophysics, and a number of other fields [13]. One of the best known reconstruction algorithms is the convolution backprojection method (CB), which is widely used in commercial CT devices [13] (with rebinning for divergent-beam projections [18]). Recently, it has been applied to spotlight-mode synthetic aperture radar image reconstruction [14, 23] in which the conventional method is the direct Fourier method (DF), i.e., Fourier-domain interpolation followed by two-dimensional (2-D) FFT [21].

Originally, CB was preferred to DF since the former provided better images [18, 20]. However, since the backprojection part of CB raises the computational complexity of the method to $O(N^3)$, while DF's complexity is $O(N^2 \log N)$, there has been interest in finding an effective implementation of DF [1, 16, 19, 24, 25, 27, 29, 30, 31]. We present here an $O(N^2 \log N)$ backprojection algorithm based on a multiscale approach, which, when used as the second half of CB, reduces the complexity of CB to $O(N^2 \log N)$. Empirical results indicate that the multiscale backprojection, together with the postprocessing step described in the second half of this paper, produces images of quality equal to or better than that of the classical backprojection algorithm and better than the quality of DF. Furthermore, multilevel methods can generally be applied under weaker regularity requirements than Fourier methods can. For example, the algorithm presented here could be adjusted to provide different resolutions for different parts of the reconstruction, even if the Radon data are equally spaced.

1.1. Relation to other ideas, publications, and patents.

The fast inverse Radon transform methods described in the present article were published first as a patent application [12], then as a report [11].

*Received by the editors July 11, 1997; accepted for publication (in revised form) October 27, 1998; published electronically December 28, 1999.
http://www.siam.org/journals/siap/60-2/32425.html

†Department of Applied Mathematics and Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel (mabrandt@weizmann.weizmann.ac.il, mannj@mc.com, mbrodski@ml.com, meirav@wisdom.weizmann.ac.il). Address correspondence to M. Galun.

In a fundamental sense, this work is part of a long-range and systematic development of multiscale computational methods of all kinds (see recent survey [3]), including multigrid (or "multilevel") solvers for partial differential equations and integro-differential equations, whose development started in the early 1970s. Of these, the one most related to the Radon transform work is the development of a multilevel solver for the standing-wave (Helmholtz) equation (see [9]; the algorithmic idea first appeared in [7, section 3.2], with more details in [6]). The wave representation on each level of this solver has a certain *spatial* resolution as well as *orientational* (or "dual space" or "momentum") resolution. *The coarser the spatial resolution at a given level, the finer its orientational resolution.* In quantum mechanics this representation idea is a manifestation of Heisenberg's uncertainty principle. In the context of *integral transforms* with oscillatory kernels, a similar multilevel representation was developed in [4, section 5].

This multilevel spatial-orientational representation idea next led to an $O(N^2 \log N)$ algorithm for calculating *all* the line integrals in an $N \times N$ grid (i.e., all integrals over straight lines, the values on each line being interpolated from the grid. By "all" we mean enough of them so that any other straight-line integral, at any possible location, orientation, and length, can immediately be interpolated from the calculated ones. See [8]; an early description of the algorithm has appeared in [15]). The original motivation for this algorithm was the task of fast detections of all edges and fibers in a picture, as a step in multiscale image processing (see also [28]). However, as a byproduct—a part of its output—the algorithm produces the Radon transform. For the particular purpose of the Radon transform, an algorithm with a similar overall structure and complexity, but different in important details, has been independently developed by Götz and Druckmüller [17] (and brought to our attention by a referee for the present article).

Next, using a related multilevel spatial-orientational organization, we developed the fast backprojection method used in the present article. The painstaking examination of the obtained pictures did not satisfy us, however. Although much faster than the classical CB, the new reconstruction yielded blurrier images: see, for example, the pictures and point-spread functions shown below. We realized, however, that the produced point-spread functions, suffering each from $O(\log N)$ random local averaging, all approximate the same Gaussian function. This led to the Fourier-domain postprocessing described below in section 5, which for little relative extra cost yields much sharper reconstructions.

The reconstructed images may be further improved and the postprocessing work reduced by using yet another type of multiscale process (see section 7 below).

Most recently, a new work has appeared [22] which presents the same fast multiscale backprojection algorithm. It lacks, however, the postprocessing part, without which blurrier pictures are produced.

2. Mathematical preliminaries. Let f be an absolutely integrable function of two variables. $P_\theta f(t)$, the Radon transform of the function at angle θ, is then defined by

$$P_\theta f(t) = \int_{-\infty}^{\infty} f(t\cos\theta - \tau\sin\theta, t\sin\theta + \tau\cos\theta)\, d\tau. \tag{2.1}$$

This is the integral of f along the line

$$x\cos\theta + y\sin\theta = t$$

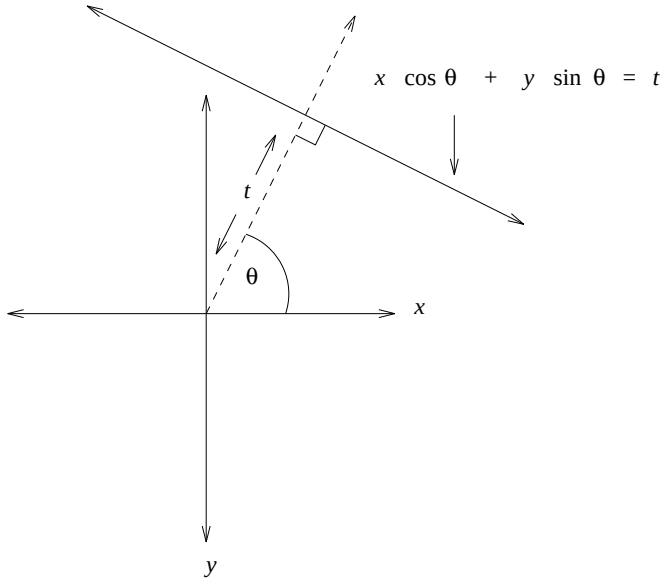


FIG. 2.1. *The Radon transform of f at angle θ and position t, denoted $P_\theta f(t)$. This is the integral of f on the line $x\cos\theta + y\sin\theta = t$.*

in the x-y plane. (See Figure 2.1.) In many imaging technologies, such as CT, the challenge is to compute f, or an approximation to it, given samples of $P_\theta f(t)$ for finitely many values of θ and t.

To understand CB, which is meant to solve this problem, we will need the Fourier transform. For $f \in L^1(\mathbb{R}^d)$, the Fourier transform $\hat{f}$ of f is defined by

$$\hat{f}(u) = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} f(x)\, e^{ix\cdot u} dx$$

for all $u \in \mathbb{R}^d$. We note that for any θ, $f \in L^1(\mathbb{R}^2)$ implies that $P_\theta f$ is well defined and belongs to $L^1(\mathbb{R}^1)$, and thus the Fourier transform of $P_\theta f$ is also well defined. Many methods for reconstructing a function from its Radon transform, including CB, rely on the Radon slice theorem [18], which states that

$$\widehat{P_\theta f}(\rho) = \sqrt{2\pi}\hat{f}(\rho\cos\theta, \rho\sin\theta). \tag{2.2}$$

Thus, if $f(x, y)$ is the image function, the 2-D Fourier transform of the image can be sampled by sampling the Fourier transform of the Radon transform of the image for different values of ρ and θ.

3. CB method.

3.1. Theory. One way to reconstruct f from its Radon transform is to compute $\hat{f}$ using (2.2) for as many values of ρ and θ as possible and then perform a 2-D inverse Fourier transform. Algorithms based on this idea are called DF. CB, however, uses the following approach. The Fourier transform inversion formula [13] can be written in polar coordinates as

$$f(x, y) = \frac{1}{2\pi} \int_0^\pi \int_{-\infty}^\infty \hat{f}(\rho\cos\theta, \rho\sin\theta)\, e^{i\rho(x\cos\theta + y\sin\theta)} |\rho|\, d\rho\, d\theta.$$

By the Radon slice theorem (2.2), this equals

$$\frac{1}{(2\pi)^{3/2}} \int_0^{\pi} \int_{-\infty}^{\infty} \widehat{P_\theta f}(\rho)\, e^{i\rho(x\cos\theta + y\sin\theta)} |\rho|\, d\rho\, d\theta.$$

In most applications, values of $P_\theta f$ or $\widehat{P_\theta f}$ are obtained from the physical data. Using the $\widehat{P_\theta f}$ values, we may compute a function $\tilde{P}_\theta f$, defined by

$$\tilde{P}_\theta f(u) = \int_{-\infty}^{\infty} \widehat{P_\theta f}(\rho)\, e^{i\rho u} |\rho|\, d\rho, \tag{3.1}$$

and then

$$f(x,y) = \frac{1}{(2\pi)^{3/2}} \int_0^{\pi} \tilde{P}_\theta f(x\cos\theta + y\sin\theta)\, d\theta, \tag{3.2}$$

which means that $f(x,y)$ can be reconstructed from the $\tilde{P}_\theta f$ values, given these values for all θ.

3.2. Implementation and computational complexity. Of course, in actual applications, $P_\theta f(t)$ is known for only a finite number of values of θ and t, and f is to be computed at a finite set of points, or pixels, (x,y). We will assume here that the pixels (x_i, y_j) to be computed lie inside a circle inscribed in an $N \times N$ square grid, with a distance d between adjacent pixel centers in both the horizontal and vertical directions; that there are Q angles θ_j at which $P_\theta f$ is known; and that at each such angle θ_j, $P_\theta f(t)$ is known at N evenly spaced values t_k of t, the difference between consecutive values t_k, t_{k+1} being d as well. Since we limit the region of reconstruction to the inscribed circle, N samples of the Radon transform in any direction are enough to cover the region. We will further assume that Q is a power of 2, though the algorithm can clearly be generalized to the case of arbitrary Q.

Computation of $\tilde{P}_\theta f$ is called the convolution step because multiplying two functions together in the Fourier domain and taking the inverse Fourier transform of the result is associated with convolution [13]. The Fourier transform and inverse Fourier transform required in (3.1) for this step are approximated in practice using the discrete Fourier transform (DFT) and its inverse. Since there are N values of $P_\theta f(t)$ for each value of θ, the computation of $\tilde{P}_\theta f$ for a single value of θ is $O(N \log N)$ using the FFT, and since there are Q values of θ, the entire convolution step is $O(QN \log N)$. If, as is usually assumed, $Q \approx N$, then the cost of the convolution step is $\mathrm{O}(N^2 \log N)$.

The integral with respect to θ in (3.2) is replaced in practice by summation over all available values of θ. Since there are $O(N^2)$ pixels at which summation is required, and Q values of θ, this final step of computing the right side of (3.2) is clearly $O(QN^2)$, or, if $Q \approx N$, $O(N^3)$. Since the contribution of $\tilde{P}_\theta f$ to a pixel (x,y) depends only on the quantity $x\cos\theta + y\sin\theta$, $\tilde{P}_\theta f(u)$ is said to be "smeared" or "backprojected" along the line $x\cos\theta + y\sin\theta = u$ for each value of u. That is, it is added to every pixel lying on that line. Therefore, this final step is called the backprojection step. Although the concept of backprojection suggests an algorithm with a different loop structure than the one suggested by (3.2)—namely, (3.2) suggests an outer loop to go through all the pixels and an inner loop to go through the values of θ, whereas the backprojection concept suggests the reverse—the backprojection concept does not change the computational complexity. Although DF is $O(N^2 \log N)$, commercial CT scanners have traditionally used CB, despite its higher computational complexity, in part because of the superior quality of the images it produces [18].

4. Multiscale backprojection. It follows from the analysis in section 3.2 that an $O(N^2 \log N)$ implementation of the backprojection step of CB would lower the complexity of the entire CB algorithm to $O(N^2 \log N)$. Such an implementation is presented here.

4.1. Basic concept. The multilevel backprojection method relies on the following reasoning. In the standard backprojection algorithm, a single $N \times N$ grid is used to sum the appropriate values of $\tilde{P}_{\theta_j} f$, for $j = 1, \ldots, Q$, for each pixel. One can, however, start with Q grids g_j^0, $j = 1, \ldots, Q$, and for each j project $\tilde{P}_{\theta_j} f$ only onto g_j. Later, the Q different grids can be added together pixelwise to produce the final image. This approach can be used to save computation in the following way.

In what follows, for all i and j, f_j^i will be a function of two continuous variables, and g_j^i will be a "grid" containing a finite set of samples of f_j^i. For all j, let

$$f_j^0(x, y) = \tilde{P}_{\theta_j} f(x \cos\theta_j + y \sin\theta_j),$$

i.e., f_j^0 is the function of two variables resulting from the backprojection of $\tilde{P}_{\theta_j} f$ along the lines $x \cos\theta_j + y \sin\theta_j = t$, for different values of t, in the x-y plane. Clearly, f_j^0 does not vary along such lines, and therefore, when collecting samples of f_j^0 to form the grid g_j^0, it is sufficient to compute and store one point value in the grid for each of the N values of t at which $\tilde{P}_{\theta_j} f(t)$ has been computed, rather than computing and storing N^2 point values. (See Figure 4.1.) In fact, g_j^0 is merely a one-dimensional (1-D) array containing all the computed values of $\tilde{P}_{\theta_j} f$, and is compiled by the convolution part of the CB algorithm. Thus, even though we are beginning the backprojection with Q grids instead of one, the total number of point values stored is $O(NQ)$. The distance between adjacent sample points of f_j^0 for any j is d, the distance between adjacent sample points of $\tilde{P}_{\theta_j} f$.

Eventually, the various initial functions f_j^0 must be added together to form the final image. As addition is commutative and associative, they may be added together in any order, and our method chooses an order that reduces the number of necessary computations. Since Q is even by assumption, our first level of grid merges will consist of adding together pairs of functions f_j^0, f_k^0 whose corresponding values of θ are close to each other, and thus the direction in which f_j^0 is constant is close to the direction in which f_k^0 is constant. The sum of these two functions, f_l^1, will vary slowly in the "in-between" direction, and thus it will only be necessary to store samples of f_l^1 in g_l^1 at a handful of widely spaced points along each line parallel to the "slow" direction in the x-y plane.

Specifically, we may assume without loss of generality that $0 \le \theta_j < \pi$ for all j and that the θ_j are ordered in such a way that $\theta_j < \theta_{j+1}$ for all $j = 1, \ldots, Q-1$. For all $j = 1, \ldots, Q$, let $\theta_j^0 = \theta_j$. For all k from 1 to $Q/2$, we will add together the functions f_{2k-1}^0 and f_{2k}^0 to obtain f_k^1. Since for any j, f_j^0 does not vary in the direction $(-\sin\theta_j, \cos\theta_j)$, it follows that f_{2k-1}^0 and f_{2k}^0 vary slowly in the direction $(-\sin\theta_k^1, \cos\theta_k^1)$, where θ_k^1 is the average of θ_{2k-1} and θ_{2k}. Therefore f_k^1 varies slowly in the $(-\sin\theta_k^1, \cos\theta_k^1)$ direction, and when forming g_k^1, it is only necessary to store samples of f_k^1 at a few widely spaced points along each line in that direction. The spacing between sample points in the fast direction, i.e., $(\cos\theta_k^1, \sin\theta_k^1)$, will again be d. The samples are computed from g_{2k-1}^0 and g_{2k}^0 by interpolation.

By similar reasoning, for $l = 1, \ldots, Q/4$, we may add f_{2l-1}^1 and f_{2l}^1 together to obtain f_l^2 in such a way that f_l^2 varies slowly in the direction $(-\sin\theta_l^2, \cos\theta_l^2)$, where

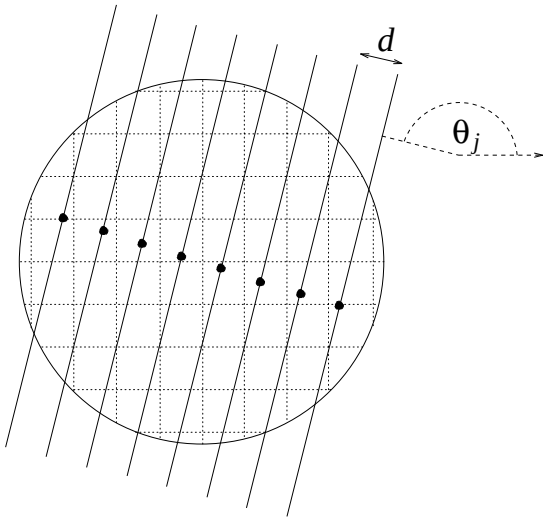


FIG. 4.1. *Construction of the zeroth-level grid g_j^0, which contains the results of a single backprojection, that of $\tilde{P}_{\theta_j} f$. Although g_j^0 is meant to contain the samples of the function f_j^0 defined on a disk in the plane, and the disk contains $O(N^2)$ pixels, shown in the figure as the intersections of dotted lines, it is sufficient to compute and store in g_j^0 samples of f_j^0 taken at only $O(N)$ points, shown in the figure as black dots, since f_j^0 only varies in the direction $(\cos\theta_j, \sin\theta_j)$.*

θ_l^2 is the average of θ_{2l-1}^1 and θ_{2l}^1, and consequently, when computing and storing samples of f_l^2 to form g_l^2, it is only necessary to store values of f_l^2 at a few widely spaced points in the direction $(-\sin\theta_l^2, \cos\theta_l^2)$. The spacing between sample points in the fast direction will again be d. f_l^2 will not vary as slowly in its "slow" direction as f_{2l}^1 varies in *its* "slow" direction, and therefore g_l^2 will require more sample points in its slow direction than g_{2l}^1 requires in its slow direction. Nevertheless, there are half as many second-level grids g_l^2 as there are first-level grids g_k^1, and as we will show, the total number of point values that must be computed and stored for all the grids at any one level of merges is $O(N^2)$. (See Figure 4.2.) Samples of f_l^2 to be stored in g_l^2 are computed from g_{2l-1}^1 and g_{2l}^1 by interpolation.

Continuing in this way, we may construct a sequence of levels of grids and functions. The functions at the ith level are constructed from pairs of functions at the $(i-1)$th level in such a way that each ith level function varies slowly in a certain direction and only a few samples along lines in that direction need to be stored in the function's grid. At the $\log_2 Q$th level, there is only one grid, and this grid represents the sum of all the original grids g^0, that is, the sum of all the backprojections of the $\tilde{P}_{\theta_j} f$. This grid is therefore the resulting reconstruction. Since $O(N^2)$ operations are needed to build the grids at each level, the overall cost of the algorithm is $O(N^2 \log Q)$.

4.2. Computing sample point spacing in the "slow" direction. As explained in section 4.1, the low computational complexity of the algorithm depends on the judicious choice of sample point spacing, and hence, the number of sample points, in a grid's slow direction. This spacing is chosen based on the following reasoning.

Since at every merge level i, two consecutive functions f_{2k-1}^{i-1} and f_{2k}^{i-1} from the previous level are merged to form f_k^i, it follows that 2^i consecutive original functions f_j^0 were merged and remerged to obtain f_k^i. We will refer to these 2^i original functions as

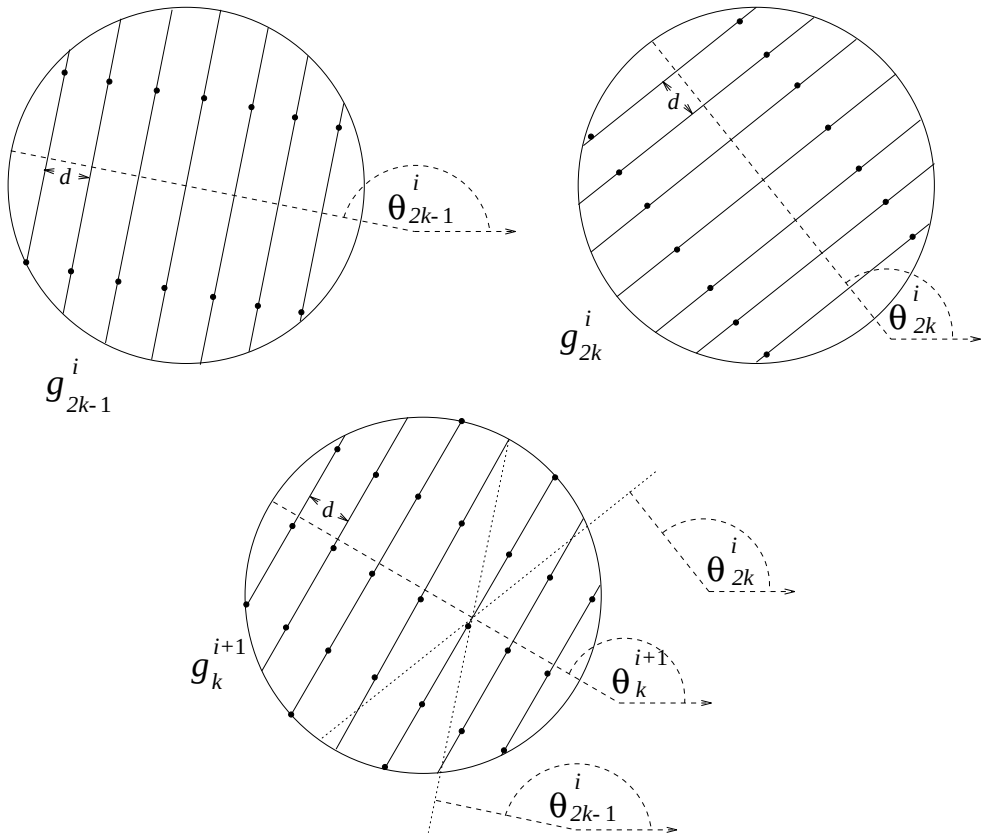


FIG. 4.2. *Merge of g^i_{2k-1} and g^i_{2k} to form g^{i+1}_k. The slow directions of f^i_{2k-1}, f^i_{2k}, and f^{i+1}_k are $(-\sin\theta^i_{2k-1}, \cos\theta^i_{2k-1}), (-\sin\theta^i_{2k}, \cos\theta^i_{2k})$, and $(-\sin\theta^{i+1}_k, \cos\theta^{i+1}_k)$ respectively, where θ^{i+1}_k is the average of θ^i_{2k-1} and θ^i_{2k}. Sample points are represented in the diagrams by black dots. g^{i+1}_k contains more samples in its slow direction than g^i_{2k-1} and g^i_{2k} do in theirs. The distance between samples in the fast direction, indicated in the diagrams by the distance between adjacent solid lines, is always d.*

In each diagram, the solid lines are parallel to the slow direction. In the bottom diagram, finely dotted lines are parallel to the respective slow directions of f^i_{2k-1} and f^i_{2k}. The slow direction of f^{i+1}_k is the average of those two slow directions.

the merged original functions. We note that the slow directions of any two consecutive original functions f^0_j and f^0_{j+1} (actually, since these are original functions, these are not just slow directions but constant directions) differ by π/Q radians, and it follows from this that the slow directions of the first and last of the merged original functions differ by $(2^i - 1)\pi/Q$ radians. It is trivial to show that the slow direction of f^i_k is halfway between the slow directions of the first and last merged original functions, and thus it differs by no more than $(2^i - 1)\pi/(2Q)$ from the slow direction of any of the merged original functions.

For any f^i_k, the fast direction is perpendicular to the slow direction and the spacing in the fast direction is d. Ideally, we would like to have samples of f^i_k at just enough points to be able to compute f^i_k at any other point by interpolation. Therefore, we wish to choose spacing in the slow direction of the function f^i_k in such a way that interpolating a value of f^i_k between two sample points A and B adjacent to each other

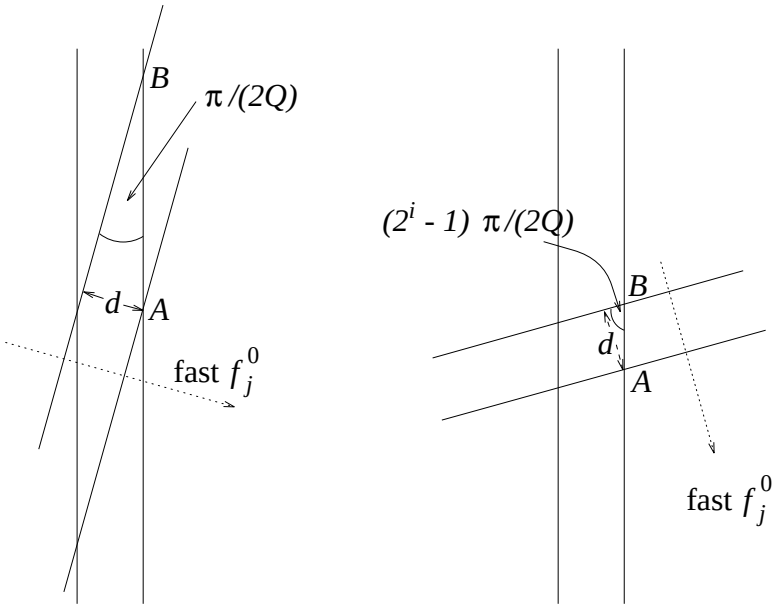


FIG. 4.3. *Choice of spacing in slow direction. Left, first merge level. Right, arbitrary merge level i. At any merge level i, the difference between the slow direction (vertical lines) of the function f_k^i, currently being formed, and the slow direction (solid diagonal lines) of any of the merged original functions is never greater than $(2^i - 1)\pi/(2Q)$. The fast direction of the merged original function is perpendicular to its slow one and its spacing in that direction is d. The distance in that direction between adjacent sample points A and B of f_k^i should not exceed d.*

on a line parallel to the slow direction would incur an error no greater than the error incurred by interpolating a value of any of the merged original functions between A and B. The spacing in the slow direction should therefore be such that if one selects a coordinate axis in the fast direction of any of the merged original functions, then the distance along this coordinate axis between A and B will be no more than d. It can be seen from Figure 4.3 that the desired distance between adjacent points in the slow direction of f_k^i should therefore be

$$\frac{d}{\sin((2^i - 1)\pi/(2Q))}. \tag{4.1}$$

4.3. Computational complexity. Since the diameter of the support of the reconstruction is Nd, the number of points necessary along a single line of f_k^i in its slow direction is Nd divided by the sampling interval, or

$$N \sin((2^i - 1)\pi/(2Q)).$$

We will assume that if this number is not an integer then the lowest integer greater than this number will be used. Since the sample spacing in the fast direction is always d, it follows that for any i and j, there are N lines parallel to the slow direction along which f_j^i will be sampled, so the total number of points in g_j^i is

$$N\lceil N \sin((2^i - 1)\pi/(2Q))\rceil.$$

Also, there are $Q/2^i$ different grids g_j^i to be computed (i.e., at the ith level of merges, j takes the values $1, \ldots, Q/2^i$). It follows from this that the total number of samples to be computed and stored at the ith level of merges is

$$\left\lceil N \sin\left((2^i - 1)\frac{\pi}{2Q}\right)\right\rceil \left(\frac{NQ}{2^i}\right) < \left(N \sin\left(\frac{2^i\pi}{2Q}\right) + 1\right)\left(\frac{NQ}{2^i}\right)$$

$$< \left(N\frac{2^i\pi}{2Q} + 1\right)\left(\frac{NQ}{2^i}\right) = \frac{\pi}{2}N^2 + \frac{NQ}{2^i}.$$

The sum of this for all levels of merges, i.e., for $i = 1, \ldots, \log_2 Q$, is less than

$$\frac{\pi}{2}N^2 \log_2 Q + NQ.$$

If, as is normally assumed, $Q \approx N$, then the order of the backprojection is $N^2 \log N$.

4.4. Practical considerations. For $i = 1, \ldots, \log_2 Q$, samples of f_l^i are computed from g_{2l-1}^{i-1} and g_{2l}^{i-1} by interpolation, but there will generally be some points where f_l^i is to be sampled that are not surrounded by sampling points of f_{2l-1}^{i-1} or of f_{2l}^{i-1}, and extrapolation must be performed there instead of interpolation. Computing a few samples of f_l^i just outside the disk in which the image is to be reconstructed will reduce the number of points at which extrapolation is necessary at the $(i+1)$th level, but will in itself require extrapolation unless a sufficient number of points outside the disk were computed at the $(i-1)$th level. A scheme with the same computational complexity is conceivable in which, at each level of grid merging, sample points outside the disk are chosen in such a way that extrapolation is never required, but we did not use such a scheme in our implementation.

In our implementation, we used one more than the

$$\lceil N \sin((2^i - 1)\pi/2Q))\rceil$$

points per line in the slow direction prescribed in section 4.3 so that the first and last sampling points on each line would be at or beyond the boundary of the disk supporting the reconstruction. This does not change the computational complexity.

We have found empirically that image quality can be improved significantly by doubling the number of samples per Radon projection by interpolation, with or without doubling the number of projections (i.e., the number of angles at which projections are computed) by interpolation. A cheaper way to improve image quality is to sample f_i^j for all i and j at no fewer than five points in the slow direction, even when (4.1) implies that fewer points are necessary in the slow directions in the first few levels of merges. However, these adjustments are less necessary when the postprocessing correction method described below is applied.

5. Postprocessing. Images resulting from the multiscale backprojection algorithm, as described up to this point, are somewhat blurred due to the many interpolations necessary. As shown below, the point spread function of the algorithm is wider than that of the classical backprojection. (Throughout this section, when we speak of the point-spread function of a given backprojection technique, we are referring to the point-spread function of the entire Radon projection-convolution-backprojection process.) In this section, we describe an $O(N^2 \log N)$ Fourier-domain correction of

the image which greatly reduces the width of the point spread function of the multiscale backprojection and enhances the resulting images. A similar correction can be performed after the classical backprojection, but with less of an improvement, and images produced by the multiscale method with Fourier-domain correction are at least as good qualitatively as those produced using the classical backprojection with Fourier-domain correction.

5.1. Basic concept. If the point-spread function of a given tomography method were shift invariant, then the obvious correction would be to divide the Fourier transform of the reconstruction by that of the point-spread function and to take the inverse Fourier transform of the result. However, the point-spread functions of both the classical and multiscale backprojection methods vary slightly over the image. Our postprocessing correction consists of dividing the Fourier transform of the image by a Gaussian which approximates the Fourier transforms of the point spread functions obtained at various points in the image. The width of the Gaussian is chosen in such a way as to optimize this approximation. As we will show, this technique is more effective for the multiscale backprojection than for the classical one, as the point-spread functions obtained for the multiscale method can be more closely approximated by a Gaussian.

5.2. Determination of Gaussian width σ_0. For each of the two methods under consideration (the classical and multiscale backprojection methods), we wished to find the 2-D Gaussian that in some sense fit a selection of point responses better than any other 2-D Gaussian. The 2-D Gaussians are of the form

$$e^{-(x^2+y^2)/\sigma^2},$$

where the width $\sigma > 0$ of the Gaussian may be chosen arbitrarily. In our context, x and y may be regarded as pixel coordinates (i.e., the x coordinates of horizontally adjacent pixels differ by 1, and the y coordinates of vertically adjacent pixels differ by 1), and the goal was to choose the value σ_0 of σ for which the resulting Gaussian best fit a *selection* (defined in section 5.3) of point responses in the following sense.

The point responses were all represented by real-valued matrices. From each such matrix, the 7×7 submatrix with the peak of the point response at its center was extracted. Let A be the *normalized* sum of these submatrices. Let the indices i and j of A run from -3 to 3. Let

$$m_{ij}(\sigma) = e^{-(i^2+j^2)/\sigma^2}.$$

For each method, σ_0 is defined as the value of σ which minimizes

$$\sum_{i,j=-3}^{3} (A_{ij} - m_{ij}(\sigma))^2.$$

5.3. Selection of point responses for computation of σ_0. We assume that backprojection is performed on a square grid of pixels or on a circular central region of this square. The computation of the Radon projections of the image to be represented by this grid, and the convolution and backprojection of the Radon projections onto the grid, can be perfomed in such a way that the entire sequence of operations is symmetric with regard to the horizontal and vertical axes of the grid and its two diagonals. In such a case, a point response anywhere in the grid is identical, up to

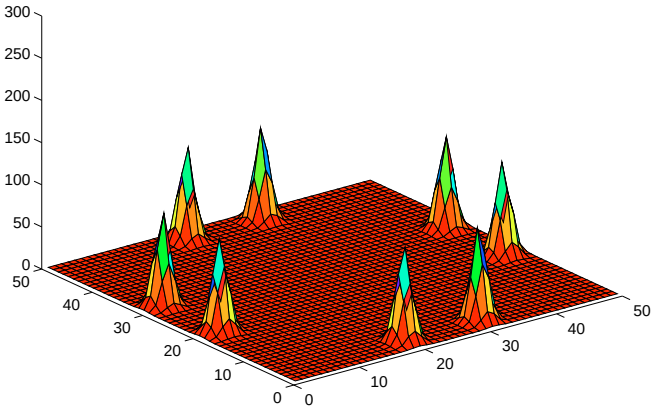


FIG. 5.1. *Eight-fold symmetry of the operations. A point response of the Radon transform-CB sequence is identical, up to rotation and reflection, to as many as seven other point responses.*

rotation and reflection, to as many as seven other point responses, whose positions are obtained from the position of the original point response by reflection across one or more of the axes of symmetry of the operations. (See Figure 5.1.) It follows that when selecting positions on the grid at which to compute point responses for the purpose of computing σ_0, one may restrict the positions to a single octant of the grid and compute the point responses at those positions, and then the point responses at the corresponding positions in the other seven octants can be obtained by rotating and reflecting the computed point responses.

In our tests, point responses were computed at 15 randomly selected points in a single octant, all inside a circle inscribed in the square grid of pixels, and the point responses at the corresponding points in the remaining octants were obtained by rotation and reflection of the original 15 point responses, as explained above.

Point responses and the resulting value of σ_0 depend on the process by which the Radon projections are obtained. In our research, Radon projection data were computed mathematically, but an apparatus implementing the method described here might obtain the projection data from physical measurements. In any case, for any such apparatus, σ_0 should be computed from point responses generated by Radon data obtained from the source from which the apparatus will obtain the Radon data in practice. The eightfold symmetry exploited for the purpose of our research may not hold for all systems.

6. Results. Figures 6.1–6.16 below were generated to compare the classical and multiscale backprojection schemes. Radon data for both methods were generated from 256×256 or 512×512 grayscale images, where the number of projections is the same as the linear size of the image. Our simulations involved two methods for calculating the Radon transform. In one method, to get the integral of the image along a line for the purpose of sampling the Radon transform at a given angle, the image is "sampled," using bilinear interpolation from neighboring pixels, at equally spaced points along the line, and the image samples are summed to form the integral. The distance between sample points is the same as the distance between pixels. This method was used in Figures 6.9–6.16. Another method for calculating the Radon transform was used in Figures 6.1–6.8. The line integral at a given angle with distance t from the origin is the sum over the intersection areas of the strip (the area between the given line and

the preceding line with distance $t-d$ from the origin) with the square pixel multiplied by the intensity value of that pixel.

For both types of backprojection, convolution was implemented simply by multiplying by $|\rho|$ in the discrete Fourier-domain (i.e., the range of the DFT), without tapering of $|\rho|$ for high values of ρ.

Image quality for both types of backprojection was found to improve when the number of samples per Radon projection was doubled using nearest-neighbor interpolation; i.e., every sample in the projection was repeated.

Referring again to (3.2), note that the classical backprojection requires that

$$\tilde{P}_{\theta_j} f(x\cos\theta_j + y\sin\theta_j) \tag{6.1}$$

be computed for every pixel (x, y) in the output image and every angle θ_j at which the Radon transform is available. In practice, for all j, $\tilde{P}_{\theta_j}(u)$ is only computed at a finite set of points u_k and $x\cos\theta_j + y\sin\theta_j$ will generally not be equal to any of them, thus requiring an interpolation scheme. In the classical backprojection whose results are shown in Figures 6.9, 6.10, and 6.13, (6.1) was computed by *linear interpolation* from the values of $\tilde{P}_{\theta_j}(u)$ at neighboring points. In the classical backprojection whose results are given in Figures 6.2 and 6.6, (3.2) is computed for every pixel (x, y) as the sum of the areas of intersection of strips with this square pixel, each area being multiplied by the corresponding $\tilde{P}_{\theta_j}(u)$, where the "upper" edge of the strip is at distance u from the origin.

As explained in section 4.1, the multiscale backprojection requires that at every merge level i, f_l^i for various l be computed from f_{2l-1}^{i-1} and f_{2l}^{i-1}. Of course, the values of f_{2l-1}^{i-1} and f_{2l}^{i-1} are only known at the points where samples were taken and stored in g_{2l-1}^{i-1} and g_{2l}^{i-1}, which means that interpolation will generally be necessary to compute f_l^i at a given point. In the implementation of the multiscale backprojection whose results are shown here, this interpolation was *bilinear*. Experiments with lower order interpolation, i.e., constant interpolation, yield worse results in the sense that the obtained picture is more noisy, although the noise amplitude is small relatively to the intensity level, approximately two percent.

As explained in section 5, our postprocessing correction consists of multiplying the Fourier transform of the image by a filter which is the Fourier transform of the delta function divided by a Fourier transform of a Gaussian with the selected σ_0. Practically, we suppress smoothly the highest frequencies of the filter. The postprocessed image is the inverse Fourier transform of this multiplication. In our presentations, the image is the absoulute value of this result.

Figures 6.1–6.4 include the Shepp–Logan "head" phantom [26], so called because of its use in testing the accuracy of reconstruction algorithms for their ability to reconstruct cross sections of the human head with computerized tomography. The image is composed of ten ellipses with different sizes and directions. Figure 6.1 shows the original 512×512 image from which Radon data were computed for the purpose of testing different variations of CB. Figure 6.2 shows the results of CB with classical backprojection. Figure 6.3 shows the results of CB with multiscale backprojection. Figure 6.4 shows the results of CB with multiscale backprojection, with doubling of the data by nearest-neighbor interpolation.

Figures 6.5–6.8 contain the Schomberg–Timmer phantom [27]. The phantom attempts to mimic a slice of the human brain. Figure 6.5 shows the original 512×512 image from which Radon data were computed. The description of Figures 6.6, 6.7, and 6.8 is, respectively, the same as the description for Figures 6.2, 6.3, and 6.4.

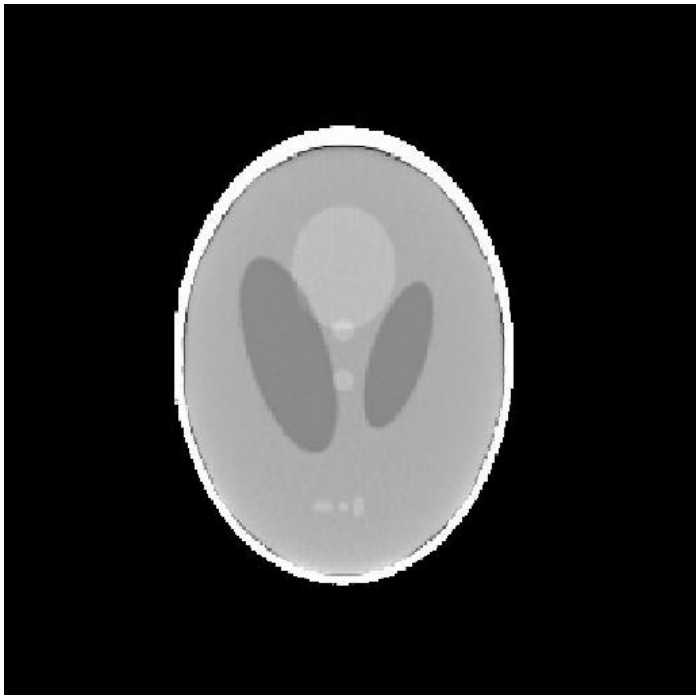


FIG. 6.1. *Shepp–Logan,* 512×512*: original image.*

FIG. 6.2. *Shepp–Logan,* 512×512*: CB,* $O(n^3)$ *version.*

FIG. 6.3. *Shepp–Logan,* 512×512*: multilevel CB.*

FIG. 6.4. *Shepp–Logan,* 512×512*: multilevel CB, doubled data.*

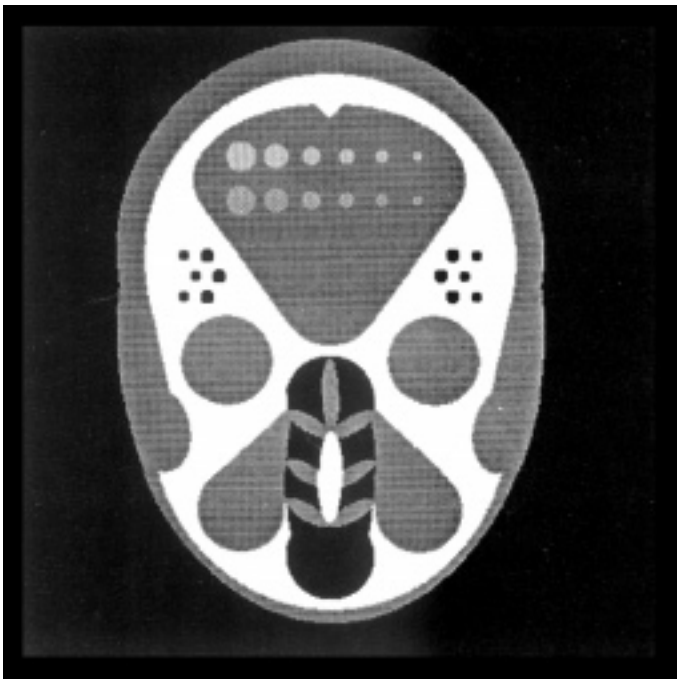


FIG. 6.5. *Schomberg–Timmer,* 512×512*: original image.*

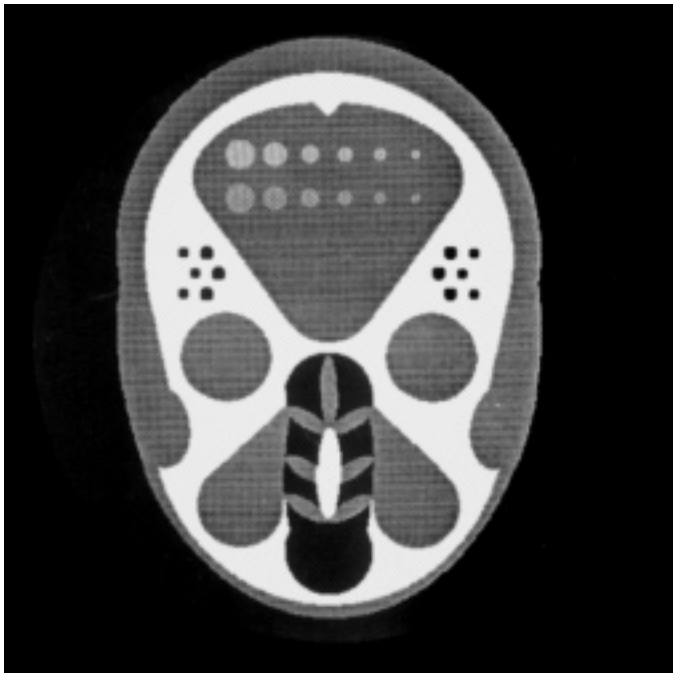


FIG. 6.6. *Schomberg–Timmer,* 512×512*: CB,* $O(n^3)$ *version.*

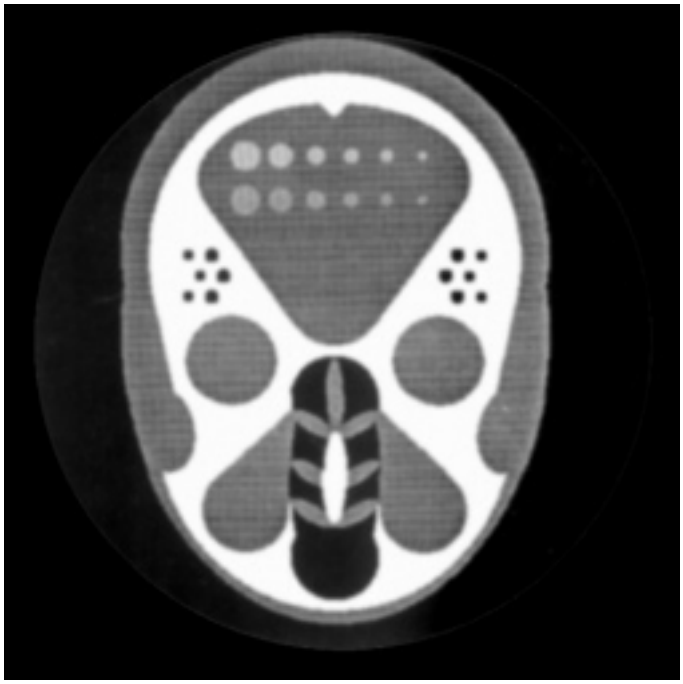


FIG. 6.7. *Schomberg–Timmer,* 512×512*: multilevel CB.*

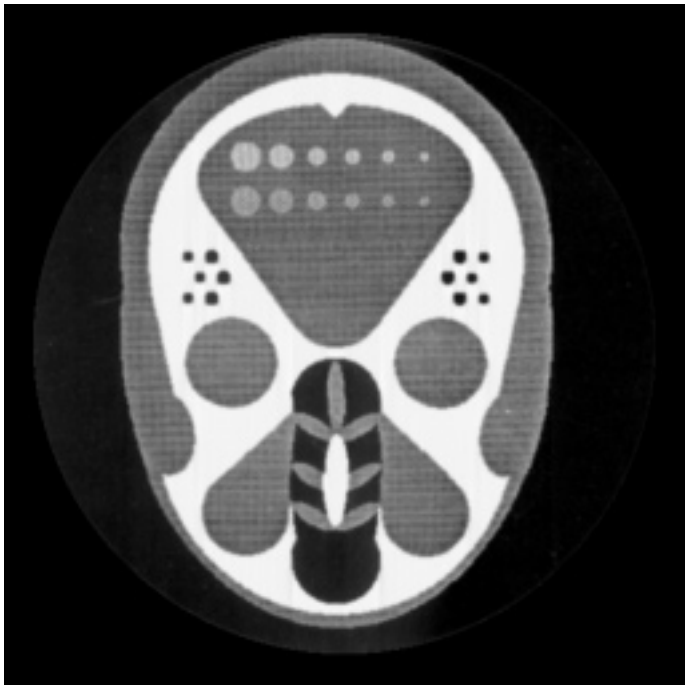


FIG. 6.8. *Schomberg–Timmer,* 512×512*: multilevel CB, doubled data.*

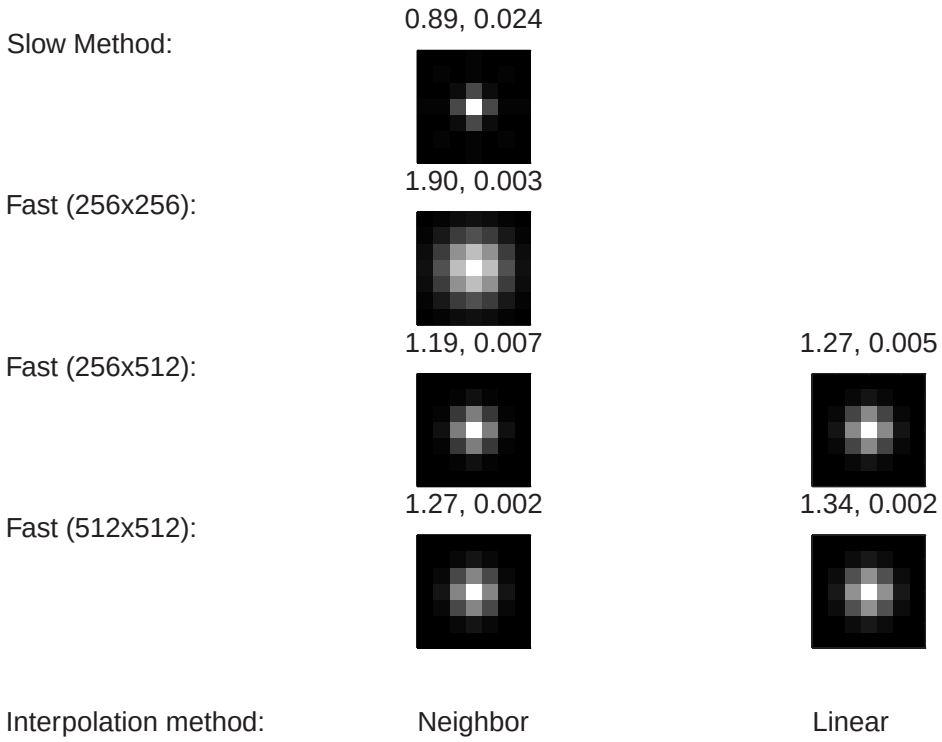
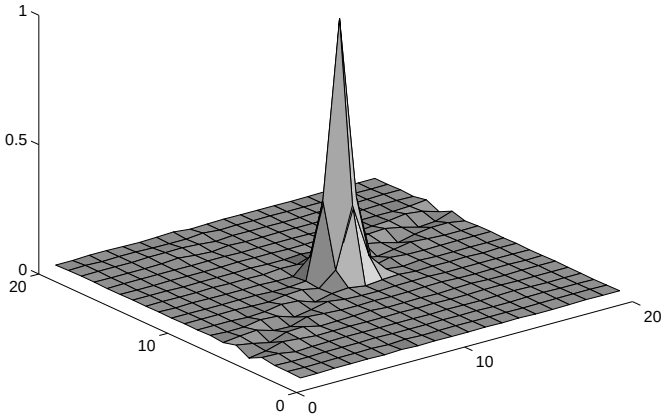


FIG. 6.9. *Reconstructions of delta-functions.*

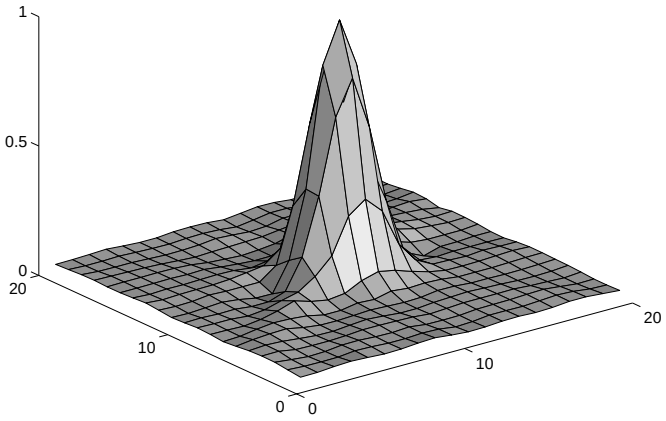
Figures 6.9–6.11 describe our point response study. Figure 6.9 indicates the optimal value σ_0 of σ and the degree to which the resulting Gaussian fits the point responses for the classical backprojections and several variants of the multiscale backprojection. Each grayscale image depicts the 7×7 center of an impulse response obtained using a particular combination of backprojection method, interpolation method, and data doubling method. Over each point response image, a pair of numbers is displayed. The first is the optimal value σ_0 of σ for the combination of methods, which is determined as described in section 5.2, and the second is the error ε given by

$$\varepsilon = \max_{i,j=-1,0,1} |A_{ij} - m_{ij}(\sigma_0)|.$$

It can be seen that while σ_0 is greater for the multiscale backprojection than for the classical backprojection, implying that multiscale backprojection without postprocessing has a wider point response and therefore produces blurrier images, the multiscale method point responses can be approximated better by a Gaussian, and therefore the multiscale backprojection with postprocessing can produce sharper images than the classical backprojection with postprocessing. The gray level numbers

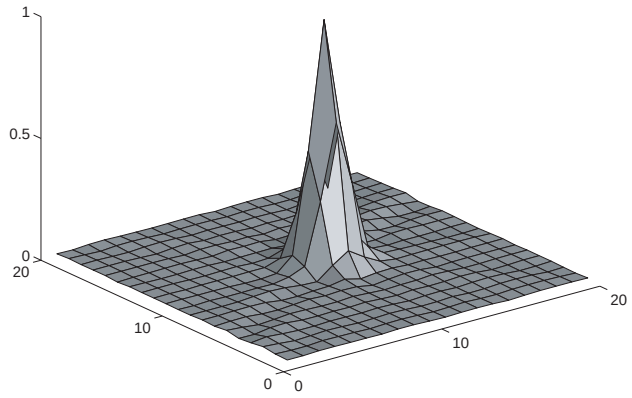


A point response of
Classical Convolution Backprojection

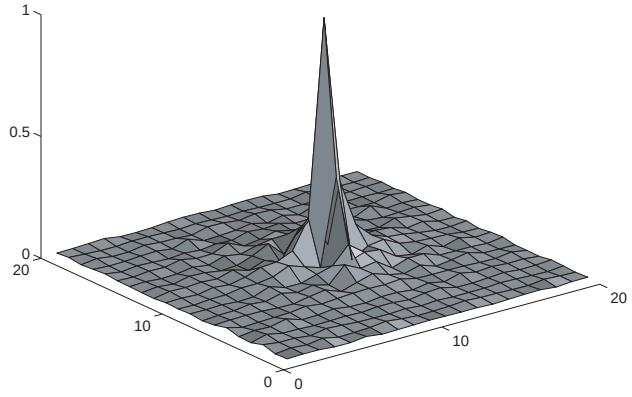


A point response of
Multilevel Convolution Backprojection

FIG. 6.10. *Point-spread functions.*



A point response of
Multilevel Convolution Backprojection with doubling



A point response of
Multilevel Convolution Backprojection with doubling and postprocessing

FIG. 6.11. *Point-spread functions.*

FIG. 6.12. *Lenna,* 256×256*: original image.*

for the impulse response windows are given in the appendix below. Figures 6.10 and 6.11 demonstrate a typical normalized point spread function in a 256×256 image as a result of each of the backprojection techniques and their variants. Figure 6.10 shows a typical normalized point response of CB with classical backprojection versus a typical normalized point response of CB with multilevel backprojection. Figure 6.11 shows a typical normalized point response of CB with multiscale backprojection with doubling of the data by nearest-neighbor interpolation, with and without postprocessing. It can be seen that the multiscale backprojection with postprocessing can produce point response at least as good as the classical backprojection point response.

Figure 6.12 shows a 256×256 Lenna image from which Radon data were computed for the purpose of testing different variations of CB. The input image was zero outside a circle inscribed in the 256×256 square, creating a sharp discontinuity at the edge of the circle, which generated a lot of energy in the higher frequencies. To reduce this high-frequency energy, a ring of nonzero pixels around the circle was added, and the grayscale decayed gradually to zero through the ring. Figure 6.13 shows the results of CB with classical backprojection. Figure 6.14 shows the results of CB with multiscale backprojection without postprocessing. Figure 6.15 shows the results of CB with multiscale backprojection, without postprocessing but with doubling of the data by means of interpolation. Figure 6.16 shows the results of CB with multiscale backprojection, *without* doubling of the data but with postprocessing.

We report simulation run times, although we would like to emphasize that the code has not been optimized in any way, and what really counts in our discussion is only the relative execution times for varying image sizes. We used an SGI challenge L machine, and the execution times were 7.63 sec and 30.41 sec for 256×256 and 512×512 images, respectively, showing as expected, a factor of approximately 4 between the two.

FIG. 6.13. *Lenna,* 256×256*: CB,* $O(n^3)$ *version.*

FIG. 6.14. *Lenna,* 256×256*: multilevel CB.*

FIG. 6.15. *Lenna,* 256×256*: multilevel CB, doubled data.*

FIG. 6.16. *Lenna,* 256×256*: postprocessing in the Fourier domain.*

7. Conclusions and potential improvements. The general conclusion of all these (and other) tests is that the bare multilevel convolution backprojection produces somewhat blurry pictures, compared with the classical, slower CB method. With carefully parameterized postprocessing (whose extra work is small compared with the overall work), however, the multilevel CB can give images which, over most of their parts, are *sharper even than the classical* CB (compare Figure 6.16 with Figure 6.13).

In some parts, mainly toward their margins, the reconstructed images are not as good. This is due to the fact that the near-Gaussian spread caused by the multilevel backprojection is not uniform over the domain. (This type of nonuniformity can be suppressed by randomly shifting each merging grid before the merge, instead of naive merging.) The *constant* Fourier-domain postprocessing should thus be replaced by a *variable* one. This cannot be done via the conventional FFT.

A general multiscale method for fast integral transforms has been presented in [4] (and previously briefly described in [2, section 8.6], and in more detail in [5, App. A] and [10]). It yields, by the way, among other transforms, a new FFT, which is slower than the conventional one by a modest constant factor (which depends on the desired accuracy) but is much more general (e.g., it does not require the transformed function to be given on a uniform grid, and it can use any number of data points). More important, the method can directly be applied to *general* transforms, including the Gaussian transform approximately describing our blurring. Such a transform is in fact performed faster than a single (conventional) FFT (in $O(N)$ instead of $O(N \log N)$ operations). With the same efficiency a similar *variable-coefficient* transform can be executed as well (the algorithm uses the smoothness of the coefficients on the scale of the grid but does not require them to be constant). As explained in [10], with essentially the same efficiency the *inverse* transforms can also be produced, as required for our postprocessing. Moreover, the fast multiscale algorithm does not even require the inverted transform to have any analytical closed-form expression (such as Gaussian); the transform can be described purely numerically (in a certain multiscale way). Hence, an efficient postprocessing can in principle be tailored directly to any given actual Radon transformer (e.g., a concrete X-ray scanner), using a sequence of training runs. Such a postprocessing may correct not only our fast backprojection blurring but also inaccuracies in the Radon machine itself.

Appendix. Gray levels of the point response windows, appearing in Figure 6.9.

TABLE A.1
Slow method.

0	0	0	0.016	0	0	0
0	0.016	0	0.016	0	0.016	0
0	0	0.047	0.28	0.047	0	0
0.016	0.016	0.28	1	0.28	0.016	0.016
0	0	0.047	0.28	0.047	0	0
0	0.016	0	0.016	0	0.016	0
0	0	0	0.016	0	0	0

TABLE A.2
Fast (256×256).

0	0.016	0.047	0.063	0.047	0.016	0
0.016	0.094	0.23	0.31	0.23	0.094	0.016
0.047	0.23	0.57	0.74	0.57	0.23	0.047
0.063	0.31	0.74	1	0.74	0.31	0.063
0.047	0.23	0.57	0.74	0.57	0.23	0.047
0.016	0.094	0.23	0.31	0.23	0.094	0.016
0	0.016	0.047	0.063	0.047	0.016	0

TABLE A.3
Fast (256×512), *neighbor*.

0	0	0	0	0	0	0
0	0	0.016	0.063	0.016	0	0
0	0.016	0.22	0.49	0.22	0.016	0
0	0.063	0.49	1	0.49	0.063	0
0	0.016	0.22	0.49	0.22	0.016	0
0	0	0.016	0.063	0.016	0	0
0	0	0	0	0	0	0

TABLE A.4
Fast (512×512), *neighbor*.

0	0	0	0	0	0	0
0	0	0.031	0.078	0.031	0	0
0	0.031	0.28	0.52	0.28	0.031	0
0	0.078	0.52	1	0.52	0.078	0
0	0.031	0.28	0.52	0.28	0.031	0
0	0	0.031	0.078	0.031	0	0
0	0	0	0	0	0	0

TABLE A.5
Fast (256×512), *linear*.

0	0	0	0	0	0	0
0	0	0.031	0.078	0.031	0	0
0	0.031	0.27	0.54	0.27	0.031	0
0	0.078	0.54	1	0.54	0.078	0
0	0.031	0.27	0.54	0.27	0.031	0
0	0	0.031	0.078	0.031	0	0
0	0	0	0	0	0	0

TABLE A.6
Fast (512×512), *linear*.

0	0	0	0	0	0	0
0	0	0.047	0.094	0.047	0	0
0	0.047	0.31	0.57	0.31	0.047	0
0	0.094	0.57	1	0.57	0.094	0
0	0.047	0.31	0.57	0.31	0.047	0
0	0	0.047	0.094	0.047	0	0
0	0	0	0	0	0	0

REFERENCES

[1] S. Alliney, S. Matej, and I. Bajla, *On the possibility of direct Fourier reconstruction from divergent-beam projections*, IEEE Trans. Med. Imag., 12 (1993), pp. 173–181.

[2] A. Brandt, *Guide to multigrid development*, in Multigrid Methods, W. Hackbusch and U. Trottenberg, eds., Springer-Verlag, New York, 1982, pp. 220–312.

[3] A. Brandt, The Gauss Center Research in Scientific Computation, Gauss Center Report WI/GC–7, March 1997. Gauss Center report WI/GC-12, Multiscale Scientific Computation: Six-Year Research Summary, 1999; available online from www.wisdom.weizmann.ac.il/ achi. Extended version in Electronic Trans. Numer. Anal., 6 (1997), pp. 1–34.

[4] A. Brandt, *Multilevel computations of integral transforms and particle interactions with oscillatory kernels*, Comput. Phys. Comm., 65 (1991), pp. 24–38.

[5] A. Brandt, *Multilevel computations: review and recent developments*, in Multigrid Methods: Theory, Applications and Supercomputing, S. F. McCormick, ed., Marcel Dekker, New York, 1988, pp. 35–62.

[6] A. Brandt, *Multiscale computational methods: research activities*, in Proceedings 1991 Hang Zhou International Conference on Scientific Computation, T. Chan and Z.-C. Shi, eds., World Scientific, Singapore, 1992, pp. 1–7.

[7] A. Brandt, *Stages in developing multigrid solutions*, in Proceedings 2nd International Congress on Numerical Methods for Engineering, E. Absi, R. Glowinski, P. Lascaux, and H. Veysseyre, eds., Dunod, Paris, 1980, pp. 23–43.

[8] A. Brandt and J. Dym, *Fast calculation of multiple line integrals*, SIAM J. Sci. Comput., 20 (1999), pp. 1417–1429.

[9] A. Brandt and I. Livshits, *Ray-way multigrid method for Helmholtz Equation*, Electronic Trans. Num. Anal., 6 (1997).

[10] A. Brandt and A. A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equations*, J. Comput. Phys., 90 (1990), pp. 348–370.

[11] A. Brandt and J. Mann, *An $O(N^2 \log N)$ Multiscale Backprojection Method*, Gauss Center Report WI/GC–6, Weizmann Institute of Science, Rehovot, Israel, 1997.

[12] A. Brandt, J. Mann, and M. Brodski, *A Fast and Accurate Radon Transform Inversion Scheme*, A Patent. Assigned to Yeda Research and Development Co. Ltd., August 1995. U.S. Patent and Trademark Office Application 08/659,595, filed 06/06/96, allowed 09/10/97. European Patent Office Application 97108722.6-2305, filed 05/30/97.

[13] S. R. Deans, *The Radon Transform and Some of its Applications*, John Wiley and Sons, New York, 1983.

[14] M. D. Desai and W. K. Jenkins, *Convolution backprojection image reconstruction for spotlight mode synthetic aperture radar*, IEEE Trans. Image Proc., 1 (1992), pp. 505–517.

[15] J. Dym, *Multilevel Methods for Early Vision*, Ph.D. thesis, Weizmann Institute of Science, Rehovot, Israel, 1994.

[16] H. Fan and J. Sanz, *Comments on "Direct Fourier reconstruction in computer tomography,"* IEEE Trans. Acous. Sp. Sig. Proc., ASSP-33 (1985), pp. 446–449.

[17] W. A. Götz and H. J. Druckmüller, *A fast digital Radon transform—an efficient means for evaluating the Hough transform*, Pattern Recognition, 29 (1996), pp. 711–718.

[18] G. Herman, *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography*, Academic Press, New York, 1980.

[19] S. Matej and I. Bajla, *A high-speed reconstruction from projections using direct Fourier method with optimized parameters—an experimental analysis*, IEEE Trans. Med. Imag., 9 (1990), pp. 421–429.

[20] R. Mersereau and A. Oppenheim, *Digital reconstruction of multidimensional signals from their projections*, Proc. IEEE, 62 (1974), pp. 1319–1338.

[21] D. C. Munson, Jr., J. D. O'Brien, and W. K. Jenkins, *A tomographic formulation of spotlight-mode synthetic aperture radar*, Proc. IEEE, 71 (1983), pp. 917–925.

[22] S. Nilsson, *Application of Fast Backprojection Techniques for Some Inverse Problems of Integral Geometry*, dissertation, Linköping University, Sweden, 1997.

[23] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1975.

[24] S. X. Pan and A. Kak, *A computational study of reconstruction algorithms for diffraction tomography: Interpolation versus filtered backpropagation*, IEEE Trans. Acous. Sp. Sig. Proc., ASSP-31 (1983), pp. 1262–1275.

[25] H. Peng and H. Stark, *Direct Fourier reconstruction in fan-beam tomography*, IEEE Trans. Med. Imag., MI-6 (1987), pp. 209–219.

[26] L. A. Shepp and B. F. Logan, *The Fourier reconstruction of a head section*, IEEE Trans. Nucl. Sci., NS-21 (1974), pp. 21–43.

[27] H. Schomberg and J. Timmer, *The gridding method for image reconstruction by Fourier transformation*, IEEE Trans. Med. Imag., 14 (1995), pp. 596–607.

[28] E. Sharon, A. Brandt, and R. Basri, *Completion energies and scale*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR–97), Puerto Rico, 1997. Report CS97–19, Weizmann Institute of Science, Rehovot, Israel. IEEE Trans. on Pattern Analysis and Machine Intelligence, submitted.

[29] H. Stark and M. Wengrovitz, *Comments and corrections on the use of polar sampling theorems in* CT, IEEE Trans. Acous. Sp. Sig. Proc., ASSP-31 (1983), pp. 1329–1331.

[30] H. Stark and J. W. Woods, *Authors' reply to "Comments on 'Direct Fourier reconstruction in computer tomography,' "* IEEE Trans. Acous. Sp. Sig. Proc., ASSP-34 (1986), pp. 379–380.

[31] H. Stark, J. Woods, I. Paul, and R. Hingorani, *Direct Fourier reconstruction in computer tomography*, IEEE Trans. Acous. Sp. Sig. Proc., ASSP-29 (1981), pp. 237–245.