A Rapid Look-Up Table Method for Reconstructing MR Images from Arbitrary K-Space Trajectories

Brian Dale, M. Wendt, *Member, IEEE*, and J. L. Duerk*, *Member, IEEE*

***Abstract*—Look-up tables (LUTs) are a common method for increasing the speed of many algorithms. Their use can be extended to the reconstruction of nonuniformly sampled k-space data using either a discrete Fourier transform (DFT) algorithm or a convolution-based gridding algorithm. A table for the DFT would be precalculated arrays of weights describing how each data point affects all of image space. A table for a convolution-based gridding operation would be a precalculated table of weights describing how each data point affects a small k-space neighborhood. These LUT methods were implemented in C++ on a modest personal computer system; they allowed a radial k-space acquisition sequence, consisting of 180 views of 256 points each, to be gridded in 36.2 ms, or, in approximately 800 ns/point. By comparison, a similar implementation of the gridding operation, without LUTs, required 45 times longer (1639.2 ms) to grid the same data. This was possible even while using a 4 $\times$ 4 Kaiser–Bessel convolution kernel, which is larger than typically used. These table-based computations will allow real time reconstruction in the future and can currently be run concurrently with the acquisition allowing for completely real-time gridding.**

***Index Terms*—Gridding, image reconstruction, k-space trajectory, tables.**

I. Introduction

THE RECONSTRUCTION of nonuniformly sampled Fourier data is a well-known problem in many fields including radio astronomy, synthetic aperture radar, and magnetic resonance imaging (MRI) [1], [2]. In MRI, this reconstruction problem arises when an image is acquired using a nonrectilinear or non-Cartesian k-space trajectory. Examples include radial k-space acquisitions for projection reconstruction (PR), spiral trajectories, rosette trajectories, or any number of other proposed trajectories. Such trajectories may be desirable for a variety of reasons including rapid acquisition times or good artifact reduction properties [3]–[5]. For example, in an interventional MRI (iMRI) setting, many trajectories could allow simultaneously high temporal and spatial resolution while providing data collection throughout a targeted region of k-space, determined by resolution requirements. This is in contrast to the currently more common keyhole techniques, which increase temporal resolution by sampling only a portion of the targeted region of k-space [6]. However, to gain their full potential, these nonrectilinear trajectories currently require access to numerically intensive special reconstruction code, sophisticated computer hardware or special reconstruction hardware, thereby limiting their use in some applications [7], [8].

The solutions to the problem of how to reconstruct nonuniformly sampled k-space data can be broadly grouped into two categories. The first are those that rely on direct application of the discrete Fourier transform (DFT) from the sampled data points. The second are those which use a gridding algorithm to estimate an equivalent set of uniformly sampled Fourier data from the nonuniformly sampled acquired data, followed by reconstruction using the fast Fourier transform (FFT) [9]–[11].

An efficient method for performing either the DFT reconstruction or data gridding operations would allow completely real-time reconstruction (i.e., limited by the acquisition time) of these nonrectilinear trajectories, which could open the door for their use in a wider variety of clinical situations. Moreover, if the method could allow the calculation of the result due to each individual data point immediately upon acquiring the data point, it would have an advantage even over current keyhole or sliding window reconstruction methods. The image could be updated after each individual data point, or after any given number of data points, to reflect the most recent information. This would be a necessary precursor to truly fluoroscopic MRI acquisitions.

The purpose of this work was to determine if the computational efficiency of precalculated look-up tables (LUTs) could be effectively exploited to achieve real time reconstruction, or real time gridding during nonrectilinear or non-Cartesian k-space sampling when using only modest conventional computational resources. The remainder of this paper will describe table-based computational methods that permit rapid reconstruction of nonuniformly sampled k-space data even on a typical, personal computer for the DFT and for some of the currently used gridding algorithms.

II. Methods

A straightforward LUT arrangement and, hence, a simple memory allocation scheme allows very predictable access into a LUT, which in turn suggests the possibility for fast table-driven computation [12]. Proposed methods to achieve an efficient DFT and gridding table method are described below.

Manuscript received March 16, 2000; revised January 9, 2001. This work is supported in part through research collaborations with Siemens Medical Systems, Radionics, Inc., Hewlett Packard, Inc., and in part through grants from the Whitaker Foundation, the American Cancer Society, Mary Ann S. Swetland Fund, and by the National Institutes f Health (NIH) under Grant R01CA81431-01. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was Z.-P. Liang. *Asterisk indicates corresponding author.*

B. Dale and M. Wendt are with the Department of Radiology and Biomedical Engineering, Case Western Reserve University and University Hospitals of Cleveland, Cleveland, OH 44106 USA.

*J. L. Duerk is with the Department of Radiology and Biomedical Engineering, Case Western Reserve University and University Hospitals of Cleveland, Cleveland, OH 44106 USA (e-mail: duerk@uhrad.com).

Publisher Item Identifier S 0278-0062(01)02778-1.

A. Description of a LUT DFT Method

A DFT table would be a precalculated table of weights describing how each data point affects the entire image space. That is, the precalculated table consists of the DFT of a delta function at the sampled location in k-space. Consider the DFT transform pair for a one-dimensional signal sampled at N equidistant points in both time and frequency

$$F(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f(n) \cdot e^{-j2\pi kn/N} \qquad k = 0, 1, \ldots, N-1$$
$$f(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} F(k) \cdot e^{j2\pi kn/N} \qquad n = 0, 1, \ldots, N-1 \tag{1}$$

where $F(k)$ is the Fourier data and $f(n)$ is the reconstructed signal in the space (or time) domain. The synthesis equation may be rewritten [see (2)] when the requirement for equidistant frequency samples is relaxed

$$f(n) = \frac{1}{\sqrt{N}} \sum_{i=0}^{Ns-1} \rho^{-1}(k(i)) \cdot F(k(i)) \cdot e^{j2\pi k(i)n/N} \qquad n = 0, 1, \ldots, N-1 \tag{2}$$

where

the $k(i)$	sampled locations in frequency;
Ns	number of frequency samples;
$\rho^{-1}(k(i))$	density compensation function at the sample locations.

The function $\rho^{-1}(k(i))$ is equal to 1.0 if the $k(i)$ are equidistant; otherwise it is large in the relatively undersampled regions and small in oversampled regions [13], [14].

Letting

$$T(i, n) = \frac{1}{\sqrt{N}} \rho^{-1}(k(i)) \cdot e^{j2\pi k(i)n/N} \qquad n = 0, 1, \ldots, N-1 \quad i = 0, 1, \ldots, Ns-1 \tag{3}$$

then (2) may be rewritten as

$$f(n) = \sum_{i=0}^{Ns-1} T(i, n) \cdot F(k(i)) \qquad n = 0, 1, \ldots, N-1. \tag{4}$$

If the $k(i)$ are known in advance, as is the case in any MRI pulse sequence, then it is possible to precalculate and store all of the $T(i, n)$, and reload them only during the acquisition/reconstruction. This is the mathematical basis of the DFT table-enhanced method. Note that all of the transcendental functions and the density compensation functions are precomputed, and the task, at run time, consists entirely of multiplication and addition. Note also that all of the $f(n)$ can be processed independently and simultaneously. Because of this, it is convenient to write the above equations in a vectorized format

$$\mathbf{f} = \sum_{i=0}^{Ns-1} \mathbf{T}(i) \cdot F(k(i))$$

where

$$\mathbf{f} = \begin{bmatrix} f(0) \\ \vdots \\ f(N-1) \end{bmatrix} \quad \text{and} \quad \mathbf{T}(i) = \begin{bmatrix} T(i, 0) \\ \vdots \\ T(i, N-1) \end{bmatrix}. \tag{5}$$

For a two-dimensional (2-D) reconstruction

$$T(i, x, y) = \frac{1}{N} \rho^{-1}(k_x(i), k_y(i)) \cdot e^{j2\pi(k_x(i)x + k_y(i)y)/N} \qquad x = 0, 1, \ldots, N-1 \quad y = 0, 1, \ldots, N-1 \quad i = 0, 1, \ldots, Ns-1$$
$$f(x, y) = \sum_{i=0}^{Ns-1} T(i, x, y) \cdot F(k_x(i), k_y(i)) \qquad x = 0, 1, \ldots, N-1 \quad y = 0, 1, \ldots, N-1 \tag{6}$$

or, in vector/matrix format

$$\mathbf{f} = \sum_{i=0}^{Ns-1} \mathbf{T}(i) \cdot F(k_x(i), k_y(i))$$

where

$$\mathbf{f} = \begin{bmatrix} f(0, 0) & \ldots & f(N-1, 0) \\ \vdots & \ddots & \vdots \\ f(0, N-1) & \ldots & f(N-1, N-1) \end{bmatrix}$$

and

$$\mathbf{T}(i) = \begin{bmatrix} T(i, 0, 0) & \ldots & T(i, N-1, 0) \\ \vdots & \ddots & \vdots \\ T(i, 0, N-1) & \ldots & T(i, N-1, N-1) \end{bmatrix}. \tag{7}$$

In MRI, each of the $F(k_x(i), k_y(i))$ is one of the acquired data points. A table-based DFT reconstruction, therefore, consists of the following steps.

1) Compute and store all of the $\mathbf{T}(i)$. This file is the precalculated LUT and may be used repeatedly for a given trajectory. That is, $\mathbf{T}(i)$ needs only to be calculated once since it can be used, for example, for any pulse sequence in which that k-space trajectory is used.
2) Load the LUT into memory.
3) Set the current estimate of $\mathbf{f}$ to zero.
4) Acquire the first data point, $F(k_x(0), k_y(0))$.
5) Calculate $\mathbf{T}(0) \cdot F(k_x(0), k_y(0))$.
6) Accumulate (add) the result into the current estimate of $\mathbf{f}$.
7) Repeat steps 4–6 for the rest of the data points.

B. Description of a LUT Method for Gridding

A table for gridding operations used prior to 2-D FFT (2D-FFT) reconstruction of nonrectilinearly sampled MRI data would be a precalculated table of weights describing how each data point contributes to a small, rectilinearly sampled, k-space neighborhood. The gridding algorithm presented by O'Sullivan consists of several steps designed to compute, not the actual Fourier data at the grid points, but a set of gridded Fourier data whose FFT is equal to the integral Fourier transform (FT) of the nonuniformly sampled function [9]. The basic steps are as follows.

1) Multiply the trajectory data, $F(k_x(i), k_y(i)) \cdot \delta(k_x - k_x(i), k_y - k_y(i))$, by an appropriate density compensation function, $\rho^{-1}(k_x(i), k_y(i))$.
2) Convolve the density compensated trajectory data by an appropriate function, $C(k_x, k_y)$. Ideally, a sinc function covering all of k-space would be used, but this would lead to a significant computational burden and, thus, finite windows, like Kaiser–Bessel functions, are chosen instead.
3) Sample the convolved function onto a rectilinear k-space grid by multiplying by the comb function

$$comb(k_x, k_y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \delta(k_x - u, k_y - v). \tag{8}$$

4) Transform the gridded data, $G(k_x, k_y)$, from the previous step using the FFT.
5) Deconvolve the transformed image by dividing it by $c(x, y)$, the FT of the convolution function used in Step 2. This step is unnecessary if a sinc function was used in the gridding convolution; for many convolution functions this correction produces minimal improvements in the central portion of the image.

The result from Step 3 of the gridding algorithm is the gridded Fourier data that lies on a rectilinear grid [10], expressed here as

$$\begin{aligned} &G(k_x, k_y) \\ &\quad = \Bigg[C(k_x, k_y) * \sum_{i=0}^{Ns-1} \\ &\qquad \cdot (\rho^{-1}(k_x(i), k_y(i)) \\ &\qquad\quad \cdot (F(k_x(i), k_y(i)) \cdot \delta(k_x - k_x(i), k_y - k_y(i)))) \Bigg] \\ &\qquad \cdot comb(k_x, k_y). \end{aligned} \tag{9}$$

Letting

$$\begin{aligned} &Tg(i, k_x, k_y) \\ &\quad = (C(k_x, k_y) * (\rho^{-1}(k_x(i), k_y(i)) \\ &\qquad\qquad \cdot \delta(k_x - k_x(i), k_y - k_y(i)))) \\ &\qquad \cdot comb(k_x, k_y) \qquad i = 0, 1, \ldots, Ns - 1 \end{aligned} \tag{10}$$

then

$$G(k_x, k_y) = \sum_{i=0}^{Ns-1} Tg(i, k_x, k_y) \cdot F(k_x(i), k_y(i)) \tag{11}$$

or, in matrix format

$$\mathbf{G} = \sum_{i=0}^{Ns-1} \mathbf{Tg}(i) \cdot F(k_x(i), k_y(i))$$

where

$$\mathbf{G} = \begin{bmatrix} G(0, 0) & \ldots & G(N-1, 0) \\ \vdots & \ddots & \vdots \\ G(0, N-1) & \ldots & G(N-1, N-1) \end{bmatrix}$$

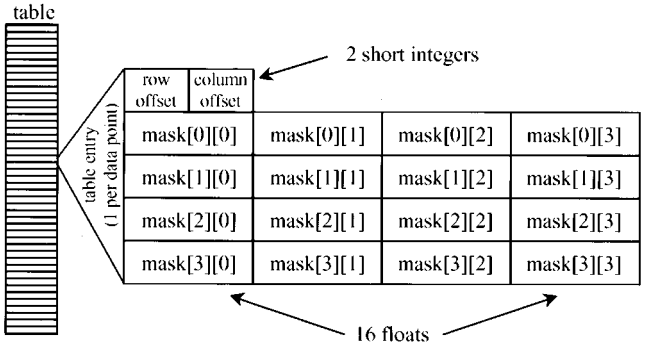

Fig. 1. Structure of the gridding table. Table example based on a 4 × 4 convolution window. The row offset and column offset values accomplish the sparse matrix representation. The mask elements are the only nonzero elements of the sparse matrix.

and

$$\mathbf{Tg}(i) = \begin{bmatrix} Tg(i, 0, 0) & \ldots & Tg(i, N-1, 0) \\ \vdots & \ddots & \vdots \\ Tg(i, 0, N-1) & \ldots & Tg(i, N-1, N-1) \end{bmatrix}. \tag{12}$$

Equation (12) has the same form as (7). Therefore, a similar table-based reconstruction process used above to calculate $\mathbf{f}$ could also be used here to calculate the gridded data, $\mathbf{G}$. Once complete, the image can be reconstructed via the 2-D (or N-D) FFT of $\mathbf{G}$. Further, any operation that can be expressed as a linear function on the sampled data points, e.g., $F(k_x(i), k_y(i))$, can be computed using the table-based reconstruction method.

Once the gridding table, $\mathbf{Tg}$, is created and stored it can be loaded at run time into memory and then used to greatly speed the processing. The table-enhanced gridding algorithm proceeds in the following steps:

1) Do a table-based reconstruction using $\mathbf{Tg}$ to reconstruct $\mathbf{G}$ (as opposed to using $\mathbf{T}$ to reconstruct $\mathbf{f}$). This covers steps 1–3 in the gridding algorithm.
2) Transform the gridded data using the FFT.
3) Deconvolve the transformed image by dividing it by $c(x, y)$, the FT of the convolution function used to calculate the table.

C. Programming Methods

For a small convolution window, the vast majority of the points in the $\mathbf{Tg}(i)$ are equal to zero. This allows use of a sparse matrix representation where the zeros need neither be stored nor calculated at run time. Thus, the tables are organized as shown in Fig. 1. The ith data point acquired from the scanner corresponds to the ith entry in the table which is, in turn, simply a sparse matrix representation of $\mathbf{Tg}(i)$. When a data point is acquired, the row and column offsets are read and used as indexes into the gridded array, which is the current estimate of $\mathbf{G}$. Then all of the weights in the table entry are multiplied by the data point and accumulated into the indicated locations in the array. The next entry in the table is then loaded and the processor waits, if necessary, to acquire the next data point. If the processor has idle time while waiting for the next data point, the size of the

convolution window may be increased or other computations may be performed without adding any time to the reconstruction. A larger convolution window generally means less energy will alias back into the image [10].

All of the gridding tables used here were calculated with the Kaiser–Bessel window (13) [9], [10]

$$C(k_x, k_y) = C(k_x) \cdot C(k_y)$$

$$C(k) = \begin{cases} \frac{1}{W} I_0 \left[\beta \sqrt{1 - \left(\frac{2k}{W} \right)^2} \right], & |k| \le \frac{W}{2} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where

I_0 zero-order modified Bessel function of the first kind;
W width of the window;
β free parameter.

β was set to the optimal value (5.7567 for $W = 4$) presented by Jackson *et al.* for windows of width 5.0 or less or otherwise to the approximately optimal value (9.4248 for $W = 6$ or 12.566 for $W = 8$) from the simple equation presented by Wajer *et al.* [10], [15].

Both radial and spiral trajectories were considered. The density compensation function was multiplication by the k-space radius of the data point for both cases [13]. The comb function used to produce the k-space grid had a sampling interval of unity as shown above; it was not oversampled as has been suggested in order to increase the field of view (FOV) and thereby move the aliasing energy away from the region of interest [10].

Three different complex number representations were attempted for the DFT table reconstruction. The first was the standard rectangular representation (i.e., $A + jB$). The second and third were polar representations ($Ae^{j\theta}$) with the phase discretized into 16 and 8 bits respectively and the magnitude being constant [equal to the value of $\rho^{-1}(k_x(i), k_y(i))$, representing the density compensation function] for each table entry. The Kaiser–Bessel window function is entirely real and, therefore, the various complex number representations used for a DFT table are neither possible nor necessary for a gridding table.

All algorithms were implemented in either the Interactive Data Language (IDL: Research Systems, Inc., Boulder CO) or in C++ (Visual C++ v5.0: Microsoft Inc., Redmond, WA). For C++ programs, the OptiVec C/C++ complex math library (by Martin Sander, shareware v1.6), which allows for complex data and includes the FFT, was used. It has been hand optimized by using assembly-level instructions and vectorized computational techniques. For example, the FFT algorithms were written in assembly level.

D. Experimental Methods

1) DFT Methods: The DFT table experiments were designed to determine differences in accuracy, computational speed, and table size of the various complex number representations. A mathematical phantom, similar to a CT resolution phantom for large data sets ($>100^2$ data points) or a single square for small data sets, was calculated based on a series of 2-D rectangular pulses of width (w_x, w_y) centered about the point (c_x, c_y) [u in (14)].

$$u_{w_x, w_y, c_x, c_y}(x, y) = u_{w_x, c_x}(x) \cdot u_{w_y, c_y}(y)$$

$$u_{w,c}(x) \xleftrightarrow{F} 4 \frac{e^{-jkc} \sin\left(\frac{kw}{2} \right)}{k}$$

where

$$u_{w,c}(x) = \begin{cases} 1, & \left(|x - c| \le \frac{w}{2} \right) \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Its analytical FT provided simulated k-space data that was uniformly sampled on a 32 × 32 square grid. LUTs were developed to reconstruct from the 32 × 32 k-space data onto a 32 × 32 image matrix. Three different LUTs were precalculated using IDL according to the three different representations of complex numbers: one using the standard rectangular representation and the other two using polar representations with the phase discretized to 8 and 16 bits respectively. The size of each of the LUTs was noted. The table-based DFT reconstruction was performed following the aforementioned method and the time required was measured using the computer's internal clock and the standard time routines. After measuring the reconstruction time with these tables, the fastest one was re-coded in C++ and compiled using the previously mentioned compiler set to optimize for speed. The deviations from use of the FFT were calculated and compared. The speed results obtained with IDL were checked by using C++ to measure the average speed of one-million multiply-accumulate (MAC) operations on random data for each of the various complex number formats. The experiment was not designed to assess or compare the computational time of the 2D-FFT versus a table-based reconstruction, but rather to explore only errors and reconstruction times associated with different table formats that might lend themselves to more rapid reconstruction.

2) Gridding Methods: All of the gridding procedures described in Section II-B and (8)–(12) were coded with the Microsoft C++ compiler. All reported reconstruction times were obtained using the standard C++ time routines with the code compiled to optimize for speed. They were performed on a Windows NT 4.0 single processor 600-MHz Pentium III personal computer with 512 MB of RAM. All of the gridding-table experiments used the same gridding code and the same 2D-FFT code for reconstruction; the only differences were in the code used to generate the precalculated LUTs and the raw data.

The first gridding-table trial measured the time required to grid simulated data resulting from sampling along a spiral trajectory. The simulated spiral trajectory data was generated from evaluation of the analytic expression for the 2D-FFT of the mathematical resolution phantom at each k-space sampling location. An Archimedian spiral trajectory was used

$$(k_x(i), k_y(i)) = \left(\text{re}\left(\frac{i}{256} e^{j\pi i/64} \right), \text{im}\left(\frac{i}{256} e^{j\pi i/64} \right) \right)$$
$$i = 0, 1, \ldots, (128 \cdot 128) - 1 \tag{15}$$

where the real part of the number represents the distance in k-space along the k_x direction and the imaginary part repre-

sents the position along the k_y direction. This spiral has 128 turns and 128 sampled data points/turn with the radius of the last data point being 64. The data were gridded onto a $(128 + W-1)^2$-point matrix where W is the width of the Kaiser–Bessel window as is dictated through the use of the convolution gridding method without oversampling. The gridded matrix was zero padded to 256^2 points as required by the OptiVec FFT routine. A reference image was generated by applying the gridding reconstruction algorithm to data obtained by sampling the transform of the mathematical resolution phantom along a 128^2 Cartesian trajectory.

These images and times were also compared to a nontable-based gridding operation. The nontable-based operation used the same code as the table-based operation except in the innermost loop where the simple MAC of the precalculated table value was replaced with the usual inline computation of the density compensation function and the Kaiser–Bessel window. This window utilized a fast Taylor-series expansion of the zero-order modified Bessel function of the first kind. Enough terms were used to ensure accuracy to one part/million. This is in contrast to the much slower function used in the table calculations, which proceeds to full machine precision. However, since this is precalculated, the increased time does not impact on the gridding/reconstruction time.

The second gridding-table test reconstructed simulated data sampled along a radial k-space trajectory, as would occur in projection reconstruction acquisitions, which are now regaining attention in a variety of MRI applications. The same mathematical resolution phantom was used to generate simulated data by evaluation of the analytic expression for its 2D-FT. The radial trajectory consisted of 180 views, each offset by 1° from the previous; 256 points were collected for each view with the 128th data point located at the origin of k-space. The data were gridded onto a $(256 + W - 1)^2$-point matrix where W is the width of the Kaiser–Bessel window. The gridded data was generated, the gridding time was measured, and the resulting times and images were compared to those obtained through a nontable-based algorithm as above.

The third and final trial of the table-based gridding algorithm reconstructed *in vivo* radial k-space data acquired in our lab. A pig's neck was scanned using a 180 view, 256 points/view radial k-space True-FISP acquisition during interventional MRI needle insertion. The pulse sequence was implemented on a Siemens 0.2T Magnetom Open MRI system (Siemens Medical Systems, Erlangen, Germany). The echo occurred at the 128th point in each view. The views were separated by approximately (but not exactly) 1° due to the MRI system's hardware truncating the gradient tables to a fixed precision. The data were again gridded onto a $(256 + W - 1)^2$-point matrix where W is the width of the Kaiser–Bessel window. The time required for table-based gridding was measured and compared with nontable-based gridding as with previous experiments.

III. Results

Results from the DFT experiment showed that the unavoidable Fourier artifacts, such as Gibbs ringing, overwhelm the miniscule differences between the FFT and the table-enhanced DFT by five orders of magnitude as would be expected. Fig. 2 shows the deviations, with respect to the FFT, generated by using the various table representations from the first DFT table experiment. The deviations in this image are calculated by taking the absolute value of the magnitude error. Before the errors were calculated, the images were scaled up to the range of bytes (gray levels) to give a reference for how visible those errors would be on a typical display device. The rectangular complex table generated the most compact error pattern with the lowest peak value. The two polar tables generated different error patterns with the errors from the 8-bit table being highest, five gray levels or approximately 2% peak error. Also note that the error for the 8-bit table was at or near its peak value (five gray levels) for many more pixels than was the 16-bit table. Unfortunately, neither of the polar representations performed better than the standard rectangular table in speed as shown in Table I. The general trend of Table I was also evident with the MAC measurements on random data in C++ (rectangular: 351 ns/MAC; 16-bit polar: 1041 ns/MAC; 8-bit polar: 972

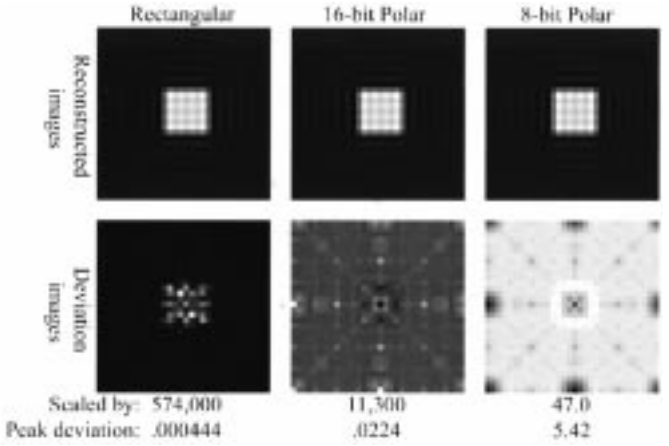

Fig. 2. Accuracy comparison between the different complex number representations for the DFT table. For the polar representations, the number of bits indicates how many bits of precision were used to discretize the phase. Deviations are scaled to the range of bytes (0–255) and are measured relative to the FFT of the same data.

TABLE I
DFT Table Reconstruction Time for Various Complex Number Representations

Representation	Reconstruction Time
Rectangular	422 ms
16-bit Polar	839 ms
8-bit Polar	731 ms
C++ (rectangular)	110 ms

TABLE II
COMPUTATION TIMES FOR VARIOUS k-SPACE TRAJECTORIES WITH LUT GRIDDING

Trajectory	W = 4[a]	W = 6[a]	W = 8[a]	2D-FFT
Spiral[b] (table-enhanced)	12.8 ms	29.3 ms	47.17 ms	44.3 ms[d]
Spiral[b] (standard)	582.3 ms	1499.5 ms	2992.9 ms	44.3 ms[d]
Radial[b] (table- enhanced)	36.2 ms	80.1 ms	131.5 ms	219.4 ms[e]
Radial[b] (standard)	1639.2 ms	4218.1 ms	8415.3 ms	218.7 ms[e]
Radial[c] (table- enhanced)	36.1 ms	80.0 ms	130.8 ms	218.4 ms[e]
Radial[c] (standard)	1636.9 ms	4211.7 ms	8399.0 ms	218.8 ms[e]

a: W is the width of the Kaiser-Bessel convolution kernel.
b: Simulated dat.
c: Actual data from pig.
d: 256^2-point 2D-FFT.
e: 512^2-point 2D-FFT.

ns/MAC). The fastest time, obtained for the optimized C++ implementation of the rectangular table, was 110 ms for a 32 $\times$ 32 reconstruction.

In Table II, the left-hand column indicates the trajectory and the top row indicates the size of the Kaiser–Bessel window used in the computations for gridded data trials. The size of the window determines the number of mask points in each table entry. These convolution kernels were generally larger than those typically used elsewhere [4], [16]–[18]. The images and results from the individual trials will be mentioned below, but it is important to note the ratio of the computation times of the table-based and the nontable-based trials. The table-enhanced gridding operation ranged from 45 to 64 times faster than the standard gridding operation, and in most cases the table-enhanced gridding was faster than the corresponding 2D-FFT while the nontable-enhanced gridding was always slower.

The spiral trajectory, the first gridding-table experiment, produced the results shown in Fig. 3 and Table II. Fig. 4(a) is the difference image between the spiral image and the reference image, scaled up by 2.7 times to improve contrast visibility. The figure shows that the only artifacts in the original image are some streaking artifacts of 93 gray levels (peak difference) on the edge or 29 gray levels (peak difference) in the central region. These differences arise from the nonuniform sampling. Fig. 4(b) is the difference image between the nongridded Cartesian data and the reference image, scaled up by 51 times to improve visibility. The peak difference is five gray levels. The peak difference between the reconstructed image obtained with the table-based gridding and the one obtained with standard gridding is one gray level. These small errors occurred slightly more frequently in the corners of the image.

The second experiment, the simulated radial k-space acquisition, resulted in the image in Fig. 5 and the computational speeds

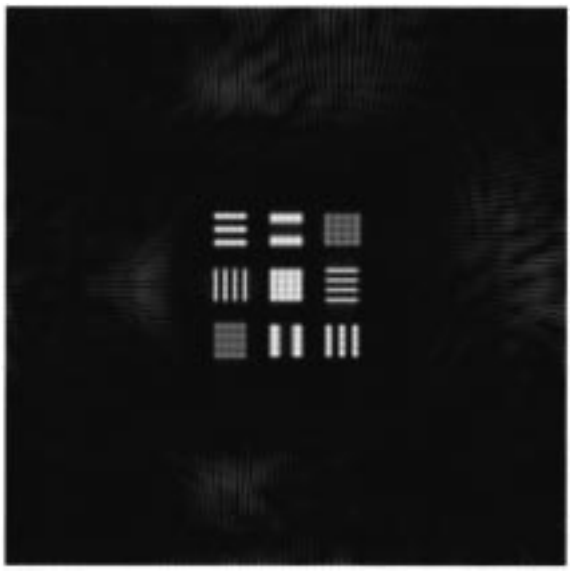

Fig. 3. Gridded and reconstructed image from the spiral trajectory sampling of the mathematical resolution phantom, gridded in 12.8 ms using the table-based reconstruction with a 4 $\times$ 4 Kaiser–Bessel convolution window. 256^2-point FFT took 44.3 ms on the same system.

given in Table II. It took nearly three times longer to reconstruct the PR sequence than the previous spiral trajectory, as expected since the radial k-space PR trajectory had nearly three times the number of data points. The artifacts in this reconstruction were similar to, but fainter (35 gray levels peak), than the artifacts in the spiral experiment.

The third and final gridding-table trial is summarized in Fig. 6 and Table II. Even using this actual MRI data, the computational speed is almost exactly the same as from the previous experiment, as was expected for two trajectories with the same number

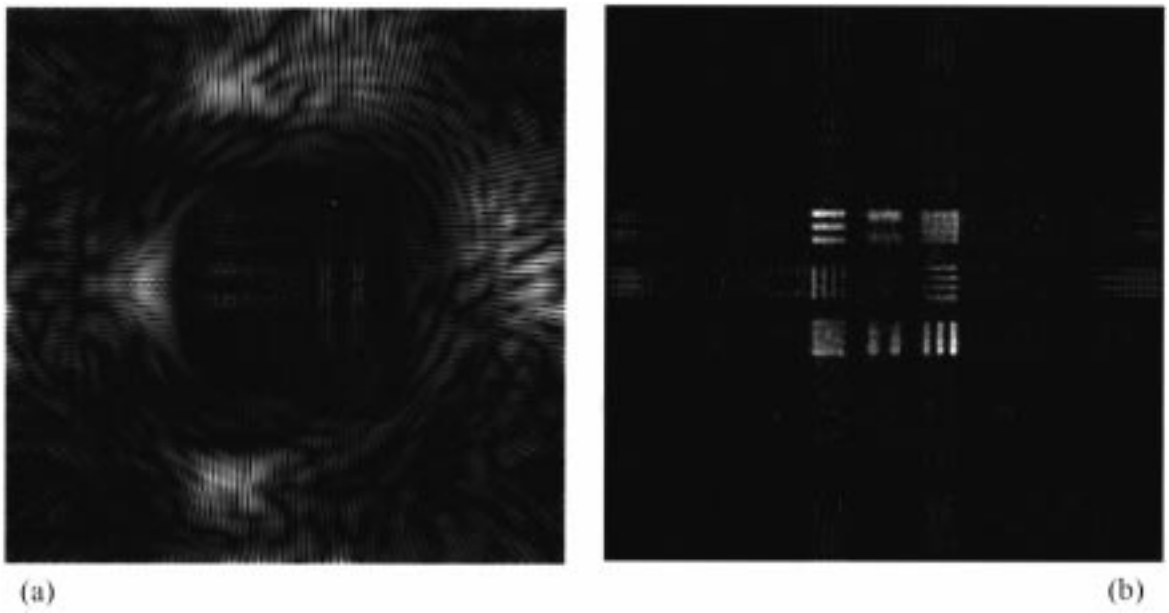

Fig. 4. Difference images with respect to a reference image, obtained by applying the gridding reconstruction algorithm to data from sampling the transform of the mathematical resolution phantom along a 128^2 Cartesian trajectory. Peak difference with the gridded/reconstructed spiral image was 93 gray levels in the outer regions and 29 in the central region. (a) The spiral versus reference difference image, which shows artifacts due to the spiral trajectory, was scaled by 2.7 to improve contrast visibility. Peak difference with the nongridded FFT-reconstructed image was five gray levels. (b) The nongridded versus reference image, which shows artifacts due only to the gridding, was scaled by 51 to improve contrast visibility. K-space data gridded using the table-based reconstruction with a 4×4 Kaiser–Bessel convolution window.

Fig. 5. Reconstructed image from the radial trajectory sampling of the mathematical resolution phantom, gridded in 36.2 ms using the table-based reconstruction with a 4×4 Kaiser–Bessel convolution window. 512^2-point FFT took 219.4 ms on the same system.

of data points. Fig. 6 also includes a filtered backprojection reconstruction of the data. Observe that most of the artifacts in the gridded image (with the notable exception of the rings in the corner) also occur in the back-projected image.

IV. Discussion

Many of the points of this paper may seem fairly obvious to some readers, but a thorough investigation regarding the applicability of LUTs to nonrectilinear MRI reconstruction in general, and these algorithms in particular, is absent in the literature. Prior to this work, it was doubtful whether such large tables would be useful in accelerating any algorithm. The conventional wisdom regarding LUTs is that they must be small enough to fit inside the cache to achieve computational efficiency, while the tables used in this manuscript are necessarily many times larger.

The DFT-table experiment, the comparison of rectangular versus polar representations, indicates that the rectangular representation is preferable over either of the polar representations (rectangular versus 16-bit polar versus 8-bit polar representations had peak deviations of 0.000 444 versus 0.0224 versus 5.42 gray levels and reconstruction times of 422 ms versus 839 ms versus 731 ms). The computational burden of the DFT reconstruction is proportional to the number of data points in the trajectory times the number of pixels in the final image (1024×1024 for the above experiments). Thus, the fastest results from these experiments (110 ms) can be extrapolated to give anticipated best-case reconstruction times for larger trajectories or image sizes. For a 128×128 trajectory and final image (or any arbitrary trajectory with 128^2 data points) the extrapolated reconstruction time is 28.16 s; for a 256×256 trajectory and final image the time is 7.5 min. These reconstruction times are much greater than would be useful in any kind of time-sensitive application. The DFT-table method is still useful to consider because, due to its inherently parallel structure, specialized hardware and advances in computer technology may eventually make the table-based DFT even faster than the FFT (discussed in further detail below). Unfortunately, the DFT is simply too computationally intensive and, thus, currently takes too long, to warrant further investigation for use in a typical situation with limited computer resources.

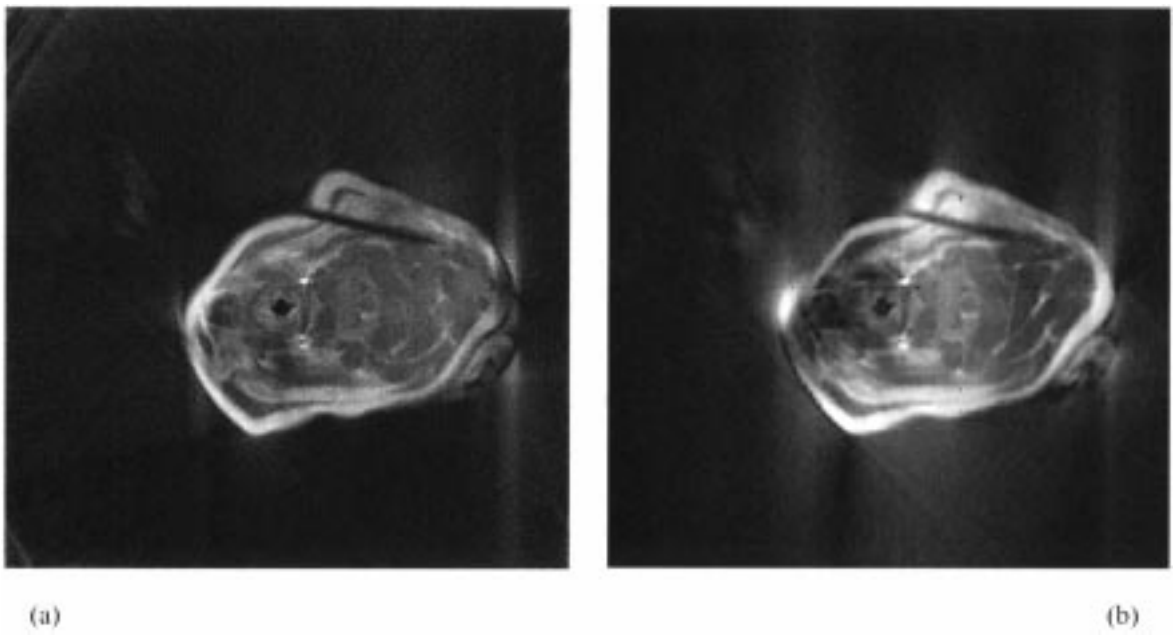

Fig. 6. (a) Reconstructed magnitude image from the *in vivo* radial acquisition of a pig neck showing inserted needle, gridded in 36.1 ms using the table-based reconstruction with a 4×4 Kaiser–Bessel convolution window. 512^2-point FFT took 218.4 ms on the same system. (b) For comparison, a filtered backprojection reconstruction of the same data is shown. The 180-view radial data was acquired using a True-FISP sequence with TE $=$ 9.8 ms and TR $=$ 20.8 ms at 0.2 T.

The first gridding experiment was the first experiment to use a non-Cartesian sampling scheme and, thus, the first experiment that needed a non-FFT reconstruction. The most interesting aspects of this experiment are the difference images between the reconstructed gridded data and the reference image and between the nongridded 2D-FFT of the Cartesian data and the reference image. Though not necessary to reconstruct the image, the use of the gridding operation on the Cartesian data allows one to separately examine the effects of the trajectory and the effects of the gridding. The spiral versus reference difference image is shown (peak difference: 93 outer region, 29 central region), scaled by 2.7 times, in Fig. 4(a). This image indicates that the artifacts in Fig. 3 are largely due to the trajectory itself. The nongridded versus reference difference image is shown (peak difference: five gray levels), scaled by 51 times, in Fig. 4(b). The most evident features of the spiral versus reference image are the radial streaks around the outer portion of the image caused by the spiral sampling of the Fourier data. However, note the distinct delineation of the central portion of the image where there is much less difference between the two images. This central portion is almost exactly 128 pixels in height and width, or it is almost exactly the central quarter of the image. With some values of β it is possible to get sharp peaks in the corners of the image. This problem could be avoided by setting the outer portion of the image to zero, keeping only the portion that is the most accurate reconstruction of the desired image (see the central portion of Fig. 4). This also supports the use of over-sampling of the Fourier data, as has been suggested in the literature, to increase the FOV prior to this cropping [10]. The low peak difference (five gray levels) in the nongridded versus reference image indicates that most of the artifacts in the table-based gridding reconstructions are attributable to the trajectory itself, rather than to the reconstruction.

The simulated radial k-space data experiment showed similar, though much less intense, artifact patterns as the previous spiral sampling. This is not surprising since the spiral ultimately generates a sampling pattern in k-space that is very similar to the radial acquisition [19]. The salient point to note from these final two experiments is the computational speed. The radial trajectories have nearly three times the number of data points as the spiral trajectory used (46 080 versus 16 384). Consequently, the gridding time for the radial acquisition was nearly three times as long as the gridding time for those other trajectories (36.2 ms versus 12.8 ms). Thus, the number of computations in the table method and, hence, the gridding time, is directly proportional to the number of data points. This is one reason for using gridding rather than the DFT tables, since with the DFT the reconstruction time should be roughly proportional to the square of the number of sampled data points [20].

The radial k-space acquisition in a living pig, the third gridding-table experiment, showed the utility of this method for actual image reconstruction. The radial pulse sequence used in this experiment requires 1300 ms to acquire all of the data while the gridding process, when performed on a modest PC, required 36.1–130.8 ms. This indicates that the entire gridding process could take place during the acquisition itself even using a single processor personal computer for the reconstruction.

If this were implemented on-line, the table-based gridding procedure could be accomplished in real time (limited by the acquisition) and the total lag time would be the time it takes to do the 2D-FFT and display the data. This would allow a variety of pulse sequences to be used in a more real-time fashion as is important in functional MRI (fMRI), perfusion, cardiac iMRI applications where currently the selection is quite limited. In addition, this efficient method would free up computational resources for other desirable activities, especially for groups with access to specialized computational resources.

When examining all of these results it should be remembered that expertise and familiarity with a particular computer language might play a significant role in affecting the speed results of any such experiment. Undoubtedly, results will vary, and the fastest possible code would come from writing, at least the critical "bottleneck" portions of code, in hand-optimized assembly language [12]. Similarly, less attention should be paid to the absolute times found in the tables, but rather the relative differences between conventional and table-based methods. The reason for this is that differences in times for any task are expected with different computer systems. However, use of speed optimized code allows comparison between relative time differences to be made. For example, if conventional regridding on a different computer requires 100 ms (and 582.3 ms here for $W = 4$: Table II), then a table-based method on this alternative hardware would likely require approximately 2.2 ms for table-based gridding since 12.8 ms was needed on our 600-MHz Pentium.

The DFT has the advantage of much simpler mathematics and direct calculation of the final reconstructed image, when compared to the various gridding approaches. However, as shown here, it is computationally intensive and, therefore, usually considered too slow for widespread use, even when using these efficient table-based methods. On the other hand, all of the gridding approaches introduce some error during the change from nonuniformly sampled data to an estimate of a uniformly sampled Fourier data set. The errors result from the need for interpolation, convolution, matrix inversion, or other such operations [9] in the data estimation process [10], [11]. Further, gridding methods create Fourier data, or k-space data, that must still be transformed into the reconstructed image by use of the FFT. The considerable numerical advantage of the FFT, however, can make the total process less computationally intensive than the corresponding DFT reconstruction, at the expense of modest image errors and artifacts.

The speed of the reconstruction really became potentially useful in dynamic settings beginning with the gridding experiments (12.8 ms to 130.8 ms). As mentioned earlier, the primary advantage of the gridding algorithm is that most of the values in $\mathbf{Tg}(i)$ are equal to zero and, using a sparse matrix representation, they need to be neither stored nor calculated. Another advantage of the gridding table over the DFT is that the DFT table must be complex while the convolution function and, therefore, the gridding table, is usually purely real (though the acquired or simulated data is still complex). This further reduces the computational burden by simplifying each complex MAC from four multiplications and four additions in the DFT table to two multiplications and two additions in the gridding table.

The final two steps (i.e., the FFT and the deconvolution) of both the normal and the table-enhanced versions of the gridding algorithm are identical and so the anticipated savings in time are in the first step. The processor is able to prepare the data for the FFT in a minimal amount of time with the table-enhanced method by reducing all of the transcendental or other functions in the first three steps to a single MAC. Even on a single processor this can mean that the gridding process can be done as fast as the data is acquired. In that case, the entire reconstruction process would only be twice as long as the reconstruction of a normal acquisition in which the row FFT is done on the line of data as it is acquired. Table II shows that nonuniform k-space sampling with a table-enhanced gridding algorithm permits gridding during the acquisition, leaving both the row and the column transformations for calculation upon completion of the acquisition.

These table-based methods are highly amenable to parallelization. With a sufficient number of processing elements the DFT-table methods could, theoretically, become even faster than the FFT. This is because the FFT must wait until the entire row is collected before beginning the row transformation. Then it must wait until all the rows are collected before beginning the column FFTs. Only these column FFTs could be parallelized for separate processors. The DFT, on the other hand, can process the result from each data point as it arrives from the scanner. All processing tasks could be equally distributed between up to as many different processing units as there are elements in the $\mathbf{T}(i)$ or $\mathbf{Tg}(i)$ matrices. Groups have already begun exploring multiprocessor and dedicated hardware systems for nonrectilinearly sampled k-space data [7], [8]. If enough elements exist to process one data point before the next arrives, then the entire process is performed in real-time and limited only by the MRI data sampling rate. The end result of using a table method in conjunction with sufficient computer power, as described above, would be the ability to reconstruct an image after the acquisition of each data point or after any arbitrary number of data points. This would even be possible for the gridding-table method, but would be somewhat more natural for the DFT-table method. Even without reconstructing a new image after every data point, the gridding process can be carried out concurrently with the image collection and, thus, the 2D-FFT could be applied as often as desired to reflect the updated data. This would be important in a MR fluoroscopy application where it would be desirable to obtain updated images in subsecond increments. Certain acquisition strategies, such as spiral or rosette, could acquire sufficient information each second to warrant an updated image reconstruction several times/s; and these table-based methods could be a means to facilitate that high rate of image reconstruction, with or without unusual computational equipment. In situations with great computational resources, this method would allow for more processing, such as motion correction, to occur on-line rather than expending all the computational resources on the gridding/reconstruction task.

The use of LUTs to improve the computational speed of algorithms is a generally accepted and common practice. In fact, many compilers, when optimizing for speed, will automatically generate small tables in order to handle complicated branching structures such as switch statements [12]. Tables often allow much more complicated numerical or logical functions to be mapped into a simple memory access function. For example, the table saves tens or hundreds of processor clock cycles (dependent on the processor) by precalculating and storing the results from complicated functions, such as Bessel, exponential and other transcendental functions. In a similar fashion, the information needed to perform complicated operations, such as convolution, can be easily stored in the table for quick use at run time. The net effect is a significant reduction in the computational burden during the reconstruction. This mapping, in

essence, trades megabytes for megaflops. This can be advantageous since it is easier to increase the maximum number of megabytes of available memory than it is to increase the number of Mflops on most available computer systems.

Other methods have been proposed to reduce the number of Mflops needed prior to reconstruction for use in gridding algorithms. Some groups have used fixed-point calculations, rather than the more accurate floating-point, in order to use the faster integer portions of the processor [21]. The table look-up methods developed here for floating point operations could also be extended to fixed-point operations and, thus, gain all of the advantages (and disadvantages) of using integer arithmetic.

The two major limitations to the speed of a table look-up operation are the size of the table and the predictability of the access pattern. Both of these relate to the probability of a given table entry being in the data cache of the processor [12]. A strong computational time penalty is incurred using the table methods each time the algorithm must wait for data from main memory, or worse, from the hard drive. The computational advantage can, therefore, be lost if the tables are poorly organized. Thus, the fastest tables are small and generally accessed in a straightforward manner [12]. Thus, the speed advantages gained by the gridding tables seem to violate some of the known stereotypes of fast tables simply because of their sheer size. The reconstruction tables in this project are not small, but their access patterns are so predictable that they enhance the computational speed significantly, even without explicit concern over cache access.

There are two principal requirements that must be satisfied to be able to use this table method. The first is that the sampled locations of k-space must be known beforehand. In MRI, this is not a difficult requirement since the sample point locations are determined by the pulse sequence, which is created beforehand. The second more limiting, requirement is that the operation replaced by the table must be a linear operation on the sampled data points. This allows a table implementation of the DFT algorithm or the commonly used convolution gridding algorithm. However, other gridding operations (e.g., interpolation) are generally not linear and, therefore, would not qualify for a table implementation as described here [9], [11]. Also, portions of the convolution gridding algorithm that are linear operations on the transformed data rather than on the original sampled data, such as the final deconvolution, cannot be implemented in this type of table. Fortunately, by combining all linear operations on the sampled data, operations like density compensation, convolution, and resampling can all be contained in a single table.

Future table-based methods could include other corrections or enhancements. For instance, Meyer *et al.* include a step for normalizing the energy in each grid point that could be included in the table with no run-time penalty [3]. Other enhancements could include corrections for eddy currents or any other influence that could be anticipated in advance and compensated for with a linear function on the original, nonuniformly sampled, k-space data.

V. Conclusion

In conclusion, a table-based gridding operation can be computed in less time than the acquisition, even when the computations are performed on a single processor PC. Using this method, the rate limiting step in generation of the MR image for many applications is the MRI pulse sequence itself since the table-based method proposed here can be run coincident with data collection. When implemented on-line, this method would allow the total reconstruction time to equal the acquisition time plus the time required for the 2D-FFT. Further improvements in reconstruction time can be achieved with this method using specially designed hardware or multi-processor systems. The current implementation of the table-based method allows for larger convolution kernels than are usually used with other methods. This in turn leads to reduced sidelobes and, therefore, less aliasing energy in the image, all within a computation time (when performed on a 600-MHz Pentium III processor with 512-MB RAM) less than many MRI acquisitions.

The LUTs require a substantial amount of memory, but the predictable access into the table allows for rapid memory access despite their size. The inclusion of the first three steps of the typical gridding algorithm in a single step and the transformation of all complicated functions into simple MACs allow for rapid computation at run time. Together the quick access and the easy computation lead to a very rapid execution even with a single processor personal computer system. LUT methods for reconstruction should be considered as an alternative method for gridding and reconstruction in nonuniformly sampled k-space MRI acquisitions.

References

[1] D. E. Hogg, G. H. MacDonald, R. G. Conway, and C. M. Wade, "Synthesis of brightness distribution in radio sources," *Astronom J.*, vol. 74, pp. 1206–1213, 1969.

[2] H. K. Choi and D. C. Munson, "Direct-Fourier reconstruction in tomography and synthetic aperture radar," *Int. J. Imag. Syst. Tech.*, vol. 9, pp. 1–13, 1998.

[3] C. H. Meyer, B. S. Hu, D. G. Nishimura, and A. Macovski, "Fast spiral coronary artery imaging," *Magn. Reson. Med.*, vol. 28, pp. 202–213, Dec. 1992.

[4] P. Irarrazabal and D. G. Nishimura, "Fast three dimensional magnetic resonance imaging," *Magn. Reson. Med.*, vol. 33, pp. 656–662, May 1995.

[5] X. Zhou, Z. P. Liang, S. L. Gewalt, G. P. Cofer, P. C. Lauterbur, and G. A. Johnson, "A fast spin echo technique with circular sampling," *Magn. Reson. Med.*, vol. 39, pp. 23–27, Jan. 1998.

[6] J. L. Duerk, J. S. Lewin, and D. H. Wu, "Application of keyhole imaging to interventional MRI: A simulation study to predict sequence requirements," *J. Magn. Reson. Imag.*, vol. 6, pp. 918–924, Nov. 1996.

[7] A. B. Kerr, J. M. Pauly, B. S. Hu, K. C. Li, C. J. Hardy, C. H. Meyer, A. Macovski, and D. G. Nishimura, "Real-time interactive MRI on a conventional scanner," *Magn. Reson. Med.*, vol. 38, pp. 355–367, Sep. 1997.

[8] H. Eggers and R. Proska, "Multiprocessor system for real-time convolution interpolation reconstruction," in *Proc. ISMRM 7th Annu. Meeting*, Philadelphia, PA, May 1999, Abstract, p. 95.

[9] J. D. O'Sullivan, "A fast sinc function gridding algorithm for Fourier inversion in computer tomography," *IEEE Trans. Med. Imag.*, vol. MI-4, pp. 200–207, Dec. 1985.

[10] J. I. Jackson, C. H. Meyer, D. G. Nishimura, and A. Macovski, "Selection of a convolution function for Fourier inversion using gridding," *IEEE Trans. Med. Imag.*, vol. 10, pp. 473–478, Sept. 1991.

[11] D. Rosenfeld, "An optimal and efficient new gridding algorithm using singular value decomposition," *Magn. Reson. Med.*, vol. 40, pp. 14–23, Jul. 1998.

[12] R. Booth, *Inner Loops: A Sourcebook for Fast 32-Bit Software Development*. Reading, MA: Addison-Wesley Developers, 1997, p. 364.

[13] R. D. Hoge, R. K. Kwan, and G. B. Pike, "Density compensation functions for spiral MRI ," *Magn. Reson. Med.*, vol. 38, pp. 117–128, Jul. 1997.

[14] J. G. Pipe and P. Menon, "Sampling density compensation in MRI: Rationale and an iterative numerical solution," *Magn. Reson. Med.*, vol. 41, pp. 179–186, Jan. 1999.
[15] F. T. A. W. Wajer, E. Woudenberg, R. de Beer, M. Fuderer, A. F. Mehlkopf, and D. van Ormondt, "Simple equation for optimal gridding parameters," in *Proc. ISMRM 7th Annual Meeting*, Philadelphia, PA, Apr. 1999, Abstract, p. 663.
[16] P. Irarrazabal, B. S. Hu, J. M. Pauly, and D. G. Nishimura, "Spatially resolved and localized real-time velocity distribution," *Magn. Reson. Med.*, vol. 30, pp. 207–212, Aug. 1993.
[17] K. Butts, J. Pauly, A. de Crespigny, and M. Moseley, "Isotropic diffusion-weighted and spiral-navigated interleaved EPI for routine imaging of acute stroke," *Magn. Reson. Med.*, vol. 38, pp. 741–749, Nov. 1997.
[18] H. Sedarat and D. G. Nishimura, "Gridding reconstruction using optimal, shift-variant interpolating kernels," in *Proc. ISMRM 7th Annual Meeting*, Philadelphia, PA, Apr. 1999, Abstract, p. 93.
[19] C. B. Ahn, J. H. Kim, and Z. H. Cho, "High-speed spiral-scan echo planar NMR imaging—I," *IEEE Trans. Med. Imag.*, vol. MI-5, pp. 2–7, Mar. 1986.
[20] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1996, p. 968.
[21] J. R. Liao, "Real-time spiral image reconstruction using fixed-point calculations," in *Proc. ISMRM 7th Annual Meeting*, Philadelphia, PA, Apr. 1999, Abstract, p. 98.