RT-GROG: Parallelized Self-Calibrating GROG for Real-Time MRI

Haris Saybasili,[1,2]* J. Andrew Derbyshire,[1] Peter Kellman,[1] Mark A. Griswold,[3] Cengizhan Ozturk,[2] Robert J. Lederman,[1] and Nicole Seiberlich[3]

A real-time implementation of self-calibrating Generalized Autocalibrating Partially Parallel Acquisitions (GRAPPA) operator gridding for radial acquisitions is presented. Self-calibrating GRAPPA operator gridding is a parallel-imaging-based, parameter-free gridding algorithm, where coil sensitivity profiles are used to calculate gridding weights. Self-calibrating GRAPPA operator gridding's weight-set calculation and image reconstruction steps are decoupled into two distinct processes, implemented in C++ and parallelized. This decoupling allows the weights to be updated adaptively in the background while image reconstruction threads use the most recent gridding weights to grid and reconstruct images. All possible combinations of two-dimensional gridding weights $G_x^m G_y^n$ are evaluated for m,n = {−0.5, −0.4, . . ., 0, 0.1, . . ., 0.5} and stored in a look-up table. Consequently, the per-sample two-dimensional weights calculation during gridding is eliminated from the reconstruction process and replaced by a simple look-up table access. In practice, up to 34× faster reconstruction than conventional (parallelized) self-calibrating GRAPPA operator gridding is achieved. On a 32-coil dataset of size 128 × 64, reconstruction performance is 14.5 frames per second (fps), while the data acquisition is 6.6 fps. Magn Reson Med 64:306–312, 2010. © 2010 Wiley-Liss, Inc.

Key words: GROG; radial trajectories; real-time MRI; parallel MRI; SC-GROG; gridding

Even though non-Cartesian trajectories have been shown to be beneficial in some clinical applications for acquiring MR data (1–3), Cartesian trajectories have gained more widespread use in the clinic as the image reconstruction process is less demanding in terms of computation and execution time. Typically, non-Cartesian data are mapped onto a Cartesian grid, a process known as gridding, prior to image reconstruction. Several methods have been proposed to perform the gridding, such as convolution gridding (4), Uniform Resampling/Block Uniform Resampling (URS/BURS) (5,6), and some iterative methods such as Iterative Next-Neighbor Regridding (INNG) (7) and Deconvolution-Interpolation Gridding (DING) (8). All these methods are computationally demanding and the reconstruction quality depends on several different parameters.

GRAPPA operator gridding (GROG) (9,10) is a gridding algorithm that uses coil sensitivity profiles to calculate k-space shifting weights in the k_x and k_y directions to shift non-Cartesian samples to their corresponding Cartesian grid locations. In GROG, density compensation is straightforward. If more than one non-Cartesian sample is mapped to the same Cartesian grid location, averaging is performed to calculate final Cartesian sample value. GROG algorithm requires an additional Cartesian dataset to use as a calibration data to calculate the weights, as in GRAPPA (11). A self-calibrating version of GROG (SC-GROG) is presented by Seiberlich et al. (12). SC-GROG uses the non-Cartesian data points themselves as a calibration dataset and provides a parameter-free, parallel-imaging-based, robust, self-calibrating, gridding algorithm that can be applied to grid both undersampled and fully sampled datasets. The quality of (SC)-GROG images is on par with conventional gridding (10,12).

In this work, we present the first real-time (RT) implementation of the SC-GROG algorithm for radial acquisitions. SC-GROG's weight-set calculation and image reconstruction steps are decoupled into two separate processes, modified for better performance, and then parallelized for a generic purpose architecture. Decoupling makes it possible to continuously update the k_x and k_y weights for each slice in the background, while the reconstruction threads use the most recent weights for gridding, as done by Guttman et al. (13) and Saybasili et al. (14). Additionally, per-sample two-dimensional (2D) shifting weights $G_x^m G_y^n$ that are normally calculated during gridding are evaluated for all possible "*non-Cartesian to Cartesian*" distance combinations in x and y directions (m,n = {−0.5, −0.4, . . ., 0, 0.1, . . ., 0.5}). The results are stored in a look-up table (LUT) by the weight calculation process. In this way, the per-sample 2D weight calculation is replaced by a simple LUT access, providing increased reconstruction performance.

MATERIALS AND METHODS

Basic Algorithm

SC-GROG uses a (N_c x N_c) weight-set to perform gridding, where N_c represents number of receiver coils used during the acquisition. Let G_1 designate unitary weight set (for unitary shifts in k-space, Δk) in one dimension. The weights for smaller shifts can be calculated by:

$$G_m = (G_1)^m \qquad [1]$$

[1] Translational Medicine Branch, National Institutes of Health/National Heart, Lung and Blood Institute (NHLBI), Department of Health and Human Services (DHHS), Bethesda, Maryland, USA.

[2] Institute of Biomedical Engineering, Bogazici University, Istanbul, Turkey.

[3] Department of Radiology, University Hospitals, Cleveland, Ohio, USA.

*Correspondence to: Haris Saybasili, Ph.D., National Institutes of Health, Building 10, Room B1D416, Bethesda, MD 20892-1061. E-mail: saybasilih@nhlbi.nih.gov

Received 1 June 2009; revised 2 November 2009; accepted 16 December 2009.

DOI 10.1002/mrm.22351
Published online in Wiley InterScience (www.interscience.wiley.com).

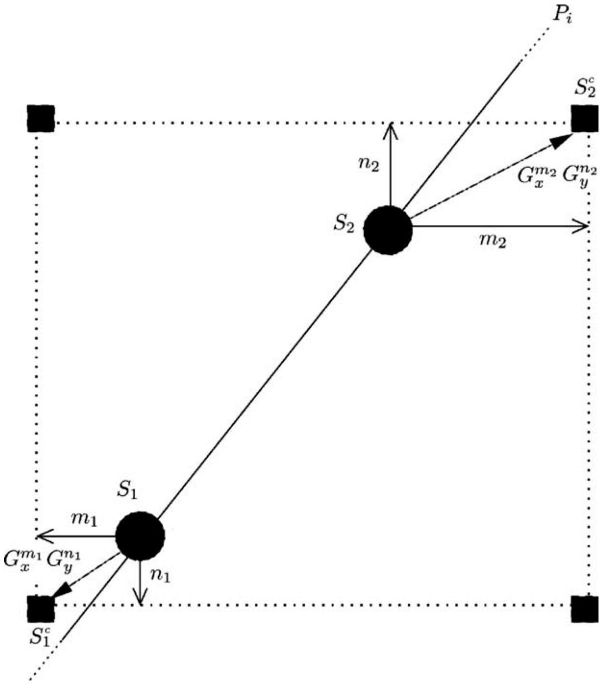

FIG. 1. GROG gridding for two samples S_1 and S_2 of a projection P_i is represented. Gridding weights G_x, and G_y are applied to the samples S_1 and S_2 according to k_x and k_y distances to their closest Cartesian grid neighbors S_1^c and S_2^c, and their contributions to these Cartesian grid points are calculated by $S_1^c = G_x^{m1} G_y^{n1} S_1$ and $S_2^c = G_x^{m2} G_y^{n2} S_2$.

where $-1 < m < 1$. Gridding of a 2D k-space sample $s(k_x, k_y)$ can be generalized using the properties of the GRAPPA operator (9):

$$s(k_x + \delta x, k_y + \delta y) = (G_x)^{\delta x} (G_y)^{\delta y} s(k_x, k_y) \qquad [2]$$

where δx and δy is the distance of this non-Cartesian sample to its nearest Cartesian grid location in the k_x and k_y direction, G_x and G_y are the unitary weights in the k_x and k_y directions, and finally $s(k_x + \delta x, k_y + \delta y)$ represents the gridded signal at the Cartesian location ($k_x + \delta x$, $k_y + \delta y$). A schematic of this operation is shown in Fig. 1. If multiple non-Cartesian samples are mapped to the same Cartesian location, the results are averaged. Therefore, no density compensation function is required.

Calculation of G_x and G_y is not straightforward. First, an angular weight-set G_{θ_i} must be calculated for each projection (i = 1,..., N_p where N_p is the number of projections). G_θ represents the unitary shift operator of size ($N_c \times N_c$) along the given radial ray: $s(\theta, r+1) = G\theta\, s(\theta, r)$. Therefore,

$$G_\theta = pinv[s(\theta, r)]s(\theta, r+1) \qquad [3]$$

where $pinv$ represents the pseudoinverse operator and $s(\theta, r)$ represents multicoil data of projection θ at position r. The set of angular weights can be decoupled into two distinct shifting weights in the k_x and k_y directions. This operation is performed by noticing that $G\theta = (G_x)^m (G_y)^n$, where m and n represent the shifts in the k_x and k_y directions between each consecutive sample along given projection:

$$\begin{aligned} G_{\theta_1} &= G_x^{m_1} G_y^{n_1} \\ G_{\theta_2} &= G_x^{m_2} G_y^{n_2} \\ &\cdots \\ G_{\theta_{Np}} &= G_x^{m_{Np}} G_y^{n_{Np}}. \end{aligned} \qquad [4]$$

Equation 4 can be linearized by taking the matrix logarithm of each side:

$$\begin{aligned} \ln(G_{\theta 1}) &= m_1 \ln(G_x) + n_1 \ln(G_y) \\ \ln(G_{\theta 2}) &= m_2 \ln(G_x) + n_2 \ln(G_y) \\ &\cdots \\ \ln(G_{\theta_{Np}}) &= m_{Np} \ln(G_x) + n_{Np} \ln(G_y). \end{aligned} \qquad [5]$$

After linearization, Eq. 5 can be reorganized as a matrix equation and solved for $ln(G_x)$ and $ln(G_y)$. Subsequently, G_x (and G_y) is calculated by taking the matrix exponentials, $G_x = \exp(ln(G_x))$ (and $G_y = \exp(ln(G_y))$, respectively).

Implementation

RT-GROG has been implemented in C++ and parallelized using Pthreads and OpenMP libraries. The weight-set calculation and reconstruction processes are decoupled to be executed asynchronously in parallel. A

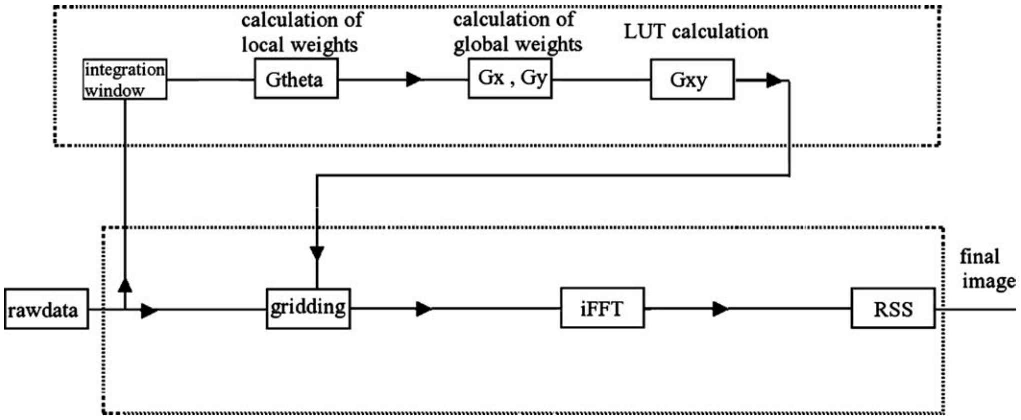

FIG. 2. Block diagram of RT-GROG is represented. Dotted rectangles represent asynchronously executed parallel regions (upper dotted rectangle: weights calculation process; lower dotted rectangle: reconstruction process).

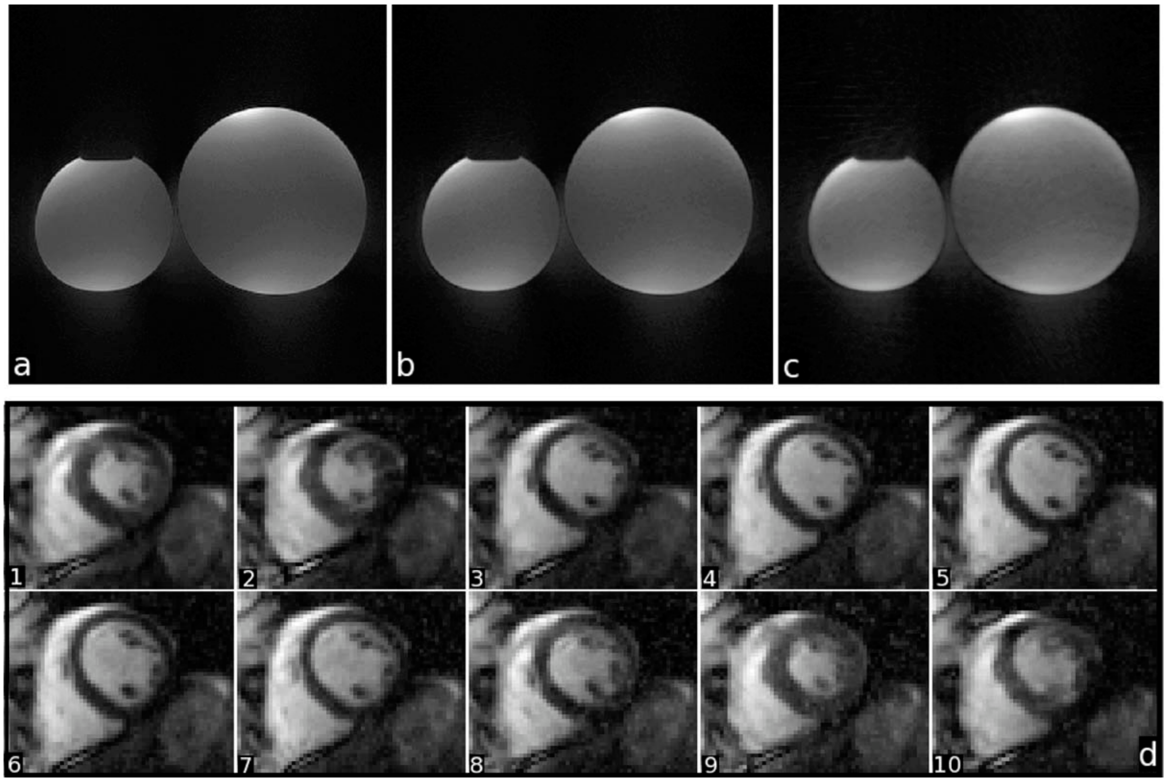

FIG. 3. RT-GROG reconstructed images are presented. Upper images represent phantom images reconstructed from a 15-coil dataset with (**a**) 256 projections, (**b**) 128 projections, and (**c**) 64 projections (256 samples on each). Lower image: (**d**) cardiac cine image series reconstructed in real time using a 32-coil dataset (64 projections, 128 samples). Acquisition time: 151 ms. Reconstruction time: 69 ms.

separate C++ class is dedicated to weights calculation (GrogWsCalculator) and image reconstruction (GrogRegridder) processes. For simple usage, another C++ class *Grog* that encapsulates weights calculation and image reconstruction objects is provided. When an instance of *Grog* is created, weights calculation and image reconstruction object instances are automatically created, and all threads are initiated. Raw data from acquired frames are fed to the instance of *Grog*, and a reconstructed image is obtained as the output of a method, *recon()*.

The weight-set calculation thread continuously updates the weights in the background to track changes in the coil sensitivities, while the reconstruction threads grid non-Cartesian samples using the latest weights and reconstruct images in real time. A total of N_c reconstruction threads are initially created by the Pthreads library, and the same threads with the same allocated memory variables are used during the whole execution to prevent continuous context switching and thus to avoid performance penalty. The weight-set calculation process uses the OpenMP library to allow dynamic change of the number of threads devoted to the weight-set calculation. The number of threads used for the weight-set calculation may be decreased dynamically after an initial set of weights, thereby devoting more resources to image reconstruction.

A block diagram of the RT-GROG implementation is shown in Fig. 2. The upper dotted rectangle describes the weights calculation process, and the lower rectangle represents the reconstruction process. Every acquired frame is fed to the integration window of the weights calculation process to obtain the calibration dataset. Following the calculation of local shifting weights G_0, G_x and G_y are determined. Subsequently, the LUT is computed. The reconstruction process uses most recent LUT to perform gridding. The final image is obtained by combining per-coil images using sum of squares.

RT-GROG was computed in parallel using eight dual-core AMD Opteron 8220 processor (2.8 GHz) on Linux-2.6.16.46-0.12-smp. Pseudoinverse operations and calculation of the eigenvectors were realized using the AMD Core Math Library (http://developer.amd.com/acml). For the matrix operations, the ATLAS library is used (http://math-atlas.sourceforge.net). The FFTW library (http://fftw.org) was used for fast Fourier transformations. The code was compiled on GCC (http://gcc.gnu.org), version 4.2.2. Matrix power, matrix logarithm, and matrix exponential operations were implemented assuming that G_x, G_y are diagonalizable matrices. Let M be a diagonalizable square matrix and V a matrix of eigenvectors of M. M can be projected into another space where its projection M' will be a diagonal matrix:

$$M' = V^{-1} M V. \qquad [6]$$

Thus,

$$\begin{aligned} \ln(M) &= V \ln(M') V^{-1} \\ \exp(M) &= V \exp(M') V^{-1} \\ M^n &= V (M')^n V^{-1}. \end{aligned} \qquad [7]$$

As M' is a diagonal matrix, the calculations in Eq. 7 are realized by replacing every number on the diagonal of M' by its logarithm, its exponential, and its power,

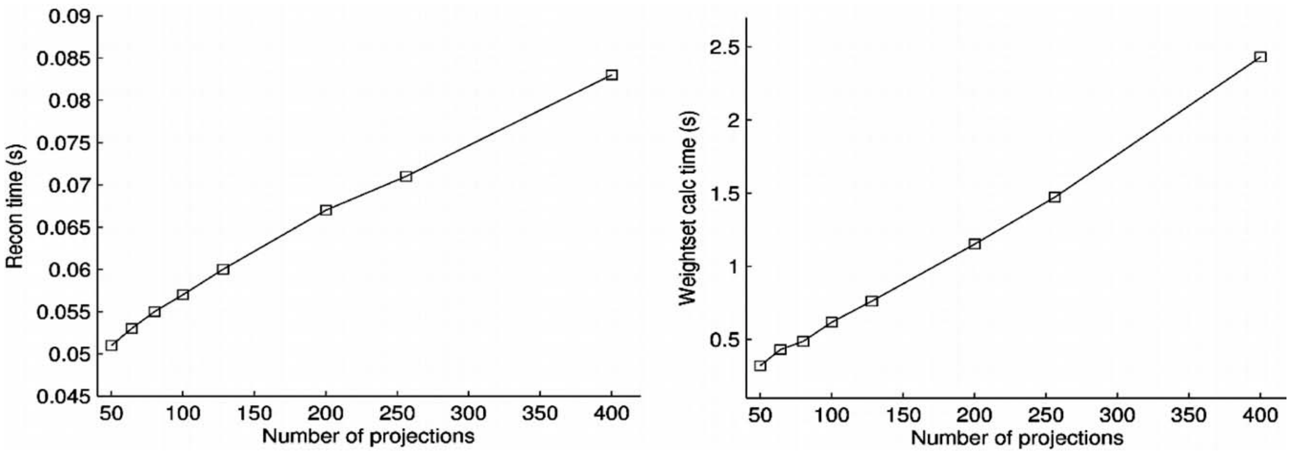

FIG. 4. Image reconstruction and weight-set calculation performances in seconds for 12-coil phantom data and different numbers of projections (number of samples = 256).

respectively. All projections are used to calculate G_x and G_y to achieve the best accuracy. We assume a step size of 0.1 (or 1/10 of unit in *k*-space) for distance measurements between a non-Cartesian sample and its closest Cartesian grid neighbor. For example, the distance of 0.123 is rounded to 0.1, while 0.167 is rounded to 0.2. This assumption makes it possible to precalculate all possible combinations of per-sample 2D weights $(G_x)^m(G_y)^n$ for m, $n = \{-0.5, -0.4,..., 0, 0.1,..., 0.5\}$ and to store them in a LUT G_{xy} by the weight-set calculation thread. In this way, Eq. 2 is simplified to

$$s(k_x + \delta x, k_y + \delta y) = G_{xy}(\delta x, \delta y)\, s(k_x, k_y) \qquad [8]$$

that eliminates per-sample $(G_x)^{\delta x}(G_y)^{\delta y}$ operations during the gridding. Reconstruction threads only calculate the distance (δx,δy) of each sample to its closest Cartesian grid neighbor and retrieve the appropriate precalculated 2D weight set from LUT. Consequently, reconstruction performance is greatly improved.

Data Acquisition

For validation, several phantom datasets were acquired using a short, wide-bore Siemens Magnetom Espree 1.5 T (Siemens Medical Solutions, Erlangen, Germany). The sequence was 2D radial True Fast Imaging with Steady State Precession (TrueFisp) with flip angle 45°, pulse repetition time 3.76 ms, field of view 340 mm. A total of six to 15 receiver coils were used to acquire 50–400 projections with 256 samples. Images were reconstructed in real time and results were analyzed in terms of performance and image quality.

Following in vitro validation, several real-time cardiac images of a healthy subject were acquired using a Siemens Magnetom Avanto 1.5 T (Siemens Medical Solutions), using the same sequence with fewer projections (64 and 96) and fewer samples (128 and 192), using 32 receiver coils (Invivo Corporation, Orlando, FL). The human subject protocol (NCT00720460) was approved by the NHLBI Institutional Review Board. All subjects consented to participate in writing. The pulse repetition time was 2.36 ms for 128 sample acquisitions and 2.86 ms for 192 sample acquisitions.

RESULTS

Image Quality

Phantom images were reconstructed using 6-15 coils datasets with 50–400 projections. Figure 3a-c represents images reconstructed from data acquired with 15 coils with $N_p = 256$, 128, 64, from left to right.

Free-breathing, nongated, short-axis, cardiac data were acquired with 32 coils (192 × 96, 128 × 64, and 192 × 64). Figure 3d represents real-time reconstructed cine images of a beating heart from a 32-coil dataset on a short axis (64 projections, 128 samples). These images show that a fixed set of SC-GROG weights may be used to grid multiple consecutive acquisitions.

Performance

The performances of the image reconstruction and weight-set calculation (in seconds) on 12-coil phantom data with different numbers of projections are represented in Fig. 4. With a 32-coil cardiac dataset, the reconstruction was up to three times faster than data acquisition. For acquisitions of 128 × 96 (192 × 96, 128 × 64, respectively), the reconstruction rate was 13.2 fps (10.0 fps and 14.5 fps respectively), while the data acquisition rate was 4.4 fps (3.6 fps and 6.6 fps, respectively).

DISCUSSION

When reconstructing images from a 12-coil phantom dataset with RT-GROG, image reconstruction speed was between 52 and 83 ms (12.0 and 19.2 fps), while the data acquisition was between 188 and 1504 ms (0.6 and 5 fps). Reconstruction speeds of 69–101 ms (10.0–14.5 fps) were achieved with a 32-coil cardiac dataset, while the data acquisition speeds were 151–274 ms (3.6–6.6 fps). The weight-set update speeds were between 321 and 2431 ms for the phantom dataset and 294 and 646 ms for the cardiac dataset. RT-GROG reconstruction was faster than the data acquisition, even with smaller matrix sizes, such as 128 × 32 (18 fps reconstruction, 13 fps acquisition). However, because SC-GROG does not fill up subsampled *k*-space data, reconstructed images suffered in terms of quality. View sharing could be used, but it is

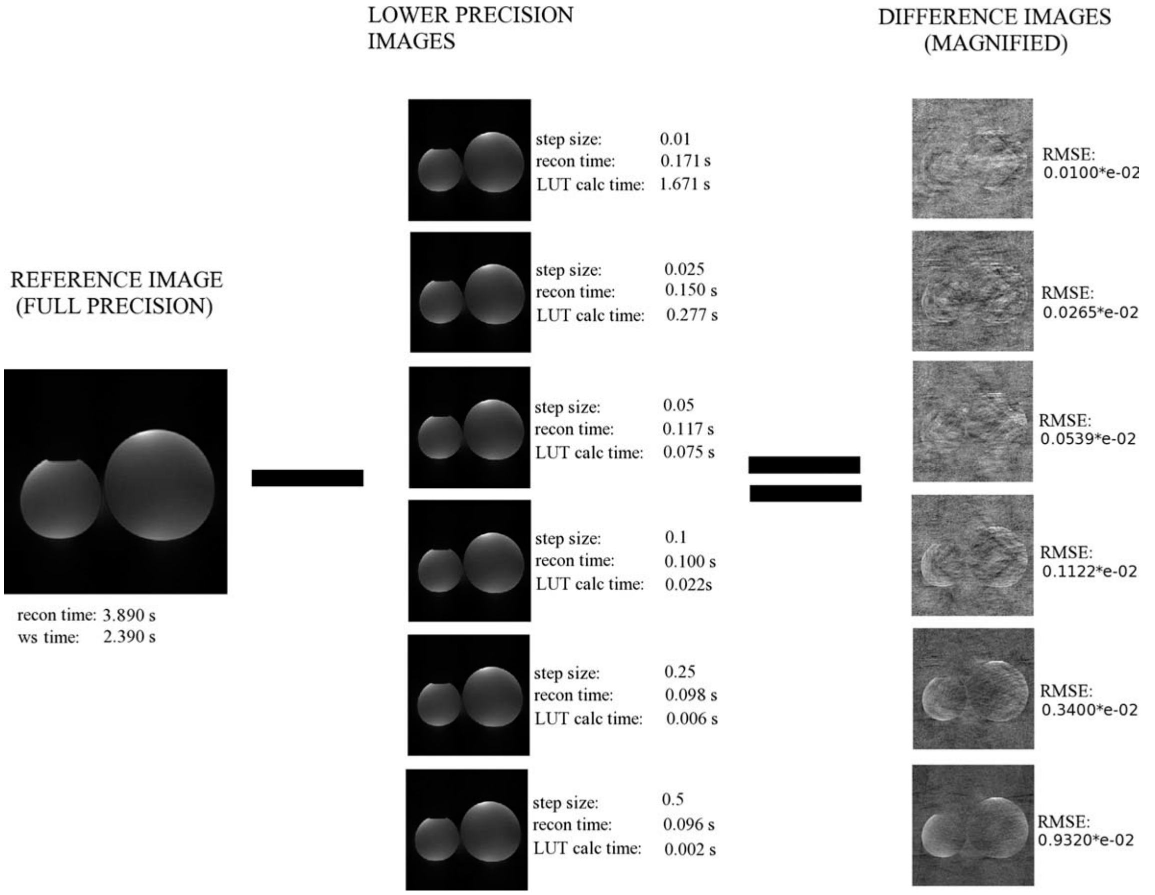

FIG. 5. The effects of different step sizes on image quality and computation times are represented. 15-coil phantom data with 256 projections (256 samples) are reconstructed using full-precision SC-GROG as a reference image (leftmost image). Subsequently, the same frame is reconstructed using different step sizes, and the difference images are calculated to compute root-mean-square-error values for each case.

not preferred because it introduces blurring due to motion. Further improved image quality on undersampled datasets may be obtained by using receiver coil sensitivity profiles to estimate missing non-Cartesian data samples prior or subsequent to gridding (15,16).

MR image reconstruction implicitly assumes that signal encoding employs perfectly linear gradients. In practice, gradient nonlinearities, spatially varying inhomogeneities, and time-varying eddy currents cause deviations from this ideal. Furthermore, MR gradient coils have independent errors and delays that can cause deviations from the expected *k*-space trajectory. In general, it is quite difficult to be certain of the particular *k*-space location for a sample and ascribe a unique value for it. Prior experience with non-Cartesian MR reconstruction suggests that the *k*-space location must be provided with significantly better precision than one *k*-space sample, but there is little to be gained by using full precision. The effect of different step sizes/precisions on image quality and performance is represented in Fig. 5. The full precision reference image of a phantom is reconstructed using conventional SC-GROG. Subsequently, the same frame is reconstructed using RT-GROG with different step sizes, and root-mean-square-error values are calculated for each case. The full precision image reconstruction is 39 times slower than RT-GROG reconstruction. In theory, changes in the step size should not affect RT-GROG reconstruction performance as only one LUT access per sample is required. However, with small step sizes, the weights calculation process uses shared system resources more aggressively and consequently slows down the reconstruction. Let t be the LUT computation time for a given step size s. Increasing the precision by n implies n^2 times bigger LUT; thus, theoretically LUT computation time increases to n^2t. In practice, the overall slowdown in LUT calculations with increased precisions is more than the theoretical values due to increased system overload. Our results indicate that a step size of 0.1 provides the best compromise between image quality and computational efficiency on our current hardware: reconstruction performance stabilizes (~0.100 sec for step sizes $\geq$0.1), and reconstructed images are virtually identical to full precision images with a step size of 0.1. Additionally, LUT computation time is negligible (22 ms) when compared to SC-GROG weights calculation time (2.390 sec). On a 32-coil dataset of size 128 × 64, SC-GROG weights are calculated in 294 ms, LUT with a step size of 0.1 is computed in 70 ms, and LUT with a step size of 0.05 in 285 ms. To further investigate our step size choice, frames were reconstructed using conventional SC-GROG (with full precision) and the discretized LUT version with a step size of

Table 1
Reconstruction Performance Comparison (in Seconds) Between Conventional Parallelized SC-GROG and RT-GROG on 32-Coil Datasets

	Matrix size			
	128 × 64	128 × 96	192 × 64	192 × 96
SC-GROG	1.412	1.972	2.285	3.4525
RT-GROG	0.069	0.076	0.101	0.123

0.1. The signal difference between these images was less than 40% of the SD of the noise of the full precision image, supporting the assertion that artifacts from using a step size of 0.1 are below the noise floor and essentially negligible. Please note that smaller step sizes should be preferred on faster computer hardware to improve image quality.

Reconstruction performance comparison between conventional SC-GROG and RT-GROG is given in Table 1. Thirty-four-fold performance increase is observed with RT-GROG for the reconstruction of 32-coil, 192 × 96 data over conventional, parallelized SC-GROG. A substantial comparison of the (SC-)GROG and conventional gridding approaches for non-Cartesian MRI reconstruction is given in the journal papers on GROG (10,12). With the GROG algorithm, gridding of a non-Cartesian sample requires N_c weighting coefficients. Convolution gridding uses a 2D kernel with variable size, *MxN*. Therefore, GROG is computationally more efficient than convolution gridding for $N_c \leq$ MxN. If $N_c >$ MxN, the convolution gridding method will map the non-Cartesian samples onto a Cartesian grid faster than GROG. However, considering the additional calculations required by convolution gridding (e.g., nontrivial density compensation with some trajectories), we believe that GROG's reconstruction performance will be better than (or at least on par with) a convolution gridding method in a clinical environment where the number of receiver coils is generally around eight to 12. Parameter-free reconstruction is beneficial in the clinic as it eliminates imaging errors due to parameterization. Therefore, RT-GROG should be preferable over conventional gridding even if it falls behind in terms of performance when high numbers of receiver coils are used.

LUT computation increased the weights calculation time by 70 ms (on average) on a 32-coil cardiac dataset. Considering the reconstruction performance gain (up to 34× on a cardiac dataset) and the fact that one fixed set of SC-GROG coefficients can grid multiple consecutive acquisitions, this additional 70 ms is considered to be reasonable.

Our results emphasize 32-coil acquisitions on a 16-core workstation. It may be difficult to provide such a workstation on a basic clinical environment. However, 8- to 12-coil acquisitions provide decent image quality, imply faster reconstruction performance, and can be reconstructed in real time on a modern quad-core workstation. If implemented on a Graphics Processing Unit (GPU), implementation costs may be further decreased. Scanner manufacturers are now providing multicore reconstruction computers. Thus, we believe that RT-GROG can be employed for clinical purposes with minimum effort. Additionally, OpenMP parallelism of the weight-set calculation thread may permit real-time synchronous SC-GROG reconstruction (a new set of weights per acquisition without delays) if enough system resources are dedicated to the weight-set calculation process. Our implementation can also be applied to constant-angular-velocity and reordered constant-linear-velocity spiral trajectories (12).

CONCLUSION

Highly parallel low latency real-time SC-GROG implementation for radial acquisitions was developed and demonstrated. The weight-set calculation for SC-GROG and the image reconstruction steps are separated into two individual processes and implemented in parallel to run asynchronously in real time on a general-purpose architecture. Following the calculation of the two one-dimensional gridding weights G_x and G_y, all possible combinations of per-sample 2D weights $(G_x)^m$ $(G_y)^n$ are precalculated for m, n = {−0.5, −0.4,... 0, 0.1,... 0.5} and stored in a LUT by the weight-set calculation process. Consequently, the per-sample 2D weight calculation during gridding is eliminated from the reconstruction process and replaced by a simple LUT access. In this way, reconstruction performance is improved to better suit real-time requirements. RT-GROG is autocalibrated, parameter free, and highly parallel to overcome slice-plane changes during real-time MRI. It is possible to dynamically change the number of threads associated to weights calculation for improved autoadaptation. Up to 34-fold reconstruction performance increase is obtained over conventional SC-GROG when reconstructing a 32-coil dataset.

ACKNOWLEDGMENT

The authors are grateful to our clinical research coordinator Annette Stine, RN. This work is supported by the Division of Intramural Research, National Heart, Lung and Blood Institute, National Institutes of Health, USA, Z01-HL005062-07 (RJL). NHLBI and Siemens Medical Systems have a Cooperative Research and Development Agreement that includes real-time MRI scan control and display.

REFERENCES

1. Glover GH, Pauly JM. Projection reconstruction techniques for reduction of motion effects in MRI. Magn Reson Med 1992;28:275–289.
2. Meyer CH, Hu BS, Nishimura DG, Macovski A. Fast spiral coronary artery imaging. Magn Reson Med 1992;28:202–213.
3. Pipe JG. Motion correction with PROPELLER MRI: application to head motion and free-breathing cardiac imaging. Magn Reson Med 1999;42:963–969.
4. Jackson JI, Meyer CH, Nishimura DG, Macovski A. Selection of a convolution function for Fourier inversion using gridding (computerised tomography application). IEEE Trans Med Imaging 1991;10: 473–478.
5. Rosenfeld D. An optimal and efficient new gridding algorithm using singular value decomposition. Magn Reson Med 1998;40:14–23.
6. Rosenfeld D. New approach to gridding using regularization and estimation theory. Magn Reson Med 2002;48:193–202.

7. Moriguchi H, Duerk JL. Iterative next-neighbor regridding (INNG): improved reconstruction from nonuniformly sampled k-space data using rescaled matrices. Magn Reson Med 2004;51:343– 352.

8. Gabr RE, Aksit P, Bottomley PA, Youssef ABM, Kadah YM. Deconvolution-interpolation gridding (DING): accurate reconstruction for arbitrary k-space trajectories. Magn Reson Med 2006;56:1182–1191.

9. Griswold MA, Blaimer M, Breuer F, Heidemann RM, Mueller M, Jakob PM. Parallel magnetic resonance imaging using the GRAPPA operator formalism. Magn Reson Med 2005;54:1553–1556.

10. Seiberlich N, Breuer FA, Blaimer M, Barkauskas K, Jakob PM, Griswold MA. Non-Cartesian data reconstruction using GRAPPA operator gridding (GROG). Magn Reson Med 2007;58:1257–1265.

11. Griswold MA, Jakob PM, Heidemann RM, Nittka M, Jellus V, Wang K, Kiefer B, Haase A. Generalized autocalibrating partially parallel acquisitions (GRAPPA). Magn Reson Med 2002;47:1202–1210.

12. Seiberlich N, Breuer F, Blaimer M, Jakob P, Griswold M. Self-calibrating GRAPPA operator gridding for radial and spiral trajectories. Magn Reson Med 2008;59:930–935.

13. Guttman MA, Kellman P, Dick AJ, Lederman RJ, McVeigh ER. Real-time accelerated interactive MRI with adaptive TSENSE and UNFOLD. Magn Reson Med 2003;50:315–321.

14. Saybasili H, Kellman P, Griswold MA, Derbyshire JA, Guttman MA. HTGRAPPA: real-time B1-weighted image domain TGRAPPA reconstruction. Magn Reson Med 2009;61:1425–1433.

15. Seiberlich N, Breuer F, Heidemann R, Blaimer M, Griswold M, Jakob P. Reconstruction of undersampled non-Cartesian datasets using pseudo-Cartesian GRAPPA in conjunction with GROG. Magn Reson Med 2008;59:1127–1137.

16. Seiberlich N, Breuer FA, Ehses P, Moriguchi H, Blaimer M, Jakob PM, Griswold MA. Using the GRAPPA operator and the generalized sampling theorem to reconstruct undersampled non-Cartesian data. Magn Reson Med 2009;61:705–715.