Implementation of SENSE with Arbitrary k-space Trajectories

Cong Zhang

Dec. 6, 2004

A. Introduction

A critical issue in MRI is the scan time. Several methods have been proposed to utilize parallel signal acquisition with multiple receiver coils for the purpose of reducing scan time. Spatial coil sensitivity is exploited to reduce the number of gradient-encoding steps, without compromising spatial resolution or the field of view (FOV). Practical implementations of parallel MR were presented by Sodickson and Manning [4] and Pruessmann, et al. [3]. Sodickson, et al. [4, 1] proposed a method known as SiMultaneous Acquisition of Spatial Harmonics (SMASH) which is a k-space implementation of parallel MR imaging. Pruessmann et al. presented a general formulation and performance analysis of the image domain sensitivity encoding method (SENSE). It is reported that the SENSE reconstruction in [3] is a computationally intensive process. For an image of size $N \times N$ being reconstructed, the complexity of a general reconstruction is $O(N^6)$. A large amounts of memory and processing time are required. Calculating a 128×128 image, e.g., would require more than 1012 operations and about 1 GB of memory. In [2], Pruessmann et al. propose an efficient implementation. It is based on the idea of performing reconstruction iteratively. By combining fast Fourier transform (FFT) with forward and reverse gridding operations, the complexity of a single iteration is reduced drastically from $O(N^4)$ to $O(N^2 \log N)$. The number of iterations necessary to achieve a given accuracy can also be reduced by using several measures of numerical optimization. Therefore, the computation time for non-Cartesian SENSE reconstruction is considerably reduced, making sensitivity-encoded imaging practical with arbitrary acquisition patterns.

The purpose of my project was to implement the SENSE image reconstruction algorithm proposed by Pruessmann in [2], evaluate its performance using MRI phantom studies, and identify computational issues that need to be addressed for improving its speed performance.

B. SENSE Reconstruction

Throughout this report, the following notations are used:

n_c the number of receiver coils

n_k the number of sampling positions in k-space

m vector of length $n_c \times n_k$, the complex sample values acquired

v vector of length N^2, the complex pixel values of the reconstructed image

E encoding matrix ($n_c n_k \times N^2$)

D density correction matrix

I intensity correction matrix ($N^2 \times N^2$)

According to [2], image reconstruction consists of three steps.

1. The intermediate image a is calculated:

$$a = IE^H Dm$$

2. An approximation solution b_{approx} of

$$(IE^H DEI)b = a$$

is determined by CG iteration.

3. An approximation solution is obtained by intensity correction of b_{approx}:

$$v_{approx} = Ib_{approx}$$

Figure 1, which is borrowed from [2], schematically shows the implementation of this procedure. The central part is a CG unit. It controls the iteration process.

B.1. Preprocessing Multi-coil Data

The multi-coil data available for this project is from a human volunteer. The raw data can be loaded with `rawloadEX.m` in MatLab. After running it, a 256×1028 matrix is filled. The matrix is basically four 256×257 matrices stacked on top of each other. Each of the four matrices represent one coil. The first row of each matrix is a “baseline” which can be removed. In other words, to obtain sampled k-space data from the four receiver coils, the following code is used:

```
d1 = d(:,1+(1:256));
d2 = d(:,2+256+(1:256));
d3 = d(:,3+512+(1:256));
d4 = d(:,4+768+(1:256));
```

The reconstruected four fully-sampled coil images are shown by Figure 2 - Figure 5

However, there is one problem with the multi-coil data. They were sampled on Cartesian grid in k-space. To evaluate the implementation, k-space raw data sampled along arbitrary trajectories are needed. Therefore, before inputting the multi-coil data to the SENSE reconstruction algorithm, some preprocessing steps are performed. First, the raw data is transformed to iamge domain. Next, the images are transformed to k-space and sampled along a specified k-space trajectory. For my project, 6 interleaved spiral k-space trajectories are used. To simulate undersampling, 3 out of 6 interleaves are discarded. The code for performing those preprocessings is given below:

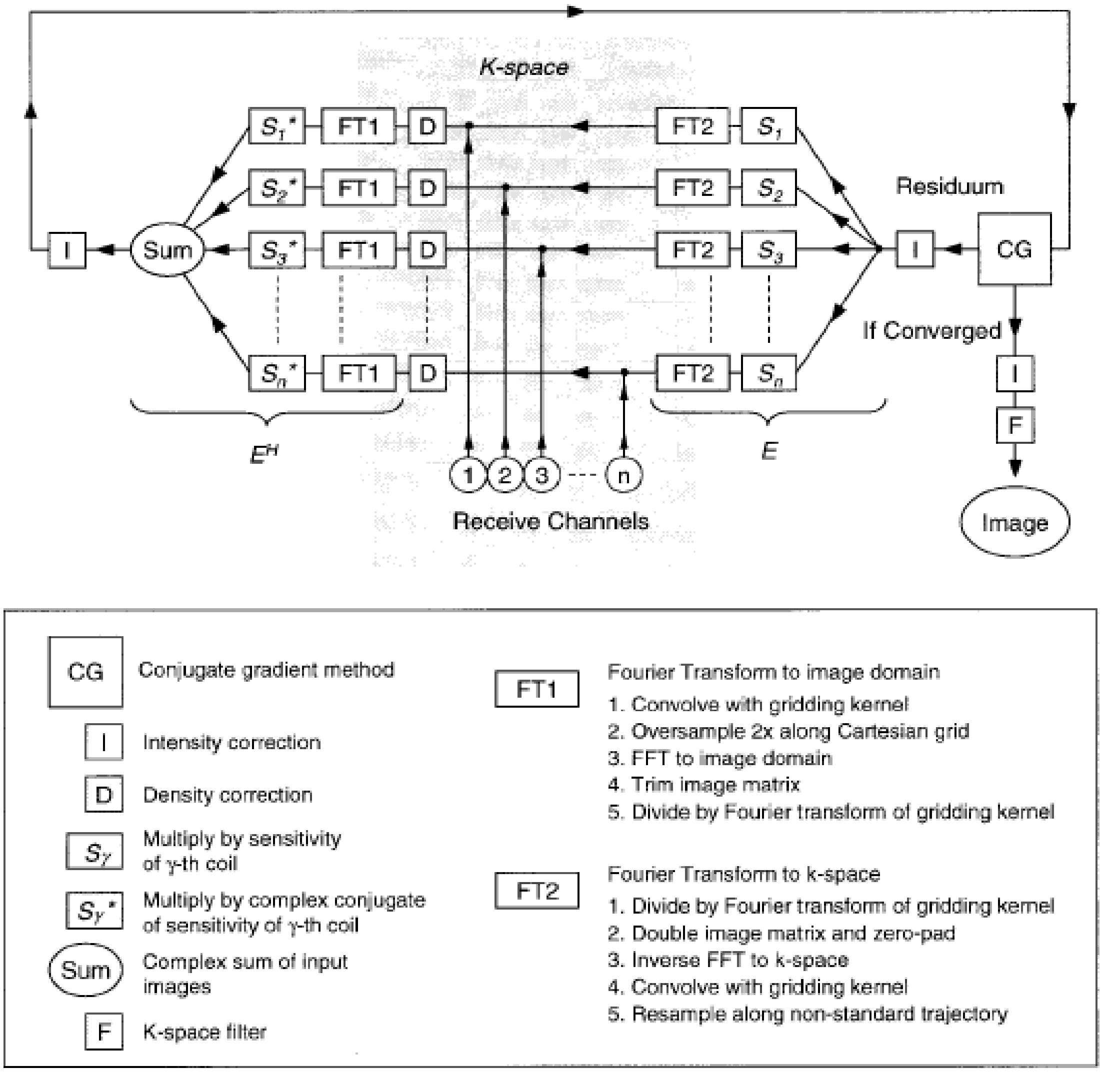

Figure 1: Implementation of iterative image reconstruction (Adapted from [2]). Conjugate gradient (CG) iteration is controlled by the central CG process.

```
% -- Fourier Transform to image domain
im1 = ift(d1);
im2 = ift(d2);
im3 = ift(d3);
im4 = ift(d4);

% -- 6-interleaved spiral k-trajectory
load rt_spiral;

% -- inverse gridding to obtain the data sampled along the spiral,
%    m1,...,m4 are the complex sample values acquired
```

Figure 2: Coil Image 1.

Figure 3: Coil Image 2.

```
ktraj = k;
ktraj = ktraj(:,1:2:6);

m1 = ft2(im1, ktraj, n);
m2 = ft2(im2, ktraj, n);
m3 = ft2(im3, ktraj, n);
m4 = ft2(im4, ktraj, n);
```

`m1, m2, m3, m4` are considered as k-space raw data undersampled along spiral trajectories.

Figure 4: Coil Image 3.

Figure 5: Coil Image 4.

B.2. Coil Sensitivity Maps

Reconstruction of the aliased image with SENSE requires accurate assessment of the coil sensitivties. Since the complex coil sensitivities depend on the coil geometry and oriention, and the sample volume of interest that alters the magnetic field distribution, it is not possible to assess absolute sensitivity values. In-vivo estimation is generally used. In my project, sensitivity maps of the four coils are estimated using the method in [3]. Specifically, single-coil full-FOV images are first reconstructed in a conventional manner. Next, normalization with a root-sum-of-squares (RSS) combined magnitude image is used to remove the image modulus. This is described by the following equation

$$\hat{s}_i(x,y) = \frac{\hat{f}_i(x,y)}{\sqrt{\sum_i |\hat{f}_i(x,y)|^2}} \approx \frac{s_i(x,y)f(x,y)}{\sqrt{\sum_i |s_i(x,y)f(x,y)|^2}} \approx \frac{s_i(x,y)}{\sqrt{\sum_i |s_i(x,y)|^2}} \left[\frac{f(x,y)}{|f(x,y)|}\right]$$

The Matlab code for determining the four coil sensitivity maps is:

```
c1 = abs(im1);
c2 = abs(im2);
c3 = abs(im3);
c4 = abs(im4);

rss = sqrt(c1 .* c1 + c2 .* c2 + c3 .* c3 + c4 .* c4);

cs_map1 = im1 ./ rss;
cs_map2 = im2 ./ rss;
cs_map3 = im3 ./ rss;
cs_map4 = im4 ./ rss;
```

To eliminate noise and avoid errors at object edges, smoothing can be done on the raw map. In [3], polynomial fit was employed.

B.3. Gridding

The FT1 and FT2 modules in SENSE reconstruction are implemented with `ft1.m` and `ft2.m`. The gridding step in FT1 is the same as conventional gridding. The sampled data along the experimental trajectory are convolved with a kaiser-Bessel kernel and resampled using a Cartesian grid. To suppress gridding artifacts, 2x oversampling is used. After FFT, the resulting image is trimmed to the actual FOV and divided by the Fourier transform of the Kaiser-Bessel kernel to compensate for apodization due to the convolution step. Similar steps are performed in FT2 module.

`gridgeneral.m` and `regridgeneral.m` are doing forward and reverse gridding respectively. By default, Kaiser-Bessel kernel with window size of 3 is assumed for both gridding steps.

C. Results

Sensitivity-encoded imaging with iterative reconstruction was evaluated using spiral k-space trajectory. The size of image matrix was 256×256. Data sets with reduced gradient encoding were created by selecting three out of 6 interleaves, corresponding to reduction factor $R = 2.0$.

Sensitivity maps were created from a conventional reconstruction including four full-FOV coil images, using the method described in B.2.. From the reduced data sets, image reconstruction was performed iteratively with density and intensity correction. The iteration was stopped when either required accuracy had been met or a specified number of iterations had been performed. Table 1 shows how CG unit converges with the number of iterations.

Iteration	1	2	3	4	5	10	15	20	25
δ	1	0.23023	0.06680	0.02884	0.01557	0.00300	0.00077	0.00029	0.00014

The resulting images are shown below.

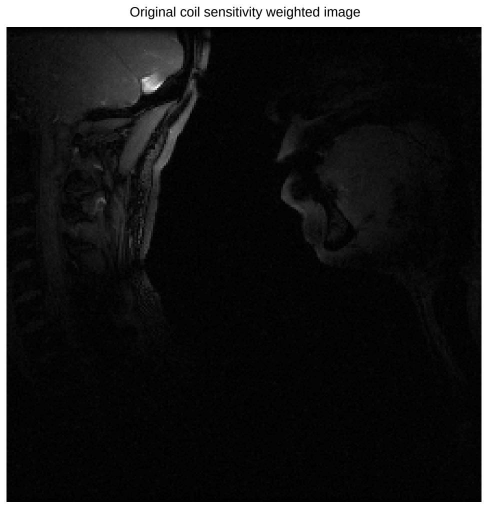

Figure 6: R = 1.

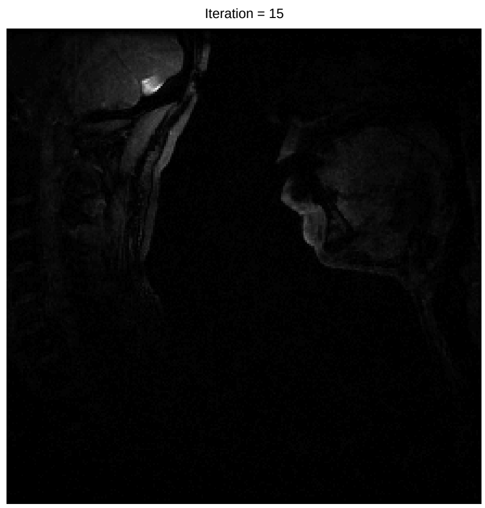

Figure 7: R = 2. After 15 iterations.

To evaluate the speed performance of my SENSE implementation, a profiing on the `sense.m` was conducted. The Profiler in Matlab is tool that shows you where an M-file is spending its time. With the profile summary, you can tell which functions consume the most time and thus where the bottleneck of the speed performance is. Figure 8 is the screenshot of the profile summary after running SENSE for 6 iterations. As can be seen, FT1 module took the most of the time. While `gridgeneral()` consumed over 60% of the total computation time. FFT operations, however, contributed 20 seconds, which is around 10% of the total time. *This finding is different from what I concluded through quantitative performance analysis of SENSE algorithm, which states that FFT consumes the most of the computation time of SENSE.*

Note: To open the Profiler, select **View** → **Profiler** from the MATLAB desktop, or type `profile viewer` in the Command Window. The MATLAB Profiler opens.

screenshot_profiling (600x500x24b bmp)

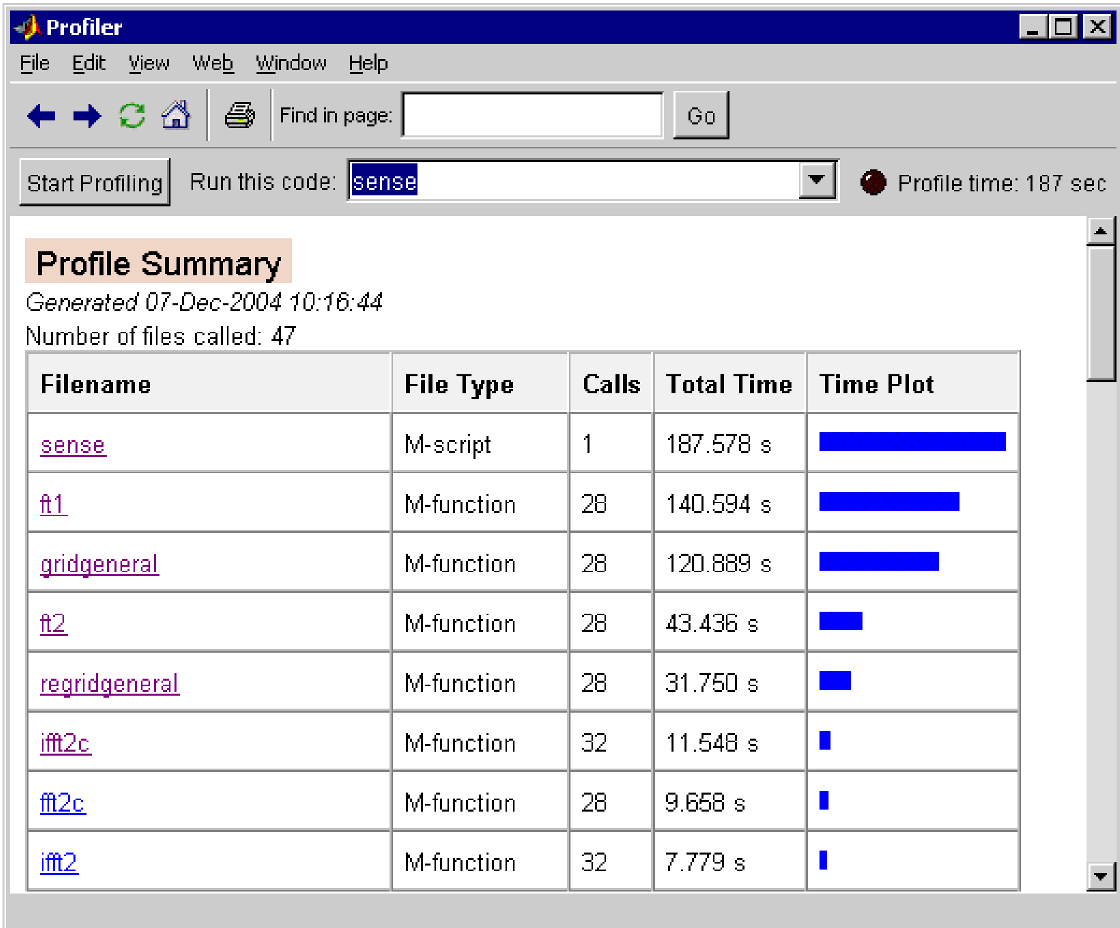

Figure 8: Screenshot of the profiling summary of SENSE.

D. Discussion

The implementation of SENSE reconstruction in [2] has shown how undersampled k-space data along an arbitrary trajectory are reconstructed into an unaliased image. The combination of the CG algorithm with forward and reverse gridding and FFT drastically reduces the computation complexity as opposed to direct methods. However, as can be seen in Figure 8, the computation time is still very long. Six iterations took 180 seconds. This is mainly due to the fact that my implementation in Matlab is not optimized. Therefore, to accelerate the computing speed of SENSE, the following three steps can be taken:

1. Identify the demanding kernel operations with the help of the Profiler. According to the profile summary, *gridding is the most demanding operation in SENSE reconstruction*.

2. Optimize the Matlab implementation. Various techniques can be used, such as avoiding unnecessary computation, changing the algorithm to avoid costly functions, and avoiding re-computation by storing results for future use, etc. This step has a great impact on the overall speed performance of an algorithm. Preliminary optimization has been done by changing the gridding implementation from using sparse matrix to turn k-space trajectory into 2D matrix to looping over the k-space samples (this is called "slow" gridding, but actually not slow), much less time (62 seconds as opposed to 120 seconds) was spent on gridding.

3. Use application-specific hardware acceleration. As pointed out by Pruessmann et al. in [2], the algorithm is perfectly suited for parallel computing. This is because the two computationally intensive operations, gridding and FFT, are carried out at the receiver channels independently. FPGA is a good candidate for hardware acceleration. It combines the flexibility of software with the high performance of custom hardware.

References

[1] W. Kyriakos. Sensitivity profiles from an array of coils for encoding and reconstruction in parallel (SPACE RIP). *Magn Reson Med*, 44(2):301–8, 2000.

[2] K. Pruessmann, M. Weiger, P. Bornert, and P. Boesiger. Advances in sensitivity encoding with arbitary k-space trajectories. *Magn Reson Med*, 46(4):638–51, 2001.

[3] K. Pruessmann, M. Weiger, M. Scheidegger, and P. Boesiger. SENSE: sensitivity encoding for fast MRI. *Magn Reson Med*, 42(5):952–62, 1999.

[4] D. Sodickson and W. Manning. Simultaneous acquisition of spatial harmonics (SMASH): fast imaging with radiofrequency coil arrays. *Magn Reson Med*, 38(4):591–603, 1997.