# CSE 591: GPU Programming

## Case Study: Iterative MRI Reconstruction

Klaus Mueller

Computer Science Department

Stony Brook University

ECE498AL

Programming Massively Parallel Processors

Lecture16-17: Application Case Study – Quantitative MRI Reconstruction





## Objective

- To learn about computational thinking skills through a concrete example
  - Problem formulation
  - Designing implementations to steer around limitations
  - Validating results
  - Understanding the impact of your improvements
- A top to bottom experience!





## Acknowledgements

Sam S. Stone[§], Haoran Yi[§], Justin P. Haldar[†], Deepthi Nandakumar, Bradley P. Sutton[†], Zhi-Pei Liang[†], Keith Thulburin*

[§]Center for Reliable and High-Performance Computing

[†] Beckman Institute for Advanced Science and Technology

*Department of Electrical and Computer Engineering*
*University of Illinois at Urbana-Champaign*

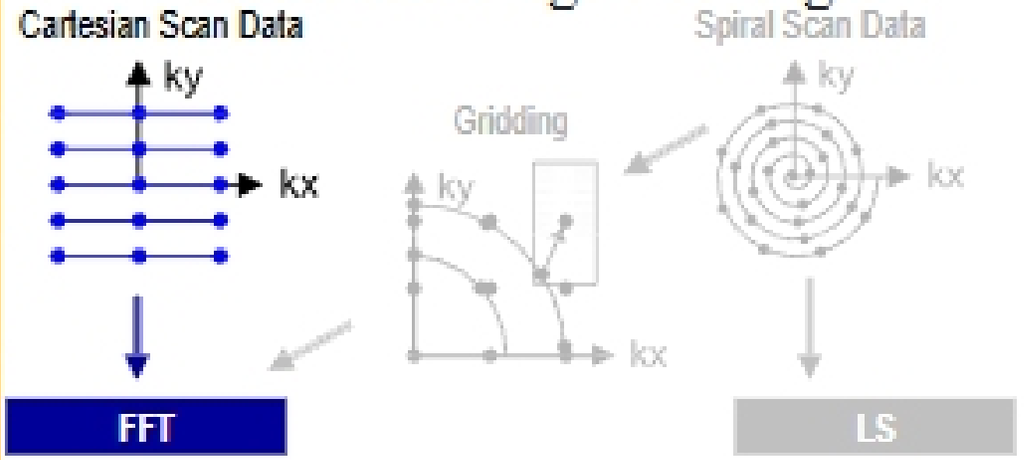
** University of Illinois, Chicago Medical Center*

# Overview

- Magnetic resonance imaging
- Non-Cartesian Scanner Trajectory
- Least-squares (LS) reconstruction algorithm
- Optimizing the LS reconstruction on the G80
  - Overcoming bottlenecks
  - Performance tuning
- Summary



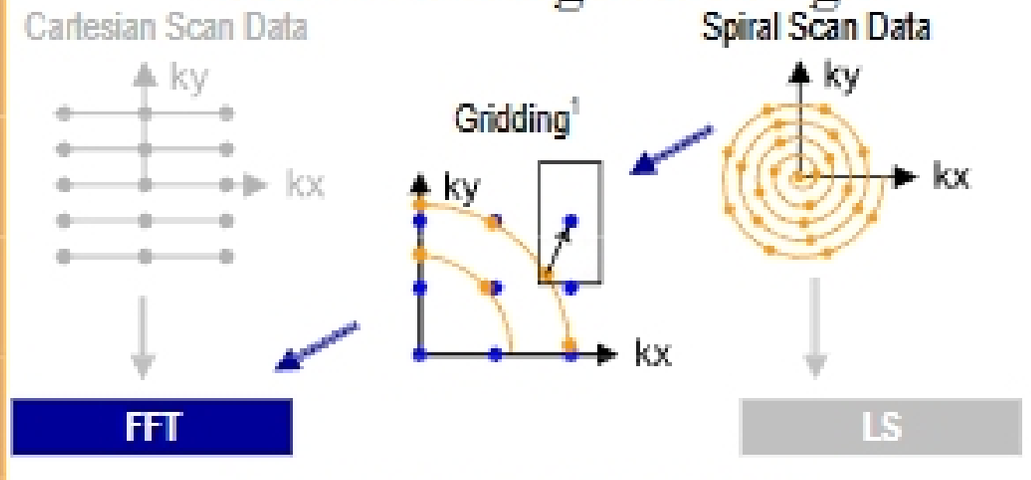

# Reconstructing MR Images




Cartesian scan data + FFT:
Slow scan, fast reconstruction, images may be poor




# Reconstructing MR Images

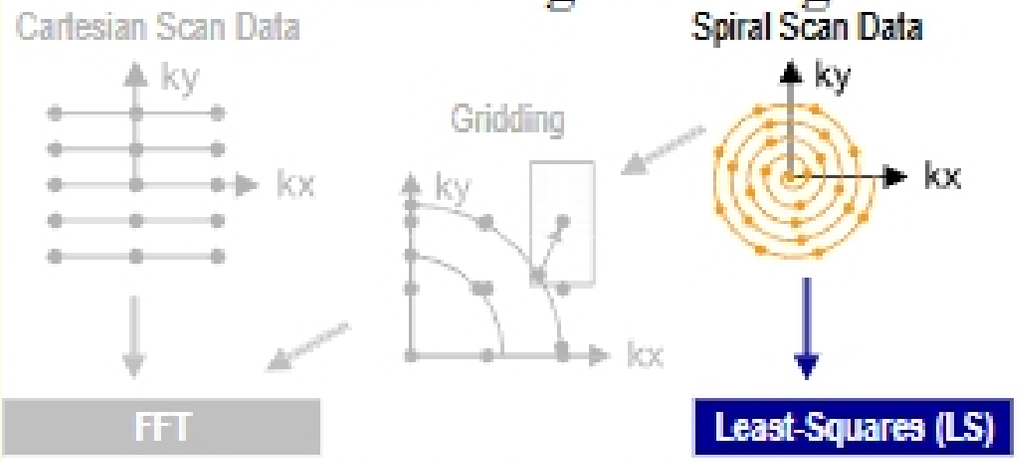



Spiral scan data + Gridding + FFT:
Fast scan, fast reconstruction, better images

[1] Based on Fig 1 of Lustig et al, Fast Spiral Fourier Transform for Iterative MR Image Reconstruction, IEEE Int'l Symp. on Biomedical Imaging, 2004

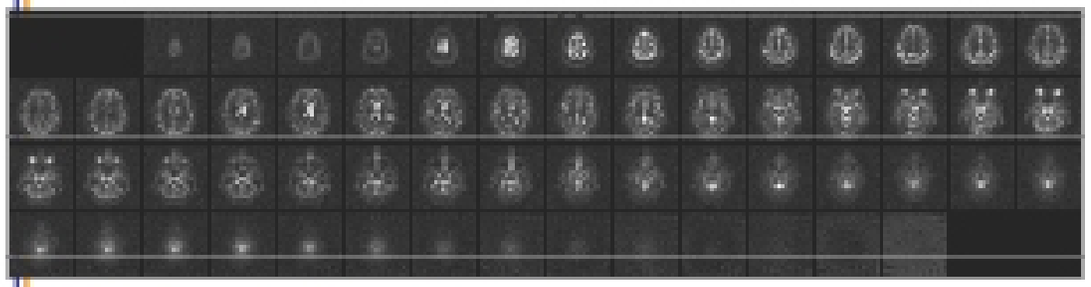



# Reconstructing MR Images




Spiral scan data + LS
Superior images at expense of significantly more computation

# An Exciting Revolution - Sodium Map of



- Images of sodium in the brain
  - Very large number of samples for increased SNR
  - Requires high-quality reconstruction
- Enables study of brain-cell viability before anatomic changes occur in stroke and cancer treatment – within days!

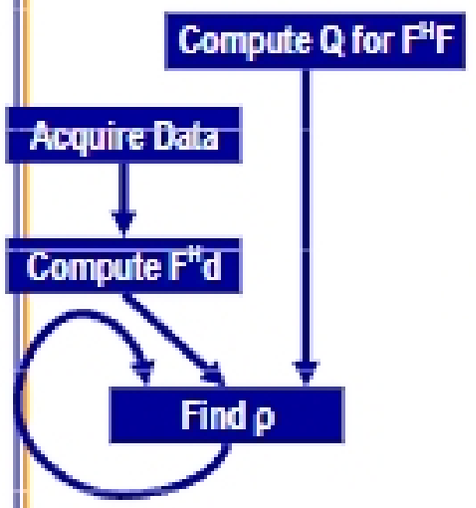
Courtesy of Keith Thulborn and Ian Atkinson, Center for MR Research, University of Illinois at Chicago





# Least-Squares Reconstruction

$$F^H F\rho = F^H d$$




- Q depends only on scanner configuration
- $F^H d$ depends on scan data
- ρ found using linear solver
- Accelerate Q and $F^H d$ on G80
  - Q: 1-2 days on CPU
  - $F^H d$: 6-7 hours on CPU
  - ρ: 1.5 minutes on CPU





```
for (m = 0; m < M; m++) {

  phiMag[m] = rPhi[m]*rPhi[m] +
              iPhi[m]*iPhi[m];

  for (n = 0; n < N; n++) {
    expQ = 2*PI*(kx[m]*x[n] +
                 ky[m]*y[n] +
                 kz[m]*z[n]);

    rQ[n] +=phiMag[m]*cos(expQ);
    iQ[n] +=phiMag[m]*sin(expQ);
  }
}
```

(a) Q computation

```
for (m = 0; m < M; m++) {

  rMu[m] = rPhi[m]*rD[m] +
           iPhi[m]*iD[m];
  iMu[m] = rPhi[m]*iD[m] -
           iPhi[m]*rD[m];

  for (n = 0; n < N; n++) {
    expFhD = 2*PI*(kx[m]*x[n] +
                   ky[m]*y[n] +
                   kz[m]*z[n]);

    cArg = cos(expFhD);
    sArg = sin(expFhD);

    rFhD[n] +=  rMu[m]*cArg -
                iMu[m]*sArg;
    iFhD[n] +=  iMu[m]*cArg +
                rMu[m]*sArg;
  }
}
```

(b) FHd computation

Q v.s. $F^H D$





# Algorithms to Accelerate

```
for (m = 0; m < M; m++) {

  rMu[m] = rPhi[m]*rD[m] +
           iPhi[m]*iD[m];
  iMu[m] = rPhi[m]*iD[m] -
           iPhi[m]*rD[m];

  for (n = 0; n < N; n++) {
    expFhD = 2*PI*(kx[m]*x[n] +
                   ky[m]*y[n] +
                   kz[m]*z[n]);

    cArg = cos(expFhD);
    sArg = sin(expFhD);

    rFhD[n] +=  rMu[m]*cArg -
                iMu[m]*sArg;
    iFhD[n] +=  iMu[m]*cArg +
                rMu[m]*sArg;
  }
```

- Scan data
  - M = # scan points
  - kx, ky, kz = 3D scan data
- Pixel data
  - N = # pixels
  - x, y, z = input 3D pixel data
  - rFhD, iFhD= output pixel data
- Complexity is O(MN)
- Inner loop
  - 13 FP MUL or ADD ops
  - 2 FP trig ops
  - 12 loads, 2 stores