Nonrigid Motion Correction in 3D Using Autofocusing with Localized Linear Translation

Cheng et al.

CIARA, SUJOY & DAVID

Motivation

Nonrigid body motion can be well approximated as simple linear translations

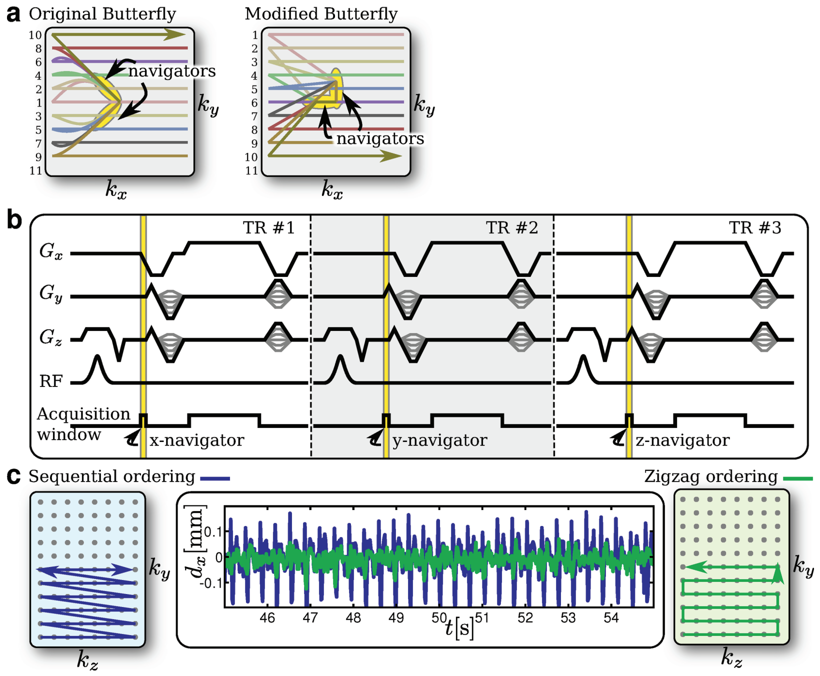
Propose a novel navigation strategy based on the so-called **"Butterfly" navigators**, modifications of the **spin-warp sequence**

With a **32-channel abdominal coil**, sufficient number of motion measurements found to approximate possible linear motion paths for every image voxel

Applied to free-breathing **abdominal patient studies** and reduction in artifacts was observed

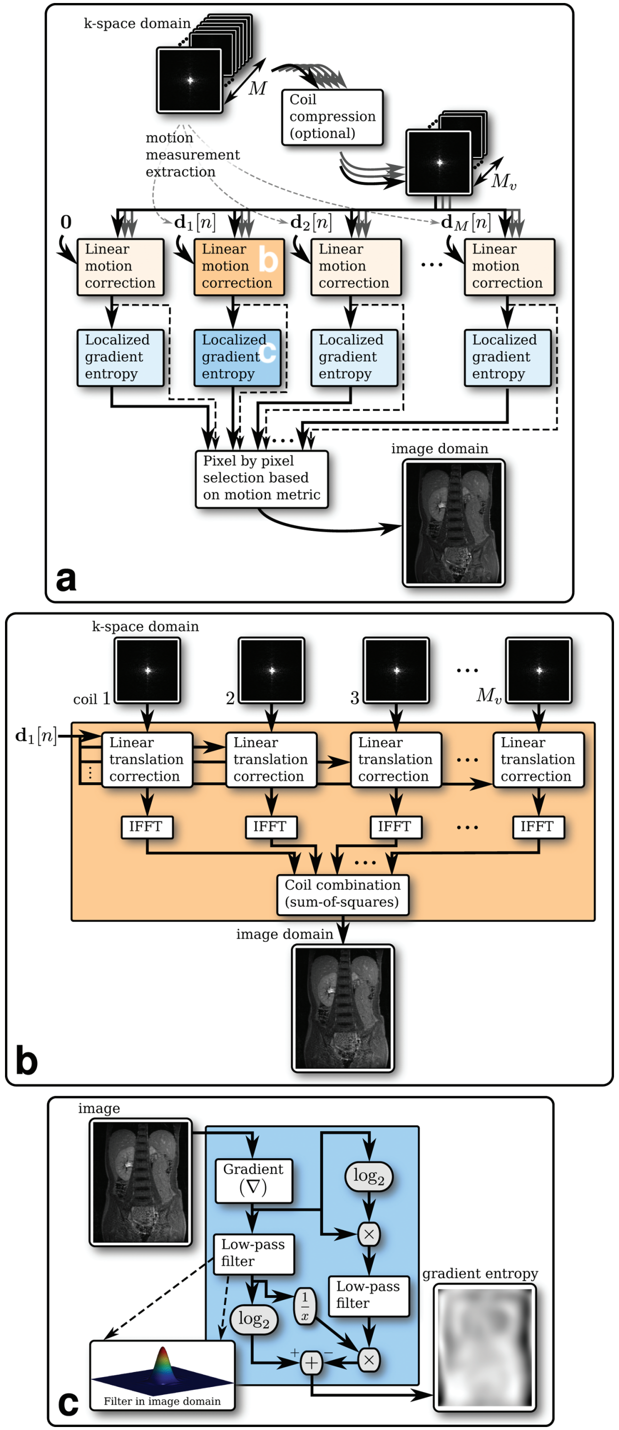
Butterfly Navigator Sequence

- (a) 2D Butterfly Trajectory
- (b) Pulse sequence timing diagram for an example 3D Butterfly trajectory
- (c) Phase/slice encoding effect on motion measurement



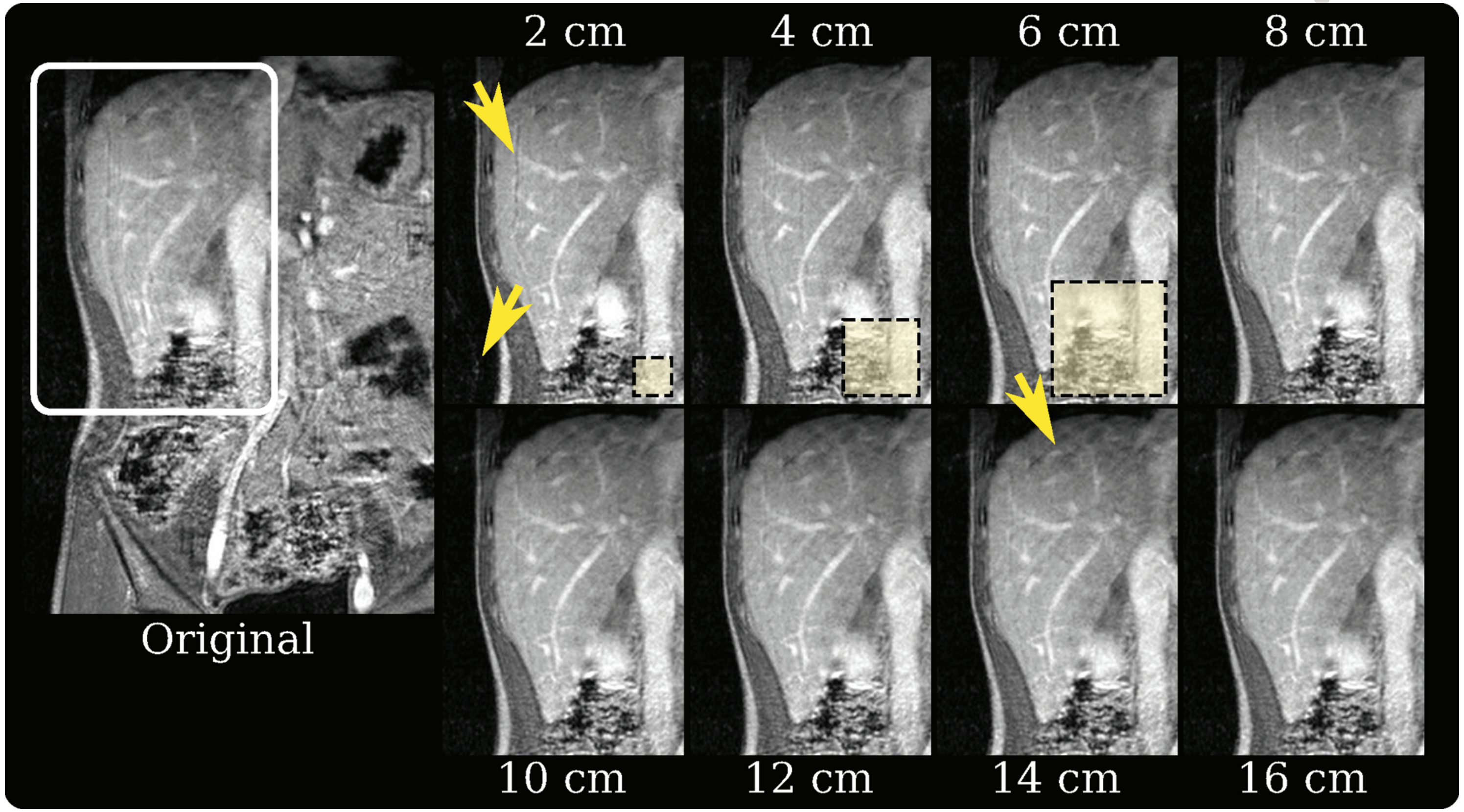
Nonrigid Motion Correction Overview

- (a) Correction scheme using data from an M-channel coil array
- (b) Linear translational motion correction suing motion measurement $\mathbf{d}_1[n]$
- (c) Localized gradient entropy calculation to determine which correction yielded the best result for a particular location



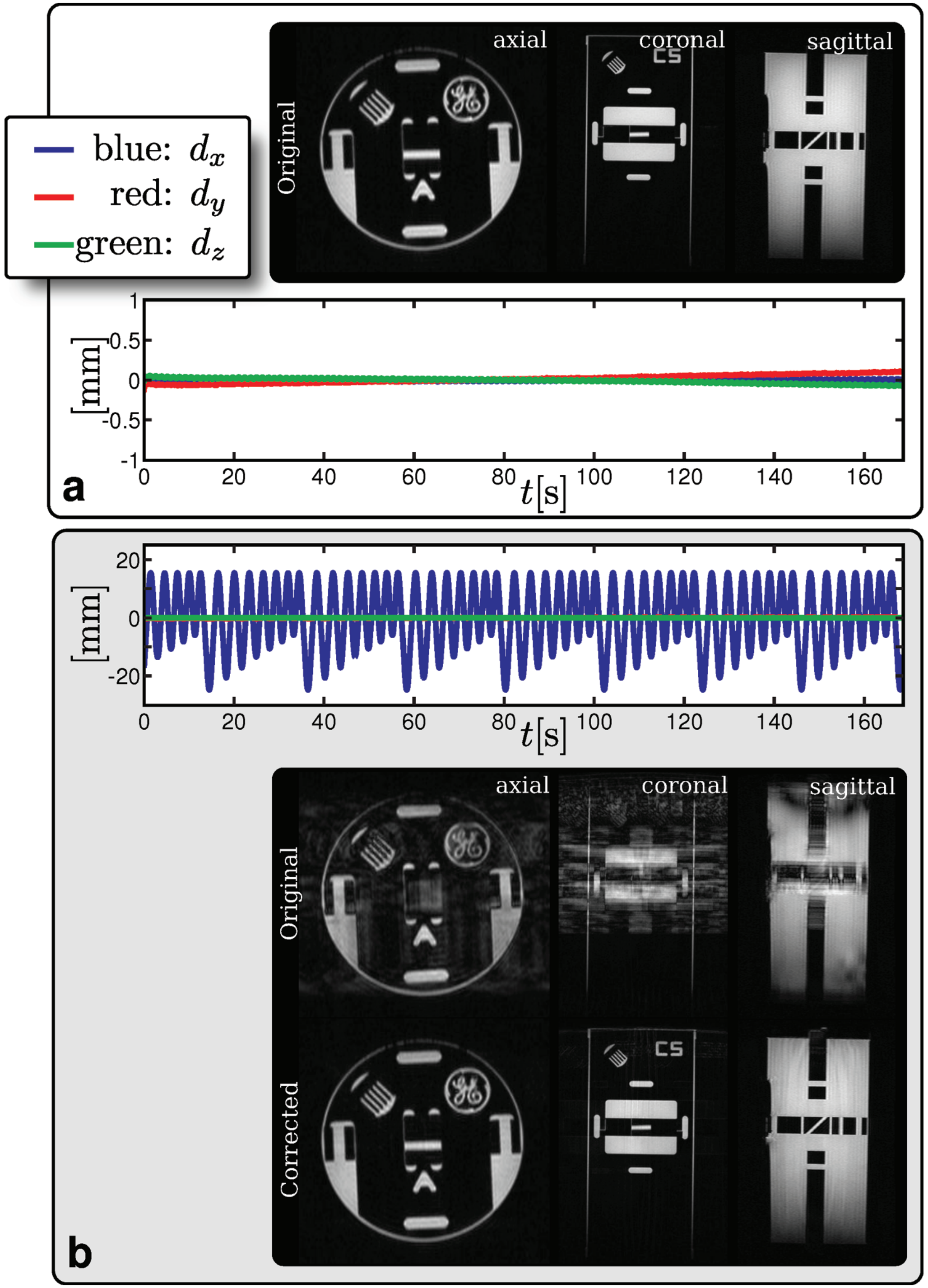
Comparison of different window widths b_c for the localized gradient entropy calculations

- When b_c=2cm, arteries appear sharper; however, ghosting artifact from fat wall is introduced. Also, noise amplification can be noticed outside the body.
- With b_c=14cm, arteries blur and a faint ghosting artifact appears above the liver
- b_c=10cm is used for correction



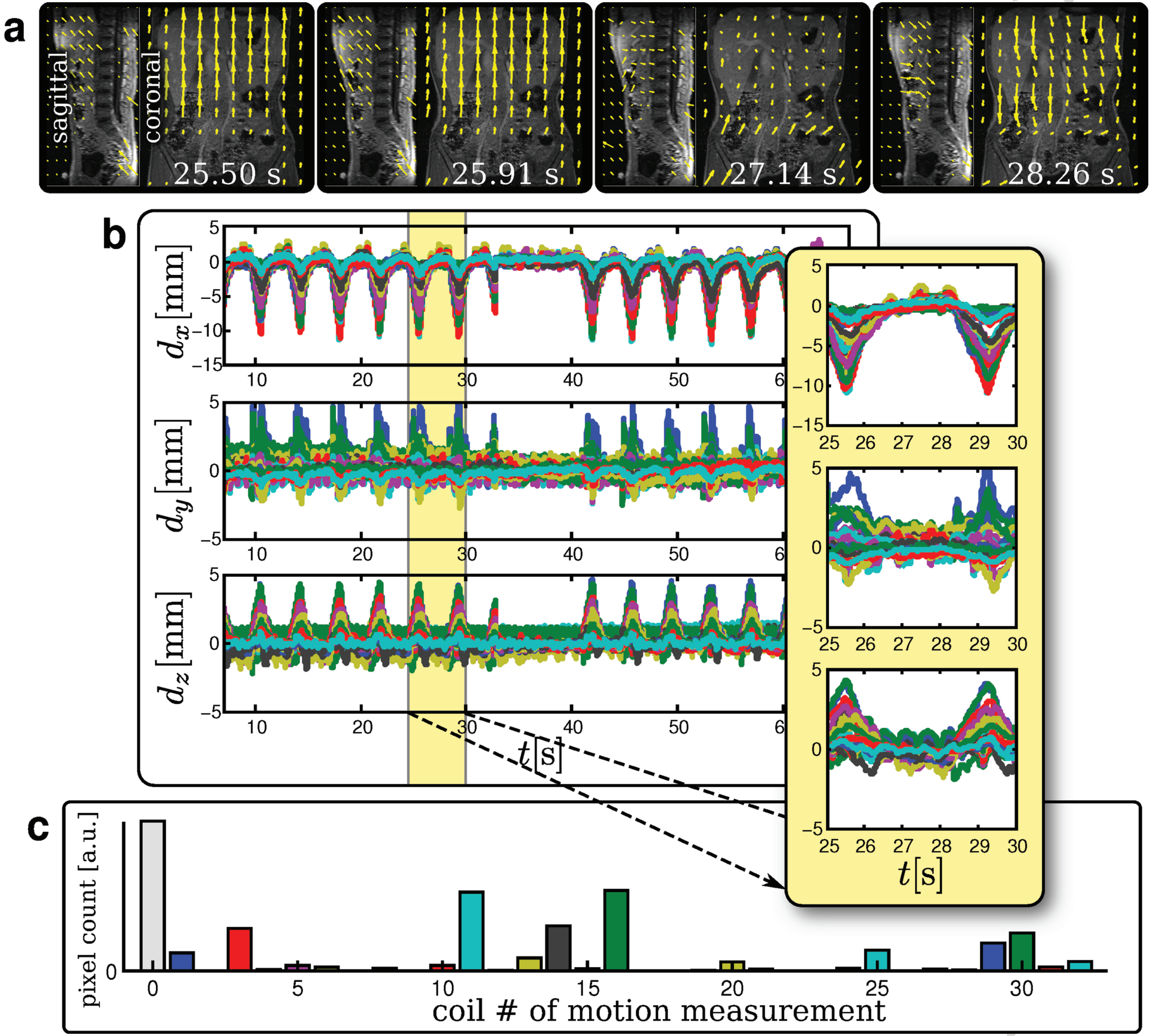
Phantom Study Results

- (a) Motionless scan
- (b) Rigid body translation validated the accuracy of the measurements and correction.



Study 1 Motion Measurements

- (a) Translation maps in sagittal and coronal slices accurately depicting respiratory motion
- (b) Motion measurements acquired from each coil
- (c) Histogram plot of number of pixels that was focused by each motion path; number of pixels gives an idea of the scan volume that was focused on by each measurement



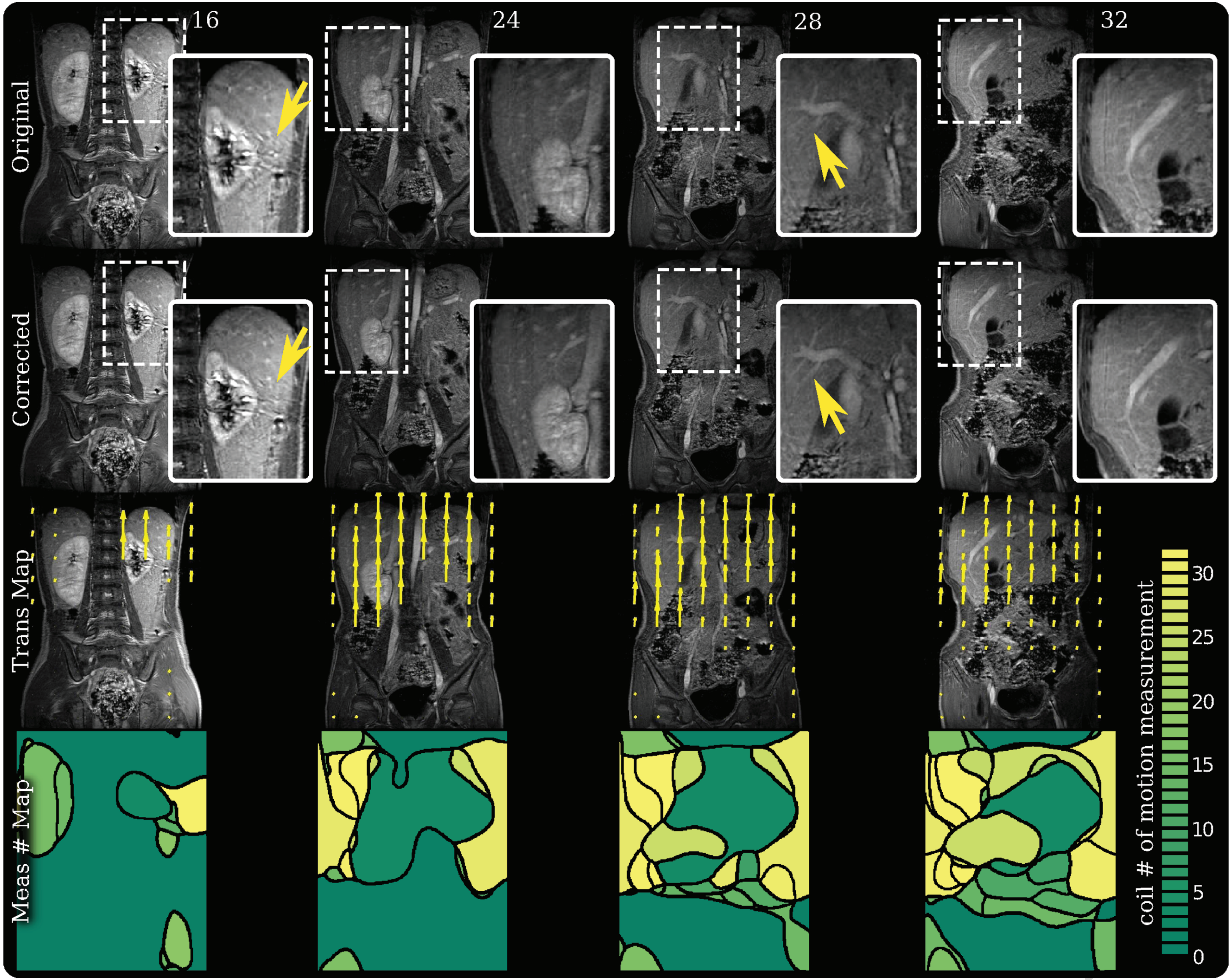
Study 1 Results

An abdominal study performed on a 6-year-old using a 3D spoiled gradient-recalled echo acquisition sequence.

1st row: Slice 16, 24, 28, 32 of original 3D volume

2nd row: Same corrected slices

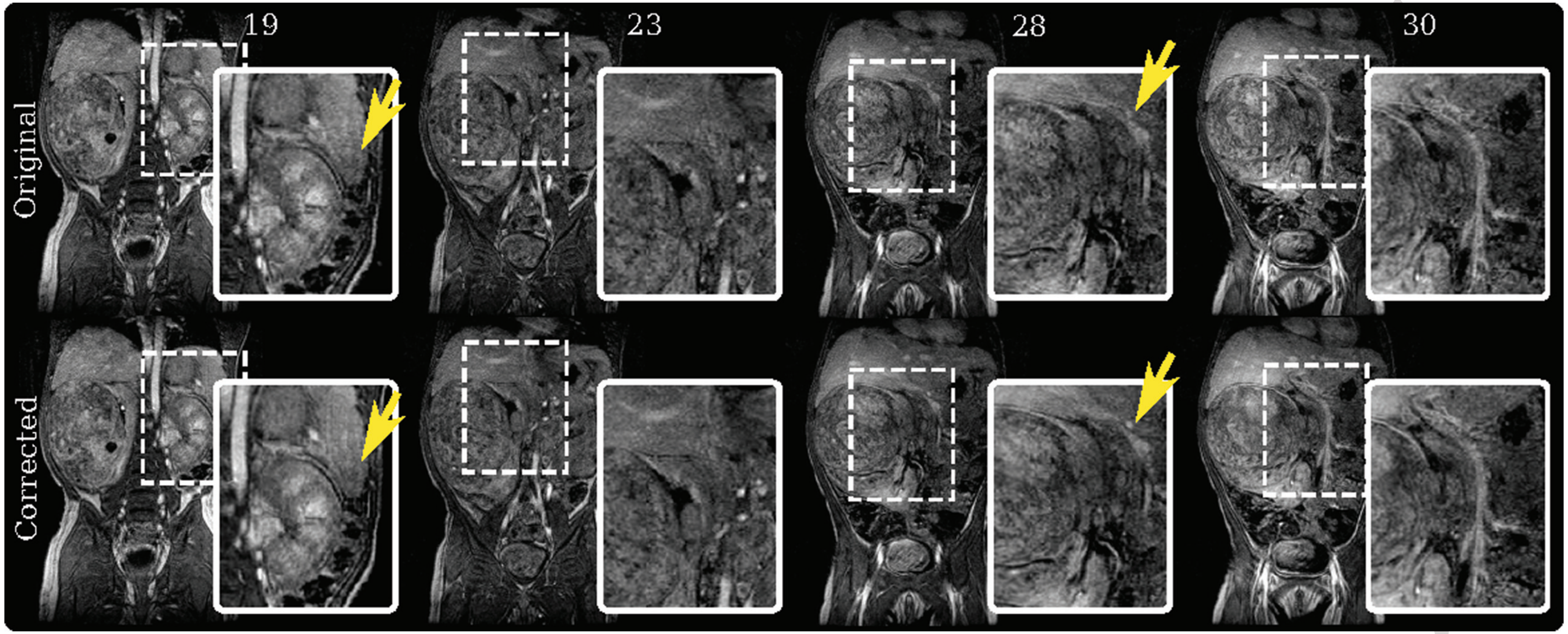
3rd row: Derived translation maps in coronal plane



Study 2 Results

Abdominal study of a 2-year-old with renal tumor scanned using a 3D spoiled gradient-recalled echo acquisition sequence.

- Ghosting artifacts in slice 19 were suppressed and tissue planes were sharpened.
- In slice 23, a lesion became better defined after correction.



David’s Analysis

David's Analysis: Time Consequences of Deleting k-Space Trajectories

- Keep it simple
- Identify modifiable components of code
- Understand and manipulate the **NAVA** and **k** matrices of example1.mat file

example1.mat (MAT-file)

Name	Value
nFRead	320
res	[0.9375,0.9375,3]
TR	5496
yorder	13312x1 double
zorder	13312x1 double
k	1x18 double
DATAA	4-D double
NAVA	18x425984 double
DATAAc	4-D double

```
% The following data structures are stored in the .mat file:
%   DATAA          -- raw k-space data with size nx*ny*nz*nc:
%                         nx -- readout length
%                         ny -- number of phase-encodes
%                         nz -- number of slice-encodes
%                         nc -- number of coils (receivers)
%   NAVA           -- raw navigator data with size nnav*nall:
%                         nnav -- navigator length
%                         nall -- total number of readouts x nc (= ny*nz*nc)
%   TR             -- [us] repitition time
%   k              -- [1/cm] k-space trajectory for navigator data
%   nFRead         -- reconstruction dimension for x
%                     (for partial k-space imaging in x)
%   res            -- [mm] image resolution
%   yorder, zorder -- y-phase encode and z-phase encode ordering
```

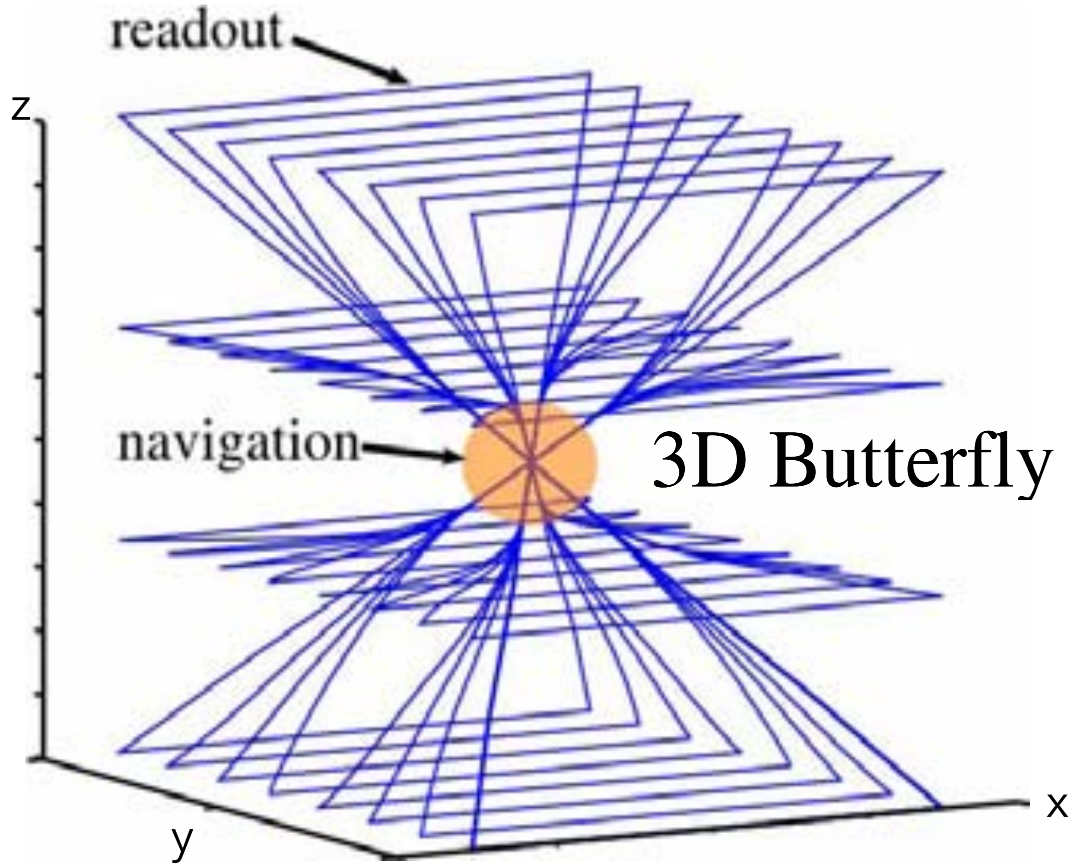
David’s Analysis: Relevant MATLAB Output

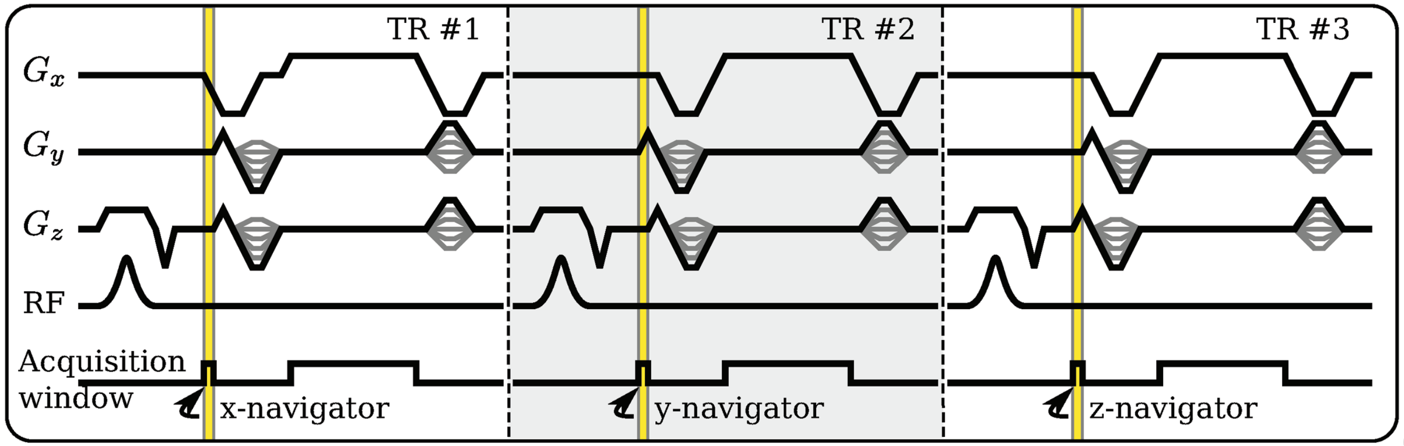
```
>> demo
runBflyMotionEstimate> Loading example data...
Elapsed time is 7.288391 seconds.
runBflyMotionEstimate> Estimating motion...
transEstAll> Compute motion estimate for coil 1: 1.73972 seconds
transEstAll> Compute motion estimate for coil 2: 0.752777 seconds
transEstAll> Compute motion estimate for coil 3: 0.728185 seconds
transEstAll> Compute motion estimate for coil 4: 0.739455 seconds
transEstAll> Compute motion estimate for coil 5: 0.779372 seconds
transEstAll> Compute motion estimate for coil 6: 0.746981 seconds
transEstAll> Compute motion estimate for coil 7: 0.754212 seconds
transEstAll> Compute motion estimate for coil 8: 0.719833 seconds
transEstAll> Compute motion estimate for coil 9: 0.711588 seconds
transEstAll> Compute motion estimate for coil 10: 0.710558 seconds
transEstAll> Compute motion estimate for coil 11: 0.695069 seconds
transEstAll> Compute motion estimate for coil 12: 0.693536 seconds
transEstAll> Compute motion estimate for coil 13: 0.696508 seconds
transEstAll> Compute motion estimate for coil 14: 0.695307 seconds
transEstAll> Compute motion estimate for coil 15: 0.690015 seconds
transEstAll> Compute motion estimate for coil 16: 0.694681 seconds
transEstAll> Compute motion estimate for coil 17: 0.721439 seconds
transEstAll> Compute motion estimate for coil 18: 0.731255 seconds
transEstAll> Compute motion estimate for coil 19: 0.743851 seconds
transEstAll> Compute motion estimate for coil 20: 0.755222 seconds
transEstAll> Compute motion estimate for coil 21: 0.712771 seconds
transEstAll> Compute motion estimate for coil 22: 0.730249 seconds
transEstAll> Compute motion estimate for coil 23: 0.742198 seconds
transEstAll> Compute motion estimate for coil 24: 0.742857 seconds
transEstAll> Compute motion estimate for coil 25: 0.706499 seconds
transEstAll> Compute motion estimate for coil 26: 0.736601 seconds
transEstAll> Compute motion estimate for coil 27: 0.729094 seconds
transEstAll> Compute motion estimate for coil 28: 0.740363 seconds
transEstAll> Compute motion estimate for coil 29: 0.684983 seconds
transEstAll> Compute motion estimate for coil 30: 0.689613 seconds
transEstAll> Compute motion estimate for coil 31: 0.685973 seconds
transEstAll> Compute motion estimate for coil 32: 0.708216 seconds
Elapsed time is 0.877619 seconds.
```

```
runMotionAutofocus> starting autofocusing...
motionAutofocus3> Recon original
motionAutofocus3> Done with recon of original: 2.056 seconds
motionAutofocus3> Correct images with motion data from each coil
motionAutofocus3> Finished measurement 1: 7.43279 seconds
motionAutofocus3> Finished measurement 2: 5.0161 seconds
motionAutofocus3> Finished measurement 3: 4.2431 seconds
motionAutofocus3> Finished measurement 4: 4.15631 seconds
motionAutofocus3> Finished measurement 5: 4.18965 seconds
motionAutofocus3> Finished measurement 6: 4.20366 seconds
motionAutofocus3> Finished measurement 7: 4.20545 seconds
motionAutofocus3> Finished measurement 8: 4.22856 seconds
motionAutofocus3> Finished measurement 9: 4.16578 seconds
motionAutofocus3> Finished measurement 10: 4.20097 seconds
motionAutofocus3> Finished measurement 11: 4.26774 seconds
motionAutofocus3> Finished measurement 12: 4.38153 seconds
motionAutofocus3> Finished measurement 13: 4.2967 seconds
motionAutofocus3> Finished measurement 14: 5.57844 seconds
motionAutofocus3> Finished measurement 15: 4.30097 seconds
motionAutofocus3> Finished measurement 16: 4.69139 seconds
motionAutofocus3> Finished measurement 17: 4.23232 seconds
motionAutofocus3> Finished measurement 18: 4.22754 seconds
motionAutofocus3> Finished measurement 19: 4.19917 seconds
motionAutofocus3> Finished measurement 20: 4.21751 seconds
motionAutofocus3> Finished measurement 21: 4.2592 seconds
motionAutofocus3> Finished measurement 22: 4.22029 seconds
motionAutofocus3> Finished measurement 23: 4.19744 seconds
motionAutofocus3> Finished measurement 24: 4.21334 seconds
motionAutofocus3> Finished measurement 25: 4.17516 seconds
motionAutofocus3> Finished measurement 26: 4.20566 seconds
motionAutofocus3> Finished measurement 27: 4.1975 seconds
motionAutofocus3> Finished measurement 28: 4.23271 seconds
motionAutofocus3> Finished measurement 29: 4.18588 seconds
motionAutofocus3> Finished measurement 30: 4.18519 seconds
motionAutofocus3> Finished measurement 31: 4.21242 seconds
motionAutofocus3> Finished measurement 32: 4.18767 seconds
runMotionAutofocus> starting homodyne recon...
Elapsed time is 5.525877 seconds.
```

David's Analysis:
Visualizing the Data

- **NAVA** (18x425984) is essentially all the blue lines
- **k** (1x18) is a descriptive matrix that contain the modified linear Butterfly trajectory (not shown)
- Goal is to modify NAVA and see the resulting effect on time.





David's Analysis: Question and Hypothesis

```
>> size(DATAA)

ans =

   192    256     52     32
```

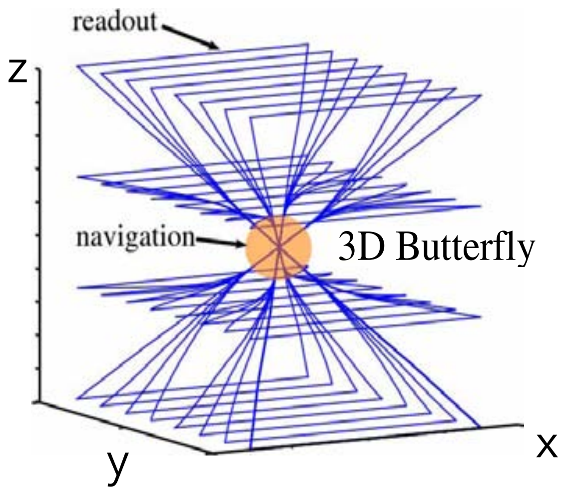
- n_x = 192 (readout length)
- n_y = 256 (number of phase-encodes)
- n_z = 52 (number of slice encodes)
- n_c = 32 (number of coils)

$n_y * n_z * n_c = 425984$, the column dimension of NAVA (18 x 425984)

- From this, I asked:

  What is the time consequence for each coil if we reduced the number of k trajectories by 1/6? 1/3? 1/2?

  I hypothesize that the overall time consequences will be faster because there is less data to process.



David's Analysis: Methods

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Manipulating the Data (YDZ Modification)

% Deleting all the even rows in both k and NAVA
 k(:,(1:6:18))=[];
 NAVA((1:6:18),:)=[];
```

Deletes 1/6 of rows for **k** and **NAVA**

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Manipulating the Data (YDZ Modification)

% Deleting all the even rows in both k and NAVA
 k(:,(1:3:18))=[];
 NAVA((1:3:18),:)=[];
```

Deletes 1/3 of rows for **k** and **NAVA**

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Manipulating the Data (YDZ Modification)

% Deleting all the even rows in both k and NAVA
 k(:,(2:2:18))=[];
 NAVA((2:2:18),:)=[];
```

Deletes 1/2 of rows for **k** and **NAVA**

David's Analysis: Acquisition Time Consequences

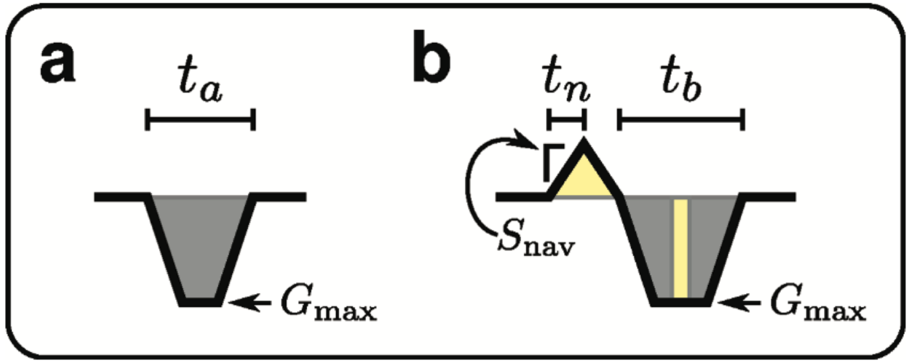


FIG. A1. Butterfly time penalty analysis. **a:** Original phase-encode gradient. **b:** Butterfly modified phase-encode gradient. For equivalent acquisitions, the total gradient areas in (a) and (b) must be equal. The gradients are designed such that the light-yellow shaded regions have the same areas and the dark-gray shaded regions have the same areas. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

$$t_n^2 S_{\text{nav}} = (t_b - t_a) G_{\text{max}} \qquad \text{[A1a]}$$

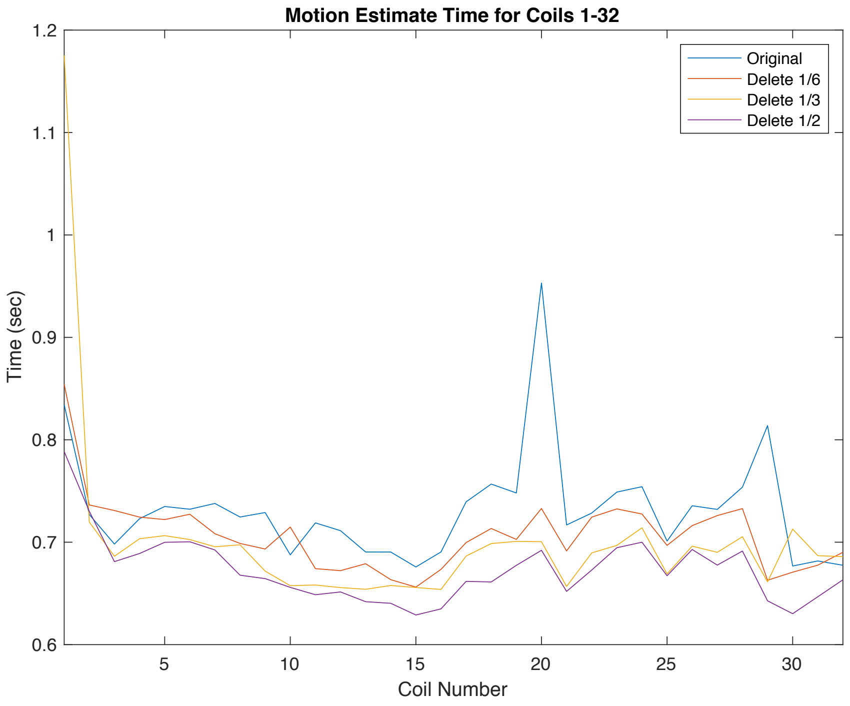
$$t_b = t_n^2 \frac{S_{\text{nav}}}{G_{\text{max}}} + t_a \qquad \text{[A1b]}$$

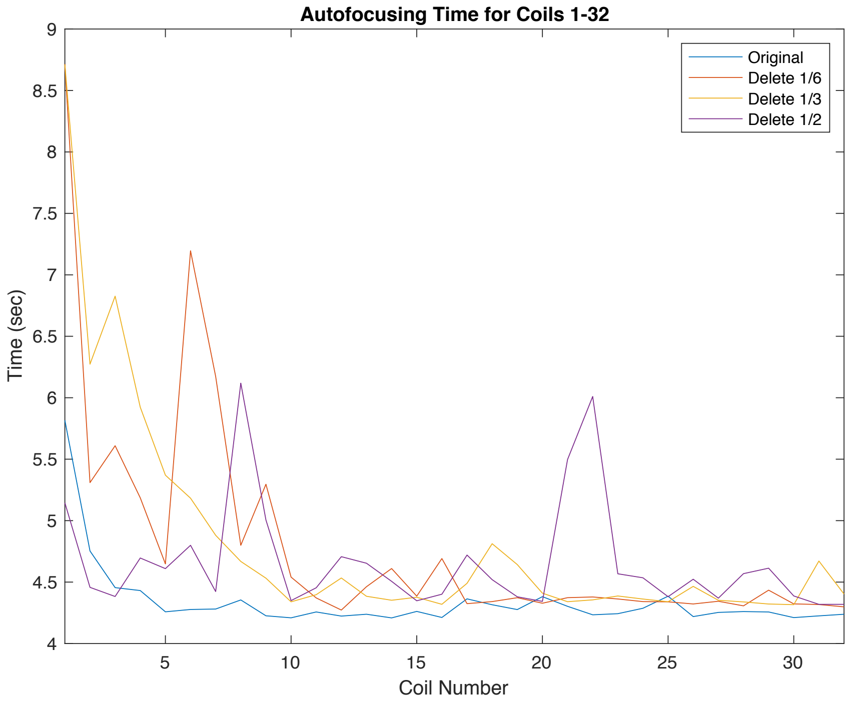
$$t_{\text{total}} = t_n + 2t_n + t_b \qquad \text{[A2a]}$$

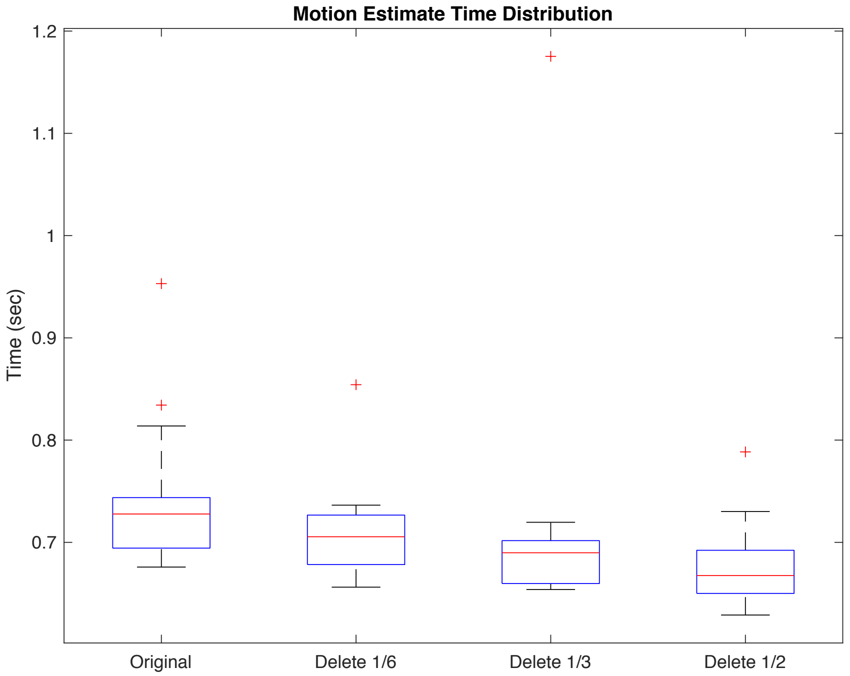
$$= 3t_n + t_n^2 \frac{S_{\text{nav}}}{G_{\text{max}}} + t_a. \qquad \text{[A2b]}$$

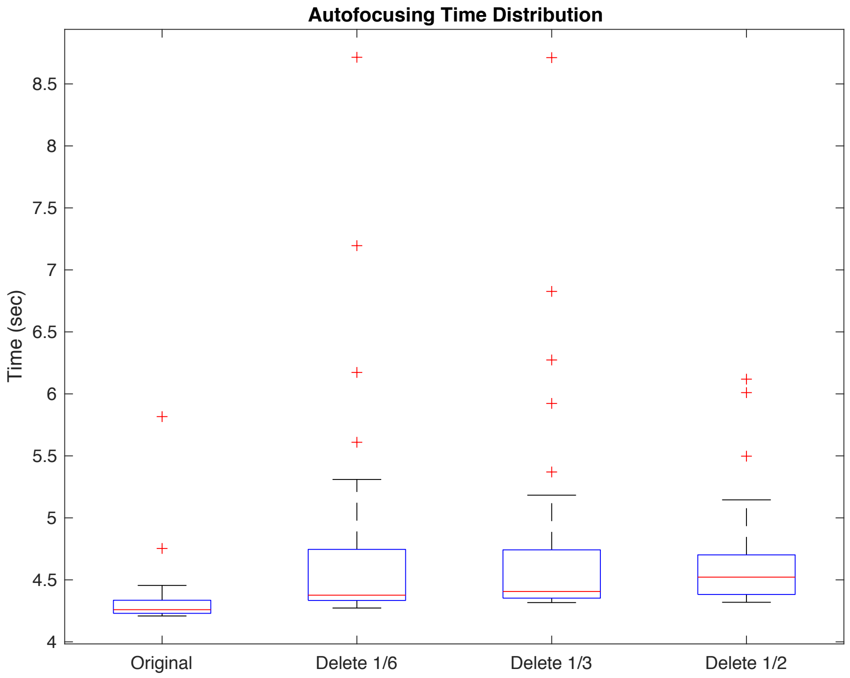
$$t_p = 3t_n + t_n^2 \frac{S_{\text{nav}}}{G_{\text{max}}}. \qquad \text{[A3]}$$

David's Analysis: Results & Discussion

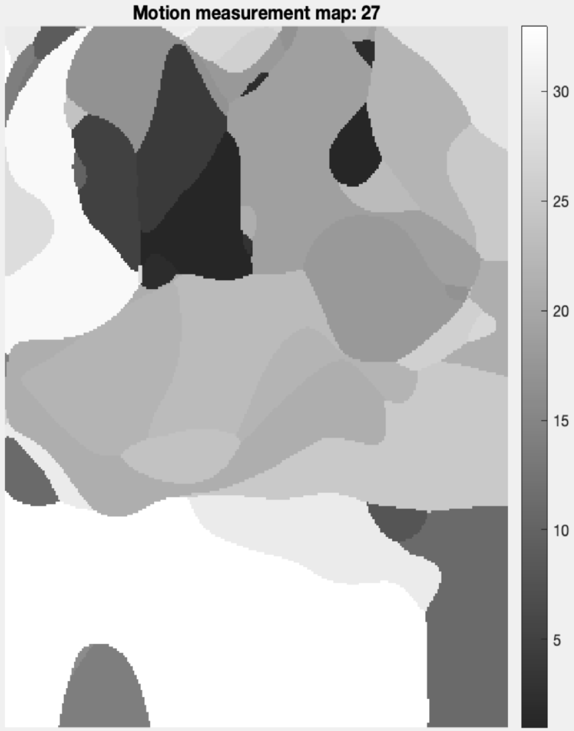




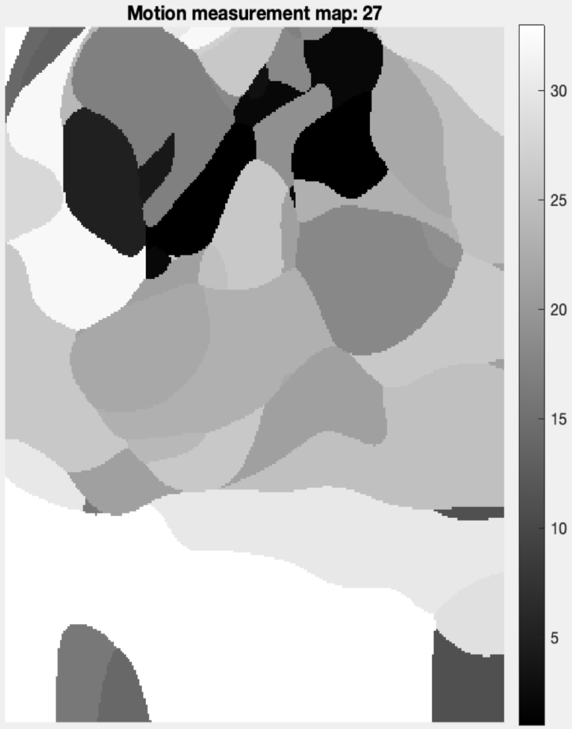




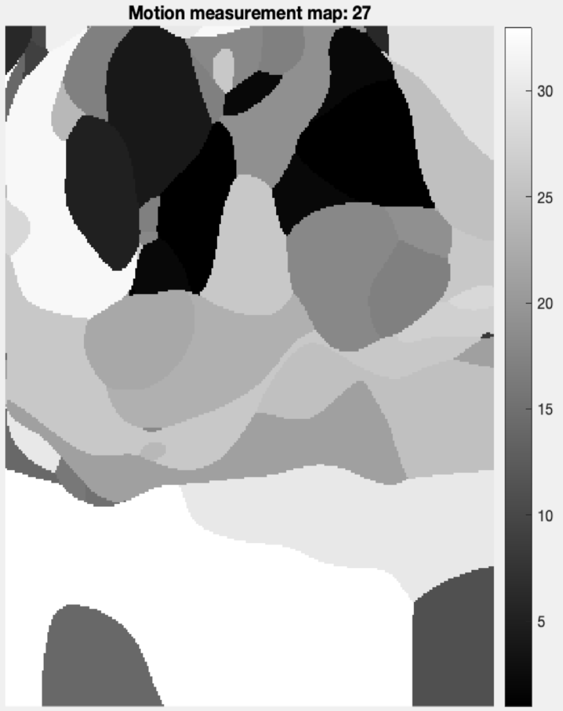
David's Analysis: Results & Discussion



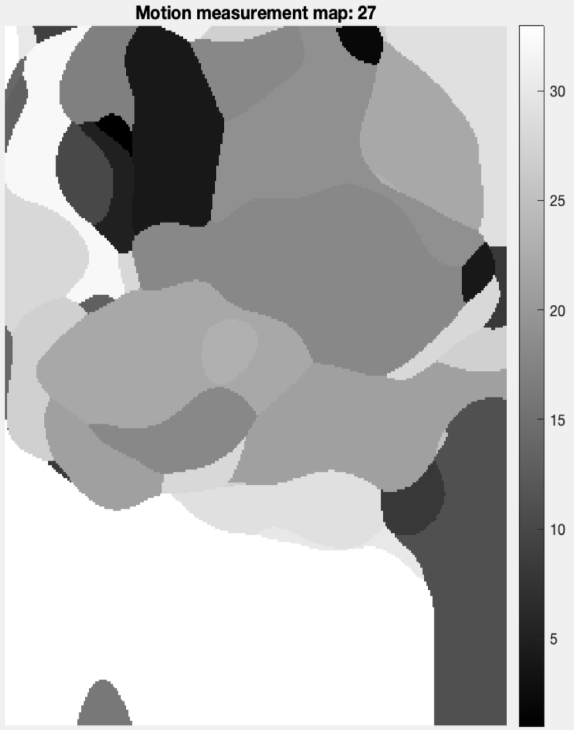
Original



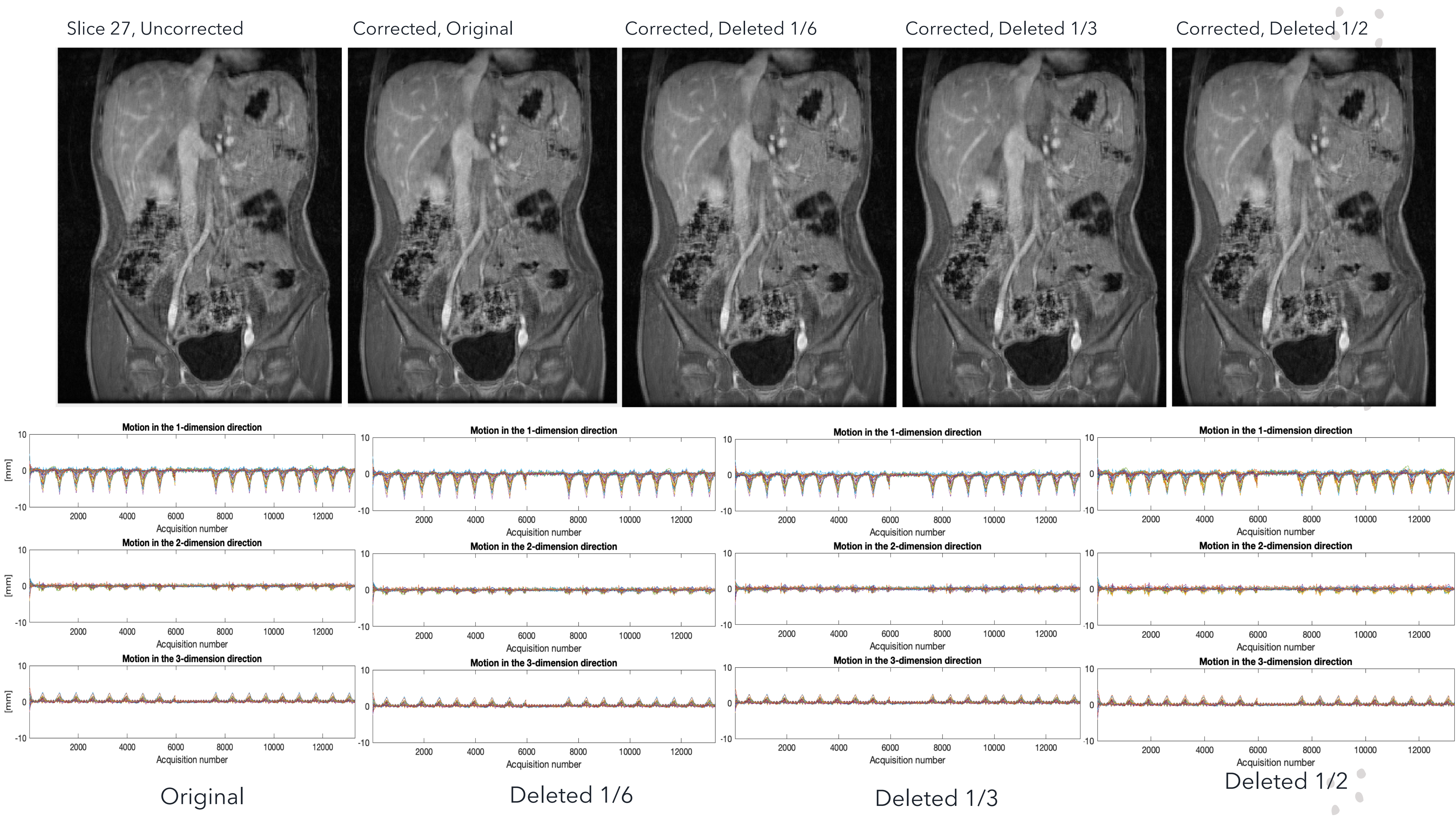
Deleted 1/6



Deleted 1/3



Deleted 1/2


Slice 27, Uncorrected
Corrected, Original
Corrected, Deleted 1/6
Corrected, Deleted 1/3
Corrected, Deleted 1/2
Motion in the 1-dimension direction
Motion in the 2-dimension direction
Motion in the 3-dimension direction
[mm]
Acquisition number
10
0
-10
2000
4000
6000
8000
10000
12000
Original
Deleted 1/6
Deleted 1/3
Deleted 1/2

Ciara’s Analysis

Linear Field Drift over time

- Field drift affects the acquired signal

$$\mathrm{s_n}\left(\mathrm{t}\right) = \mathrm{e}^{2\pi \mathrm{i n} \Delta\nu_{\mathrm{scan}} \mathrm{t}} \int_{-\infty}^{\infty} \mathrm{M_0}\left(\mathrm{x}\right) \mathrm{e}^{2\pi \mathrm{i k_n x}} \mathrm{dx}$$

Tal, A., & Gonen, O. (2013).

- Typical drift around 0.1ppm/hour

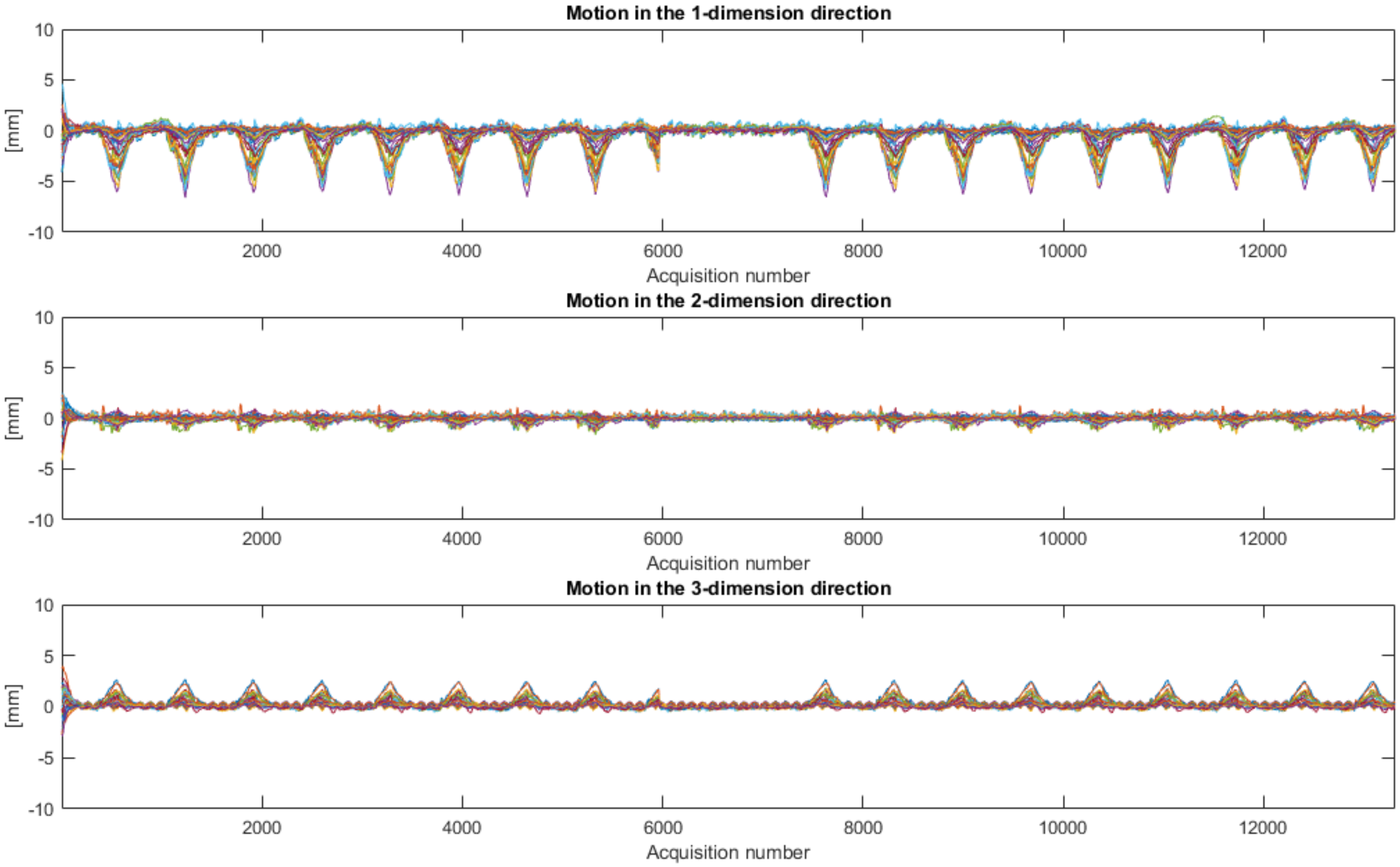
- For the nth scan:

$$W_n = e^{2\pi i n v 5\mathrm{e}-7}$$

- For each element in the raw k-space data

$$K'_n = W_n K_n$$

Original motion estimate



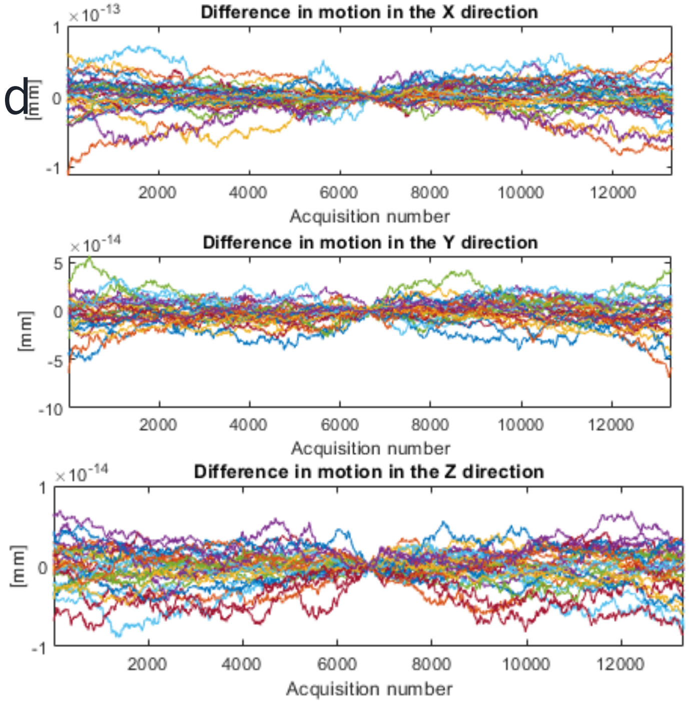
Motion estimate with field drift over time

Applied weighting values

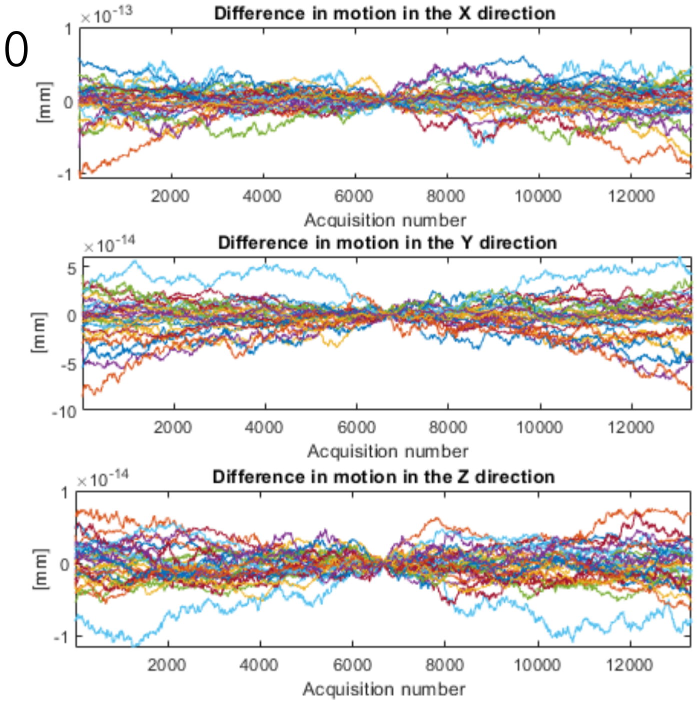
to navigator and scan data

- Same order of magnitude across relative field drift 17 orders of magnitude

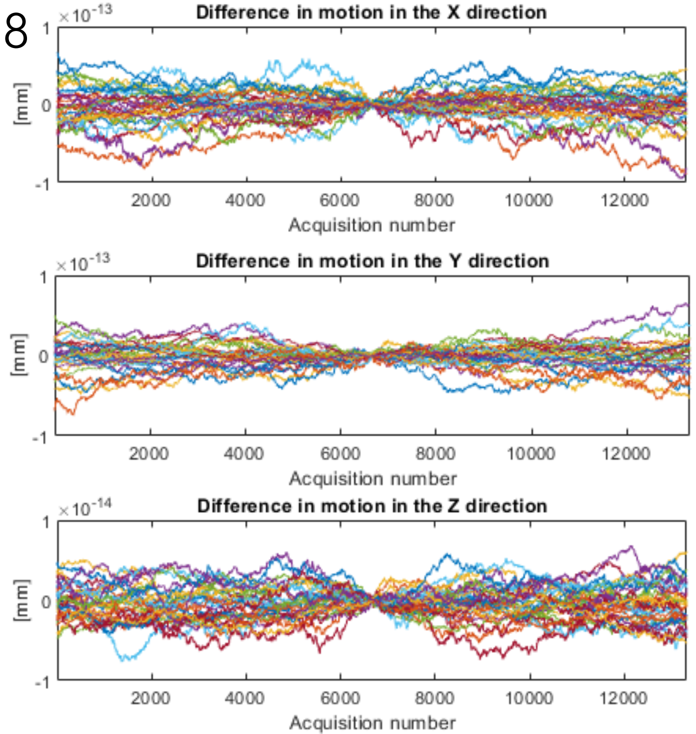
V=1



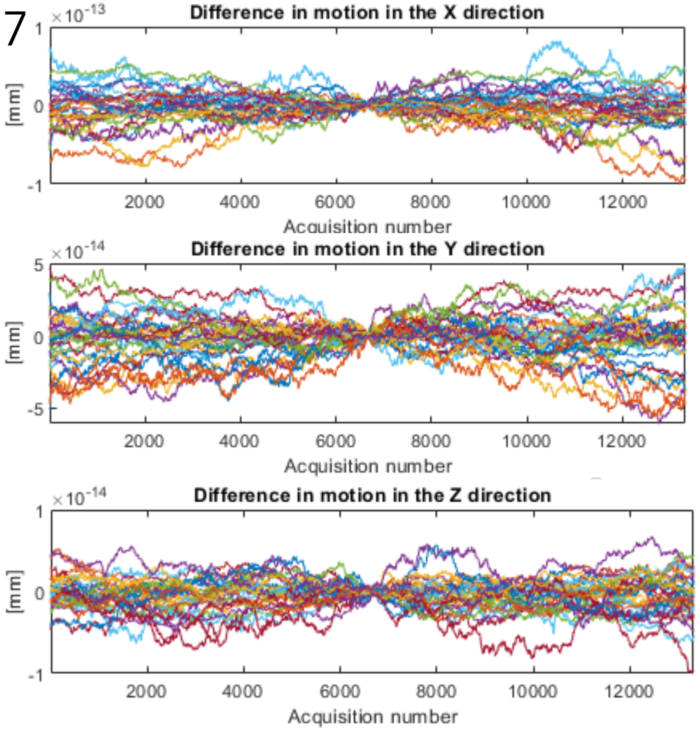
V=1000



V=1e8



V=1e17



Average difference in motion estimate

V	x	y	z
1	9.9416e-17	-8.8128e-16	-1.5524e-16
1000	4.5283e-16	-1.6724e-15	-2.3828e-16
1e8	-1.3459e-15	-1.0845e-15	-2.9971e-16
1e17	-1.1362e-15	1.5156e-15	-4.3958e-16

Average difference in motion estimate magnitude

V	x	y	z
1	1.3366e-14	8.5005e-15	1.6051e-15
1000	1.2559e-14	1.0827e-14	1.6865e-15
1e8	1.4309e-14	1.0069e-14	1.6275e-15
1e17	1.3936e-14	1.0210e-14	1.4504e-15

Field Drift over time

- Original modification weight
- For the nth scan:

$$W_n = e^{2\pi i n v 5\text{e}-7}$$

$$W_n = vne^{2\pi i n v 5\text{e}-7}$$

$$W_n = vn$$

$$K'_n = W_n K_n$$

Average difference in motion estimate magnitude

$$W_n = e^{2\pi i n v 5\text{e}-7}$$

V	x	y	z
1	1.3366e-14	8.5005e-15	1.6051e-15
1000	1.2559e-14	1.0827e-14	1.6865e-15
1e8	1.4309e-14	1.0069e-14	1.6275e-15
1e17	1.3936e-14	1.0210e-14	1.4504e-15

$$W_n = vne^{2\pi i n v 5\text{e}-7}$$

V	x	y	z
1	1.5517e-14	1.0599e-14	1.6411e-15
1000	1.4720e-14	9.3875e-15	1.5109e-15

Average difference in motion estimate magnitude

$$W_n = e^{2\pi inv5e-7}$$

V	x	y	z
1	1.3366e-14	8.5005e-15	1.6051e-15
1000	1.2559e-14	1.0827e-14	1.6865e-15
1e8	1.4309e-14	1.0069e-14	1.6275e-15
1e17	1.3936e-14	1.0210e-14	1.4504e-15

$$W_n = vn$$

V	x	y	z
1000	1.3269e-14	1.1723e-14	1.4825e-15
1e8	1.1569e-14	9.5154e-15	1.5577e-15

Summary Linear Field Drift

- Algorithm robust to linear field drift across acquisitions
- Algorithm robust to drift over time
- Differences in navigator data between acquisitions can account for uniform field inhomogenities
- Would likely be robust to field inhomogeneities caused by e.g. metallic objects

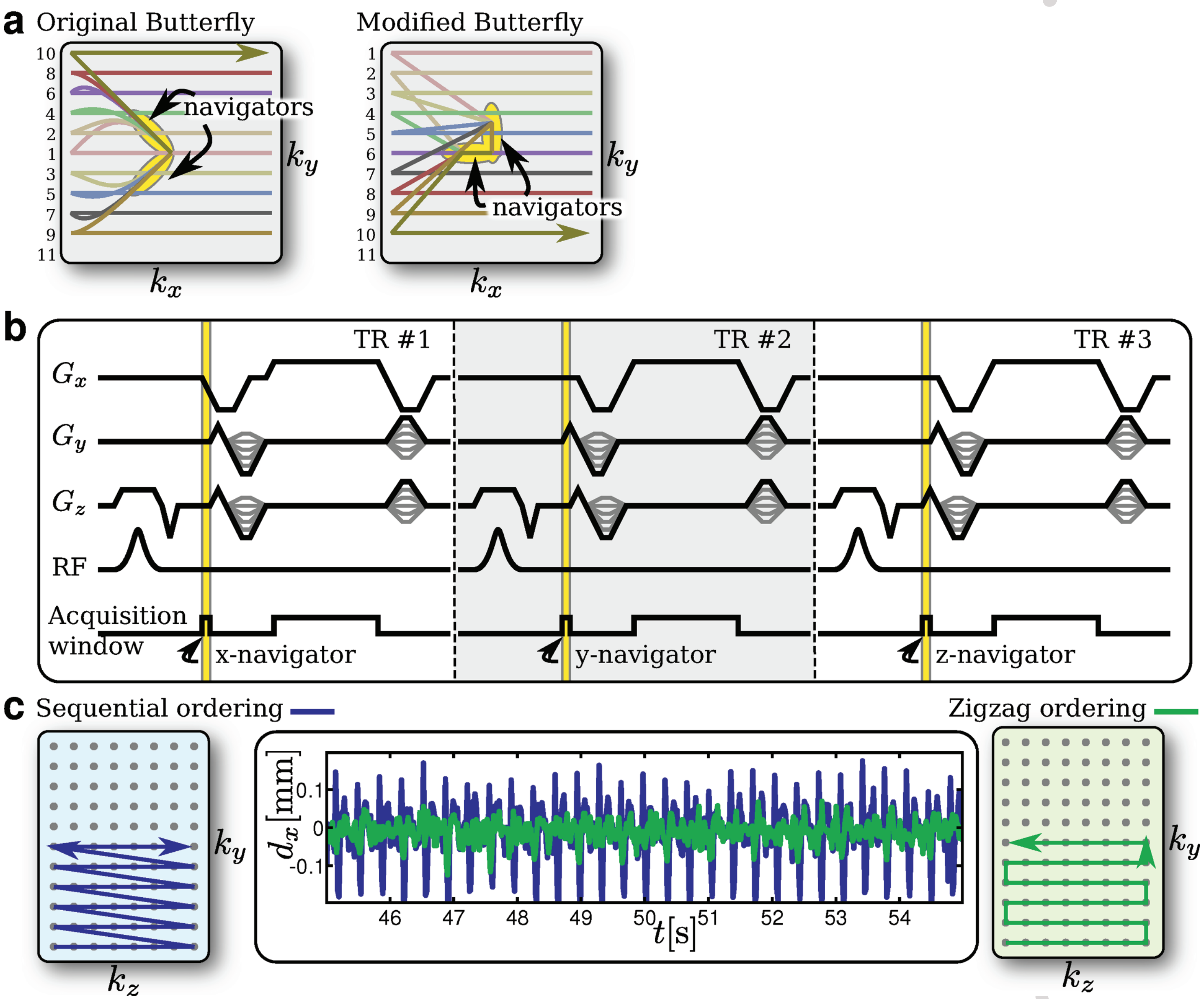
References

Tal, A., & Gonen, O. (2013). Localization errors in MR spectroscopic imaging due to the drift of the main magnetic field and their correction. *Magnetic resonance in medicine*, *70*(4), 895–904. https://doi.org/10.1002/mrm.24536

Modified Trajectory

- 3D trajectory along each gradient axis
- Traversed to minimize distance in k space

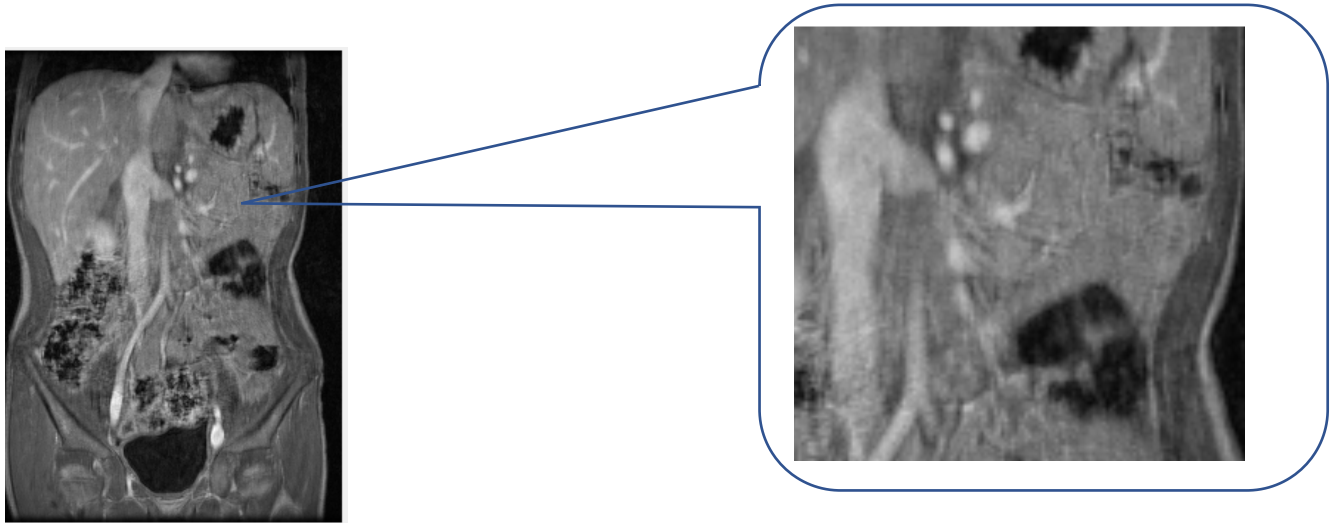
- Benefits
- Robust to system errors e.g. timing delays
- Reduce motion estimate complexity
- Flexible phase / slice order

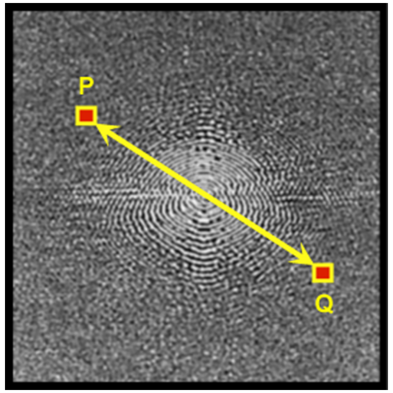


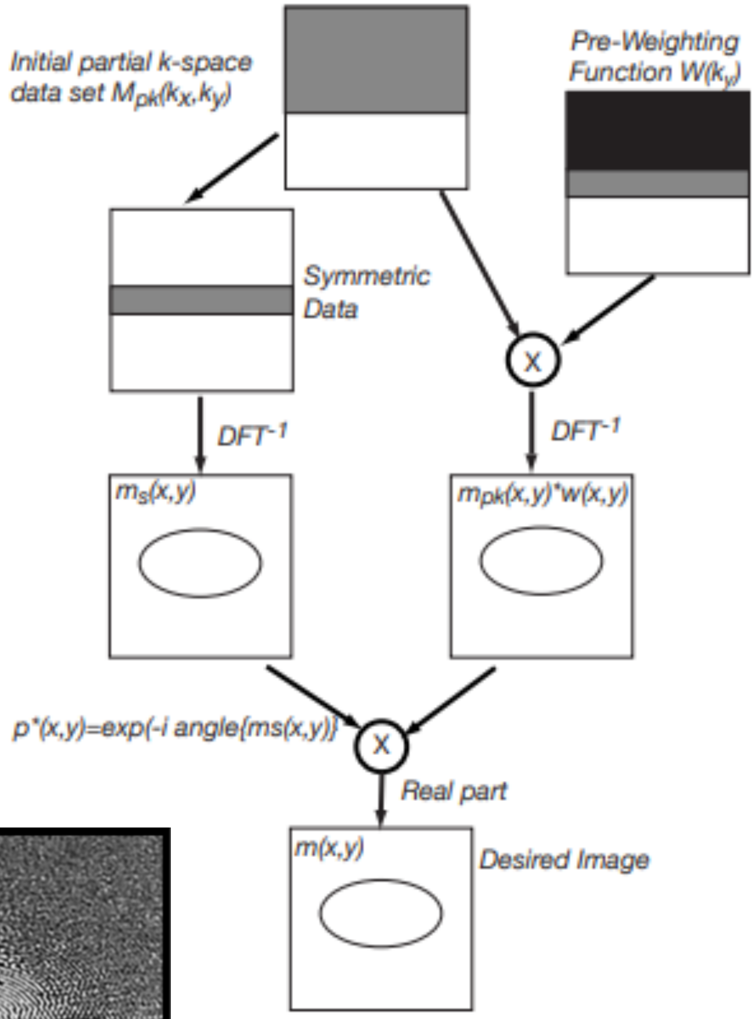
Sujoy’s Analysis

Homodyne reconstruction

- Homodyne Reconstruction
  - Fast single iteration technique.
  - Corrects most of the artifacts.
  - Computational Time=65.7946 sec

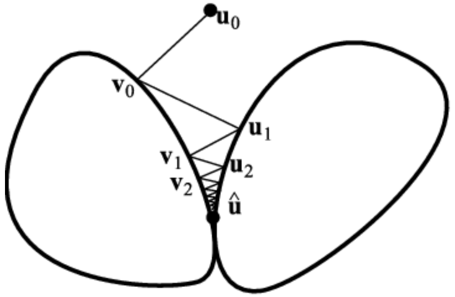


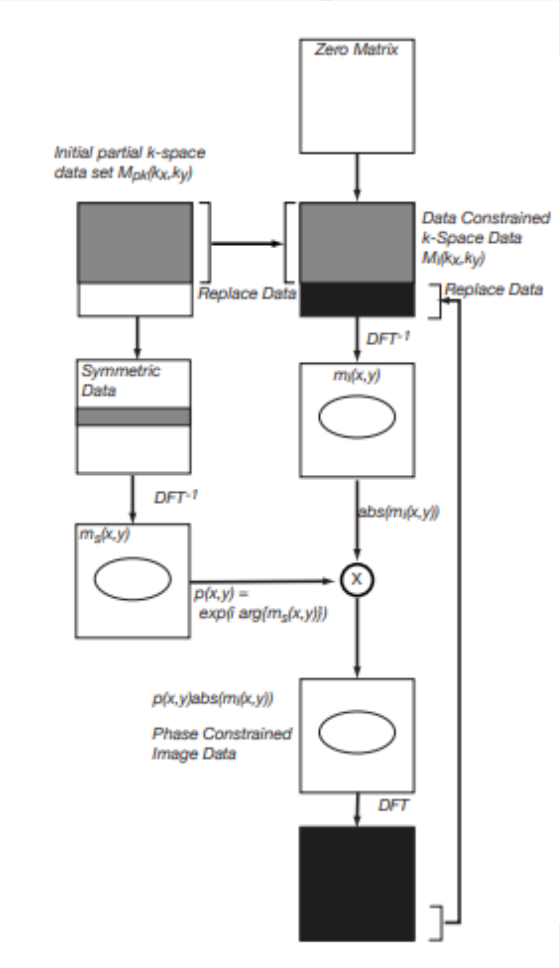




Projection onto Convex Sets (POCS)

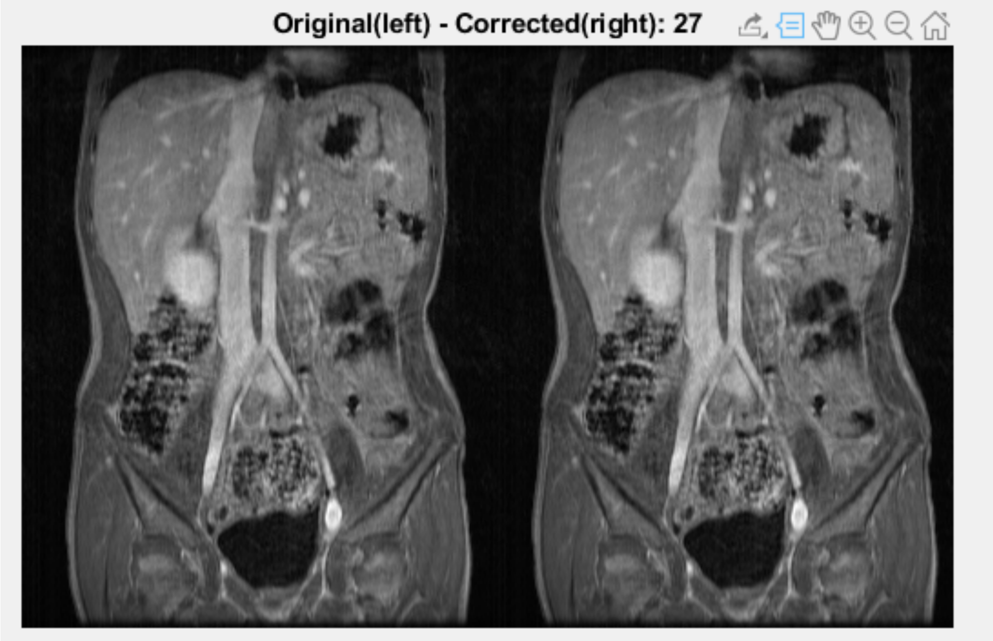
- Iterative technique
- Higher accuracy requires long time.
- Time= 5615.004906 seconds
- Background component of the complex noise suppressed



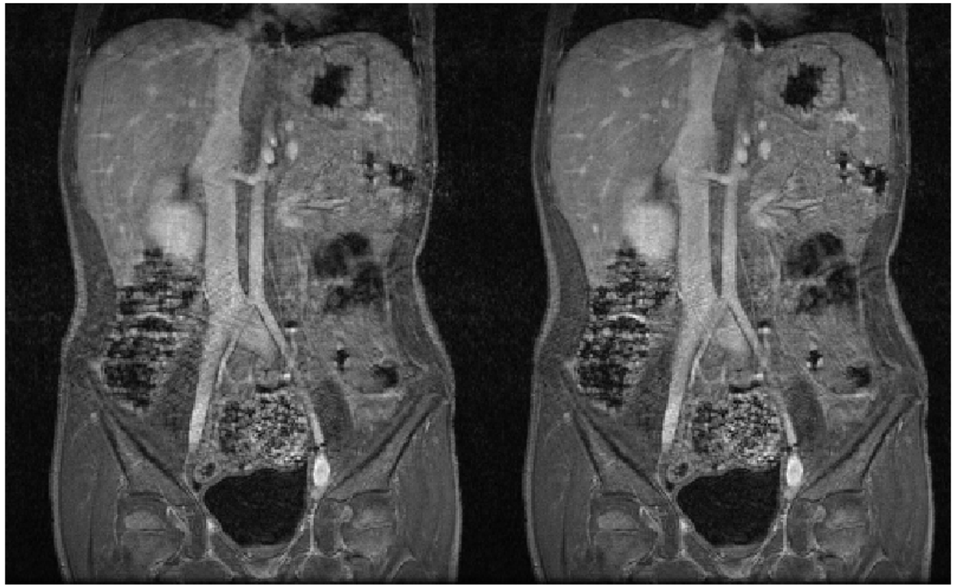


Results

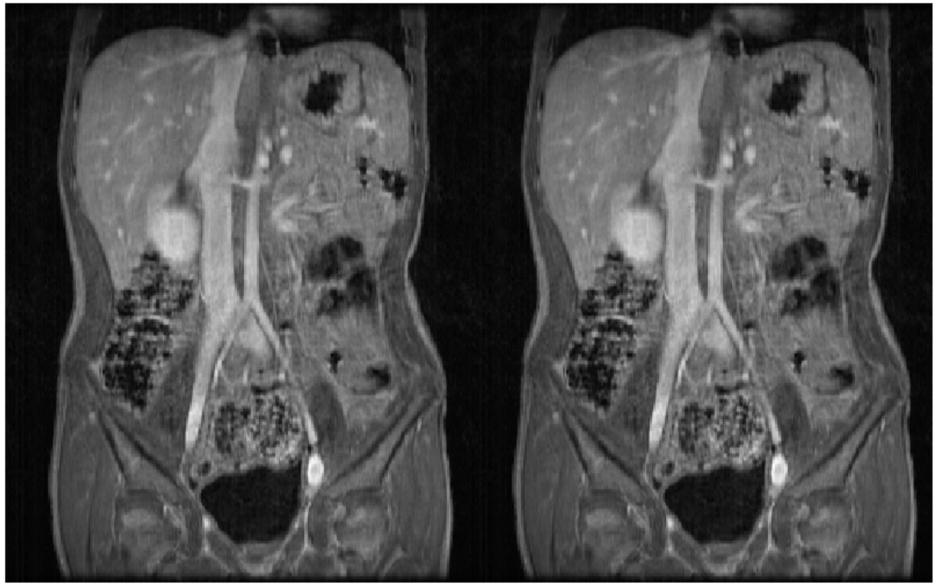
Autocorrected image



Homodyne

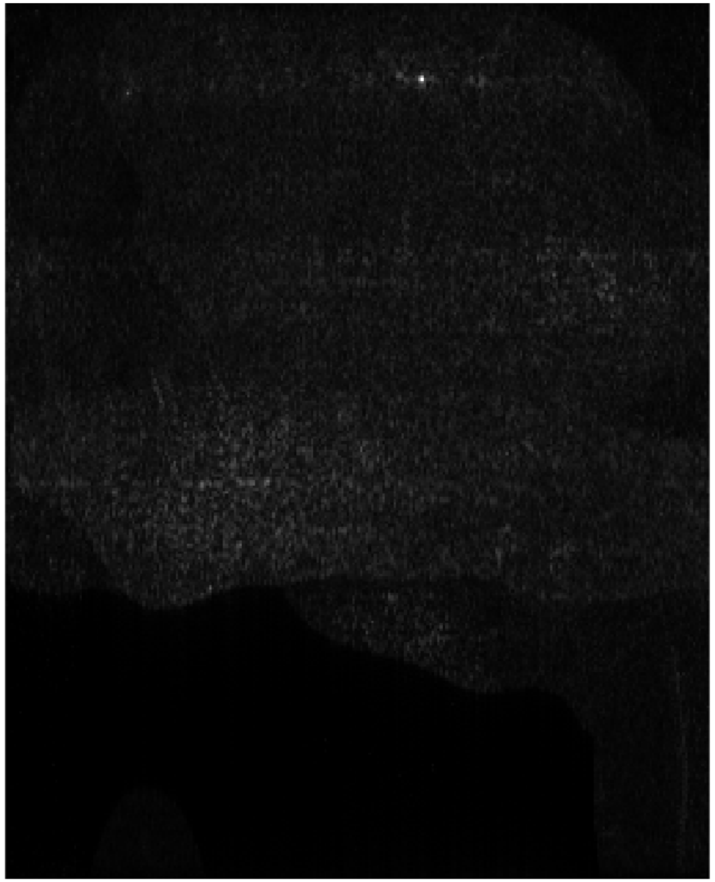


POCS



Remanent phase

Homodyne



Summary

- Homodyne reconstruction provides faster output necessary for motion artifact correction.
- POCS takes longer time but the accuracy can be further improved by Classification or regression techniques.
- Deep learning techniques to recognize anomalies can also be used.