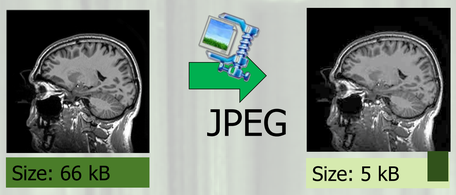
CATEGORY: MEDICAL IMAGING - MI02

POSTER P6177 | CONTACT NAME Jonas Vylder: jonas.devylder@telin.ugent.be

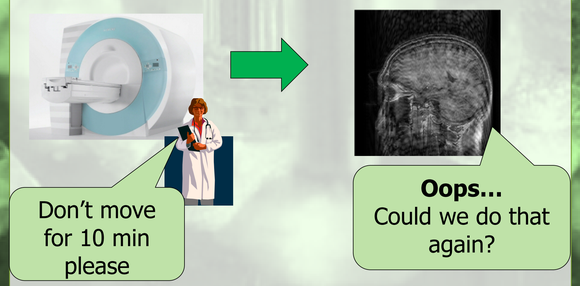
GPU TECHNOLOGY CONFERENCE

MRI

We know raw images contain much redundant information. It's the reason JPEG was invented:

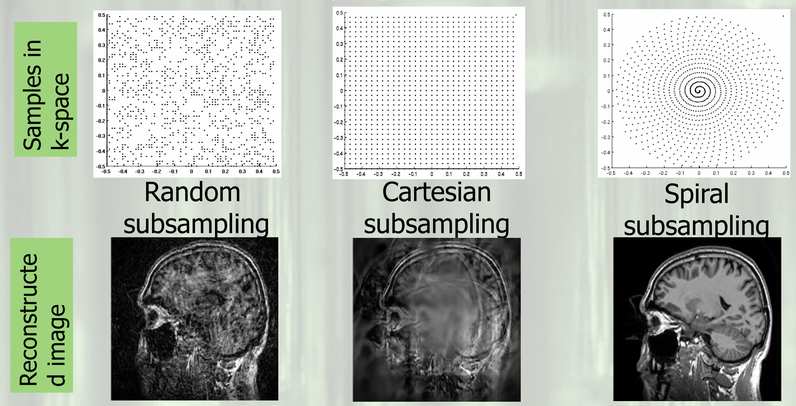


If these images contain so much redundant information, why do we spend so much TIME acquiring it all?



Data reduction

Scan less data points (in k-space) and fill in the blanks in an intelligent way.
Well, you can do this in a large number of ways. Here are some for 25% subsampling:



Check

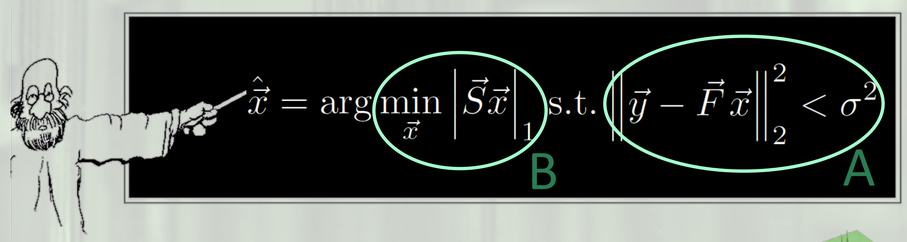
Demos and further explanation can be found at:

http://gepura.io

or come and see us at **boot #826**

MRI reconstruction

We prefer the image x that: A. Fits the 25% acquired data y
B. Introduces the minimal amount of stripes and structures Sx



- The shearlet transform S **optimally** decomposes an image x into flat regions and stripes and structures.
- Noise = stripes and structures (**built-in noise suppression!!**).

- Solved using convex optimization: iterative methods with fast convergence.

Due to the non-uniform sampling, each optimization iteration has to use the non-uniform fast Fourier transform (NUFFT), which is not as fast as the name makes believe. GPU acceleration greatly speeds-up MRI reconstruction research.

GPU acceleration with the Gepura suite

Quasar is a new **programming language** (incl. compiler, runtime optimization, …) for fast development on heterogeneous hardware.

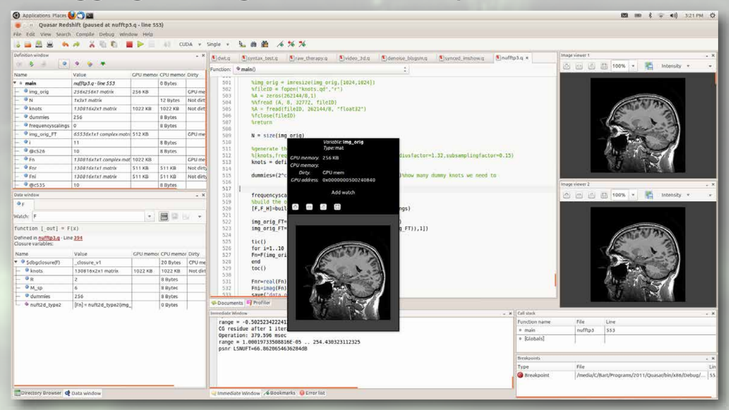
```
function [knots,frequencyscalings] = definespiralgrid(N,radiusfactor=1.32,subsamplingfactor=0.25)

    radius=radiusfactor/N[1];
    numsamples=prod(N)*subsamplingfactor;

    krak=(pi/radius)/(numsamples-1);
    theta=0..krak..pi/radius;

    x=radius*theta/2/pi.*cos(theta+pi*(1+radius)/radius);
    y=radius*theta/2/pi.*sin(theta+pi*(1+radius)/radius);

    knots=cat(reshape(y,[numel(y),1]),reshape(x,[numel(y),1]))

    frequencyscalings = transpose(1/N[0].*krak*radius*theta/2/pi);
    frequencyscalings = (frequencyscalings)/max(frequencyscalings)*nunel(frequencyscalings)
end
```

Compact code

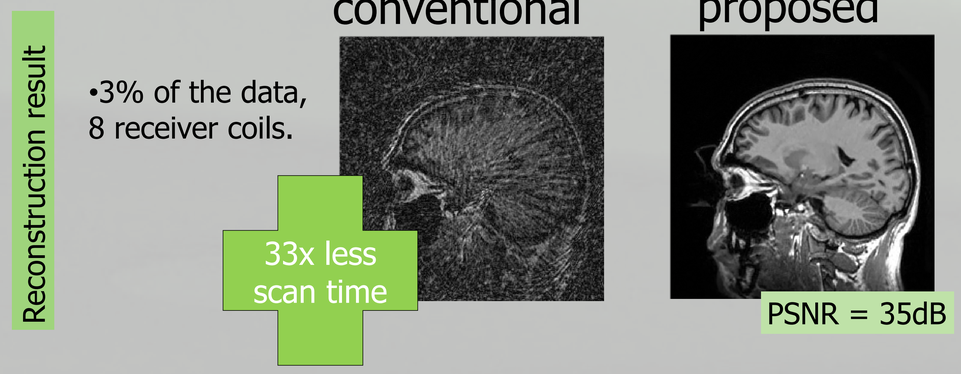
Matrix notation & operations

Automatic parallelization

The Redshift **IDE** allows for efficient debugging and algorithmic development:



Results



Future work

Quasar allows **low-cost** development of fast GPU implementation:

- a fast GPU implementation is **desirable** for 'compressed sensing MRI' applications

A fast GPU implementation **enables** further algorithm development:

- more complex/advanced regularization parameter optimization
- auto-calibration MRI
- parallel MRI
- 3D MRI
- demanding non-uniform K-space trajectories