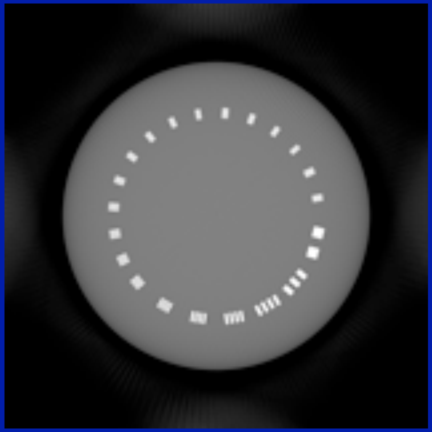
High Resolution Iterative CT Reconstruction using Graphics Hardware



NSS-MIC HPMI 2009

October 27th, 2009

Benjamin Keck[1,2], Hannes G. Hofmann[1], Holger Scherl[2], Markus Kowarschik[2] and Joachim Hornegger[1]

[1] Pattern Recognition Lab (Computer Science 5)
Friedrich-Alexander-University Erlangen-Nuremberg, Germany

SIEMENS

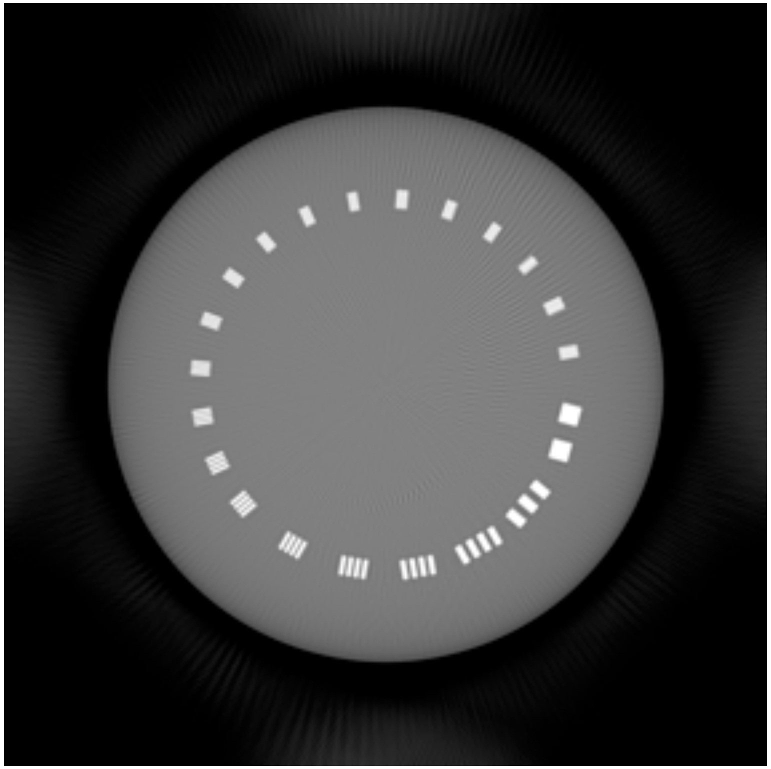
[2] Siemens Healthcare, CV,
Medical Electronics & Imaging Solutions, Erlangen, Germany

Motivation

- **Iterative reconstruction methods can lead to better reconstruction results than analytical methods:**
  - particularly for the case of noisy/incomplete data
- **Computational complexity of iterative methods is much higher for iterative methods than for standard FBP-type algorithms**
- **Fast GPU hardware allows for efficient development and evaluation of iterative reconstruction approaches**
- **Focus of this work: approaches to circumvent existing hardware limitations**

Outline

- **GPU-accelerated SART using CUDA[1]**
  - back-projection
  - forward-projection
- **Limitations**
- **Possible solutions**
- **Performance comparison**
- **Proof of concept**

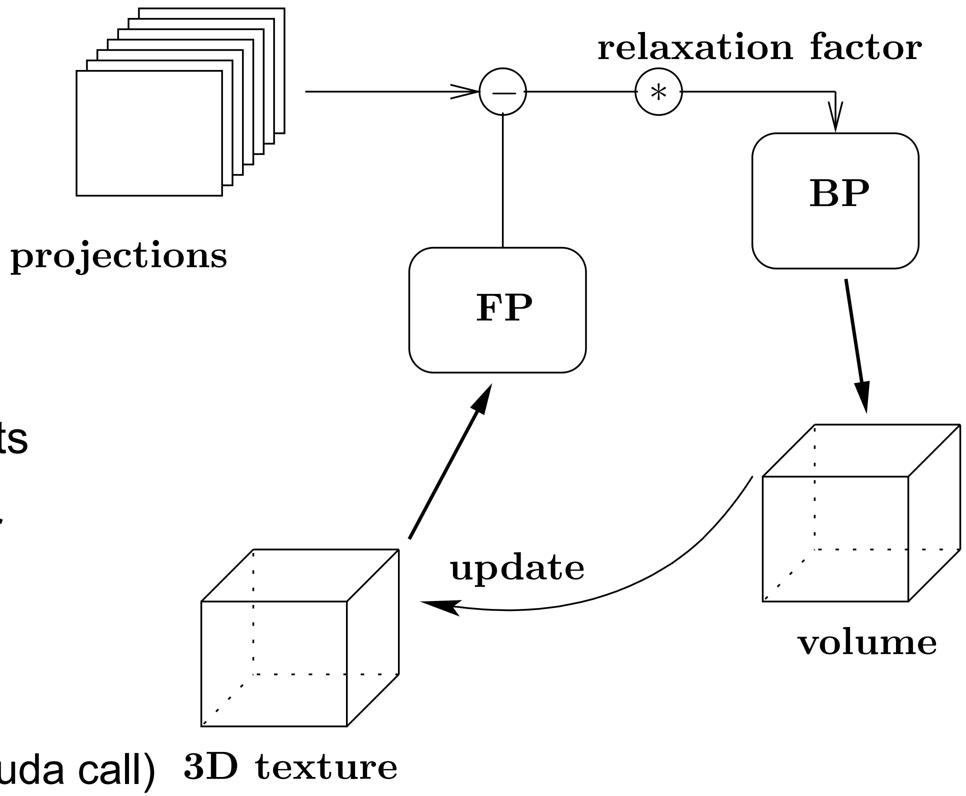


Iterative reconstruction of the Catphan CTP528 phantom

[1]Keck, B., Hofmann, H.G., Scherl, H., Kowarschik, M., and Hornegger, J., "GPU-accelerated SART reconstruction using the CUDA programming environment," in [Proceedings of SPIE Conference, Lake Buena Vista 2009], Samei E., Hsieh J., eds., 72582B (2009).

GPU-accelerated SART

- Back-projection (BP): voxel-driven approach (Scherl et al.[2])
- Forward-projection (FP):
  - based on ray casting
  - CUDA 2.0 supports 3-D textures
  - enabled hardware support for trilinear interpolation of sample points
- Un-matched pair forward-projector and back-projector (Zeng et al.[3])
- Texture update procedure:
  - copy whole volume into texture (one cuda call)



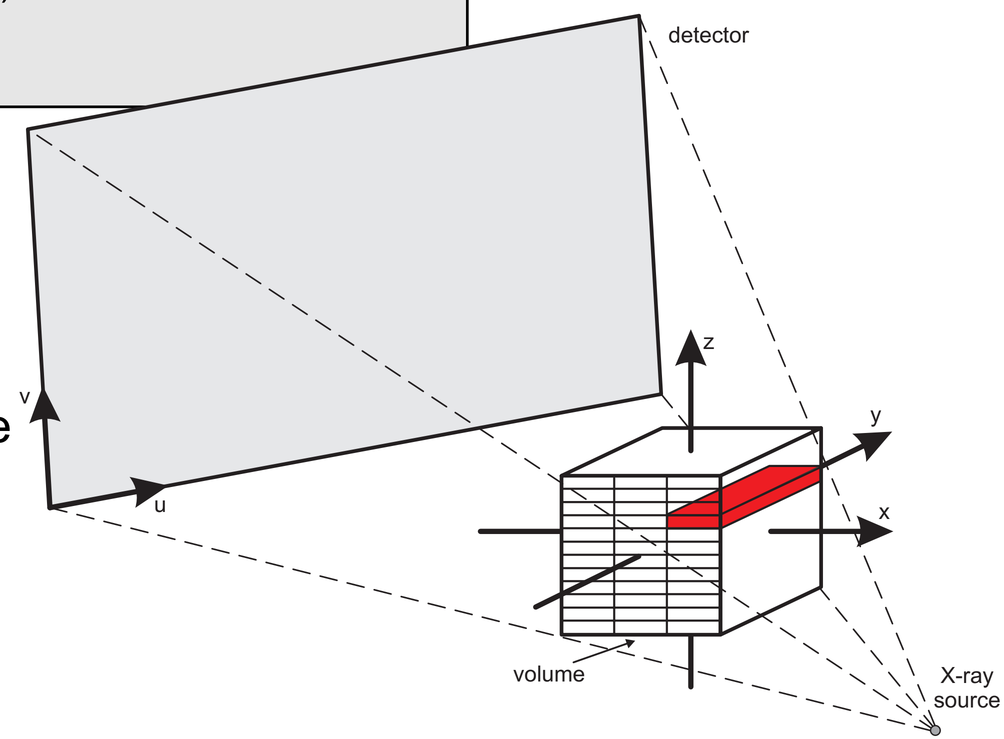
[2]Scherl, H., Keck, B., Kowarschik, M., and Hornegger, J., “Fast GPU-Based CT Reconstruction using the Common Unified Device Architecture (CUDA),” in [Nuclear Science Symposium, Medical Imaging Conference 2007], Frey, E. C., ed., 4464–4466 (2007).
[3]Zeng, G. and Gullberg, G., “Unmatched projector/backprojector pairs in an iterative reconstruction algorithm,” IEEE Transactions on Medical Imaging 19, 548–555 (May 2000).

Back-projection using CUDA

Host:
For selected projections P_j
    Call kernel;

Kernel:
Compute voxel x and z coordinate;
For all voxels (x,y,z), y=[0 N_y[... number of voxels in y-direction
    Compute the coordinates (u,v) of voxel (x,y,z) in projection P_j
    Get the projection value at position (2-D texfetch)
    Add the weighted value to voxel

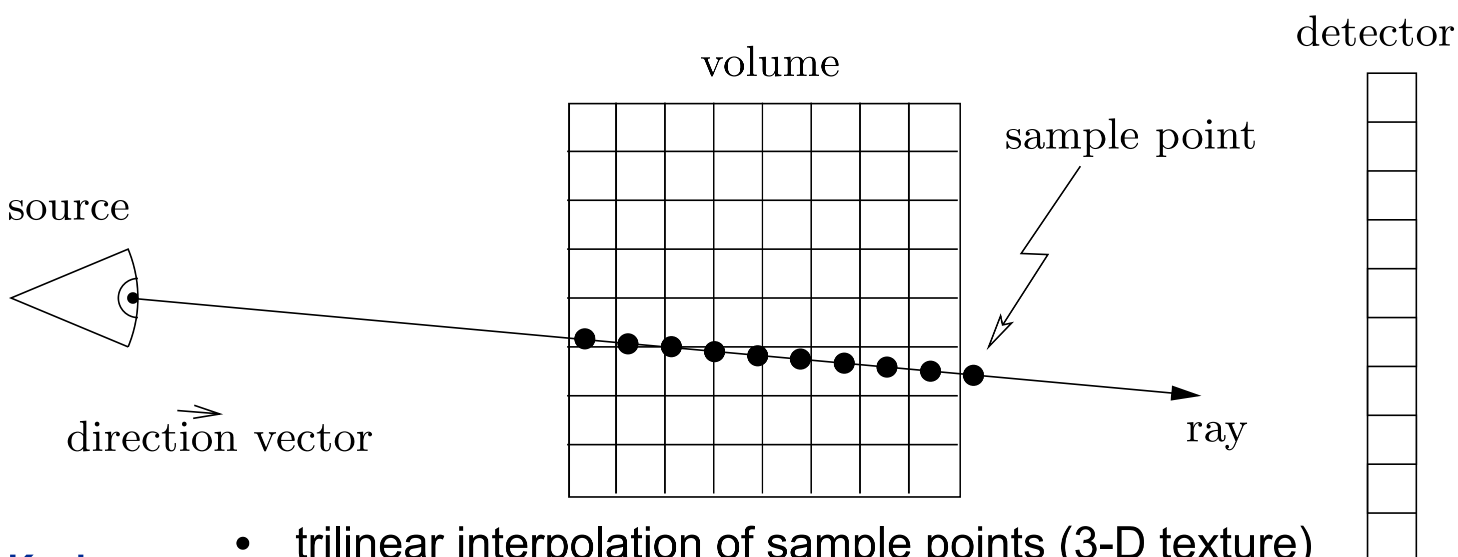
- Whole writable volume in device memory
- Current projection / corrective image in 2-D texture memory



Forward-projection using ray casting

Host:
For selected projections P_j
    Compute source position out of projection matrix;
    Compute inverted projection matrix;
    Call kernel;

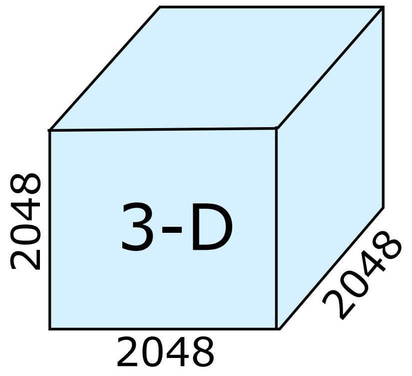
Kernel:
    Compute pixel u and v coordinate and the normalized ray direction;
    Compute entrance and exit point of the ray to the volume;
    Perform ray casting: see illustration;
    Normalize pixel value to world coordinate system units;

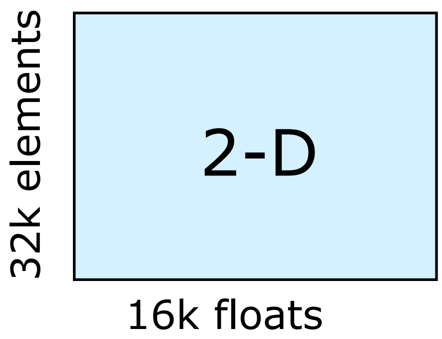


- trilinear interpolation of sample points (3-D texture)

Limitations

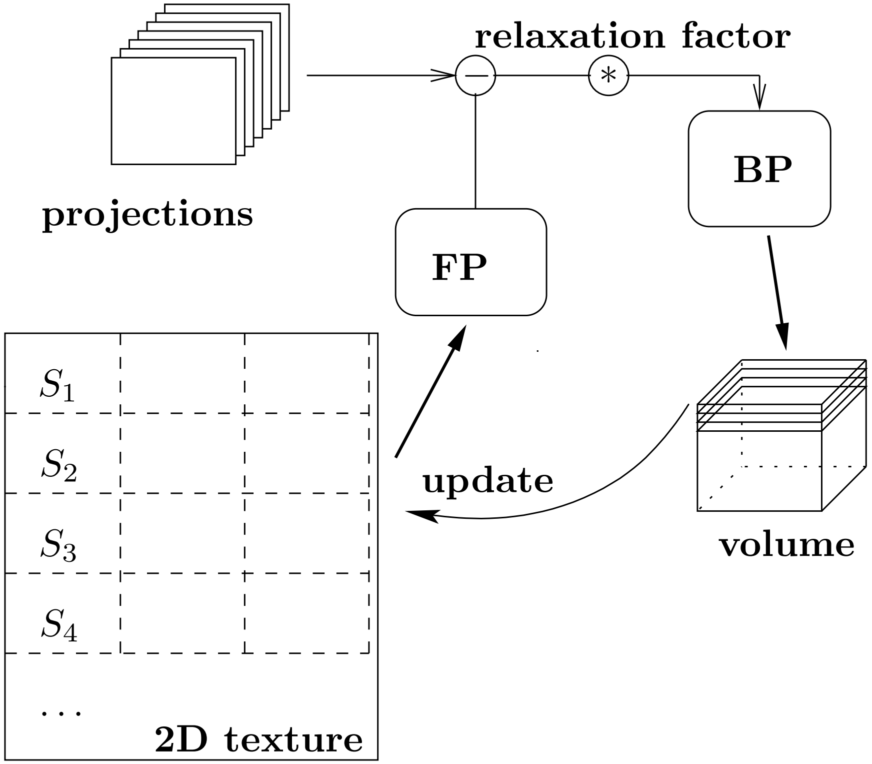
- GPU device memory:
  - QuadroFX 5600: 1.5 GB
  - QuadroFX 5800 / Tesla C1060: 4 GB
- Texture size limitations:
  - 3-D arrays in CUDA 2.0 and OpenGL: 2048^3 elements
  - 2-D texture arrays in CUDA 2.0: 16k × 32k float elements
  - 1-D texture arrays in CUDA 2.0: 8k elements
  - 1-D linear texture $2^{27} = 512^3$ elements: (512 MB float)
- High resolution example from 3-D mammography:
  - 3072 × 2048 × 50 (≈ 1.2 GB)
  - fits into device memory
  - slice resolution exceeds 2048 × 2048 elements
  - ERROR in forward-projection: 3-D texture limitation





Possible solution I: CUDA 1.1 approach[1]

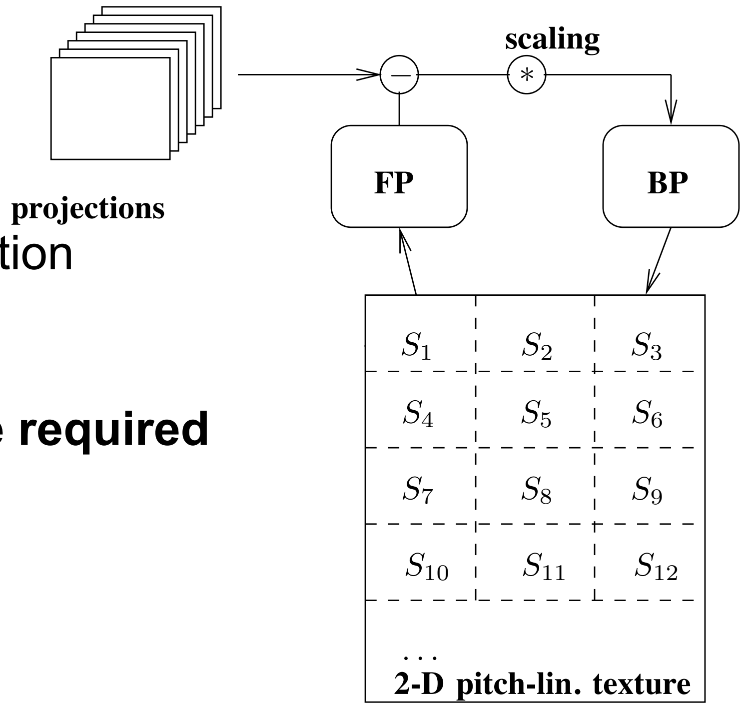
- **Forward-projection (FP) from 2-D texture array:**
  - spread volume slices S_i into 2-D texture array
  - fetch two bilinear interpolated (hardware) values from proximate slices
  - kernel computes sample point by linear interpolation (software)
- **Texture update procedure:**
  - slice-wise copy
  - slow (~1s for a 512^3 vol.)



[1] Keck, B., Hofmann, H.G., Scherl, H., Kowarschik, M., and Hornegger, J., "GPU-accelerated SART reconstruction using the CUDA programming environment," in [Proceedings of SPIE Conference, Lake Buena Vista 2009], Samei E., Hsieh J., eds., 72582B (2009).

Possible solution II: new approach

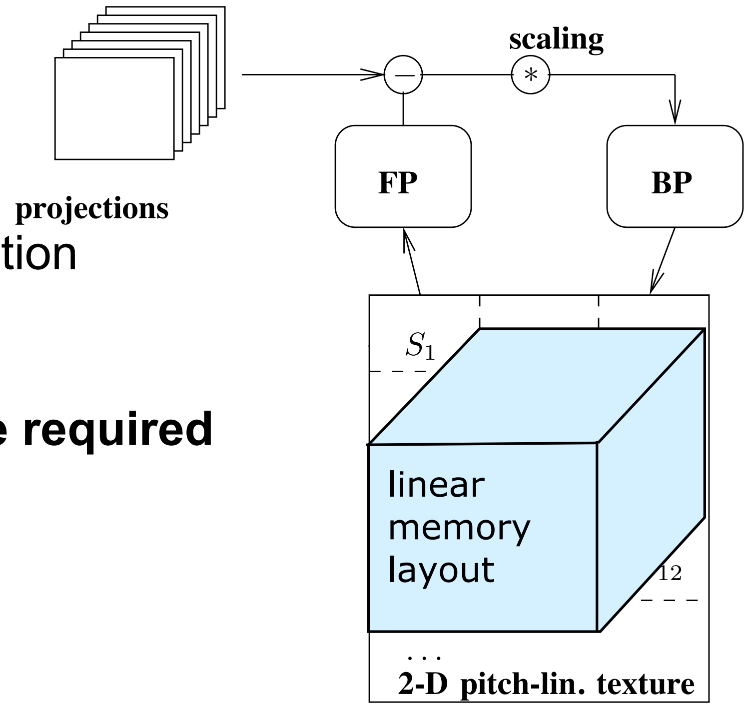
- **FP: 2-D texture from pitchlinear memory:**
  - CUDA 2.2 feature (released May 2009)
  - 16k × 32k float elements (2 GB)
  - hardware-accelerated bilinear interpolation
  - linear interpolation in software
  - single volume copy: **no texture update required**



- **BP: different memory layout**
  - adapt memory address computation due to chosen layout

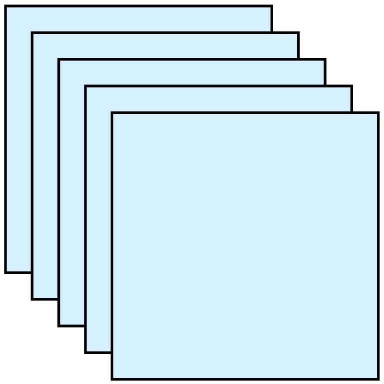
Possible solution II: new approach

- **FP: 2-D texture from pitchlinear memory:**
  - CUDA 2.2 feature (released May 2009)
  - 16k × 32k float elements (2 GB)
  - hardware-accelerated bilinear interpolation
  - linear interpolation in software
  - single volume copy: **no texture update required**

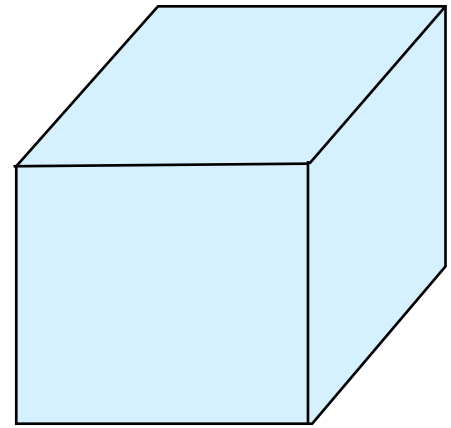


- **BP: different memory layout**
  - adapt memory address computation due to chosen layout
  - final memory resort to linear layout

Experimental setup for performance comparison



Projections:
228 projections
à 256x128 pixel



Volume:
512x512x350

- Performing 20 iterations
- Step size used in ray cast algorithm: 0.3 of uniform voxel size

Compared systems:

GPU:
NVIDIA
QuadroFX 5600

GPU:
NVIDIA
Tesla C1060

Reconstruction time comparison

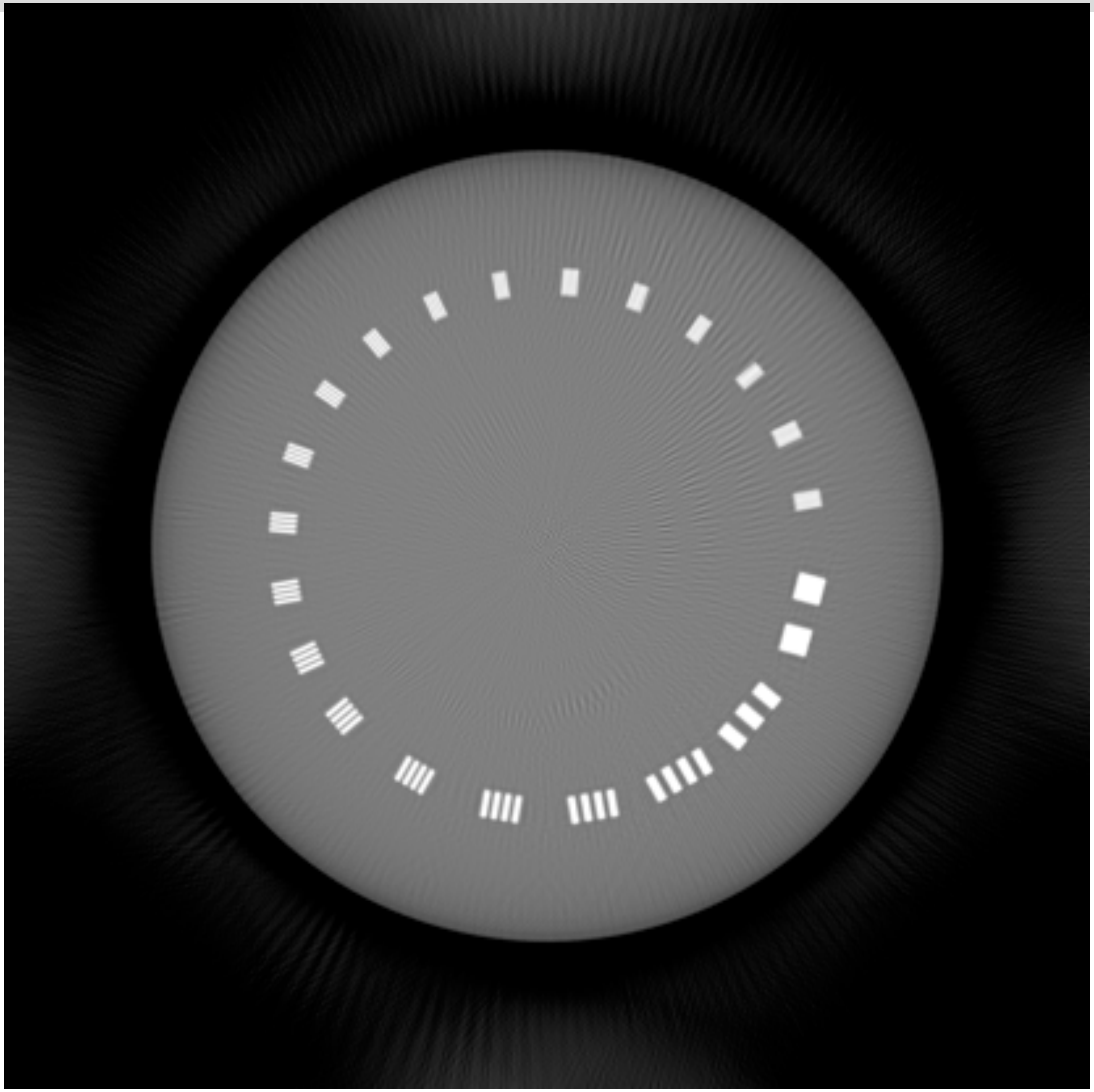
Hardware	QuadroFX 5600			Tesla C1060
volume representation (FP)	**2-D texturearray**	**3-D texturearray**	**2-D pitch-linear texture**	**2-D pitch-linear texture**
volume representation (BP)	global memory (linear)	global memory (linear)	global memory (spec. arrangement)	global memory (spec. arrangement)
device memory required [MB]	700	700	350	350
volume synchr. needed	YES	YES	NO	NO
required CUDA version	≥ CUDA 1.1	≥ CUDA 2.0	≥ CUDA 2.2	≥ CUDA 2.2
SART performance in [s]*	**4234**	**844**	**1488**	**955**

*preliminary results

Proof of concept

- **High resolution phantom:**
  - Phantomlab Catphan CTP 528
  - 21 high contrast line pairs

- **SART reconstructions:**
  - 400 simulated phantom projections à 1024x128 pixel
  - 20 iterations



Proof of concept

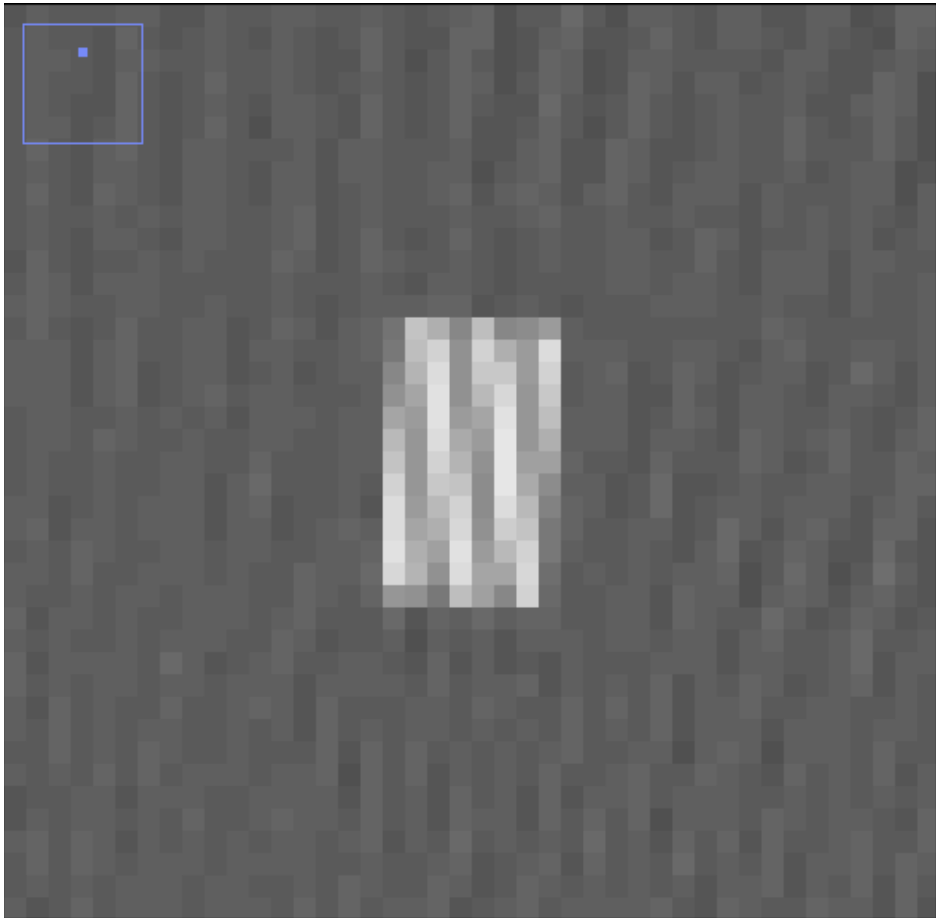
- **High resolution phantom:**
  - Phantomlab Catphan CTP 528
  - 21 high contrast line pairs
- **SART reconstructions:**
  - 400 simulated phantom projections à 1024x128 pixel
  - 20 iterations

Hardware	**Tesla C1060**			
volume resolution	512^2 x 100	1024^2 x 100	2048^2 x 100	3072 x 2048 x 50
voxel size in mm	0.4 x 0.4 x 0.1	0.2 x 0.2 x 0.1	0.1 x 0.1 x 0.1	0.075 x 0.1 x 0.1
device memory required [MB]	100	400	1600	1200
SART per-formance in [s]*	**1166**	**2407**	**11353**	**4951**

***preliminary results**

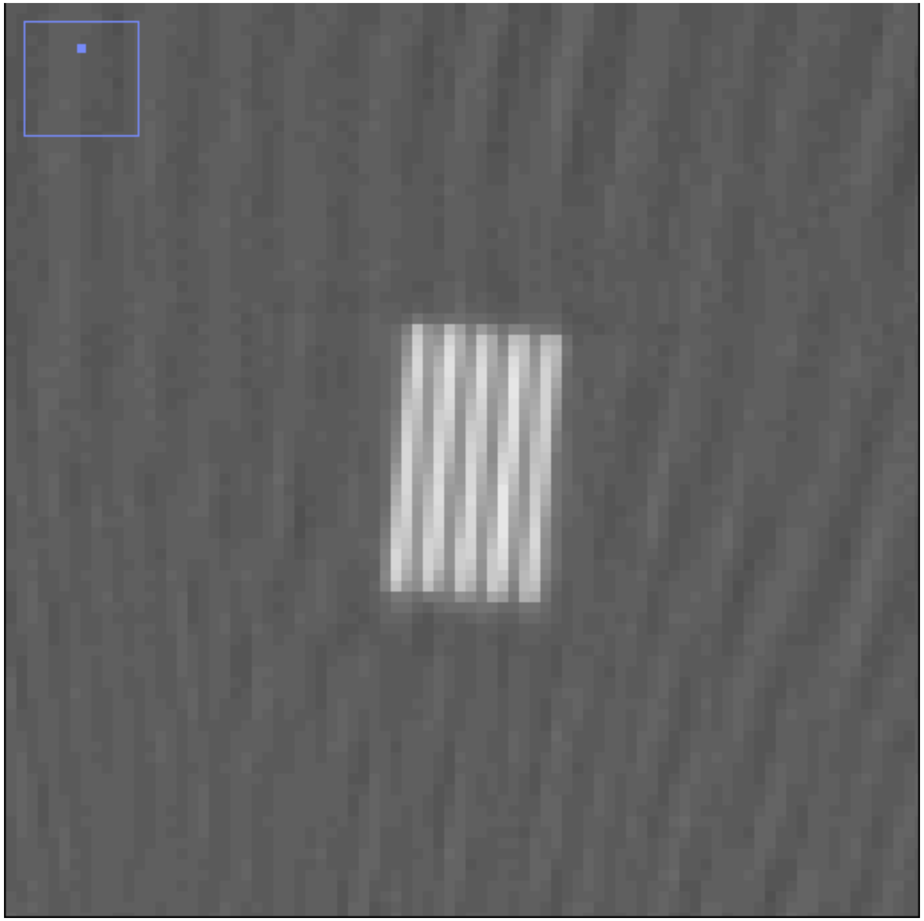
Benjamin Keck

Proof of concept



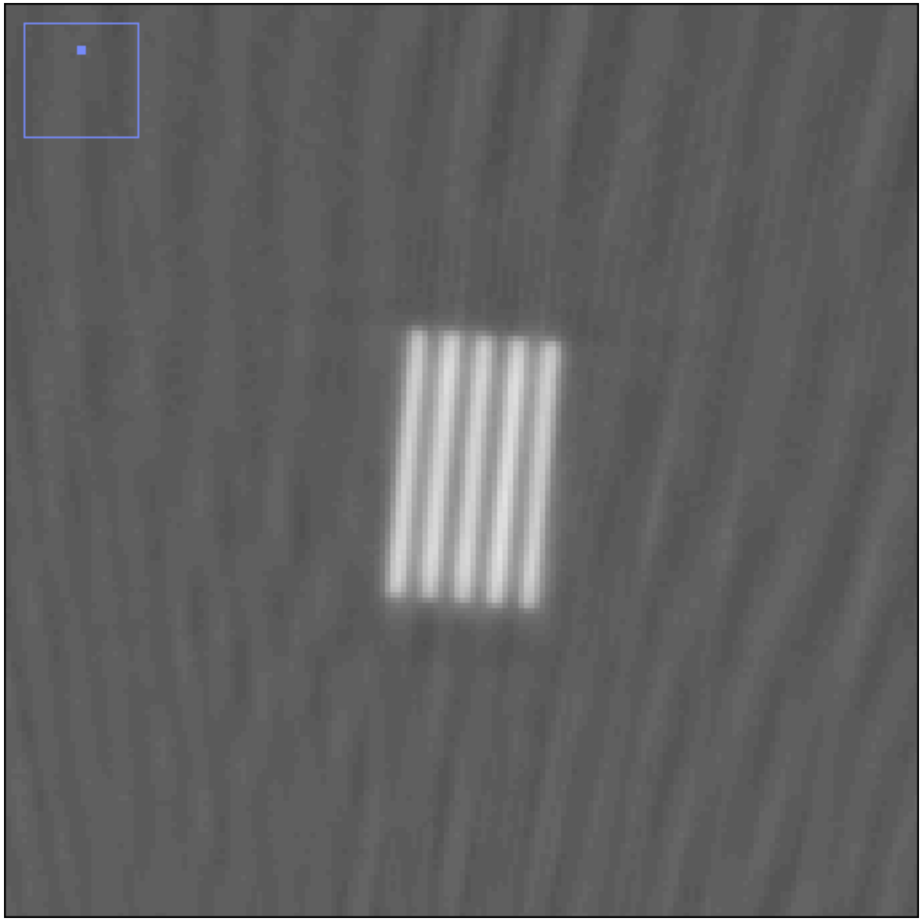
512^2

Proof of concept



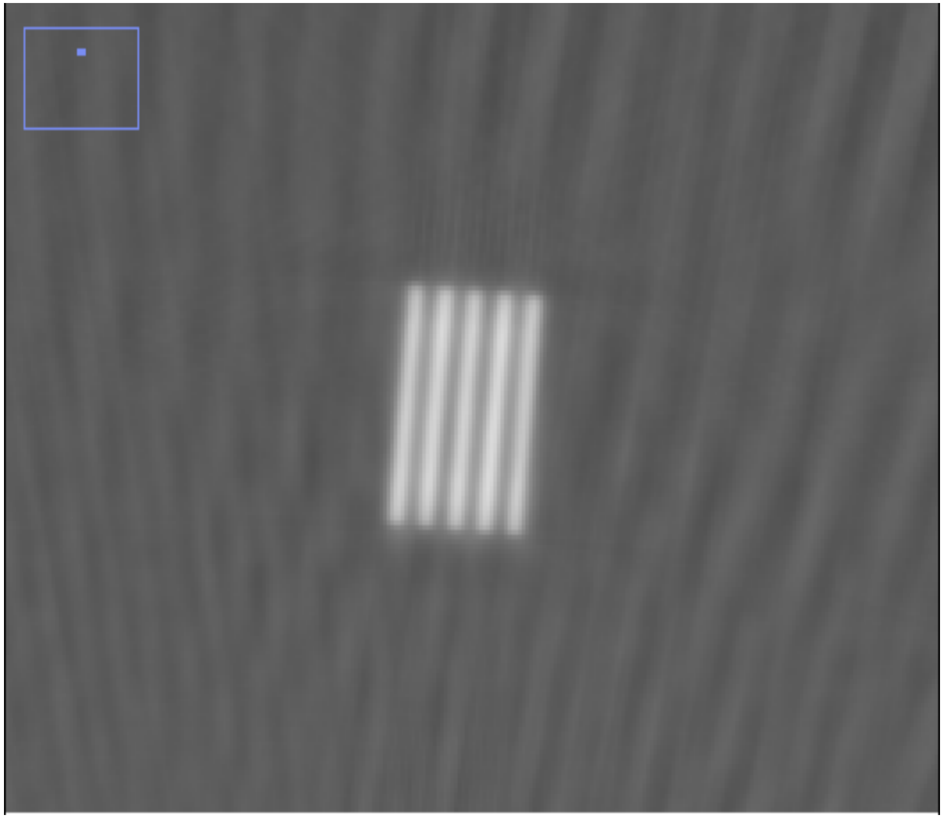
1024^2

Proof of concept



2048^2

Proof of concept



3072 x 2048

Conclusion

- enhanced GPU-accelerated SART
- pro/cons of 3-D texture usage
- trade-off solution for high (non-compatible) resolutions
- proof of concept

Thanks to HPMI for the travel grant

Thanks for your attention