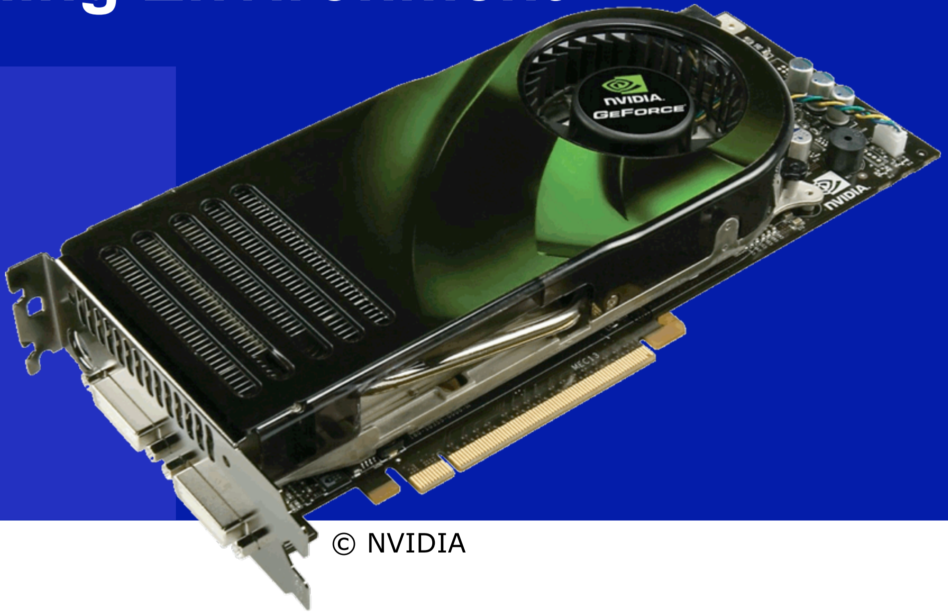
GPU-Accelerated SART Reconstruction Using the CUDA Programming Environment

SPIE 2009

February 12th, 2009



© NVIDIA

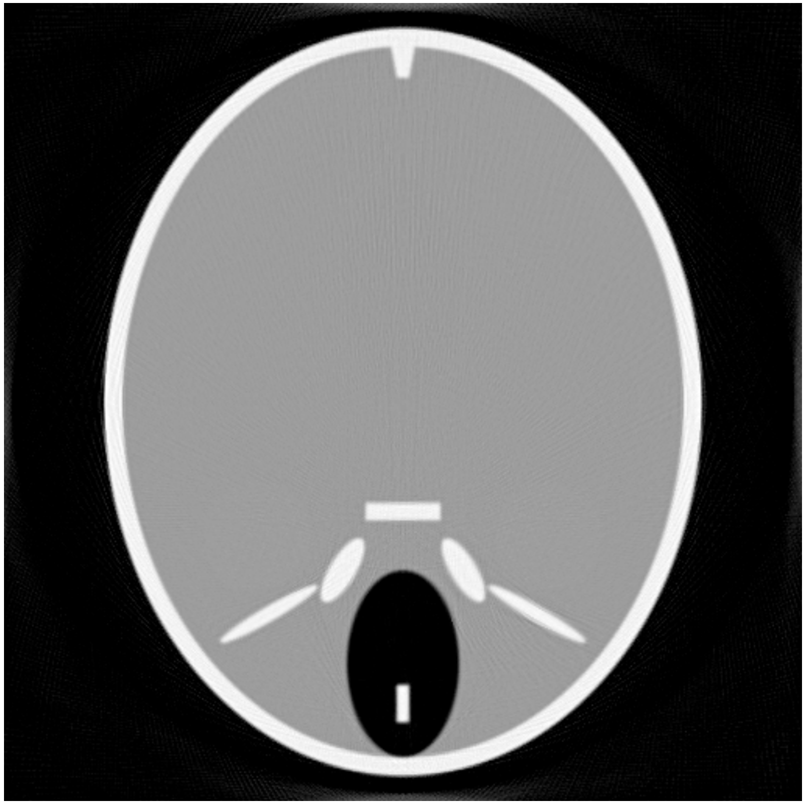
Benjamin Keck[1,2], Hannes Hofmann[1], Holger Scherl[2], Markus Kowarschik[2] and Joachim Hornegger[1]

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

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

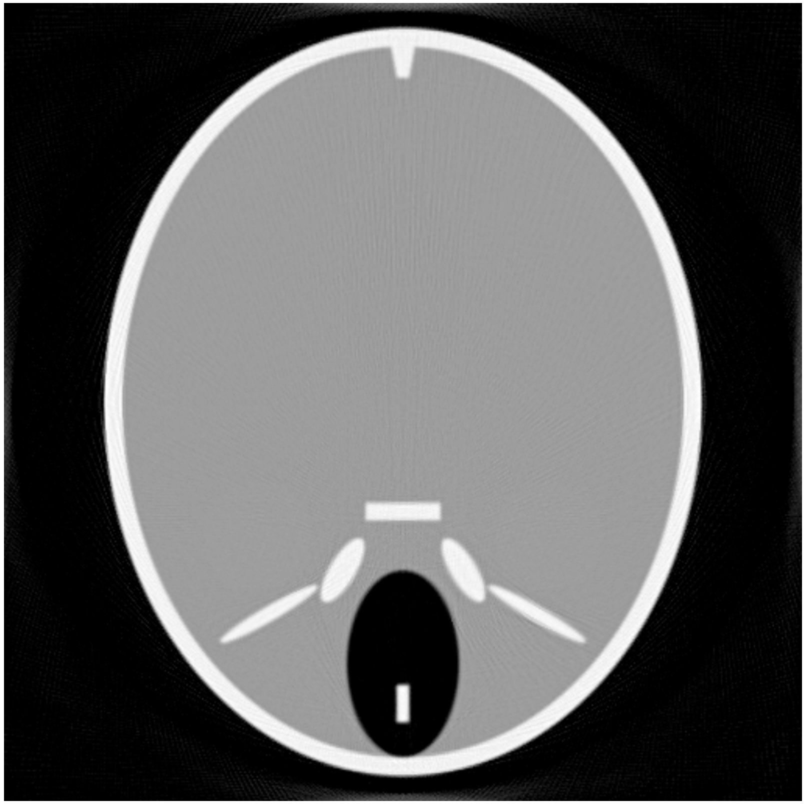
Outline

- Motivation
- Algebraic Reconstruction Techniques
- First Approach (CUDA 1.1)
- Second Approach (CUDA 2.0)
- Experimental Setup & Results
- Discussion & Conclusion
- Outlook



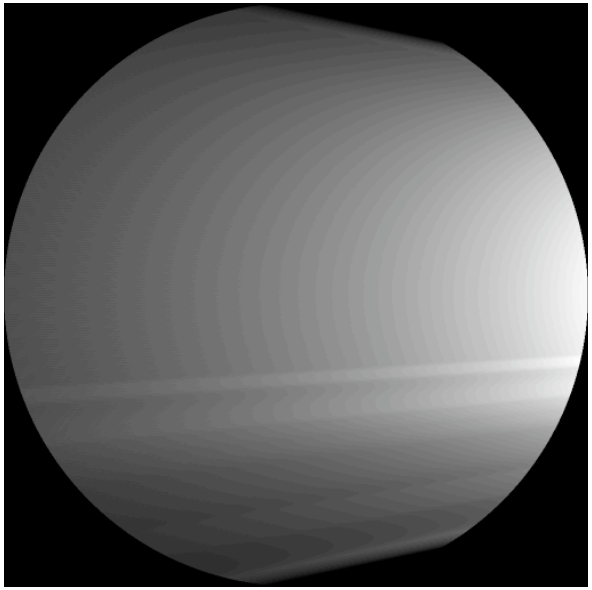
Outline

- **Motivation**
- Algebraic Reconstruction Techniques
- First Approach (CUDA 1.1)
- Second Approach (CUDA 2.0)
- Experimental Setup & Results
- Discussion & Conclusion
- Outlook



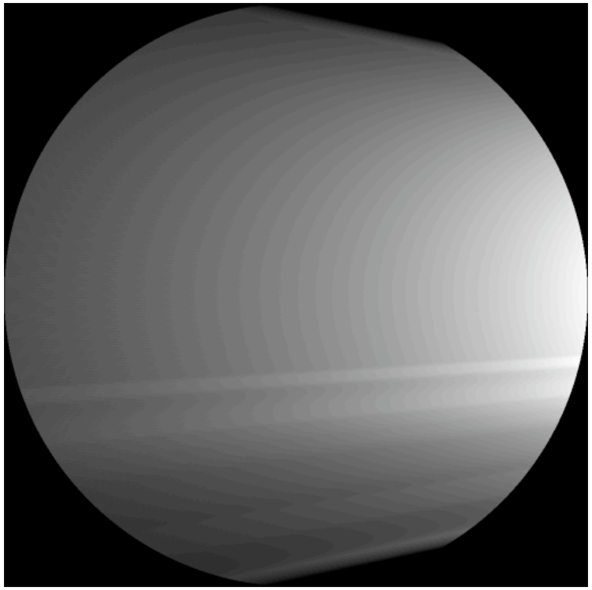
Motivation

- Simultaneous Algebraic Reconstruction Technique (SART):
  - well-studied reconstruction method for cone-beam CT scanners
  - rarely used due to its computational demands
  - many advantages in specific scenarios over the more popular FBP



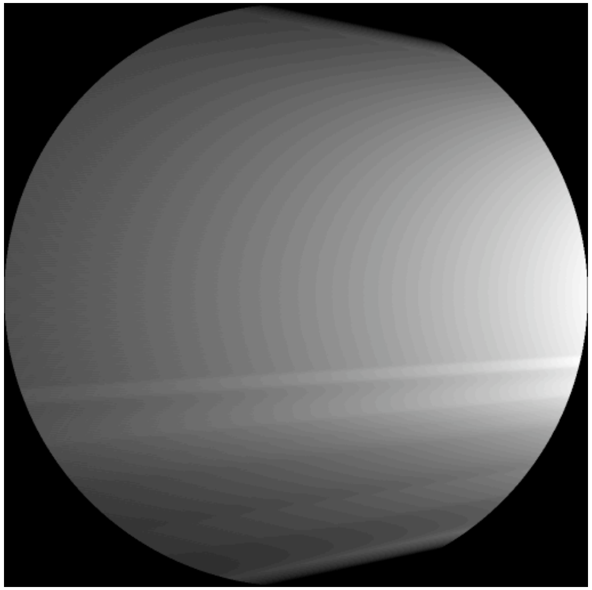
Motivation

- Simultaneous Algebraic Reconstruction Technique (SART):
  - well-studied reconstruction method for cone-beam CT scanners
  - rarely used due to its computational demands
  - many advantages in specific scenarios over the more popular FBP
- Geometry setup:
  - volume size: 512x512x350 voxels
  - 228 projections, each 256x128 pixels



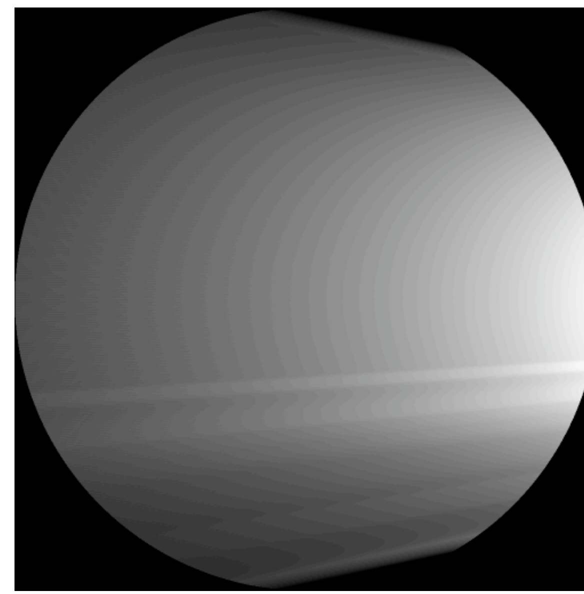
Motivation

- Simultaneous Algebraic Reconstruction Technique (SART):
  - well-studied reconstruction method for cone-beam CT scanners
  - rarely used due to its computational demands
  - many advantages in specific scenarios over the more popular FBP
- Geometry setup:
  - volume size: 512x512x350 voxels
  - 228 projections, each 256x128 pixels
- SART runtime with 20 iterations:
  - about 9 hours on off-the-shelf dual-core PC
  - about 2 hours on 8-core workstation



Motivation

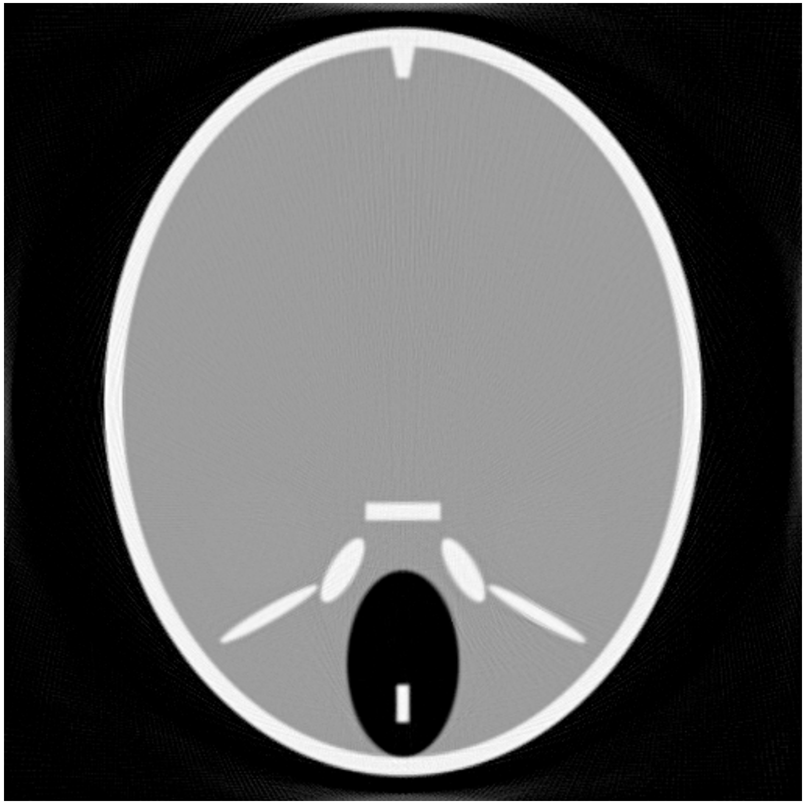
- Simultaneous Algebraic Reconstruction Technique (SART):
  - well-studied reconstruction method for cone-beam CT scanners
  - rarely used due to its computational demands
  - many advantages in specific scenarios over the more popular FBP
- Geometry setup:
  - volume size: 512x512x350 voxels
  - 228 projections, each 256x128 pixels



- SART runtime with 20 iterations:
  - about 9 hours on off-the-shelf dual-core PC
  - about 2 hours on 8-core workstation
- Accelerate reconstruction using
  NVIDIAs Common Unified Device Architecture (CUDA)

Outline

- Motivation
- **Algebraic Reconstruction Techniques**
- First Approach (CUDA 1.1)
- Second Approach (CUDA 2.0)
- Experimental Setup & Results
- Discussion & Conclusion
- Outlook



Algebraic Reconstruction Techniques

- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Algebraic Reconstruction Techniques

- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Method:	ART	SART	SIRT	Ordered Subsets (OS)
Update current volume estimation by computation of				

Algebraic Reconstruction Techniques

- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Method:	ART	SART	SIRT	Ordered Subsets (OS)
Update current volume estimation by computation of	each ray			

Algebraic Reconstruction Techniques

- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Method:	ART	SART	SIRT	Ordered Subsets (OS)
Update current volume estimation by computation of	each ray	each projection		

Algebraic Reconstruction Techniques

- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Method:	ART	SART	SIRT	Ordered Subsets (OS)
Update current volume estimation by computation of	each ray	each projection	all projections	

Algebraic Reconstruction Techniques

- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Method:	ART	SART	SIRT	Ordered Subsets (OS)
Update current volume estimation by computation of	each ray	each projection	all projections	a subset of all projections

Algebraic Reconstruction Techniques

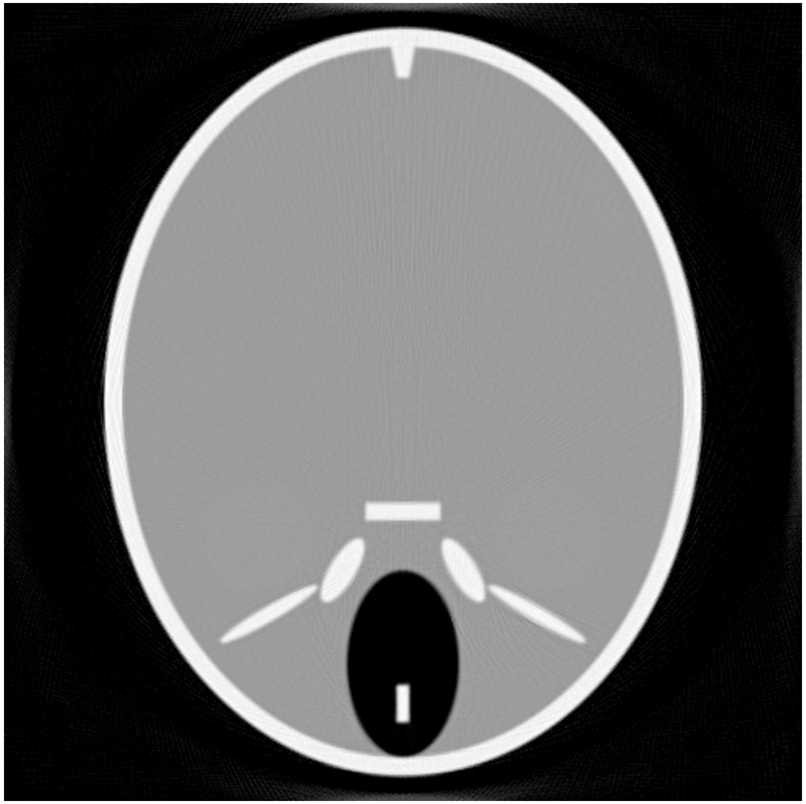
- Solve a system of linear equations according to the Kaczmarz method
- The followings methods can be distinguished by their update rule:

Method:	ART	SART	SIRT	Ordered Subsets (OS)
Update current volume estimation by computation of	each ray	each projection	all projections	a subset of all projections

- Each method consists of two computationally intensive parts:
  - correction image computation
(including forward-projection and weighting)
  - back-projection of correction image

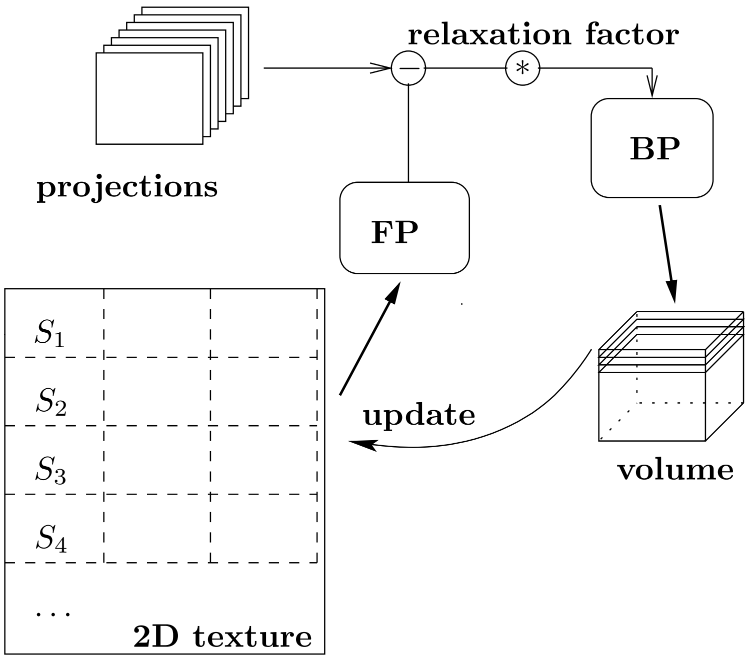
Outline

- Motivation
- Algebraic Reconstruction Techniques
- **First Approach (CUDA 1.1)**
- Second Approach (CUDA 2.0)
- Experimental Setup & Results
- Discussion & Conclusion
- Outlook



First Approach (CUDA 1.1)

- Back-projection (BP): voxel-driven approach (Scherl et al.[1])
- Forward-projection (FP):
  - based on ray casting (eligible on GPUs)
  - numerical error comparable to other popular interpolation and integration methods used in CT (Xu et al.[2])
- Unmatched pair forward-projector and back-projector (Zeng et al.[3])

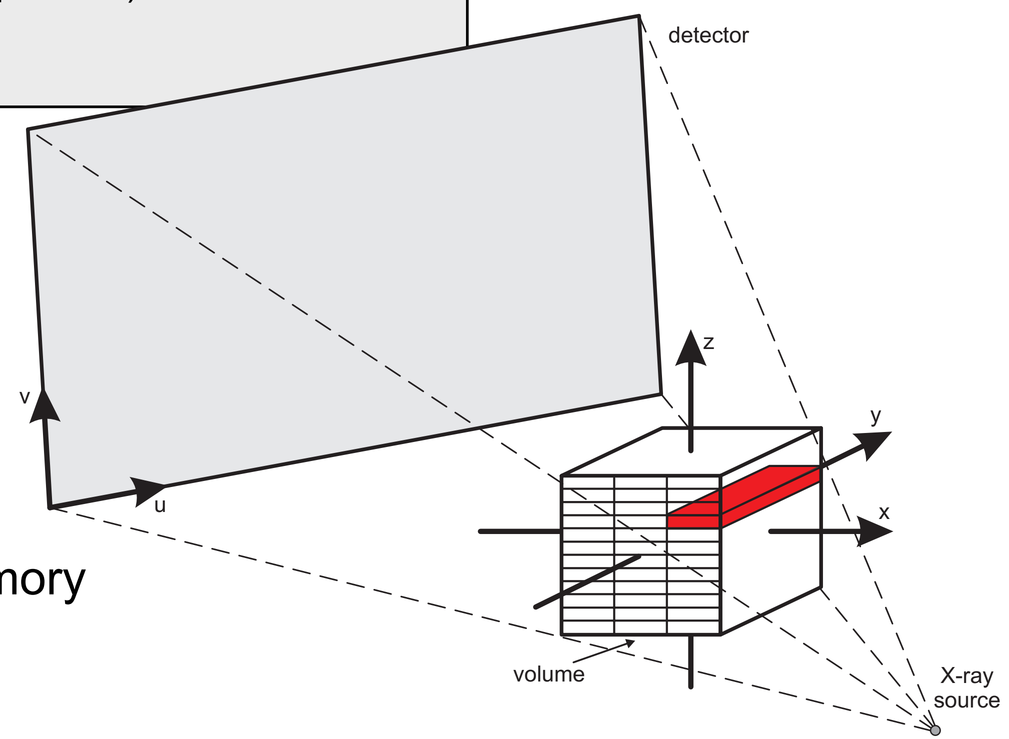


[1]Scherl, H., Keck, B., Kowarschik, M., and Hornegger, J., "Fast GPU-Based CT Reconstruction using the Common Unified Device Architecture (CUDA)," in [Nuclear Science Symposium, Medical Imaging Conference 2007], Frey, E. C., ed., 4464–4466 (2007).
[2]Xu, F. and Mueller, K., "A comparative study of popular interpolation and integration methods for use in computed tomography," Biomedical Imaging: Nano to Macro, 2006. 3rd IEEE International Symposium on, 1252–1255 (April 2006).
[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 (Scherl et al.[1])

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 (i,j) of voxel (x,y,z) in projection
    Get the projection value at position (texture interpolation)
    Add the weighted value to voxel

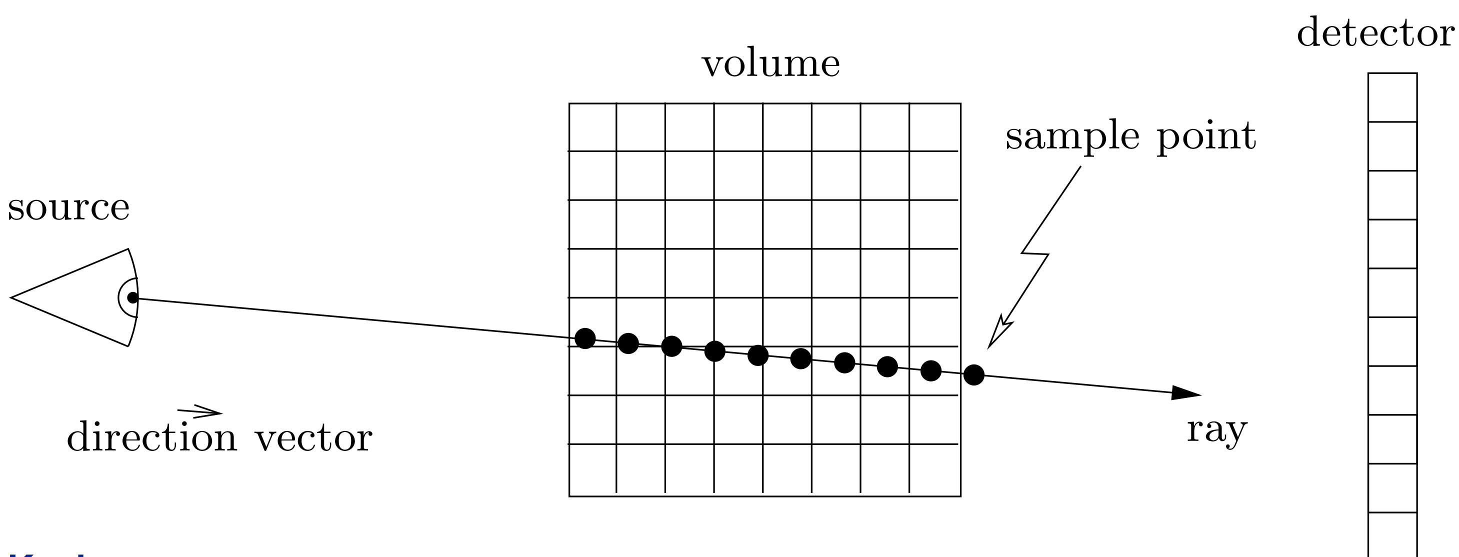


- Same implementation for both (CUDA 1.1, CUDA 2.0)
- Complete volume in device memory
- Current projection in 2D texture memory

Forward-projection using CUDA

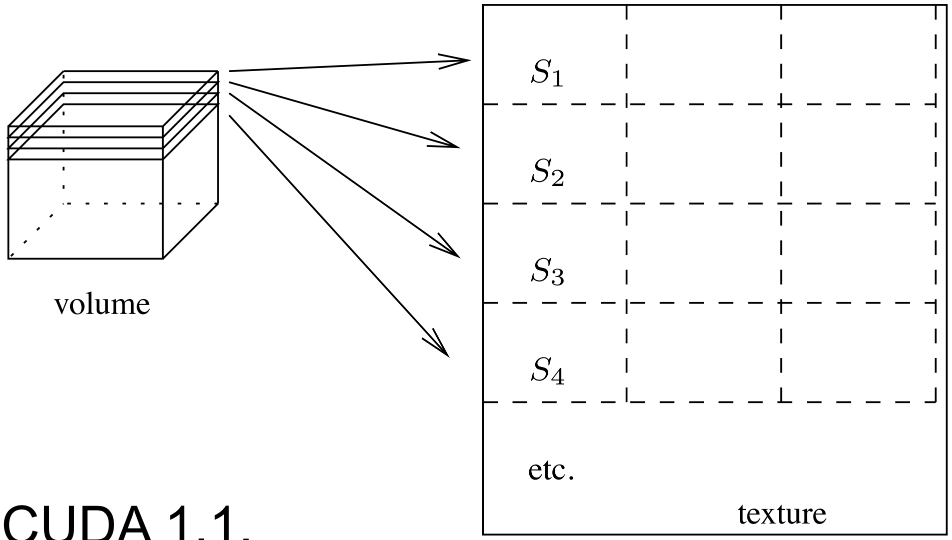
Host:
For selected projections P_j
Compute source position out of projection matrix;
    Compute inverted projection matrix;
    Call kernel;

Kernel:
Compute pixel u and v coordinate and 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



Sample Point Interpolation

- Recent graphics cards' hardware supports texture interpolation (1D, 2D, 3D)
- CUDA 1.1 supports only 1D, 2D textures, no 3D textures
- CUDA 1.1 workaround:
  - spread volume slices S_i into 2D texture
  - fetch two bilinear interpolated values from proximate slices
  - kernel computes sample point by linear interpolation



- Comparison of ray casting using CUDA 1.1, CUDA 2.0 and OpenGL see Weinlich et al.[4]

[4] Weinlich, A., Keck, B., Scherl, H., Korwarschik, M., and Hornegger, J., "Comparison of High-Speed Ray Casting on GPU using CUDA and OpenGL," in [High-performance and Hardware-aware Computing (HipHaC 2008)], Buchty, R. and Weiss, J.-P., eds., 25–30 (2008).

Texture Update Procedure

- Texture memory used by forward-projection is read-only
- Back-projection updates volume in global memory (r/w)
- Texture memory has to be synchronized with global memory
  - spread whole volume from global memory into 2D texture
  - expensive task in CUDA 1.1
    (approx. 1.15 sec for one update of a 512^3 volume with float values)
- Slightly increased number of FP and BP between two texture updates:
  - results in OS scheme
  - decreases number of texture updates and cuts total time
  - convergence remains almost at the same level (Xu et al.[5])

[5] Xu, F., Mueller, K., Jones, M., Keszthelyi, B., Sedat, J., and Agard, D., "On the Efficiency of Iterative Ordered Subset Reconstruction Algorithms for Accelerations on GPUs," (2008). Workshop on High-Performance Medical Image Computing and Computer Aided Intervention (HP-MICCAI 2008).

SART - OS distinction

SART

- ...
- **Texture update**
- Forward-projection
- Back-projection
- **Texture update**
- Forward-projection
- Back-projection
- **Texture update**
- ...

SART - OS distinction

SART

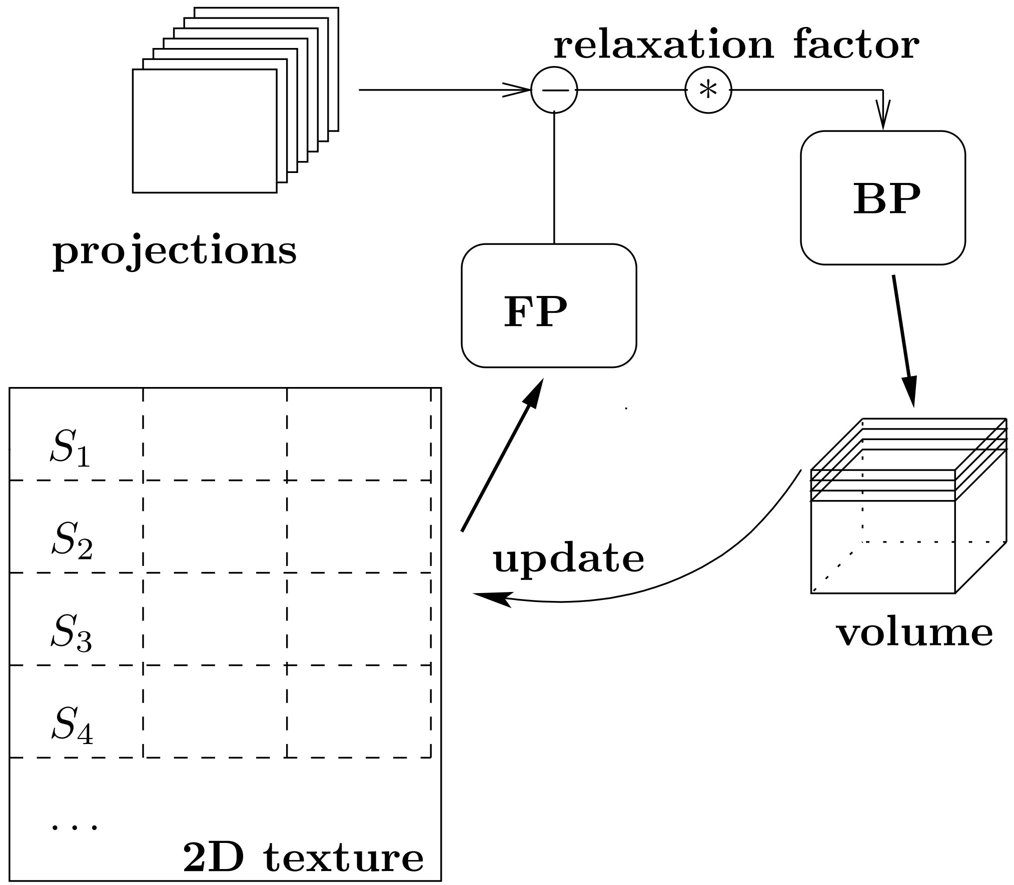
- ...
- **Texture update**
- Forward-projection
- Back-projection
- **Texture update**
- Forward-projection
- Back-projection
- **Texture update**
- ...

OS (2.proj)

- ...
- **Texture update**
- Forward-projection
- Back-projection

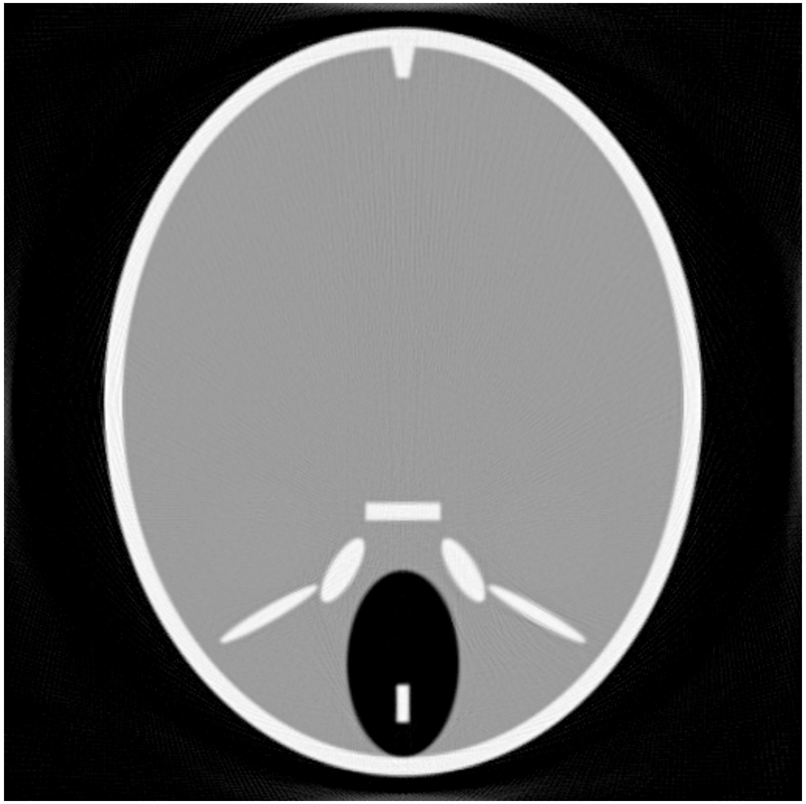
- Forward-projection
- Back-projection
- **Texture update**
- ...

First Approach (CUDA 1.1) - Concept



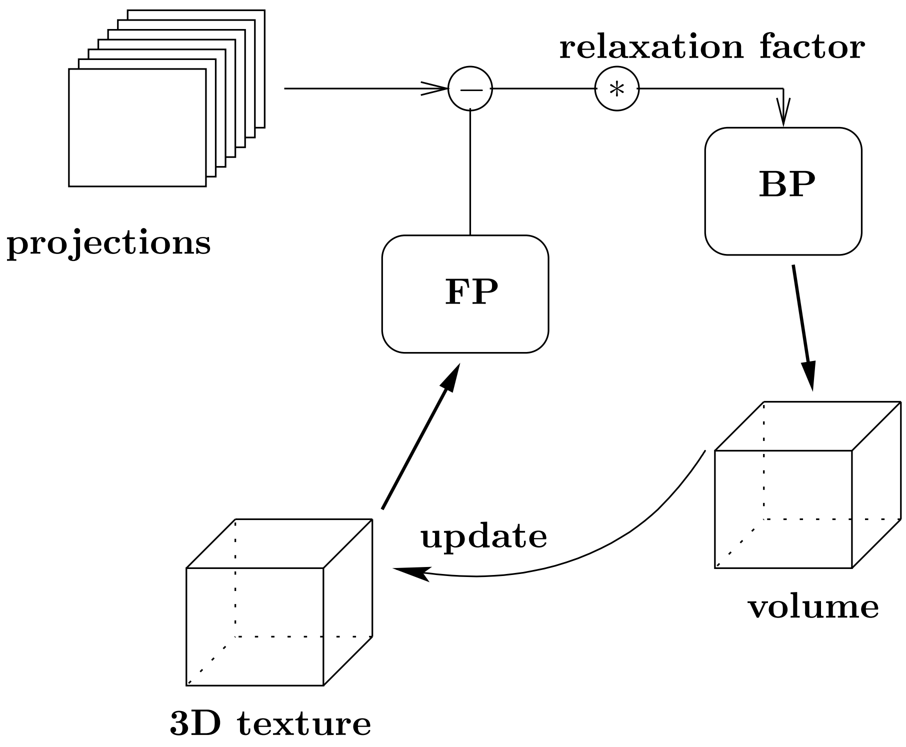
Outline

- Motivation
- Algebraic Reconstruction Techniques
- First Approach (CUDA 1.1)
- **Second Approach (CUDA 2.0)**
- Experimental Setup & Results
- Discussion & Conclusion
- Outlook



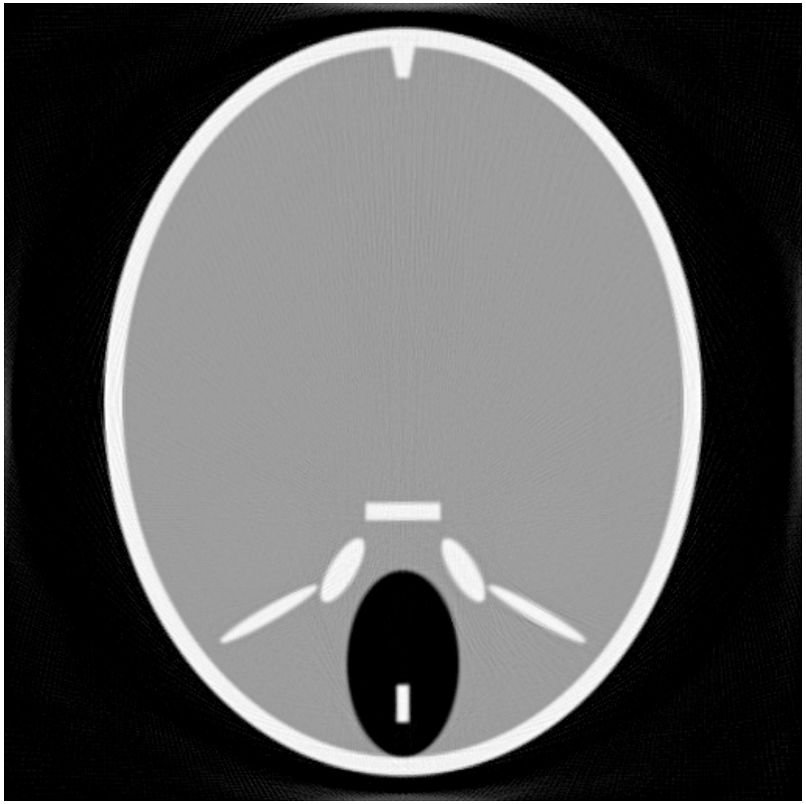
CUDA 2.0 Approach

- Back-projection remains same implementation
- Difference in forward-projection
  - CUDA 2.0 supports 3D textures
  - enabled hardware support for trilinear interpolation
- Easier texture update procedure
  - single instruction copy
  - update approx. 10 times faster



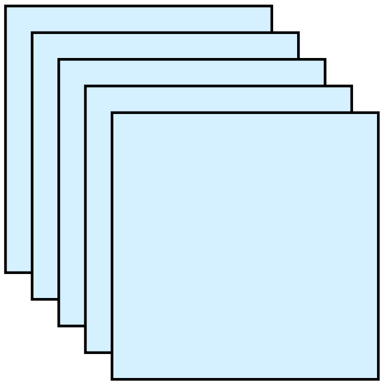
Outline

- Motivation
- Algebraic Reconstruction Techniques
- First Approach (CUDA 1.1)
- Second Approach (CUDA 2.0)
- **Experimental Setup & Results**
- Discussion & Conclusion
- Outlook



Experimental Setup

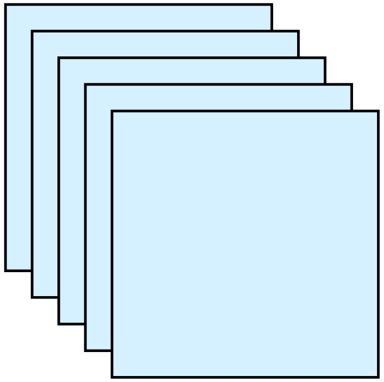
Experimental Setup



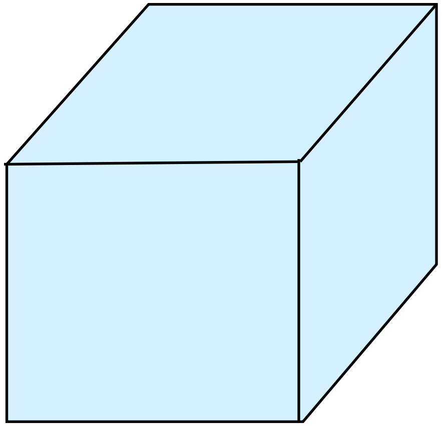
Projections:
228 projections
à 256x128 pixel

Benjamin Keck

Experimental Setup

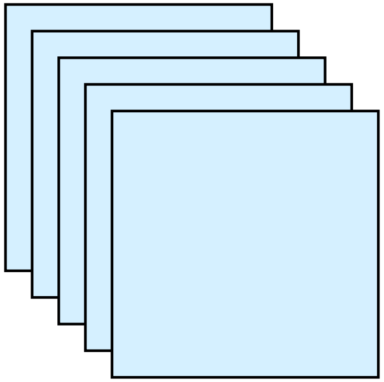


Projections:
228 projections
à 256x128 pixel

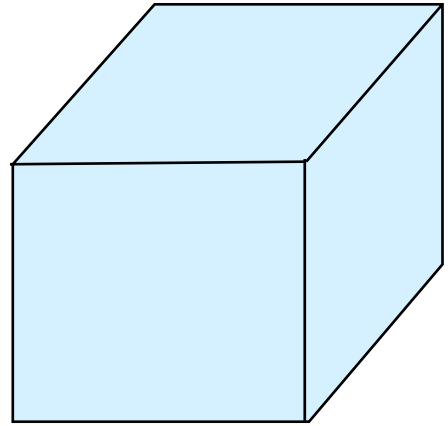


Volume:
512x512x350

Experimental Setup



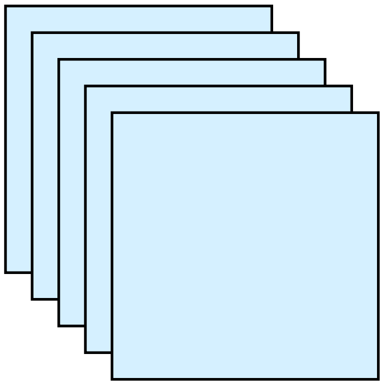
Projections:
228 projections
à 256x128 pixel



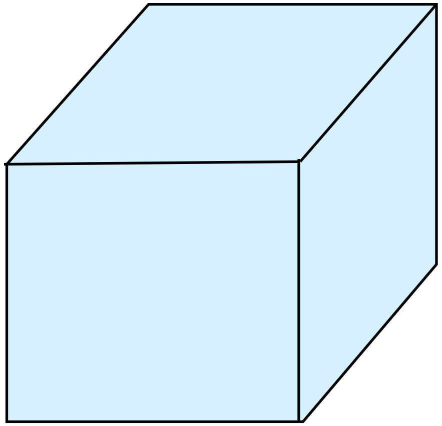
Volume:
512x512x350

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

Experimental Setup



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:

Off-the-shelf PC:	Workstation:	GPU:
Intel Core2Duo	Two Intel QuadCore	NVIDIA
@ 2 GHz	@ 2.33 GHz	QuadroFX 5600

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**			
OS(2proj.)	”	”		
OS(5proj.)	”	”		
OS(7proj.)	”	”		
OS(10proj.)	”	”		

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630		
OS(2proj.)	"	"		
OS(5proj.)	"	"		
OS(7proj.)	"	"		
OS(10proj.)	"	"		

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	
OS(2proj.)	"	"		
OS(5proj.)	"	"		
OS(7proj.)	"	"		
OS(10proj.)	"	"		

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	$2\times$Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	
OS(2proj.)	”	”	2435	
OS(5proj.)	”	”		
OS(7proj.)	”	”		
OS(10proj.)	”	”		

Results

	512 × 512 × 350 voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	
OS(2proj.)	"	"	2435	
OS(5proj.)	"	"	1359	
OS(7proj.)	"	"		
OS(10proj.)	"	"		

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	
OS(2proj.)	"	"	2435	
OS(5proj.)	"	"	1359	
OS(7proj.)	"	"	1156	
OS(10proj.)	"	"		

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	
OS(2proj.)	”	”	2435	
OS(5proj.)	”	”	1359	
OS(7proj.)	”	”	1156	
OS(10proj.)	”	”	998	

Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	844
OS(2proj.)	”	”	2435	661
OS(5proj.)	”	”	1359	551
OS(7proj.)	”	”	1156	530
OS(10proj.)	”	”	998	**514**

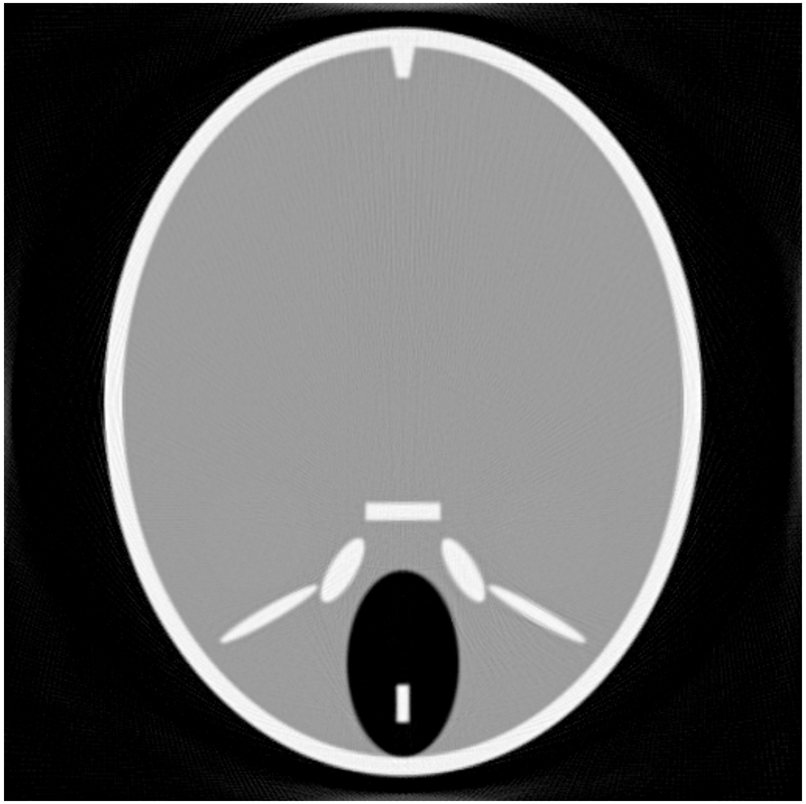
Results

	$512 \times 512 \times 350$ voxels			
Hardware/ Method	Intel Core2Duo 2 GHz Time [s]	2×Intel Xeon QuadCore 2.33 GHz Time [s]	QuadroFX 5600 CUDA 1.1 Time [s]	QuadroFX 5600 CUDA 2.0 Time [s]
SART	**32968**	6630	4234	844
OS(2proj.)	”	”	2435	661
OS(5proj.)	”	”	1359	551
OS(7proj.)	”	”	1156	530
OS(10proj.)	”	”	998	**514**

- OS optimization reduces GPU specific runtime up to 76% (CUDA 1.1), 39% (CUDA 2.0)
- CUDA 2.0 implementation (SART) outperforms CUDA 1.1 (OS 10proj.)
- Speedup factor GPU vs. CPU: 64x - 12x (PC resp. Workstation)

Outline

- Motivation
- Algebraic Reconstruction Techniques
- First Approach (CUDA 1.1)
- Second Approach (CUDA 2.0)
- Experimental Setup & Results
- **Discussion & Conclusion**
- Outlook

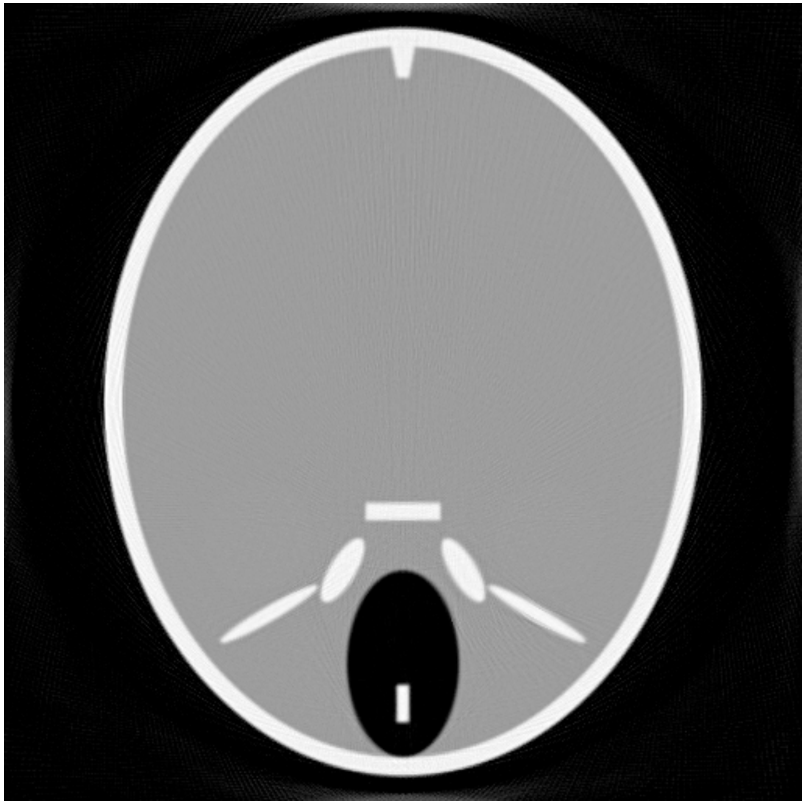


Discussion & Conclusion

- SART can be effectively performed on GPU using CUDA
- Texture memory usage:
  - benefit from hardware-accelerated interpolation
  - drawback due to necessary synchronization (especially CUDA 1.1)
- OS reduces number of time consuming synchronizations
- Significant progress between CUDA 1.1 and CUDA 2.0 for SART
- GPU implementation is already applicable for specific usage in the clinical environment (runtime < 9 minutes)

Outline

- Motivation
- Algebraic Reconstruction Techniques
- First Approach (CUDA 1.1)
- Second Approach (CUDA 2.0)
- Experimental Setup & Results
- Discussion & Conclusion
- **Outlook**



Outlook

- Most presented results on hardware-optimized reconstruction are not comparable due to variations in data acquisitions
- Open platform RabbitCT (www.rabbitCT.com)
  - back-projection performance
  - back-projection ranking (includes reference, website, paper)
  - reference implementation available
  - in-vivo dataset of a rabbit
- Computational complexity
  - Volume size (128^3, 256^3, 512^3, 1024^3)
  - 496 projections of size 1248x960

http://www.rabbitCT.com

Arnd Dörfler, Neuroradiology, University-Clinic Erlangen

Acknowledgements

- Thanks to the support by Siemens Healthcare, CV, Medical Electronics & Imaging Solutions

- The International Max Planck Research School for Optics and Imaging (IMPRS-OI)

- The Erlangen Graduate School in Advanced Optical Technologies (SAOT)

- Special thanks to Dr. Holger Kunze who supported us with his software framework for iterative reconstruction using multi-core CPUs.

Thank you for your Attention!