Visual Medicine: Part Two – Advanced Topics in Visual Medicine

CT Reconstruction and Functional Imaging

Klaus Mueller

Computer Science,
Stony Brook University

mueller@cs.sunysb.edu

Purpose and Aspirations

Purpose: Give insight into

- modalities
- acquisition devices
- algorithms

that are commonly used to generate the medical datasets that our community seeks to visualize

Aspiration 1: Deeper knowledge might be helpful to develop more target-oriented visualizations

Aspiration 2: Appreciate programmable commodity graphics hardware (even more) for GP-GPU

Overview

More on principles of Computed Tomography (CT)

- modalities: anatomical vs. functional
- analytical and iterative reconstruction methods

Architecture and Programming of GPUs

- how to take advantage of hardware features

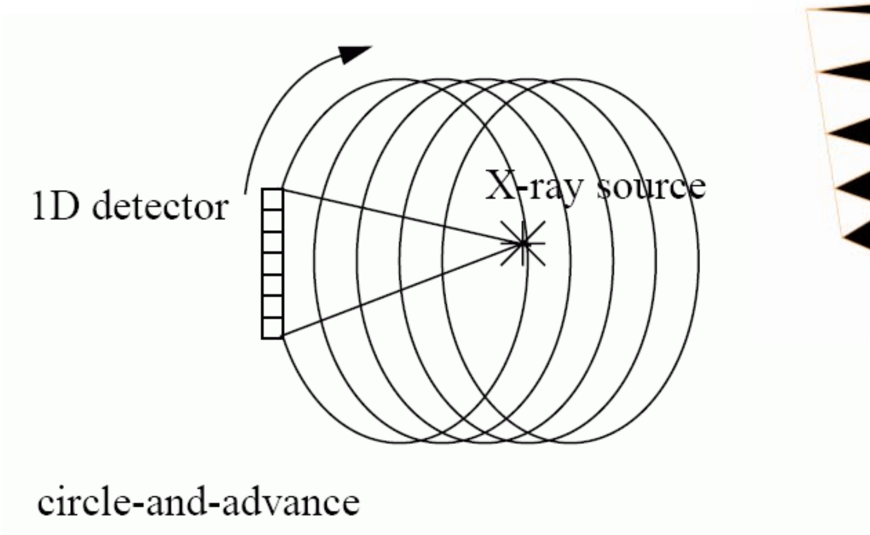
Mapping CT Algorithms to the GPU

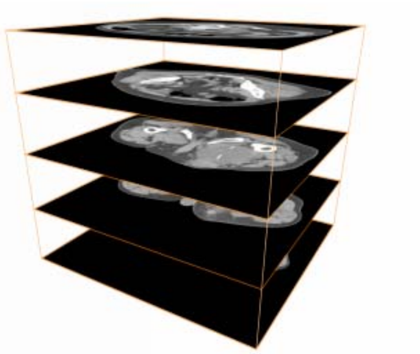
- Feldkamp, OS-EM, SART

Results, Discussion, Future Prospects

Data Acquisition: Fan-Beam CT

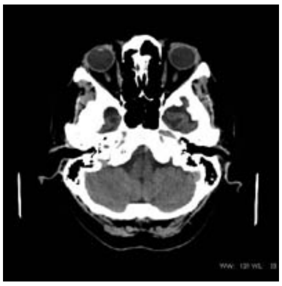
With stop-motion (no longer in use):

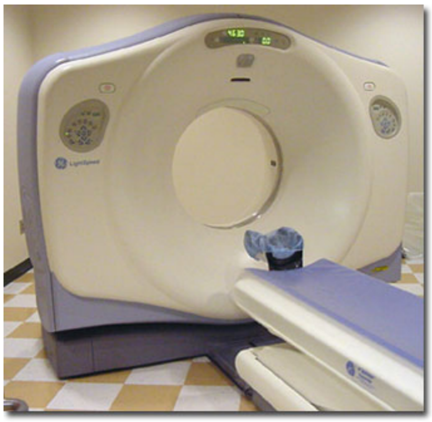


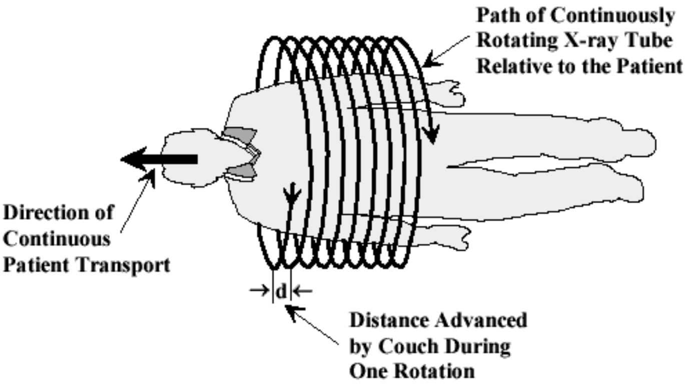


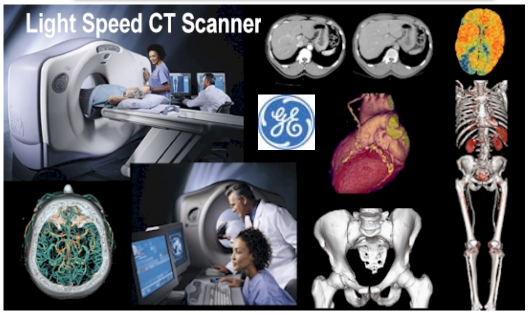
Data Acquisition: Helical (Spiral) CT

VIS 2006 BALTIMORE·MARYLAND·USA



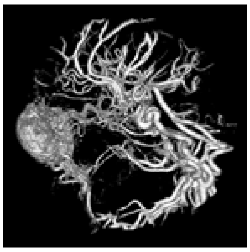


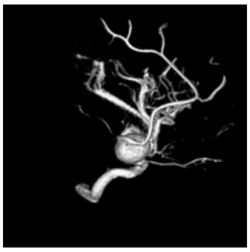


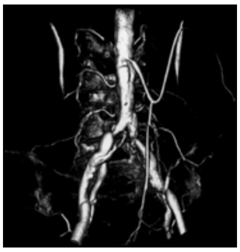


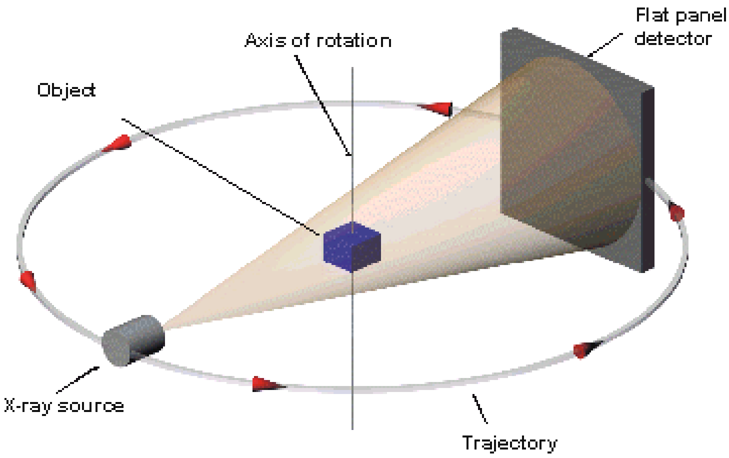
Data Acquisition: Cone-Beam

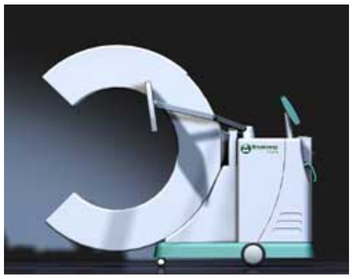
VIS 2006 BALTIMORE·MARYLAND·USA











Breakaway Imaging

Comparison: Helical (Spiral) CT

Multislice (16, 32, now 64)

Requires multiple (fast) rotations around the patient

- head-to-toe: 10s
- high-definition heart: 5 beats (5s)
- price: $$$ (but less expensive than MRI)

Comparison: Cone-Beam CT

One rotation covers a large portion of the body

- true volumetric acquisition

Can use conventional X-ray source and 2D detector

Needs fast, stable gantry (price $$)

Applications in:

- (interactive) patient setup and procedures
- trauma and emergency unit
- dental

Functional Imaging: Overview

SPECT: Single Photon Emission Tomography

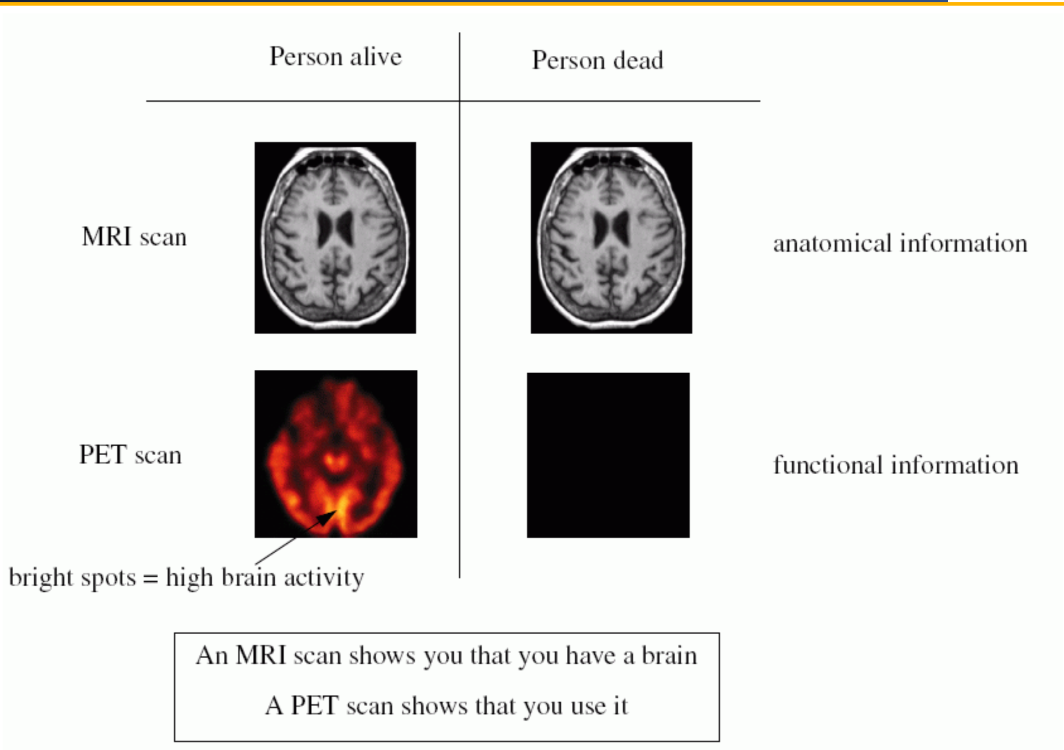
PET: Positron Emission Tomography

Also called “Metabolic Imaging”

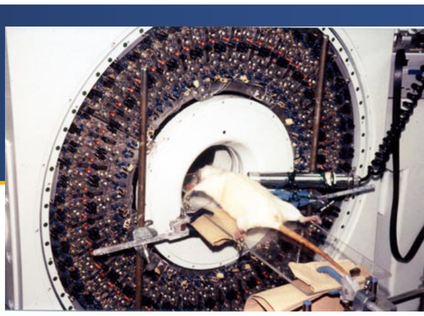
Idea:

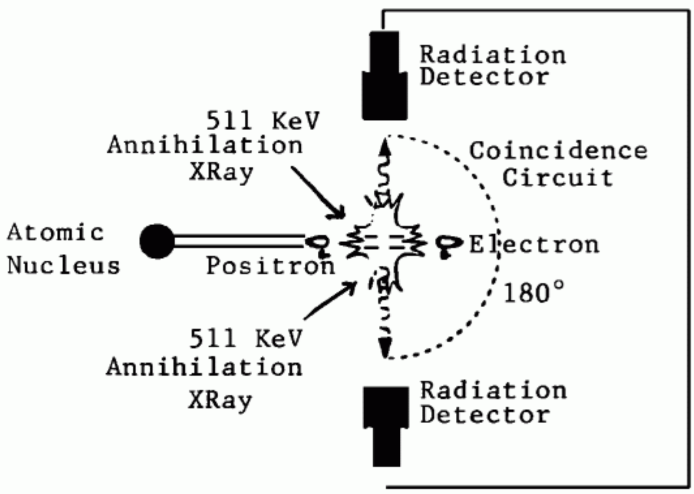
- inject (into the bloodstream) a pharmaceutical labeled with a radionucletide tracer
- pharmaceutical will go to an anatomic site with metabolic activity (e.g., an area in the brain)
- tracer will lead to photons that can be detected

Functional Imaging: Overview

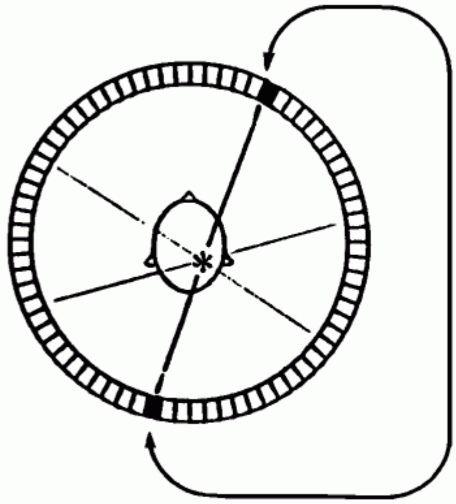


PET: Concept





Principles of Decay and Detection

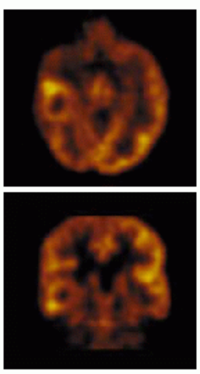


PET Detector Ring Coincidence Imaging

PET: Case Study

Usually displayed pseudo-colored:

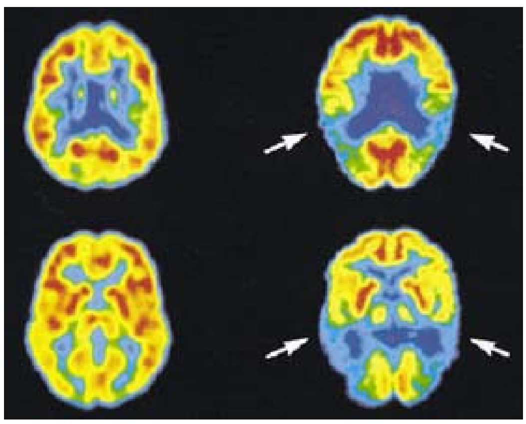
- red, yellow: high activity
- green, blue: low activity



raw

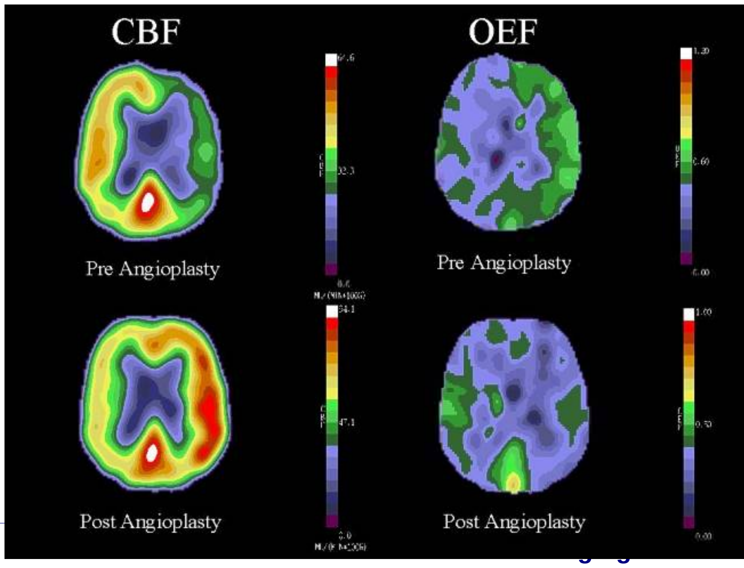
Pseudo-colored

normal Alzheimer’s



PET: Case Study

Reduced Cerebral Blood Flow (CBF) and elevated compensatory Oxygen Extraction (OEF) before and after carotid artery angioplasty (stroke risk)



SPECT: Concept

A labeled tracer (e.g., glucose) is injected into the blood stream:

- only a single photon is emitted
- slower decay than PET

Applications:

- measure blood flow through arteries and veins
- brain, heart, renal

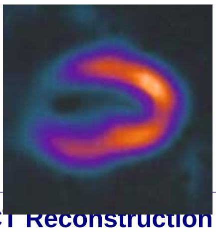
SPECT: Case Studies

Brain: uncontrolled complex partial seizures

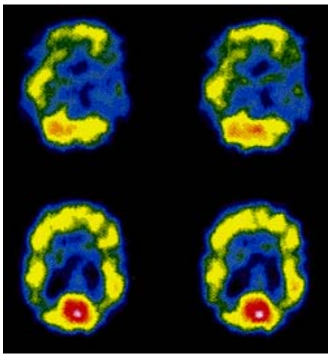
- left temporal lobe has less blood flow than right
- indicates nonfunctioning brain areas causing the seizures

Heart: perfusion of heart muscle

- orange, yellow: good perfusion
- blue, purple: poor perfusion



heart

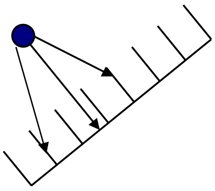


brain metabolism

PET vs. SPECT (1)

SPECT:

- single photon is produced (need collimator on the detector to determine its path)
- low resolution (6-8 mm)
- tracer decay slower
    - therefore longer-lasting effects can be monitored
    - tracers don't have to be produced on site
    - but also takes longer scan times



PET vs. SPECT (2)

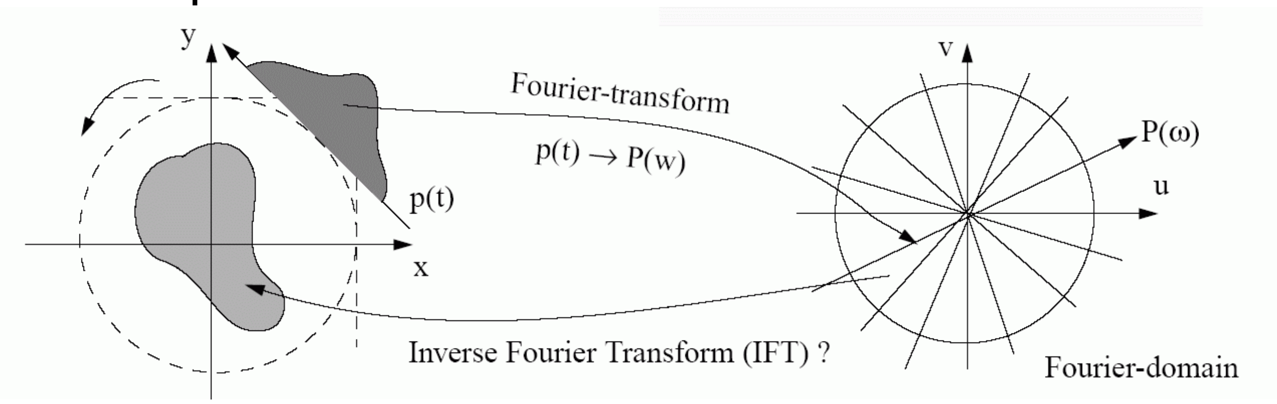
PET:

- no collimators needed -- annihilated positrons yield detectable dual gamma rays 180° apart
- tracers decay fast
  - transient processes can be monitored
  - scan time short (less than a minute)
  - tracers must be produced in nearby cyclotrons
- more expensive equipment (detector hardware)
- higher resolution than SPECT (2-3 mm)
- best for the study of brain receptors with particular neurotransmitters (over fMRI)

Reconstruction: Backprojection

Concept:

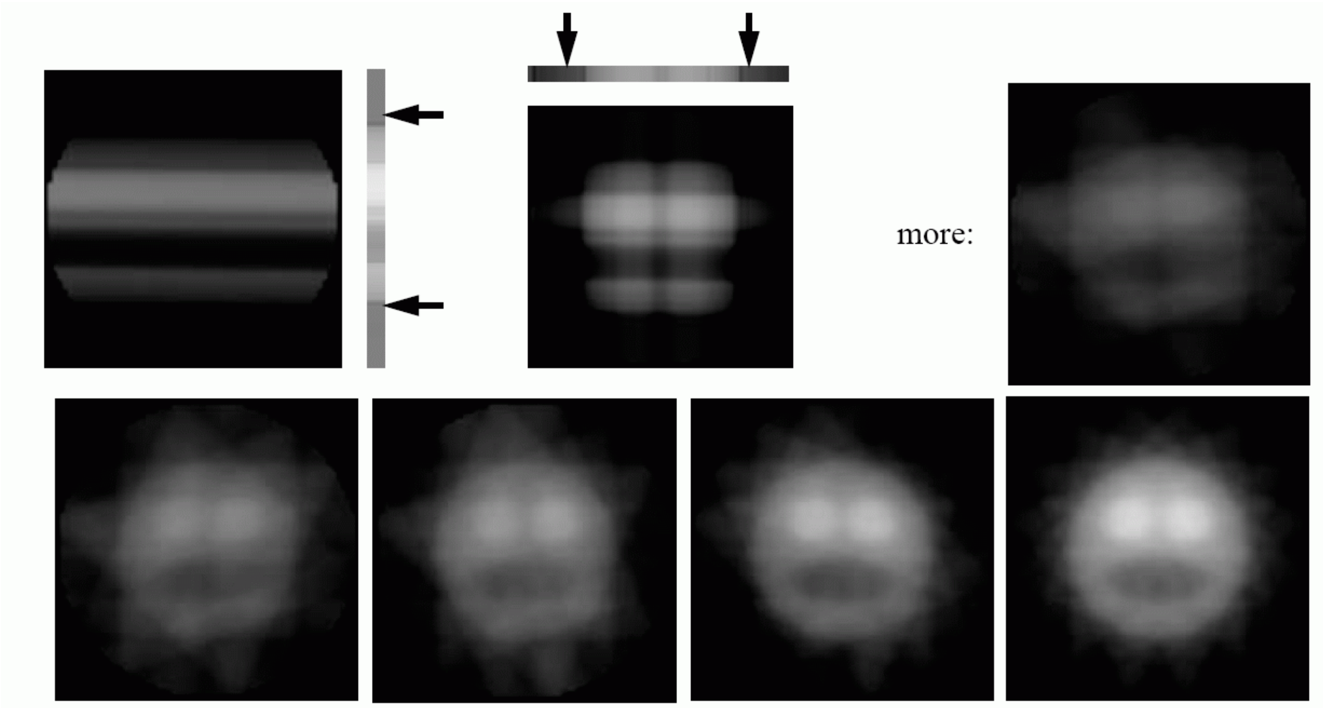
Fourier Slice Theorem



Inverse FT requires interpolation in frequency space: prone to artifacts

Use backprojection in the spatial domain

Reconstruction: Backprojection

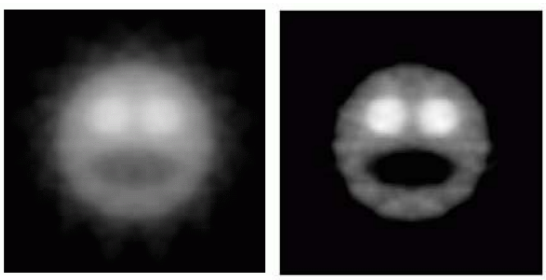


Reconstruction: Filtered Backprojection (FBP)

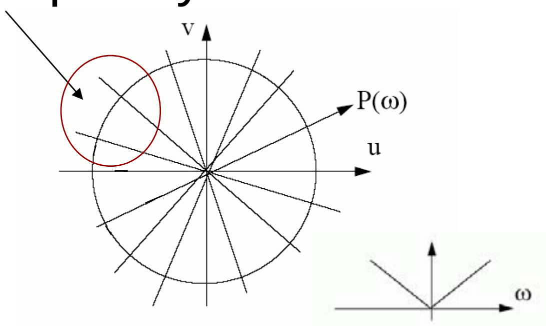
Notice the blurring?

Apply ramp filtering (in frequency space) before backprojection in spatial domain

Intuitive explanation: This makes up for the lack of undersampling in the high frequency bands



not filtered filtered



Reconstruction: Iterative Methods

Iterative methods are advantageous in these cases:

- limited number of projections
- irregularly-spaced and -angled projections
- non-straight ray paths (example: refraction in ultrasound imaging)
- corrective measures during reconstruction (example: metal artifacts)
- presence of statistical (Poisson) noise and scatter (mainly in functional imaging: SPECT, PET)

Simultaneous Algebraic Reconstruction Technique (SART)

Iteratively solves W V=P

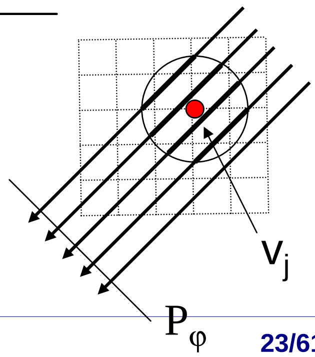
$$v_j^{k+1} = v_j^k + \lambda \frac{\sum_i \frac{p_i - \sum_j v_j^k w_{ij}}{\sum_j w_{ij}} w_{ij}}{\sum_i w_{ij}}$$

Simultaneous Algebraic Reconstruction Technique (SART)

Projection

Projection (into pixel)

$$v_j^{k+1} = v_j^k + \lambda \frac{\sum_i \dfrac{p_i - \sum_l v_l^k w_{il}}{\sum_l w_{il}} w_{ij}}{\sum_i w_{ij}}$$



Simultaneous Algebraic Reconstruction Technique (SART)

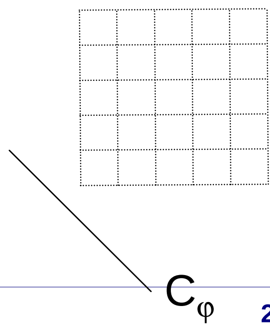
Correction factor computation

Projection (into pixel)

Scanned pixel

Normalization at pixel *i*

$$v_j^{k+1} = v_j^k + \lambda \frac{\sum_i \dfrac{p_i - \sum_l v_l^k w_{il}}{\sum_l w_{il}} w_{ij}}{\sum_i w_{ij}}$$



Simultaneous Algebraic Reconstruction Technique (SART)

Backprojection

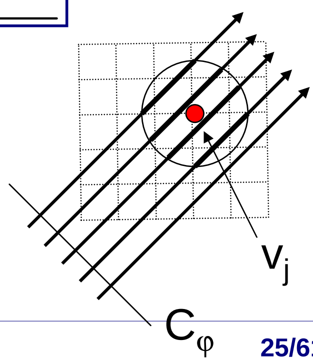
Projection (into pixel)

Backprojection (into voxel)

Scanned pixel

Normalization at pixel *i*

$$v_j^{k+1} = v_j^k + \lambda \frac{\displaystyle\sum_i \frac{p_i - \sum_l v_l^k w_{il}}{\sum_l w_{il}} w_{ij}}{\displaystyle\sum_i w_{ij}}$$



Simultaneous Algebraic Reconstruction Technique (SART)

Voxel normalization

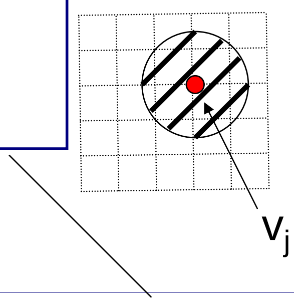
Projection (into pixel)

Backprojection (into voxel)

Scanned pixel

Normalization at pixel *i*

$$v_j^{k+1} = v_j^k + \lambda \frac{\displaystyle\sum_i \frac{p_i - \sum_l v_l^k w_{il}}{\sum_l w_{il}} w_{ij}}{\displaystyle\sum_i w_{ij}}$$



Normalization at voxel *j*

Simultaneous Algebraic Reconstruction Technique (SART)

Voxel update

Projection (into pixel)

Backprojection (into voxel)

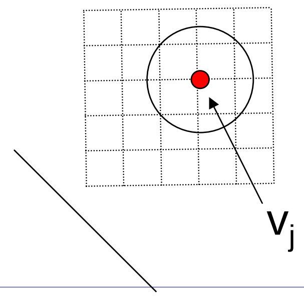
Scanned pixel

Normalization at pixel *i*

$$v_j^{k+1} = v_j^k + \lambda \frac{\sum_i \frac{p_i - \sum_l v_l^k w_{il}}{\sum_l w_{il}} w_{ij}}{\sum_i w_{ij}}$$

New (*k*+1) and previous (*k*) values of voxel *j*

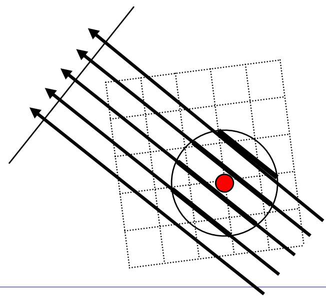
Normalization at voxel *j*



Simultaneous Algebraic Reconstruction Technique (SART)

Next projection

$$v_j^{k+1} = v_j^k + \lambda \frac{\sum_i \frac{p_i - \sum_l v_l^k w_{il}}{\sum_l w_{il}} w_{ij}}{\sum_i w_{ij}}$$



Maximum Likelihood Expectation Maximization (ML-EM)

Maximizes the likelihod of the values of voxels *j*, given values at pixels *i*

$$v_j^{k+1} = \frac{v_j^k}{\sum_i w_{ij}} \sum_i \frac{p_i}{\sum_j v_j^k w_{ij}}$$

Maximum Likelihood Expectation Maximization (ML-EM)

Maximizes the likelihod of the values of voxels *j*, given values at pixels *i*

New (*k*+1) and previous (*k*) values of voxel *j*

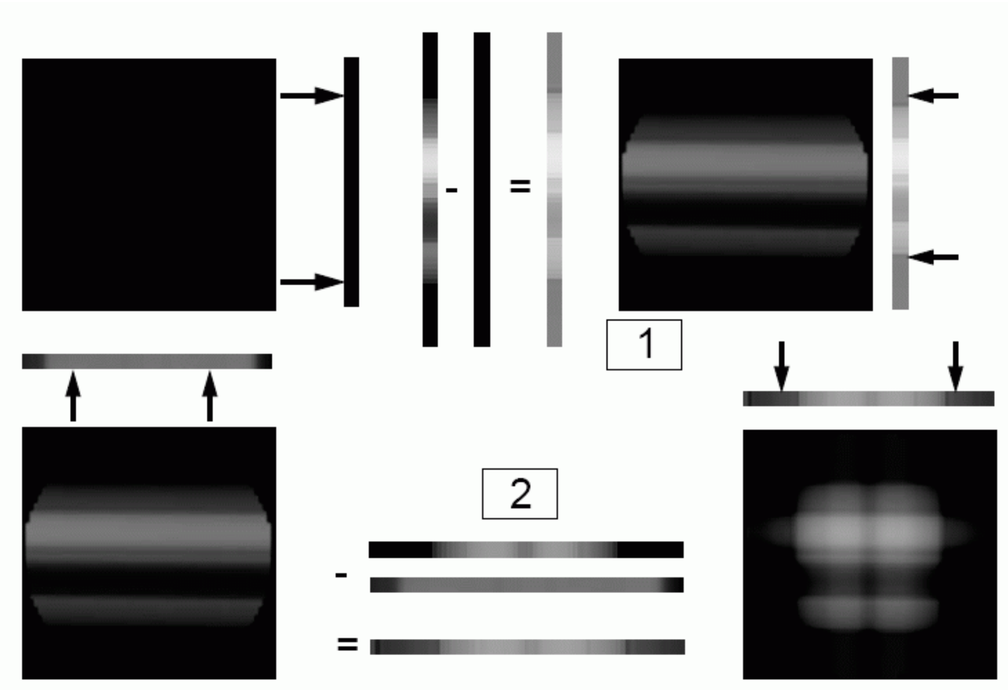
Backprojection (into voxel *j*)

$$v_j^{k+1} = \frac{v_j^k}{\sum_i w_{ij}} \sum_i \frac{p_i}{\sum_j v_j^k w_{ij}}$$

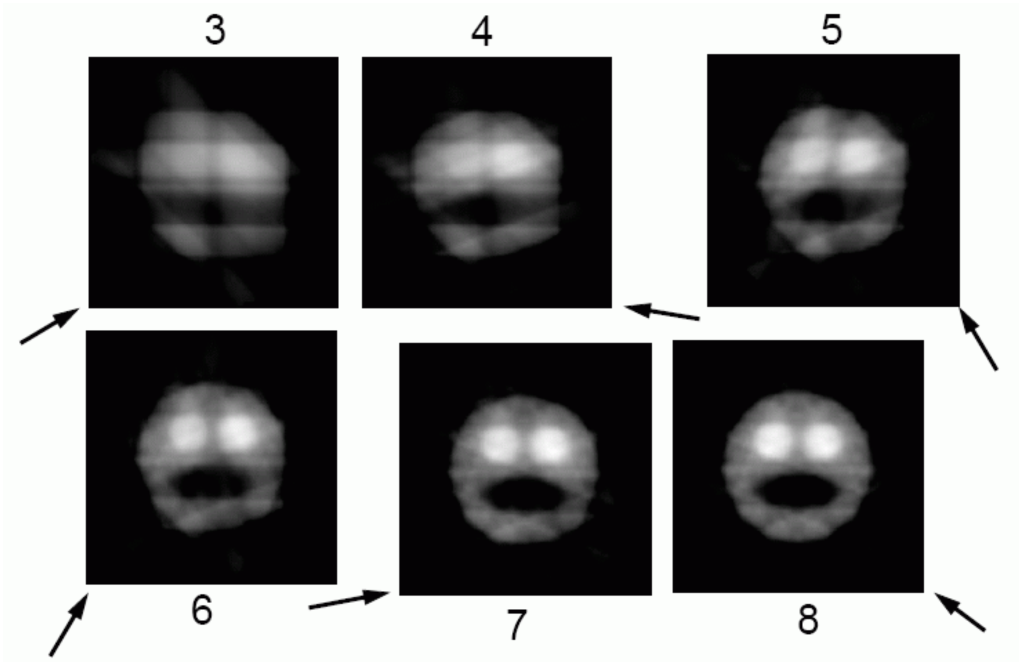
Normalization at voxel *j*

Projection (into pixel *i*)

Iterative Reconstruction Demonstration: SART



Iterative Reconstruction Demonstration: SART



Remarks: Algorithms

SART:

- projection ordering important
- ensure that consecutively selected projections are approximately orthogonal
- random selection works well in practice

EM:

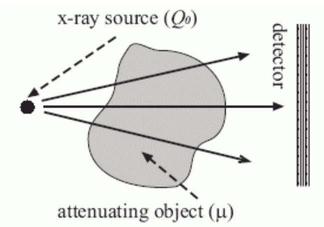
- convergence slow if all projections are applied before voxel update
- use OS-EM (Ordered Subsets EM): only a subset of projections are applied per iteration

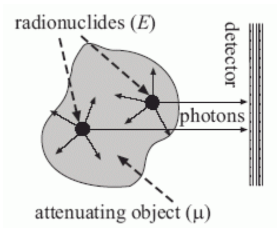
Remarks: Image Generation

There is a strong resemblance to volume rendering

Transmission CT:

- known external source
- unknown density volume
- X-ray rendering





Functional CT:

- unknown emissive volume
- known density volume (prior transmission CT)
- full volume rendering most appropriate

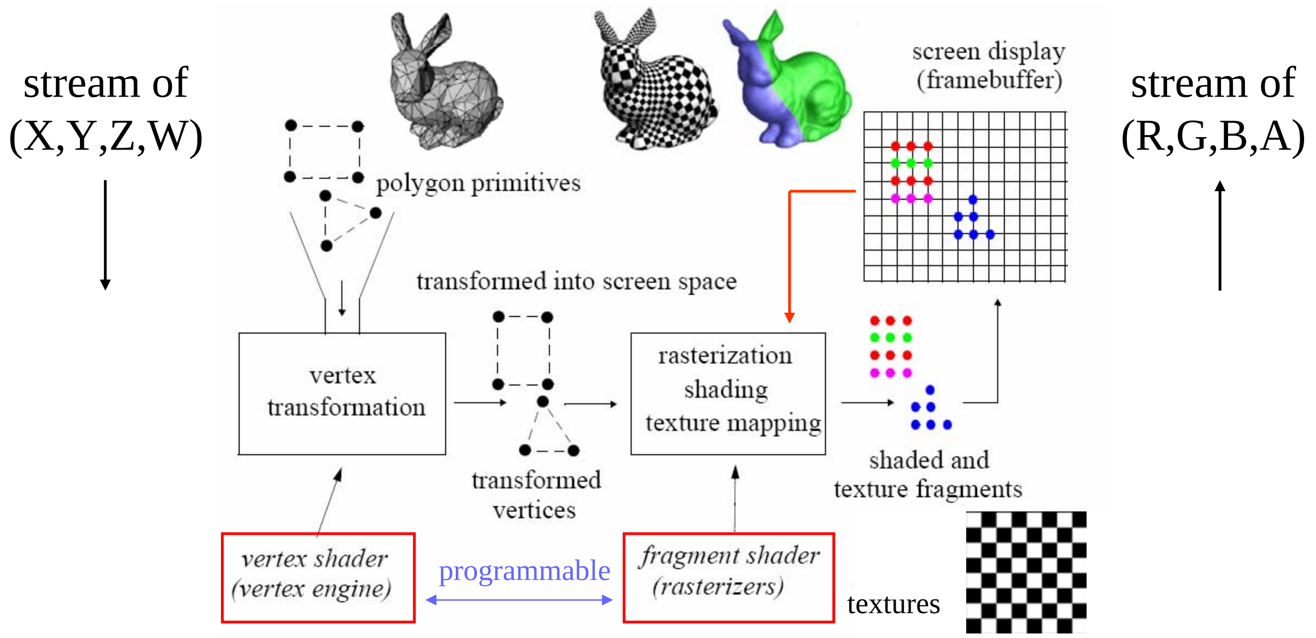
GPU Acceleration: Preparation

The goal is to develop a library of functions with which all algorithms can be implemented

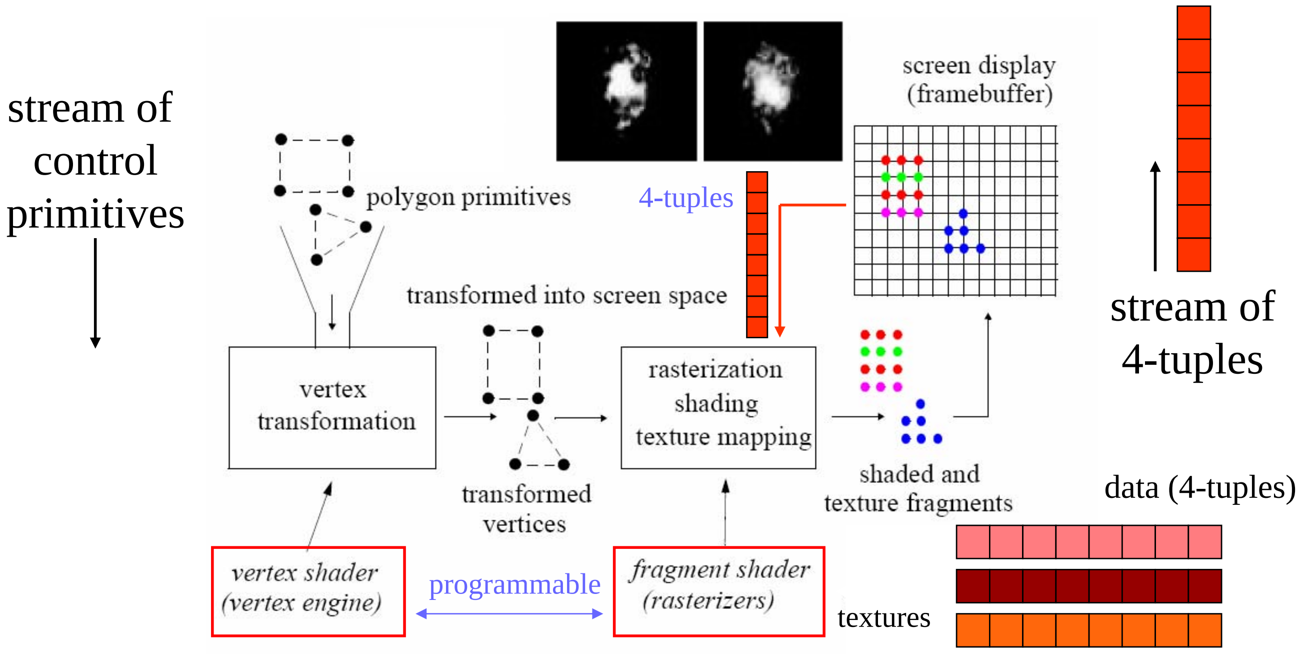
Common operators:

- Projections along straight ray paths
- Back-projections along straight ray paths
- Normalization of pixels or voxels
- Grid correction factor computation per pixel
- Projections and back-projections occur on an image basis
- Parallel or perspective (cone-beam) viewing

GPU: Graphics Point of View



GPU: Computational Point of View



GPU Highlights

Four-fold task parallel (RGBA)

Stream processing -- one instruction decode per data vector

SIMD -- many parallel pipelines per instruction

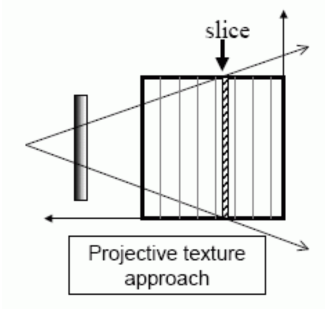
All standard graphics ops are hardwired, some arithmetic (+, -, *), but mostly at 8 bit precision (blending at 16-bit float precision)

GPU fully programmable via fragment programs, at 16 / 32 bit floating point precision

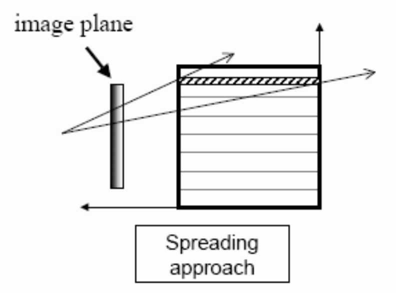
Acceleration Options

Projection and backprojection:

- Projective textures: vertical volume slices
  - easier to incorporate attenuation correction
  - uses full floating point precision



- Section spreading: horizontal volume slices
  - allows for a faster 2-pass byte/float approach



Acceleration Options

Pixel and voxel level operations

- normalizations, correction factors
- can be done via texture streaming using a relatively simple fragment program
- not further discussed here

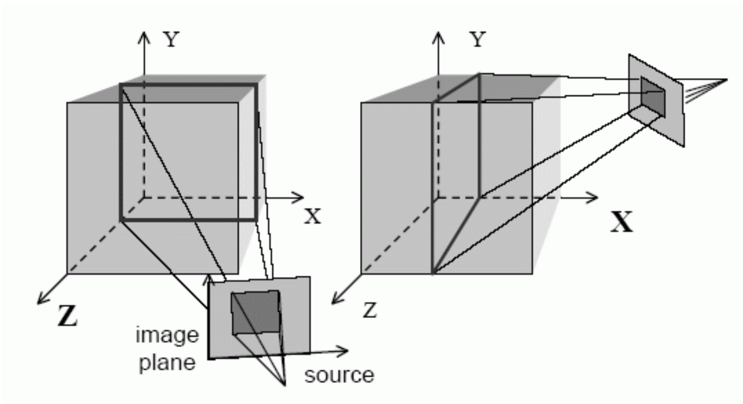
In the following, we shall explain the algorithm via the backprojection operations

For projections, simply switch roles of sender and receiver

Backprojection with Projective Textures: Concept

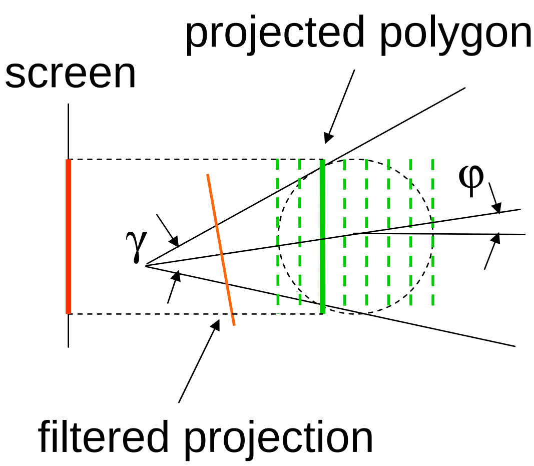
Requires two texture stacks

- one for x-major and one for z-major projections
- requires small overhead for stack swapping when next projection uses a different major axis



Backprojection with Projective Textures: Algorithm

Init texture matrix
$TM = T_1 R P S T_2$



For each volume slice

Pass 1:

associate 3D texture coordinate with each poly vertex

render tex-mapped poly to pbuffer (use TM to map texture coordinates onto projective texture)

Pass 2:

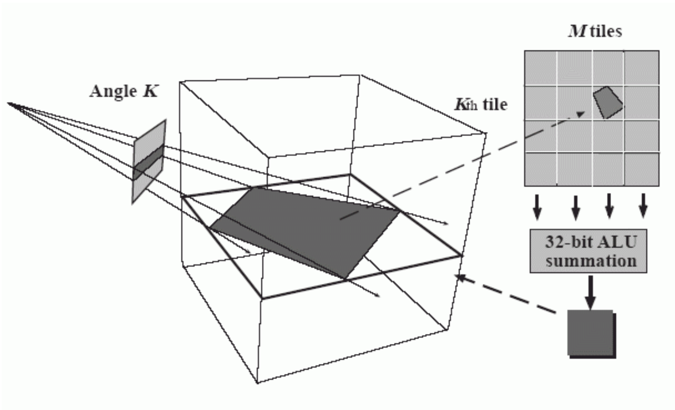
volume slice texture += pbuffer

Backprojection with Section Spreading: Concept

Allocate a 2D float texture holding *M* tiles of size N^2. This tile will hold the backprojections for *M* angles via (speedy) hardwired byte arithmetic

Pass 1 renders image sections into the tiles

Pass 2 accumulates these tiles in floating point (using hardwired floating point adders)



Backprojection with Line Spreading: Algorithm

For each volume slice

Pass 1: for each of the *M* available projection (byte) images

determine the projection shadow of the image on the slice

associate the texture coordinates of the shadowing image section with the shadow polygon on the slice

render the texture mapped slice into the M^{th} tile

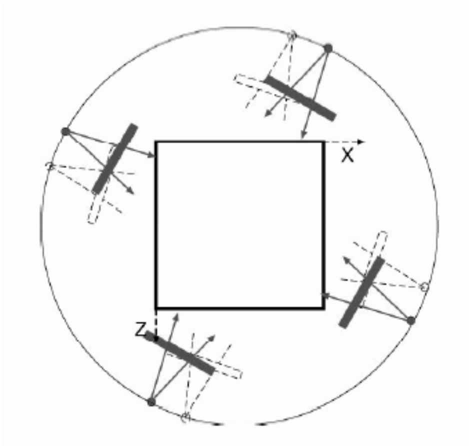
Pass 2: accumulate the *M* tiles to form the sum of all backprojections

Note: this algorithm is only meaningful for OS-EM and FBP

Enhancements: Speedup

4-fold speedup: use RGBA channels

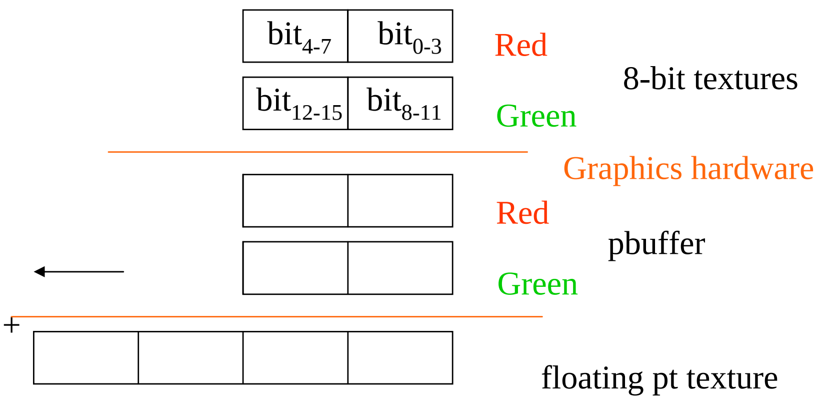
- accumulate orthogonal channels in parallel
- perform appropriate rotations in accumulation stage



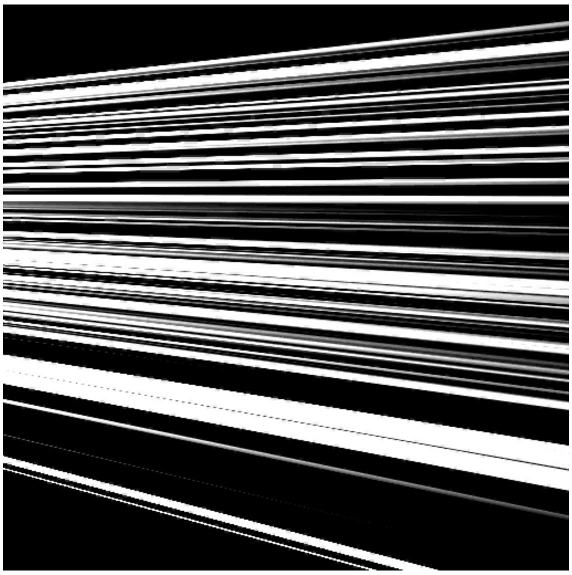
Enhancements: Accuracy

Extend precision to (pseudo-)16-bit

- Break 16-bit data up into lower / higher 8-bits
- Render into separate tiles
- Accumulate with proper shifts



Reconstruction Demo



Enhancements: Attenuation Modeling

Projection:

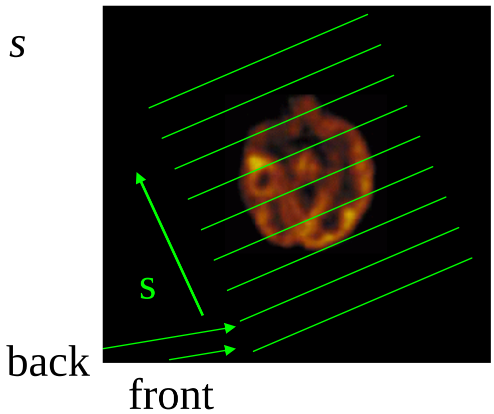
$E(s)$ emission at s

$\mu(s)$ density at s

$$P = \int_{s=0}^{L} E(s) e^{-\int_{t=0}^{s} \mu(t)dt} ds$$

$$\approx \sum_{i=0}^{L/\Delta s} E(i\Delta s) e^{-\sum_{j=0}^{i-1} \mu(j\Delta t)\Delta t} \Delta s$$

$$\approx \sum_{i=0}^{L/\Delta s} E(i) \prod_{j=0}^{i-1} (1-\mu(j))$$



with $(1-\mu(j\Delta t)) = t_{front/back}$:

$$E_{front} = E_{back} t_{front} + E_{front}$$

$$t_{front} = t_{front} t_{back}$$

Enhancements: Attenuation Modeling

This is similar to the typical volume rendering compositing equation for front-to-back traversal

We can therefore composite the interpolated slices in hardware, mapping the densities in the CT volume to transparencies t

Backprojection requires two buffers in tandem mode:

- keep track of current t_{front} and current E_{front}
- after backprojecting E_{front} update t_{front} for next slice

Or, first compute all E_{front} and then backproject all

Results

Timings have been obtained on an Nvidia FX 5950 with 256MB memory

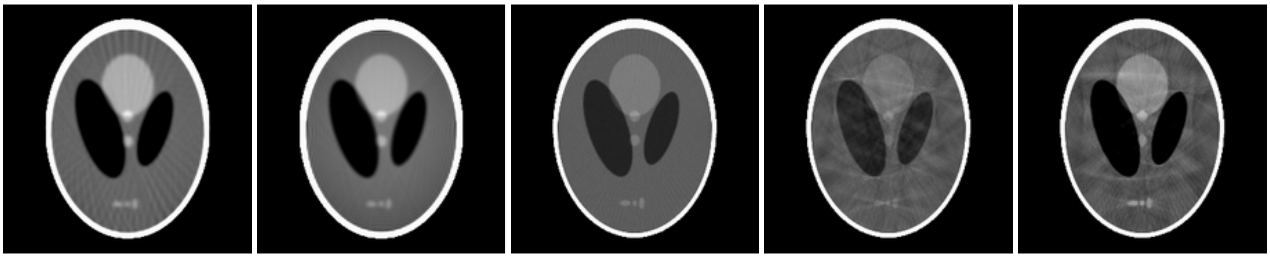
Results: Filtered Backprojection

80 projections ← 160 projections →

← 0.5% contrast → 1% contrast

← floating point → ← 16-bit → ← 8-bit →

← 128^3 volume → ← 256^3 volume →



0.3 s | 0.6 s | 1.2 s | 0.6 s | 0.6 s

5 min for CPU full floating point (speedup = 250) ↑ (1.2 s)

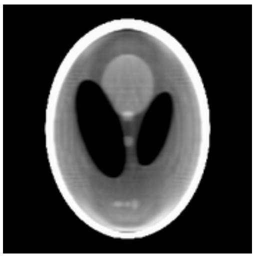
0.1 s for 128^3 ↑ (0.6 s)

Filtered Backprojection: Larger Data

Volume	Projections	Precision	timings
256^3	256	32-bit	2.0s
256^3	256	8-bit x2	1.9s
512^3	512	32-bit	32s
512^3	512	8-bit x2	30s

Note: with current drivers the pseudo-16 bit method is not much faster than the 32-bit (floating point) solution

Results: SART

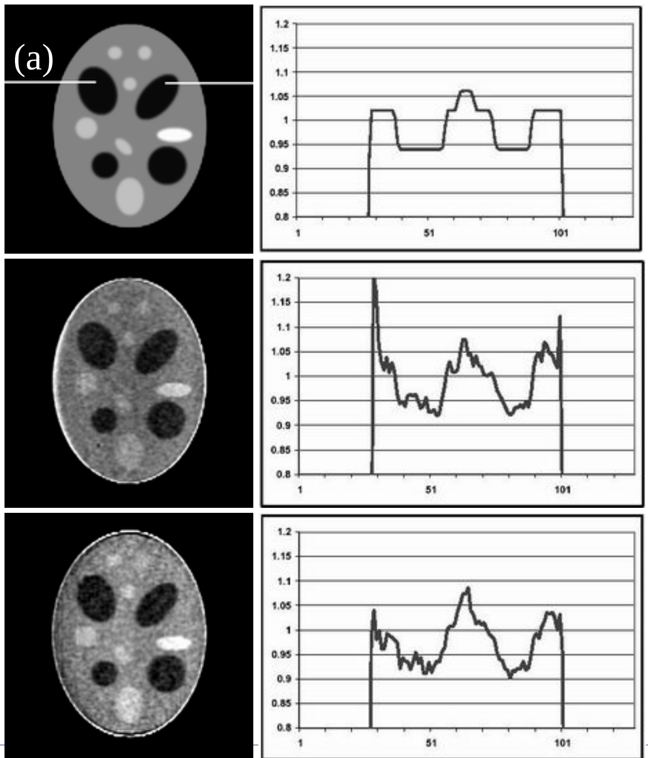


80 projections
2 s
CPU: 7.5 min (speedup: 225)
SGI (1998): 3.1 min

160 projections
4 s
CPU: 15 min (speedup: 225)

3 iterations, 128^3 grid, full floating point precision

Results: OS-EM



floating point precision
128^3 grid
80 projections
3 iterations

Original

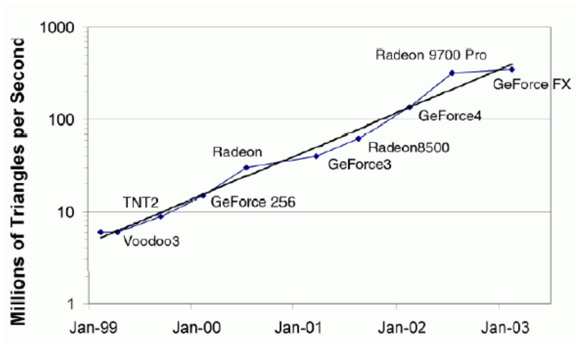
with Poisson noise added
GPU-reconstructed in 63 s

profiles are densities
across the line (a)

reconstructed on the CPU

Immense Performance Growth

GPUs grow at triple of Moore's law

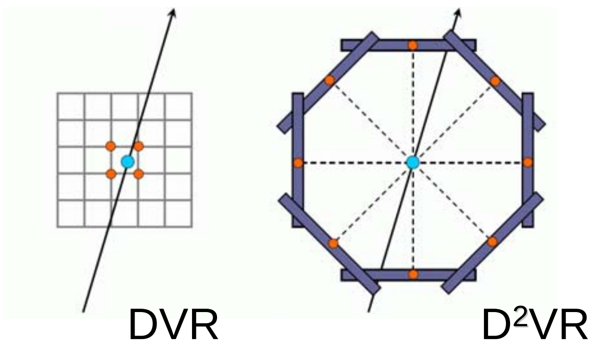


Current reconstruction speeds with NVidia 7800 FX run 4 times faster than with NVidia 5950

- dual/quad-board SLI gives a further factor 2/4
- further fragment code optimizations can give additional significant speedups

D^2VR: Visualization from Projections

GPU-acceleration of CT also enables near-interactive volume (1-2 fps) rendering directly from the projection data



128 x 128^2 projections

256^2 viewport

Conclusions (1)

GPUs show excellent capabilities for rapid tomographic reconstruction at low cost (allowing fast tomography even for niche markets)

Programmability allows new capabilities to be added easily (also for prototyping in the lab)

Immediate accelerated visualization (3D rendering, slicing) of reconstruction results is straightforward

Speed enables applications in patient positioning and procedure setup, interactive radiotherapy, and others

Conclusions (2)

Sustained growth rate at triple of Moore's law leaves much hope for future performance ratios

Emergence of SLI-mediated quad-GPU setups add to this hope

GPU clusters with Sepia bus (or other) make affordable CT supercomputers

Software soon to be available at www.rapidCT.com

References (1)

More information is available here:

F. Xu and K. Mueller, "Accelerating popular tomographic reconstruction algorithms on commodity PC graphics hardware," *IEEE Transactions on Nuclear Science*, vol. 52, no. 3, pp. 654-663, 2005.

F. Xu and K. Mueller, "RapidCT: Acceleration of 3D computed tomography on the GPU," *ACM Workshop on General-Purpose Computing on Graphics Processors 2004 (GPGPU '04)*, Los Angeles, August 2004.

F. Xu and K. Mueller, "Ultra-fast filtered backprojection on commodity graphics hardware", *2004 IEEE International Symposium on Biomedical Imaging,* Arlington, VA, April 2004.

F. Xu and K. Mueller, "Towards a Unified Framework for Rapid Computed Tomography on Commodity GPUs", *IEEE Medical Imaging Conference (MIC) 2003*, Portland, OR, October 2003.

K. Mueller and R. Yagel, "Rapid 3D cone-beam reconstruction with the Algebraic Reconstruction Technique (ART) by using texture mapping hardware," vol. 19, no. 12, pp. 1227-1237, *IEEE Transactions on Medical Imaging*, 2000.

References (2)

F. Xu and K. Mueller, "A comparative study of popular interpolation and integration methods for use in computed tomography," *IEEE 2006 International Symposium on Biomedical Imaging (ISBI '06*), Arlington, VA, April 6-9, 2006.

K. Mueller and F. Xu, "Practical considerations for GPU-accelerated CT, " *IEEE 2006 International Symposium on Biomedical Imaging (ISBI '06),* Arlington, VA, April 6-9, 2006.

P. Rautek, B. Csébfalvi, S. Grimm, S. Bruckner, E. Groeller, "D^2VR: High-quality volume rendering of projection-based volumetric data," *Joint Eurographics-IEEE TCVG Symposium on Visualization (EuroVis),* pp. 211-218, 2006.

F. Xu and K. Mueller, "GPU-Accelerated D^2VR," Volume Graphics 2006, pp. 23-30, Boston, MA, August 2006.

Most papers are available at http://www.cs.sunysb.edu/~mueller/research/ct.html

Acknowledgments

This work was partially funded by:

NSF, NIH

Keck Foundation, UCSF

Breakaway Imaging, LLC