MIC-GPU: High-Performance Computing for Medical Imaging on Programmable Graphics Hardware (GPUs)

SPIE Medical Imaging

CT Reconstruction Pipeline Components

Klaus Mueller

Computer Science
Center for Visual Computing
Stony Brook University

Course Schedule

1:30 – 2:00:	Introduction
2:00 – 2:30:	GPU architecture, programming model, and programming facilties
2:30 – 3:00:	GPU programming examples (image processing)
	Coffee Break
3:30 – 4:00:	CT reconstruction pipeline components
4:00 – 4:30:	GPU-acceleration of individual components
4:30 – 5:00:	Various CT reconstruction pipelines, load balancing and load estimation
5:00 – 5:30:	Reconstruction visualization and final remarks

CT Reconstruction Pipeline

A CT reconstruction pipeline is typically composed of a number of serial components

Example 1: Filtered Backprojection

- projection filtering
- backprojection
- post-weighting

Example 2: Iterative 3D reconstruction in blocks

- backprojection of volume into set's views
- correction factor computation
- backprojection of correction factors
- post-weighting (normalization)

Kernel-Centric Decomposition

We can consider each of these steps to be a SIMD kernel, as follows:

Example 1: Filtered Backprojection

- projection filtering → *filtering kernel*
- backprojection → *backprojection kernel*
- post-weighting → *post-weighting kernel*

Example 2: Iterative 3D reconstruction in blocks

- backprojection of volume into set's views → *projection kernel*
- correction factor computation → *correction factor kernel*
- backprojection of correction factors → *backprojection kernel*
- normalization → *normalization kernel*

Kernel-Centric Decomposition

SPIE Medical Imaging

We can consider each of these steps to be a SIMD kernel, as follows:

Example 1: Filtered Backprojection

- projection filtering → *filtering kernel*
- backprojection → *backprojection kernel*
- post-weighting → *post-weighting kernel*

Example 2: Iterative 3D reconstruction in blocks

- backprojection of volume into set's views → *projection kernel*
- correction factor computation → *correction factor kernel*
- backprojection of correction factors → *backprojection kernel*
- normalization → *normalization kernel*

vector operations   projector with interpolation

Kernel Scheduling

SPIE Medical Imaging

SIMD can only execute one kernel at a time

- this prohibits kernel overlap, even if mathematically correct
- we may merge kernels if targets are identical → this favors load balancing and the reduction of passes
- but recall that scattering to multiple targets is undesirable

Therefore a decomposition of a reconstruction pipeline into components is advisable

- develop an optimized kernel for each component
- overlap (=hide) the loading of data (if needed) with execution of a prior kernel (or within kernel)
- also optimize what platform to run the computations (CPU, GPU), but then consider transfer of data

Popular CT Reconstruction Pipelines

SPIE Medical Imaging

We will discuss:

- analytical schemes (Feldkamp)
- various algebraic schemes (SART, SIRT)
- statistical schemes (EM, OS-EM)
- in terms of anatomical and metabolic (functional) CT
- for various beam geometries: parallel, fan, cone

The projection/backprojection is typically the most expensive operation

- it is part of every algorithm and application
- with variations in
  - beam geometry
  - modeling of tissue (attenuation, scattering) and detector effect
  - each is implemented with a dedicated kernel
  - each such kernel is loaded into the GPU on demand

Terminology

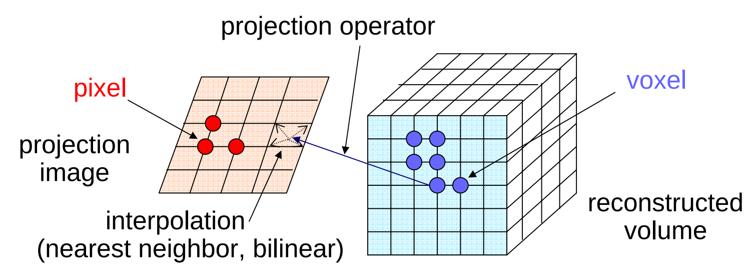
SPIE Medical Imaging

We shall discuss all material in terms of 3D reconstruction

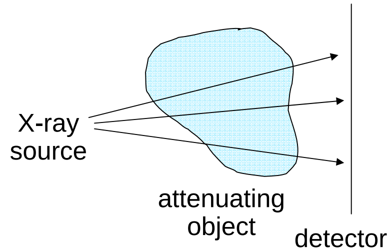
- the reduction to 2D slice reconstruction is straightforward

Pixels: the basis elements (point samples) of the projection image (the photon measurements)

Voxels: the basis elements (point samples) of the reconstruction volume (the attenuation densities or the tracer photon emissions)

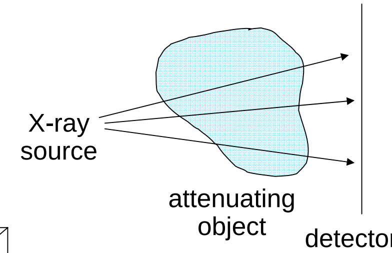


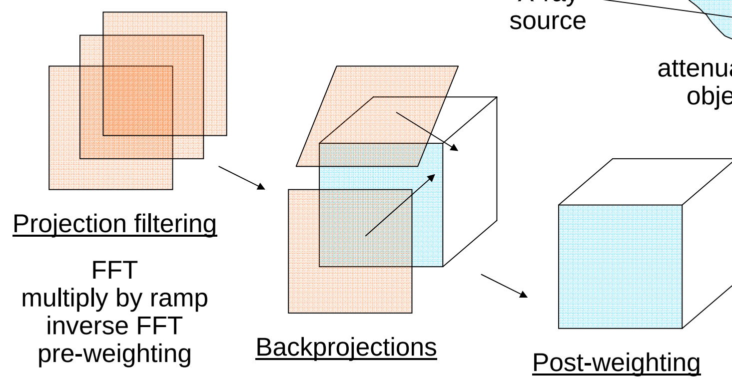
Transmission CT



Transmission CT

A typical reconstruction algorithm
is Filtered Backprojection

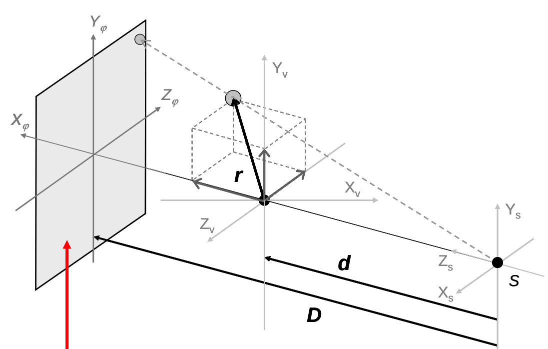




Example

Feldkamp-Davis-Kress (FDK) Cone-beam reconstruction

FDK: Filtering



filtered
projection data

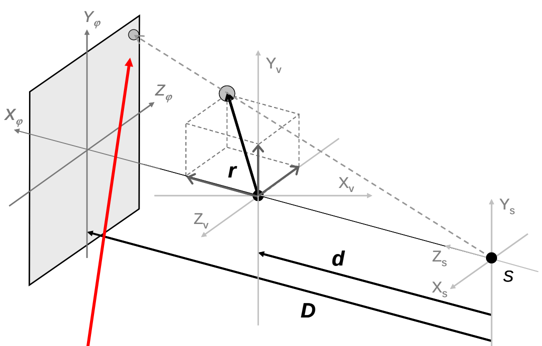
$$\hat{P}_\phi(Y,Z) = \frac{D}{\sqrt{D^2+Y^2+Z^2}} P_\phi(Y,Z) ** g(Y)$$

circular pre-weighting

projection data

ramp filter

FDK: Backprojection

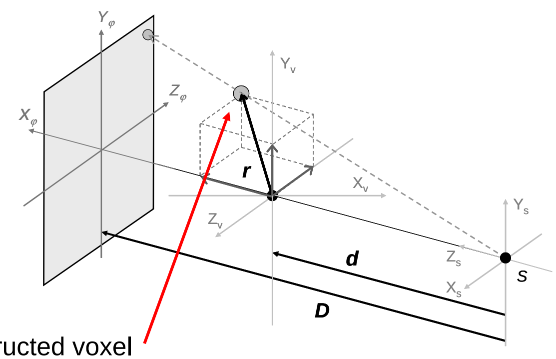


$$\hat{P}_\phi(\mathbf{r}) = \hat{P}_\phi(Y(\mathbf{r}), Z(\mathbf{r})), \quad Y(\mathbf{r}) = \frac{\mathbf{r}\cdot y_\phi}{d + \mathbf{r}\cdot x_\phi} D, \quad Z(\mathbf{r}) = \frac{\mathbf{r}\cdot z_\phi}{d + \mathbf{r}\cdot x_\phi} D$$

voxel→projection mapping

projection coordinates of mapped voxel

FDK: Accumulation, Depth-Weighting



reconstructed voxel

$$f(\mathbf{r}) = \frac{1}{4\pi^2} \int_0^{2\pi} \frac{d^2}{(d + \mathbf{r}\cdot x_\phi)^2} \hat{P}_\phi(\mathbf{r}) d\phi$$

accumulation for all projections

depth-weighting

Other Analytical Algorithms

Similar concepts apply for other analytical CT algorithms

- modified FDK, multi-orbit cone-beam CT
- helical CT with exact and non-exact algorithms

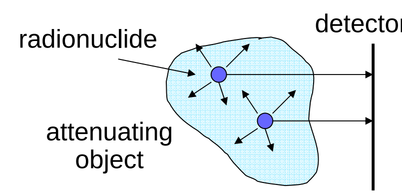
Always a sequence of serial steps

- projection filtering, possibly rebinning
- backprojection
- accumulation and weighting

Only backprojection (and rebinning) requires interpolation

- can make use of graphics texture-mapping facilities
- but can also be implemented as SIMD (fragment) programs (GPGPU)
- the remaining operations are straight vector arithmetic (GPGPU)

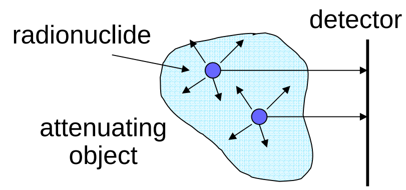
Emission CT

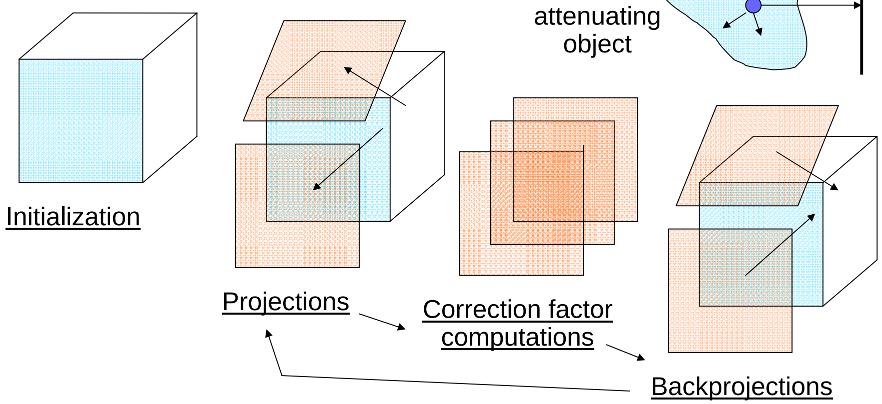


Emission CT

SPIE Medical Imaging

Typically reconstructions are done in an iterative manner





Example: EM (OS-EM)

SPIE Medical Imaging

EM: volume correction after each projection

- tends to converge slowly

OS-EM: volume correction after a set of projections

- converges faster

Example: EM (OS-EM)

SPIE Medical Imaging

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*)

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

Correction value

weight factor

Normalization at voxel *j*

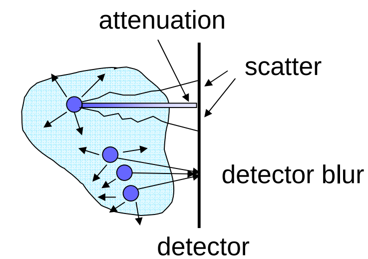
Projection (into pixel *i*)

The Weight Factor

SPIE Medical Imaging

The weight factor w_{ij} can model various effects:

- interpolation filter factors (nearest neighbor, bilinear, Gaussian, Bessel, etc)
- detector geometric response (blurring due to off-angle photon contributions)
- photon attenuation (requires an attenuation map **μ** obtained via transmission CT)
- photon scattering (requires a gradient map, typically the transmission CT reconstruction)



Attenuation Modeling: Theory

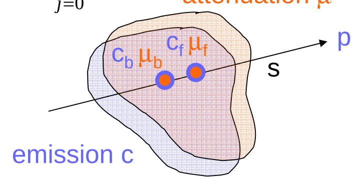
Forward projection (front-to-back):

- the energy arriving at a detector pixel is: $p = \int_{s=0}^{l} c(s)\, e^{-\int_{t=0}^{s} \mu(t)dt}\, ds$
- in discrete terms:

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

- using a Taylor series approximation:

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



note: all values are normalized to [0,1]

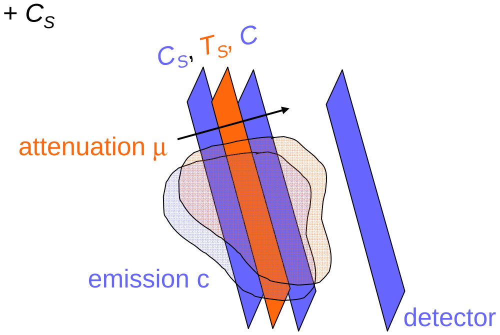
- as a recursive equation: $c_f = c_f + c_b t_f \quad t_f = t_f(1-\mu_b) = t_f t_b$
- equivalent back-to-front compositing:

$$c_b = c_b(1-\mu_f) + c_f = c_b t_f + c_f$$

Attenuation Modeling: Practice

Forward projection (back-to-front traversal):

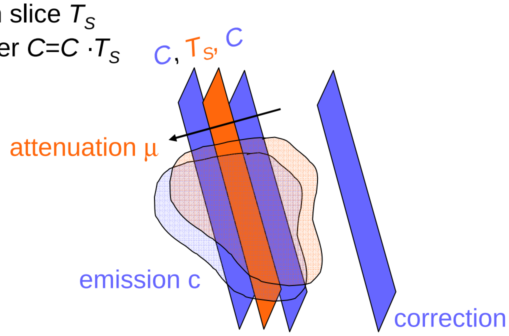
- emission buffer C=0
- step from back to front, at each step:
  - interpolate emission slice C_S and attenuation slice T_S
  - composite $C = C \cdot T_S + C_S$



Attenuation Modeling: Practice

Backprojection (front-to-back traversal):

- initialize correction buffer C
- step from front to back, at each step:
  - spread (and add) C into emission volume affected by slice
  - interpolate attenuation slice T_S
  - update correction buffer $C = C \cdot T_S$



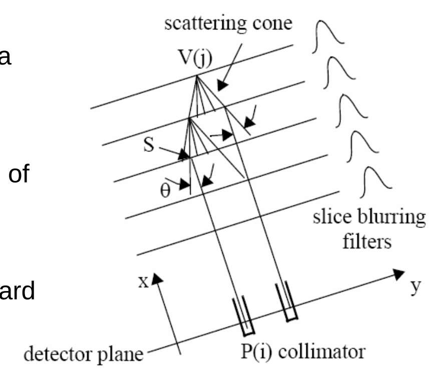
Scatter Modeling: Theory

Idea:

- scattering can be modeled by a phase function resembling a Gaussian
- the anatomical density map determines the parameters (σ) of this Gaussian

Approach:

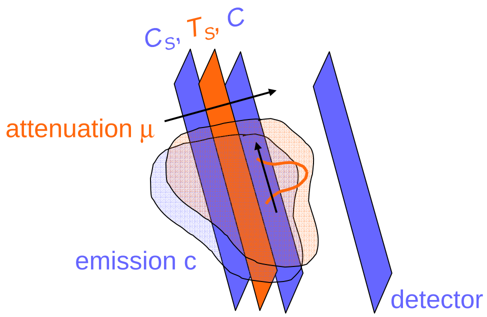
- scattering of emissions in forward projection is a back-to-front diffusion process (see figure)
- scattering of backprojected correction factors is a front-to-back diffusion process



Scatter Modeling: Practice

Forward projection (back-to-front traversal):

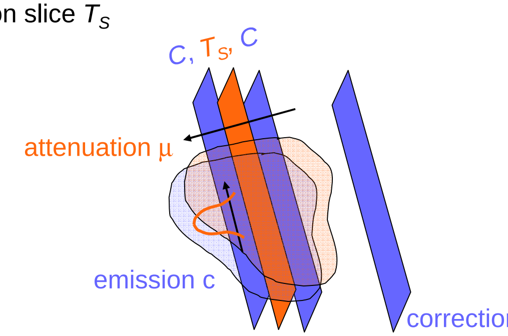
- emission buffer $C = 0$
- step from back to front, at each step:
  - interpolate emission slice C_S and attenuation slice T_S
  - blur C using T_S
  - $C = C + C_S$



Scatter Modeling: Practice

Backprojection (front-to-back traversal):

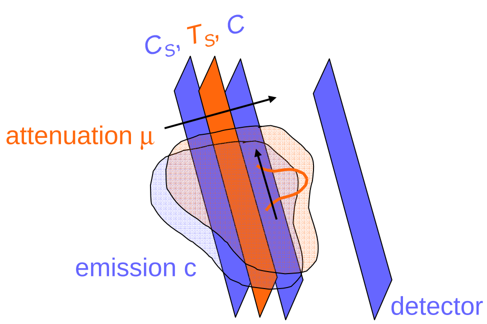
- initialize correction buffer C
- step from front to back, at each step:
  - spread (and add) C into emission volume
  - interpolate attenuation slice T_S
  - blur C using T_S



Combining Both Effects

Forward projection (back-to-front traversal):

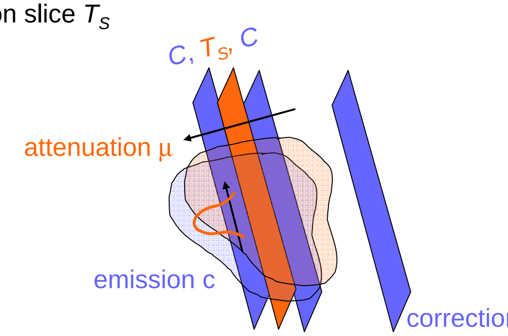
- emission buffer C=0
- step from back to front, at each step:
  - interpolate emission slice C_S and attenuation slice T_S
  - blur C using T_S
  - $C = C \cdot T_S + C_S$



Combining Both Effects

Backprojection (front-to-back traversal):

- initialize correction buffer C
- step from front to back, at each step:
  - spread (and add) C into the emission volume
  - interpolate attenuation slice T_S
  - blur C using T_S
  - update $C = C \cdot T_S$



Example: SART/SIRT

SPIE Medical Imaging

Projection (into pixel *i*)

Backprojection (into voxel *j*)

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*

Correction value

Normalization at voxel *j*

Course Schedule

SPIE Medical Imaging

1:30 – 2:00:	Introduction
2:00 – 2:30:	GPU architecture, programming model, and programming facilties
2:30 – 3:00:	GPU programming examples (image processing)
	Coffee Break
3:30 – 4:00:	CT reconstruction pipeline components
4:00 – 4:30:	GPU-acceleration of individual components
4:30 – 5:00:	Various CT reconstruction pipelines, load balancing and load estimation
5:00 – 5:30:	Reconstruction visualization and final remarks