Models for biomedical image reconstruction based on Newton-Cotes formulae for integral approximations

Thesis Defense

by: Daniel Heilper

Supervisor: Prof. Dan Gordon

Haifa University, Computer Science

Outline

Introduction

Problem Definition

Approximation of line integral of function using sampled data

Framework

Tomographic image reconstructions using projection data

CT, PET, Electron tomography, etc..

Introduction

Problem Definition

Approximation of line integral of function using sampled data

Framework

Tomographic image reconstructions using projection data

CT, PET, Electron tomography, etc..

Introduction

Problem Definition

Approximation of line integral of function using sampled data

Framework

Tomographic image reconstructions using projection data

CT, PET, Electron tomography, etc..

The Reconstruction Problem

The intensity after traversal of Ray_i is
$I_i = \sum_j |Ray_i \cap p_j| \cdot \rho_j$

- p_j - j'th pixel
- ρ_j - value of the attenuation function in p_j

This is a system of linear equations $\mathbf{Ax} = \mathbf{b}$
where $x_j = \rho_j$, $b_i = I_i$ and
$a_j^i = |Ray_i \cap p_j|$

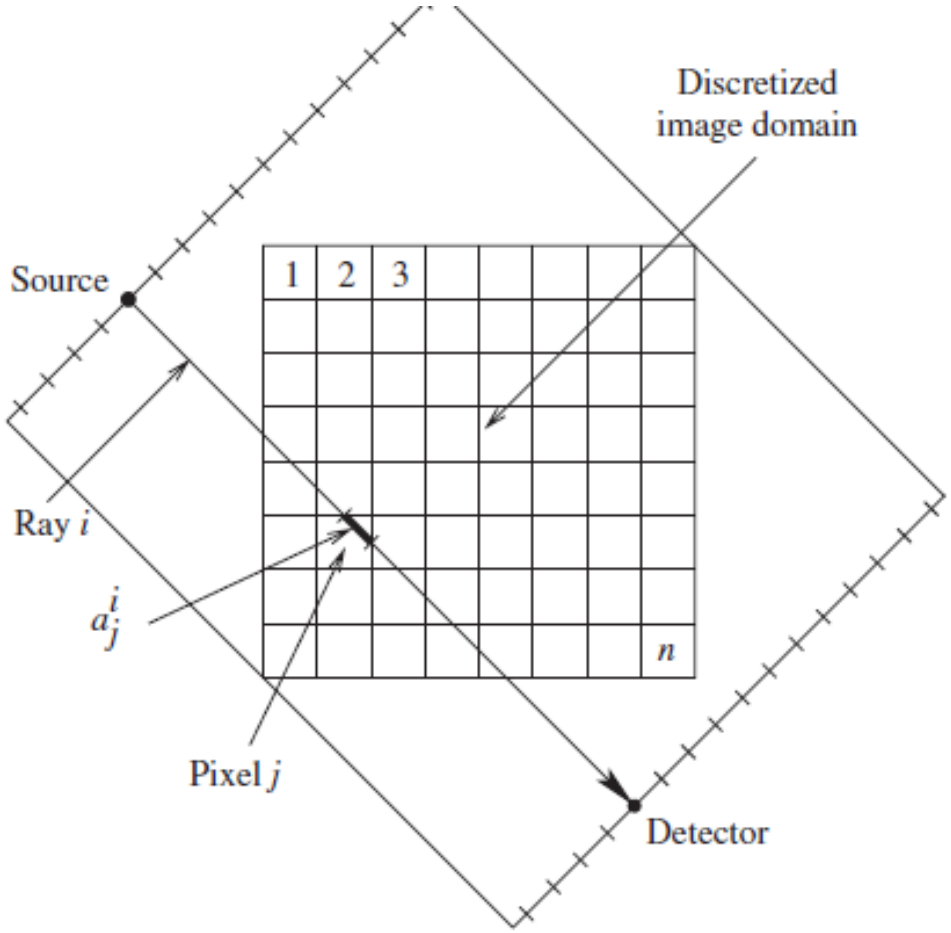


Image basis functions for image reconstruction

Goal

- Find some approximating function $\mathbf{g}(\mathbf{x}, \mathbf{y})$ such that
  - $\mathbf{g}(\mathbf{x}, \mathbf{y})$ is represented as a linear combination of certain basis functions
  - The line integrals are a "good" approximation to the measurements

Image basis functions for image reconstruction

Goal

- Find some approximating function $\mathbf{g}(\mathbf{x}, \mathbf{y})$ such that
  - $\mathbf{g}(\mathbf{x}, \mathbf{y})$ is represented as a linear combination of certain basis functions
  - The line integrals are a "good" approximation to the measurements

Image basis functions for image reconstruction

Goal

- Find some approximating function $\mathbf{g}(\mathbf{x}, \mathbf{y})$ such that
  - $\mathbf{g}(\mathbf{x}, \mathbf{y})$ is represented as a linear combination of certain basis functions
  - The line integrals are a "good" approximation to the measurements

Most used Basis functions

PBF Pixel basis function

Blob Radially symmetric modifications of the Kaiser-Bessel window functions

	PBF	**Blob**
Computation	cheap	expensive
Model's complexity	simple	complex
Usage	trivial	complex tuning
Geometry	voxels	hexagons
Side effects	salt&pepper	smoothed images
Noise situation	very sensitive	reasonable
Contrast	low	high

1D approximations for quadratures of sampled functions

Target

Approximate $\int_{\mathbf{a}}^{\mathbf{b}} \mathbf{f}(\mathbf{x})\mathbf{dx}$ using blending functions $\mathbf{g_i}(\mathbf{x})$, and a set of sampling points

$$f(x) \approx g(x) = \sum_{i=1}^{n-1} \alpha_i g_i(x)$$

1D approximations for quadratures of sampled functions

If $a = a_0 < a_1 < \ldots < a_n = b$ is a partition of the interval $[a, b]$ then

$$\int_a^b g(x)dx = \sum_{i=1}^{n-1} \alpha_i \int_{a_i}^{a_{i+1}} g_i(x)\, dx = \sum_{i=1}^{n-1} \alpha_i (a_{i+1} - a_i) \int_0^1 g_i \left(\frac{x - a_i}{a_{i+1} - a_i} \right)$$

so we may consider functions

$$\ell(x) = g_i \left(\frac{x - a_i}{a_{i+1} - a_i} \right)$$

defined for x on interval $[0, 1]$

1D approximations and the corresponding models

Rectangle rule: Sample value taken at $x = \frac{1}{2}$

$$\boldsymbol{\ell(x)} = s_1(x) = \begin{cases} 1 & \text{for } 0 \leq x \leq 1, \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{g(x)} = \sum_{i=1}^{n-1} f(a_i + \frac{h_i}{2})\, s_1(\frac{x - a_i}{h_i})$$

Trapezoidal rule: Sample values taken at $x = 0$ and $x = 1$

$$\boldsymbol{\ell_0}(x) = 1 - x \text{ and } \boldsymbol{\ell_1}(x) = x$$

$$\mathbf{g(x)} = \sum_{i=1}^{n-1} \left[f(a_i)\boldsymbol{\ell_0}\left(\frac{\mathbf{x - a_i}}{\mathbf{h}}\right) + f(a_{i+1})\boldsymbol{\ell_1}\left(\frac{\mathbf{x - a_i}}{\mathbf{h}}\right) \right]$$

quadratic rule (Simpson) : Samples taken at $x = 0, 1, 2$

Newton-Cotes

The blending functions are chosen as interpolation polynomials for the points $0, 1$, etc

- $n = 2$
  - $x_1 = 0,\ x_2 = 1$
  - $\ell_0(x) = x, \quad \ell_1(x) = 1 - x, \quad \mathbf{0 \leq x \leq 1}$
- $n = 3$
  - $x_1 = 0,\ x_2 = 1,\ x_3 = 2$
  - $q_0(x) = \frac{1}{2}x^2 - \frac{3}{2}x + 1, \quad q_1(x) = -x^2 + 2x,$
    $q_2(x) = \frac{1}{2}x^2 - \frac{1}{2}x, \quad \mathbf{0 \leq x \leq 2}$

Extension to 2D interpolation

- **rectangle**: $\mathbf{g_{ij}(x,y) = f(i+\frac{1}{2}, j+\frac{1}{2}) s_1(x-i) s_1(y-j)}$
- **trapezoid**:

$$\begin{aligned} \mathbf{g_{ij}(x,y)} &= \mathbf{f(i,j)}\ell_{00}\mathbf{(x-i, y-j)} + \mathbf{f(i+1,j)}\ell_{10}\mathbf{(x-i, y-j)} + \\ & \quad \mathbf{f(i,j+1)}\ell_{01}\mathbf{(x-i, y-j)} + \mathbf{f(i+1,j+1)}\ell_{11}\mathbf{(x-i, y-j)} \end{aligned}$$

  - $\ell_{00}(x,y) = (1-x)\cdot(1-y)$, $\ell_{01}(x,y) = (1-x)\cdot y$, $\ell_{10}(x,y) = x\cdot(1-y)$, $\ell_{11}(x,y) = x\cdot y$

- **quadratic** (Simpson): **9** basis functions defined over $[0,2]\times[0,2]$: $\mathbf{q_{ij}(x,y) = q_i(x) q_j(y)} \quad i,j \in \{0,1,2\}$
  - adjacent pixels **overlapping** interpolation range

Extension to 2D interpolation

- **rectangle**: $\mathbf{g_{ij}(x,y) = f(i+\frac{1}{2}, j+\frac{1}{2}) s_1(x-i) s_1(y-j)}$
- **trapezoid**:

$$\begin{aligned} \mathbf{g_{ij}(x,y)} \quad = \quad & \mathbf{f(i,j)}\ell_{00}\mathbf{(x-i, y-j)} + \mathbf{f(i+1,j)}\ell_{10}\mathbf{(x-i, y-j)} + \\ & \mathbf{f(i,j+1)}\ell_{01}\mathbf{(x-i, y-j)} + \mathbf{f(i+1,j+1)}\ell_{11}\mathbf{(x-i, y-j)} \end{aligned}$$

  - $\ell_{00}(x,y) = (1-x)\cdot(1-y)$, $\ell_{01}(x,y) = (1-x)\cdot y$, $\ell_{10}(x,y) = x\cdot(1-y)$, $\ell_{11}(x,y) = x\cdot y$

- **quadratic** (Simpson): **9** basis functions defined over $[0,2]\times[0,2]$: $\mathbf{q_{ij}(x,y) = q_i(x) q_j(y)} \quad i,j \in \{0,1,2\}$
  - adjacent pixels **overlapping** interpolation range

Extension to 2D interpolation

- **rectangle**: $\mathbf{g_{ij}(x,y) = f(i+\frac{1}{2}, j+\frac{1}{2}) s_1(x-i) s_1(y-j)}$
- **trapezoid**:

$$\begin{aligned}\mathbf{g_{ij}(x,y)} \quad = \quad & \mathbf{f(i,j)}\ell_{00}\mathbf{(x-i, y-j)} + \mathbf{f(i+1,j)}\ell_{10}\mathbf{(x-i, y-j)} + \\ & \mathbf{f(i,j+1)}\ell_{01}\mathbf{(x-i, y-j)} + \mathbf{f(i+1,j+1)}\ell_{11}\mathbf{(x-i, y-j)}\end{aligned}$$

  - $\ell_{00}(x,y) = (1-x)\cdot(1-y)$, $\ell_{01}(x,y) = (1-x)\cdot y$, $\ell_{10}(x,y) = x\cdot(1-y)$, $\ell_{11}(x,y) = x\cdot y$

- **quadratic** (Simpson): **9** basis functions defined over $[0,2]\times[0,2]$: $\mathbf{q_{ij}(x,y) = q_i(x) q_j(y)} \quad i,j \in \{0,1,2\}$
  - adjacent pixels **overlapping** interpolation range

Computation of line integrals

For the integral of the *blending functions*

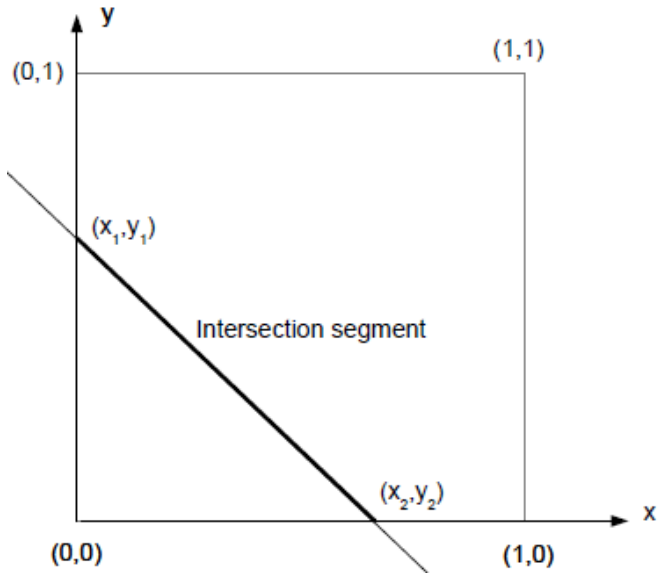
$$\mathbf{c_{ij}} = \int_{P_1}^{P_2} \ell_{ij}(x,y)ds = \int_{(x_1,y_1)}^{(x_2,y_2)} \ell_{ij}(x,y)ds$$

use the parametrization

$$(x(t),y(t)) = (1-t)\cdot P_1 + t\cdot P_2 \qquad 0 \le t \le 1$$

Then:

$$\mathbf{c_{ij}} = ||P_2 - P_1|| \cdot \int_0^1 \ell_{ij}(x(t),y(t))dt$$



Computation of line integrals(continued)

If we denote

$$dx = x_2 - x_1 \qquad dy = y_2 - y_1$$

$$\widetilde{x}_1 = 1 - \widetilde{x}_1 \qquad \widetilde{y}_1 = 1 - \widetilde{y}_1$$

then

$$\begin{array}{lcllll}
\mathbf{c_{00}} & = & +\frac{1}{3} \cdot dx \cdot dy & -\frac{1}{2} \cdot \widetilde{x}_1 \cdot dy & -\frac{1}{2} \cdot dx \cdot \widetilde{y}_1 & +\widetilde{x}_1 \cdot \widetilde{y}_1 \\
\mathbf{c_{10}} & = & -\frac{1}{3} \cdot dx \cdot dy & -\frac{1}{2} \cdot x_1 \cdot dy & +\frac{1}{2} \cdot dx \cdot \widetilde{y}_1 & +x_1 \cdot \widetilde{y}_1 \\
\mathbf{c_{01}} & = & -\frac{1}{3} \cdot dx \cdot dy & +\frac{1}{2} \cdot \widetilde{x}_1 \cdot dy & -\frac{1}{2} \cdot dx \cdot y_1 & +\widetilde{x}_1 \cdot y_1 \\
\mathbf{c_{11}} & = & +\frac{1}{3} \cdot dx \cdot dy & +\frac{1}{2} \cdot x_1 \cdot dy & +\frac{1}{2} \cdot dx \cdot y_1 & +x_1 \cdot y_1
\end{array}$$

Calculation of weights

Input: (ρ, θ) - ray parameters
G - grid definition

Output: w_{ij} - weights in sparse representation

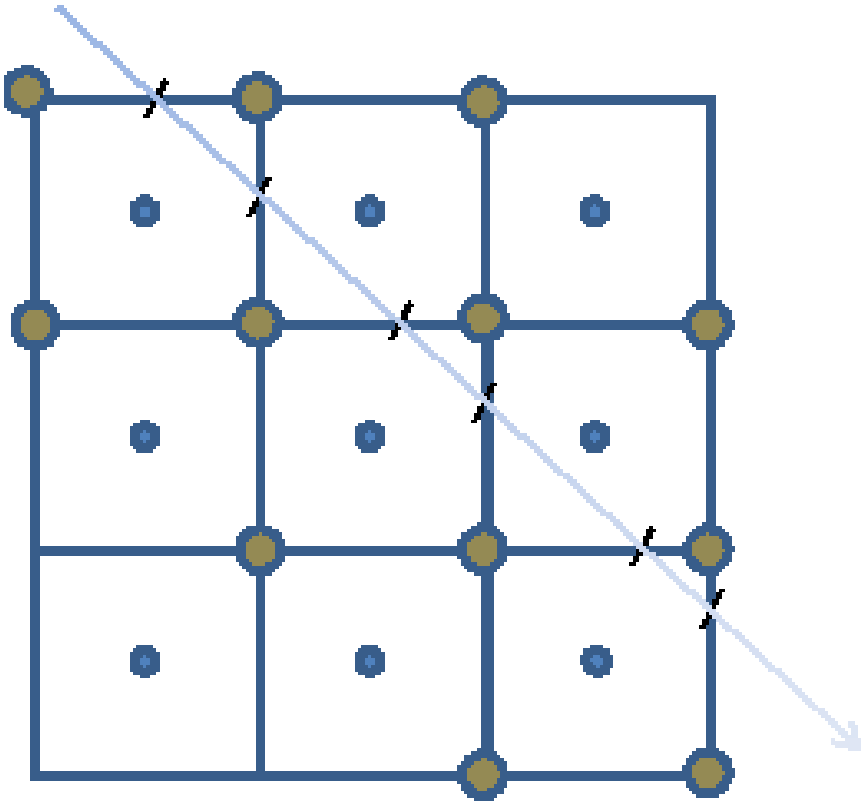
- General interface for any order of the Newton-Cotes basis functions
- Pure computational geometry routine (depends only on the grid and rays configuration)

General Schema

```
for each ray do
    Traverse the grid and get all ray-pixel intersections
    for each intersected pixel do
        Apply the blending function integral
        Calculate ray's weights
    end for
end for
```

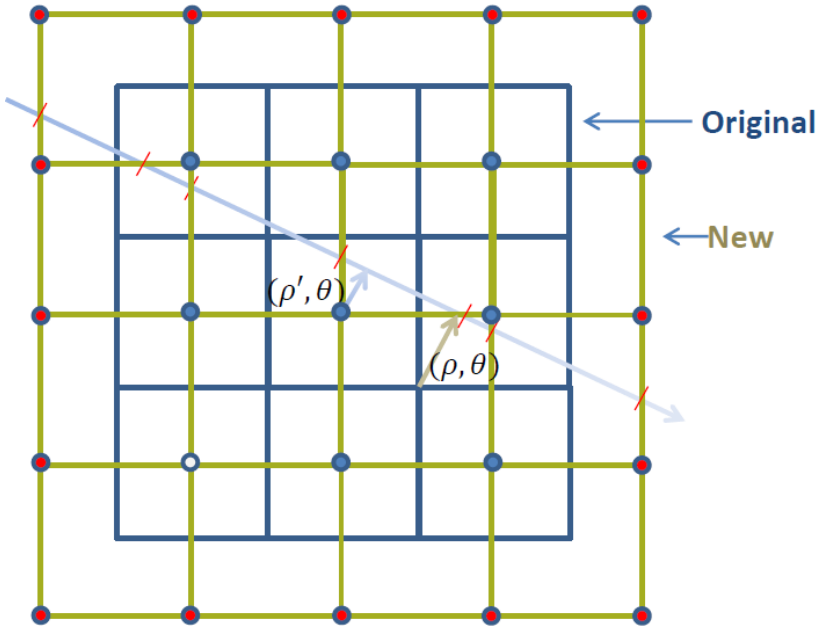
Model's Geometry

Problem: Linear (and higher order) interpolation on the grid corners will produce biased weights



Model's Geometry (continued)

Solution: Define a new grid



- **Edge pixels** requires special handling
- Frame of reference changed

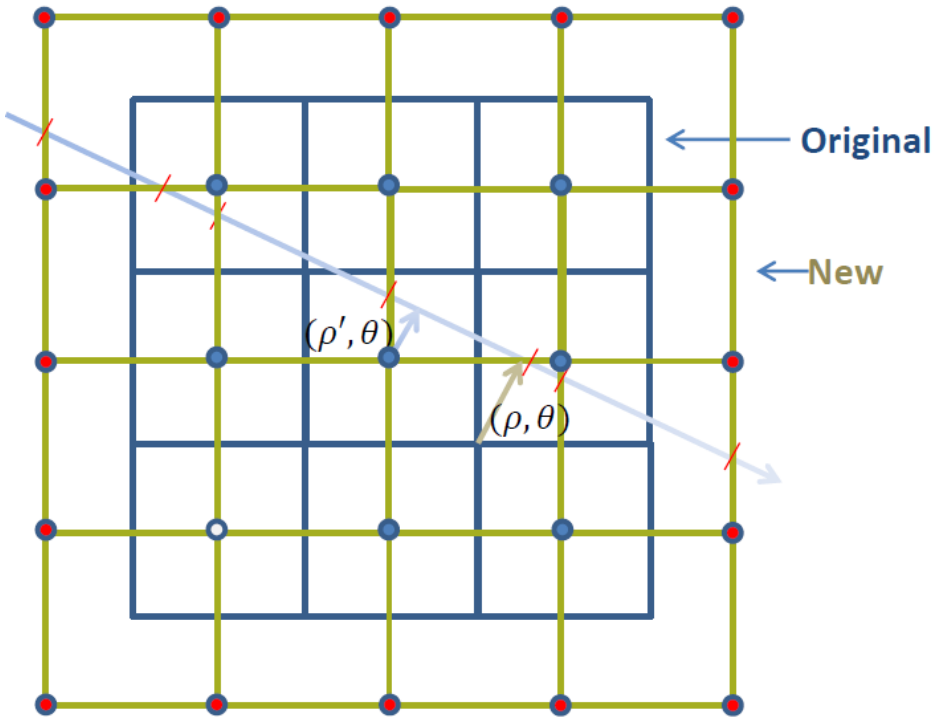
Grid traversal algorithms

DDA (Digital Differential Analyzer)

- sequential
- accumulates errors (truncation, round-off)
- rays are translated

Our proposed algorithm

- parallel
- no error accumulation
- grid is translated



Ray translation

- Normal form: $\mathbf{x}\cos\theta + \mathbf{y}\sin\theta = \rho$
- Vector form: $\mathbf{r}\cdot\mathbf{u} = \rho$, ,$\mathbf{r}'\cdot\mathbf{u} = \rho'$ $\mathbf{u} = (\cos\theta, \sin\theta)$
- The translation is $\mathbf{r}' - \mathbf{r} = \Delta\mathbf{r} = (\Delta x, \Delta y)$
- Line-normal intersection points: $\mathbf{a} = \rho\mathbf{u}$, $\mathbf{a}' = \rho'\mathbf{u}$
- For $(\Delta x, \Delta y) = (\frac{\mathbf{1}}{\mathbf{2}}, \frac{\mathbf{1}}{\mathbf{2}})$

$$\Delta\mathbf{a} = -\frac{\mathbf{1}}{\mathbf{2}}(\cos\theta + \sin\theta)\,(\cos\theta, \sin\theta)$$

Our proposed algorithm

Schema:

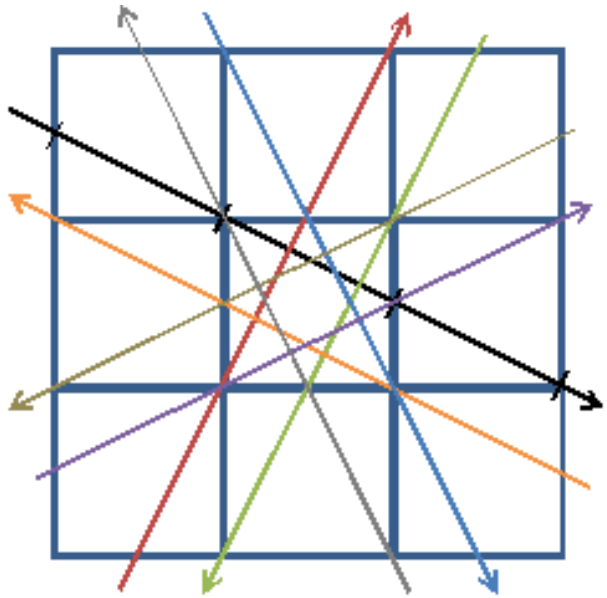
- Ray: $\mathbf{P_t} = \mathbf{P_0} + \mathbf{t} \cdot \mathbf{dir}$
- Grid: $\mathbf{x} = \mathbf{t_{x_i}}$, $\mathbf{y} = \mathbf{t_{y_j}}$
- Determine bounding box in $\mathbf{t}$: $[\mathbf{t_{x_{low}}}, \mathbf{t_{x_{high}}}]$, $[\mathbf{t_{y_{low}}}, \mathbf{t_{y_{high}}}]$
- Collect the 2 lists of $\mathbf{t}$ values of the ray intersections with grid lines
- *Merge the 2 ordered lists*, and get the pixels and their intersection length

Weights calculation

- A pixel may sum weights from 1 to 3 intersections
- For sparse representation, **efficient** summation is non-trivial (requires tracking previous calculated pixels)
- Blending function integration involves additional arithmetic operations
- Handling outlier pixels

Symmetry

- Symmetry groups of order 8 (in 2D)
- Computation reduction is factor of group's order, but depends on selection of projection angles
- Weights mapping is simple (coordinates substitutions, linear transformations)
- Mapping LUT



Weights storage

- In iterative reconstruction, ray projection usually applied numerous times
- Weights re-calculation in each projection drastically degrades performance
- For an almost determined system over $\mathbf{M} \times \mathbf{N}$ image: $\mathbf{O}(\mathbf{MN}(\max(\mathbf{M}, \mathbf{N}))$
- Symmetry usage
- May be infeasible in strip rays, higher dimensions, polynomial order, resolution

Parallel ray-detector CT simulation

- The linear model (**LBF**) was implemented in 2D in SNARK09 . Additional implemention - grid traversal, storage
- Tested and compared to pixel model (**PBF**) and **Blob**
  - Mathematical phantoms (Shepp-Logan, Herman's Head and others)
  - Configuration such as noise, detector and projection resolution, and phantom contrast
  - Quantitative measurement
  - 2 Reconstruction algorithms

Tested reconstruction algorithms

ART - Algebraic reconstruction technique

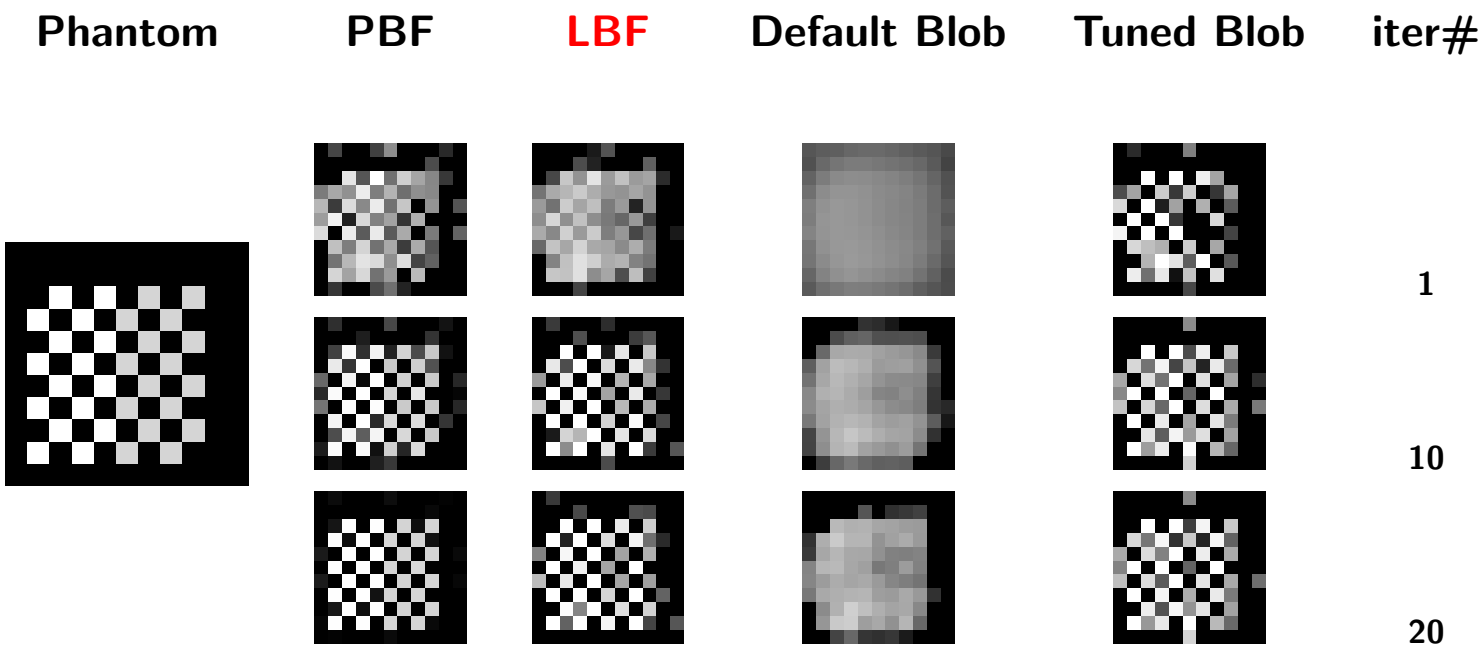
- Least square minimization

EMAP - Expectation maximization with log likelihood

- Regularization

- ART is not well suited to underdetermined system and noisy data
  - selective smoothing was used in some cases

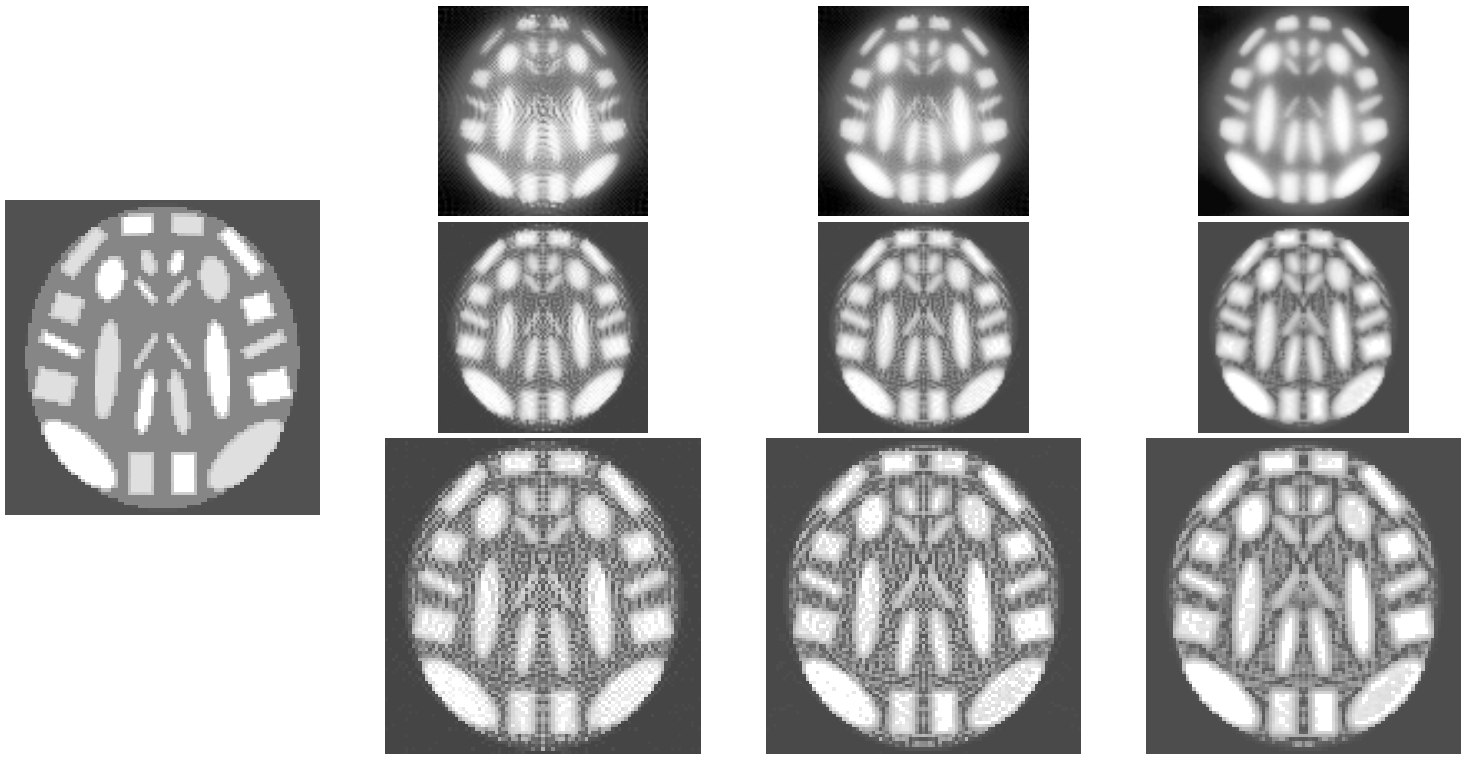
Spatial aliasing



- Blob can be destructive in highly detailed images

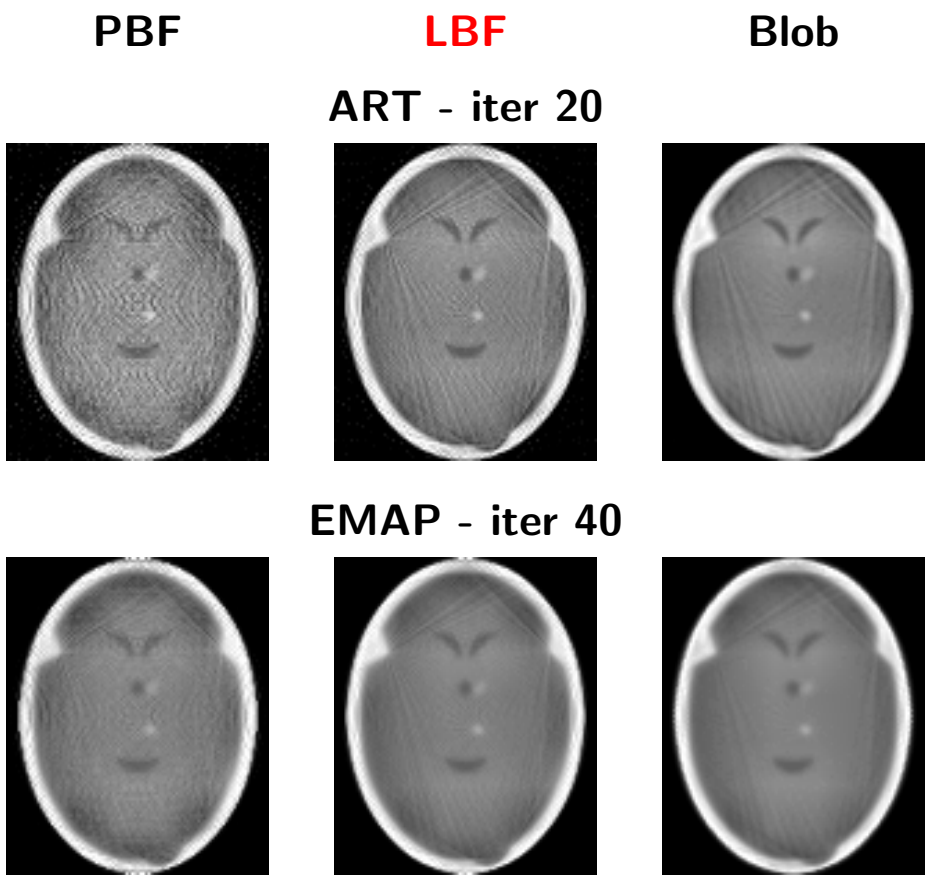
Low contrast

Identify region's **activity** (white or grey)

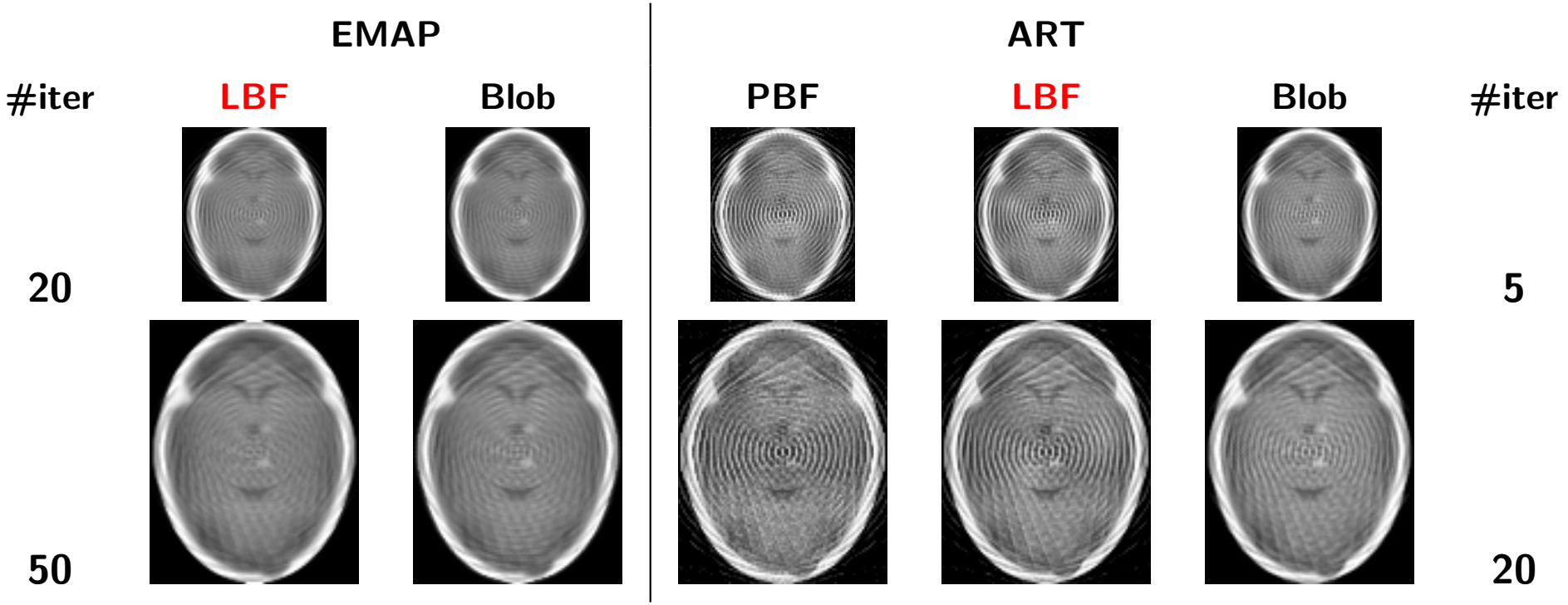


Low projection resolution

- Test for $\mathbf{N} \times \mathbf{N}$ image
  - $|projections| = \frac{\mathbf{N}}{2}$
  - $|equations| = \frac{\mathbf{N}^2}{2}$



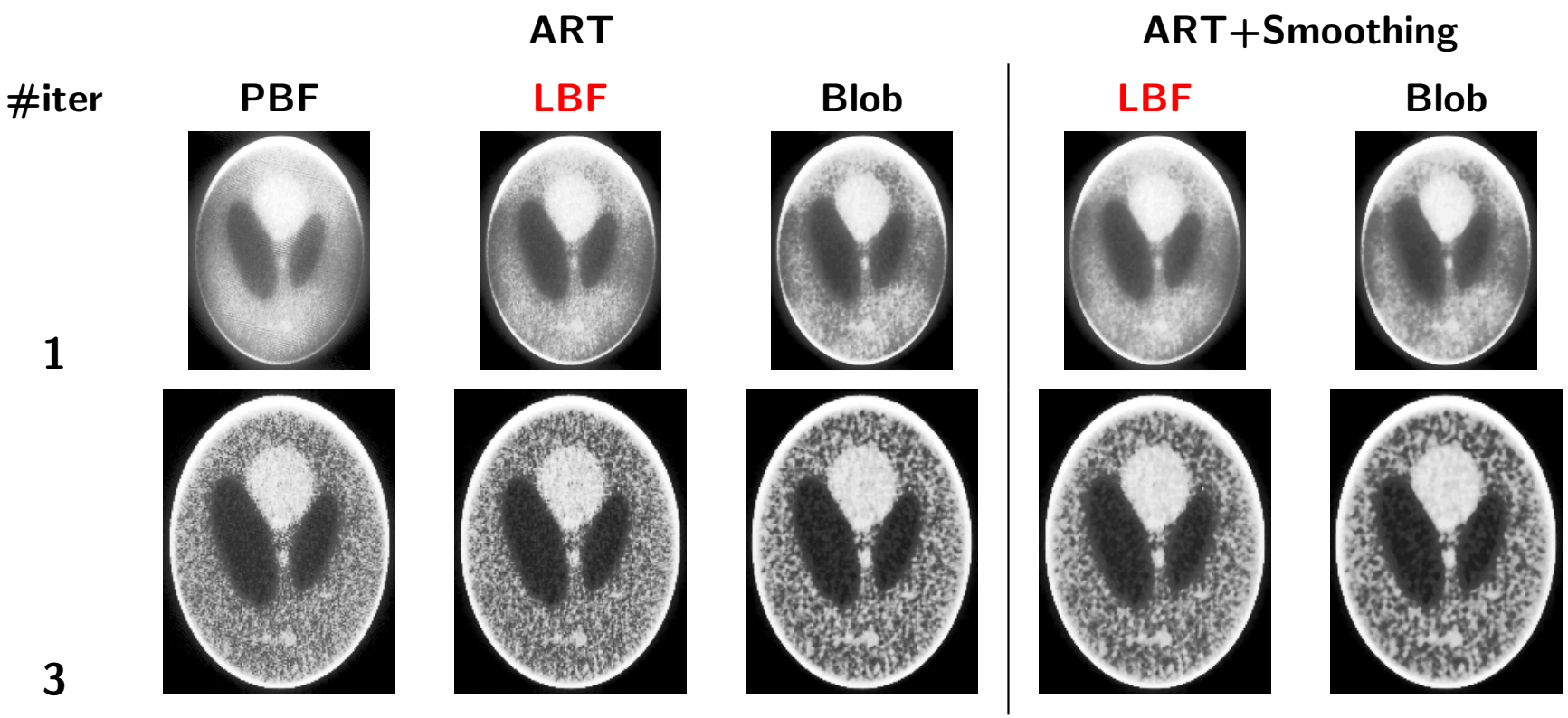
Low ray-detector resolution



- Regularization is essential
- LBF is comptetitive to Blob

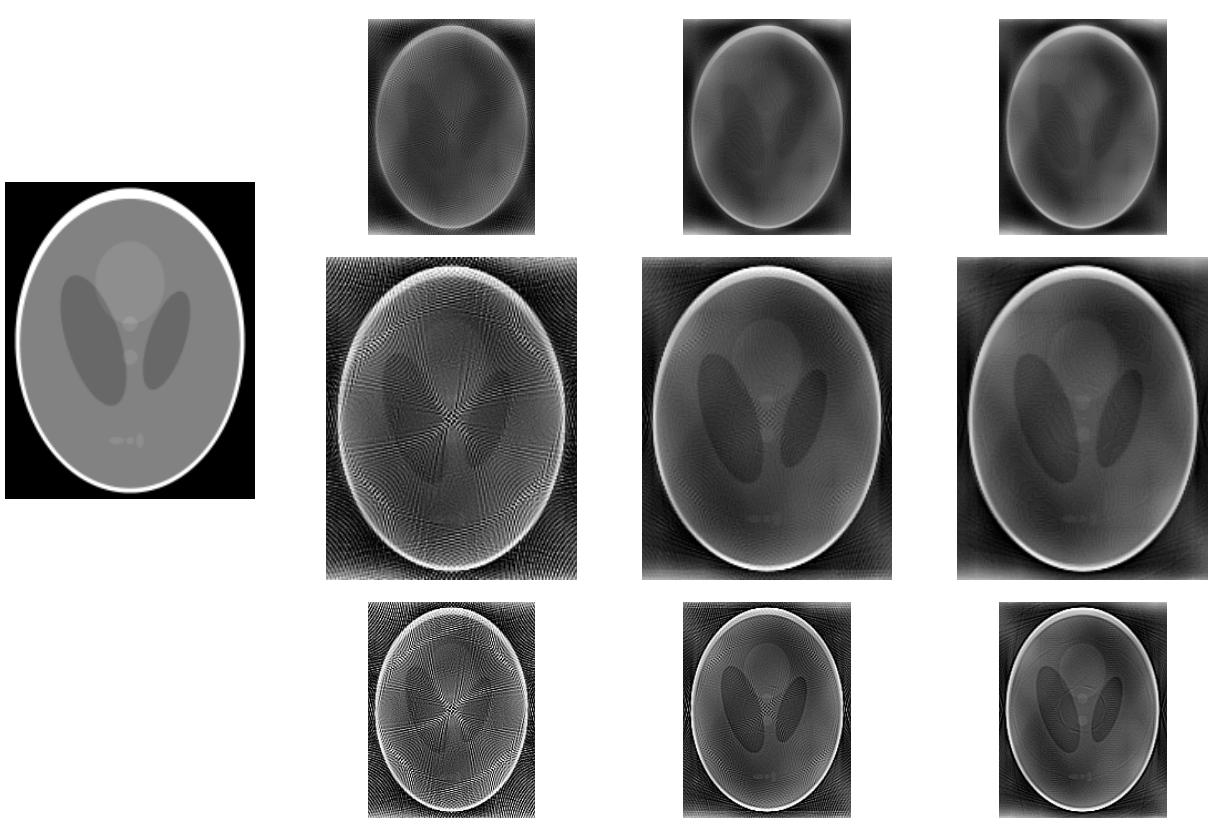
Multiplicative noise

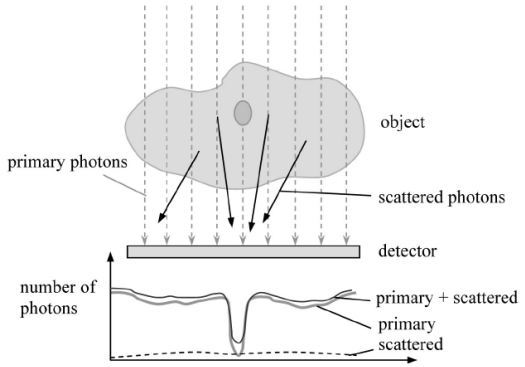
- by-product of signal amplification
- simulation: $P_{new} = P_{old} gauss(\widehat{\sigma}, \widehat{\rho})$



Scatter noise (Compton effect)

- Main artifacts: streaks and reduced contrast





Conclusions

- In almost any situation, the LBF produced better reconstrution then PBF
- LBF vs Blob: In underdetermined system and noise, some reconstruction algorithm will gain more from using LBF, while others more from Blob
- Computational complexity: PBF $<$ **LBF** $<$ Blob

Conclusions

- In almost any situation, the LBF produced better reconstrution then PBF
- LBF vs Blob: In underdetermined system and noise, some reconstruction algorithm will gain more from using LBF, while others more from Blob
- Computational complexity: PBF $<$ **LBF** $<$ Blob

Conclusions

- In almost any situation, the LBF produced better reconstrution then PBF
- LBF vs Blob: In underdetermined system and noise, some reconstruction algorithm will gain more from using LBF, while others more from Blob
- Computational complexity: PBF $<$ **LBF** $<$ Blob

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals

Future Work

- Test clinical sinograms
- Experiment more reconstruction algorithms
- Computation optimization
- Super-resolutions grids
- Quadratic model
- Fully 3D reconstructions
- Applications to PET and electron tomography
- Experiment other approximation methods
- Other types of ray integrals