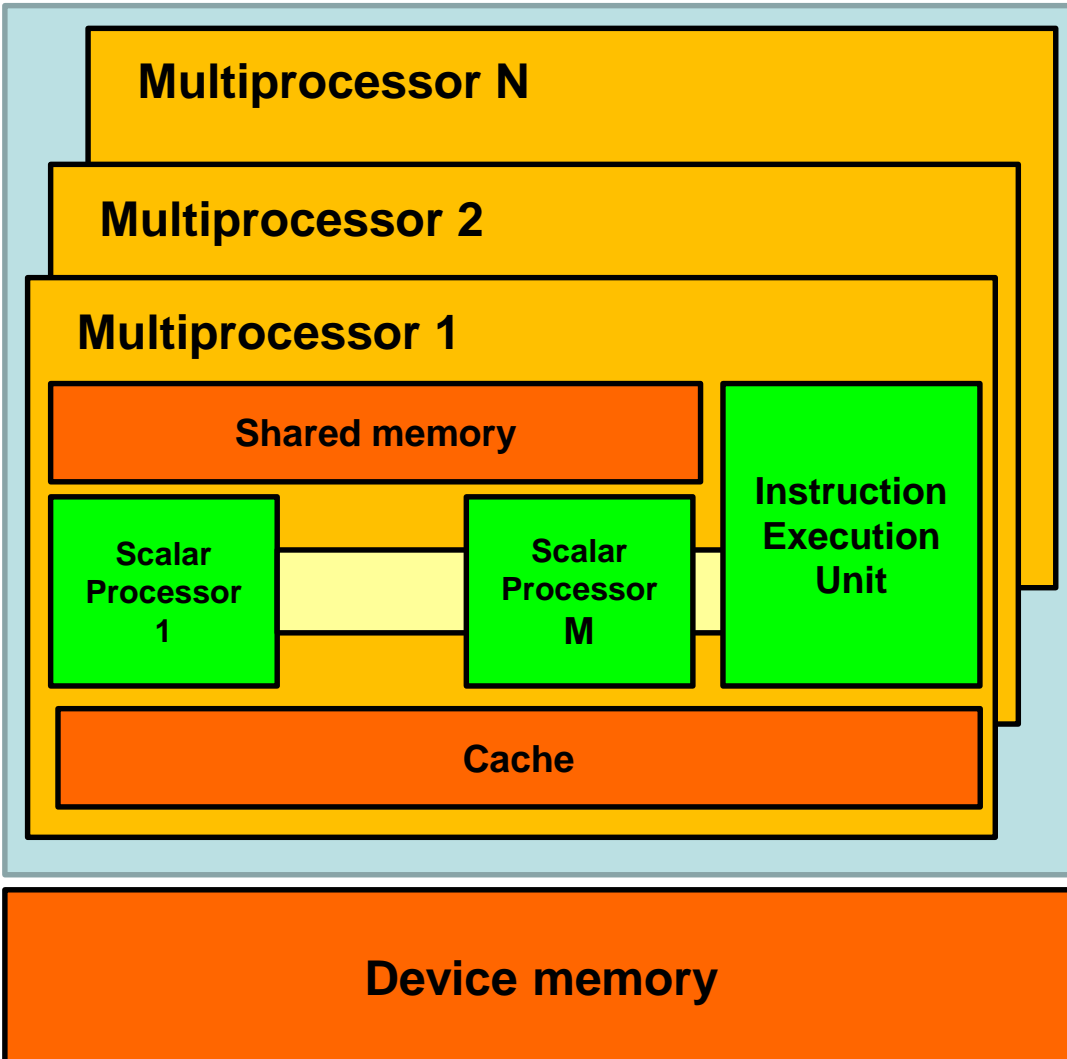


Massively Parallel GPU-friendly Algorithms for PET

Szirmay-Kalos László, <http://cg.iit.bme.hu>,
Budapest, University of Technology and Economics

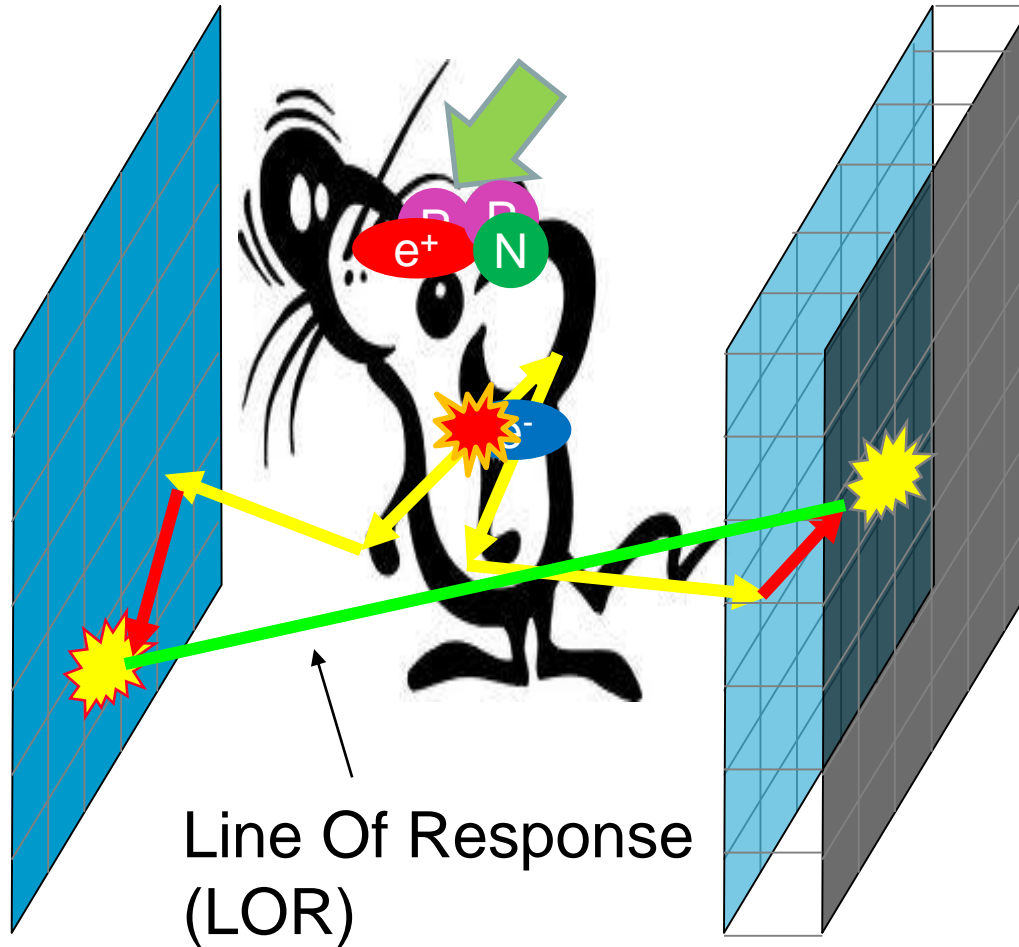


(GP)GPU: CUDA (OpenCL)



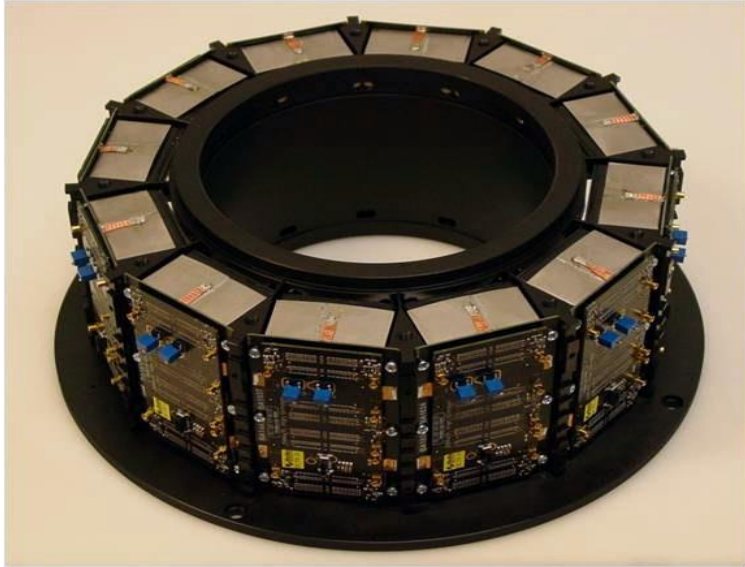
- Massively parallel: #threads $> 10^4$
- Independence: synchronization and write collisions should be avoided
- SIMD: conditional statements are not welcome
- Coalesced memory access

PET physics



Mediso

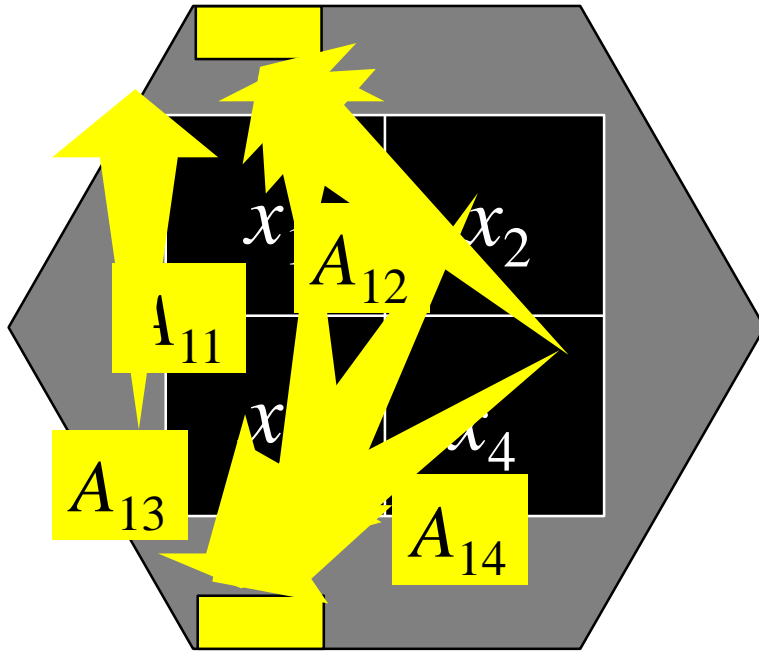
nanoScan PET



AnyScan PET



Maximum-likelihood reconstruction



$$\tilde{y}_1 = A_{11}x_1 + A_{12}x_2 + A_{13}x_3 + A_{14}x_4$$

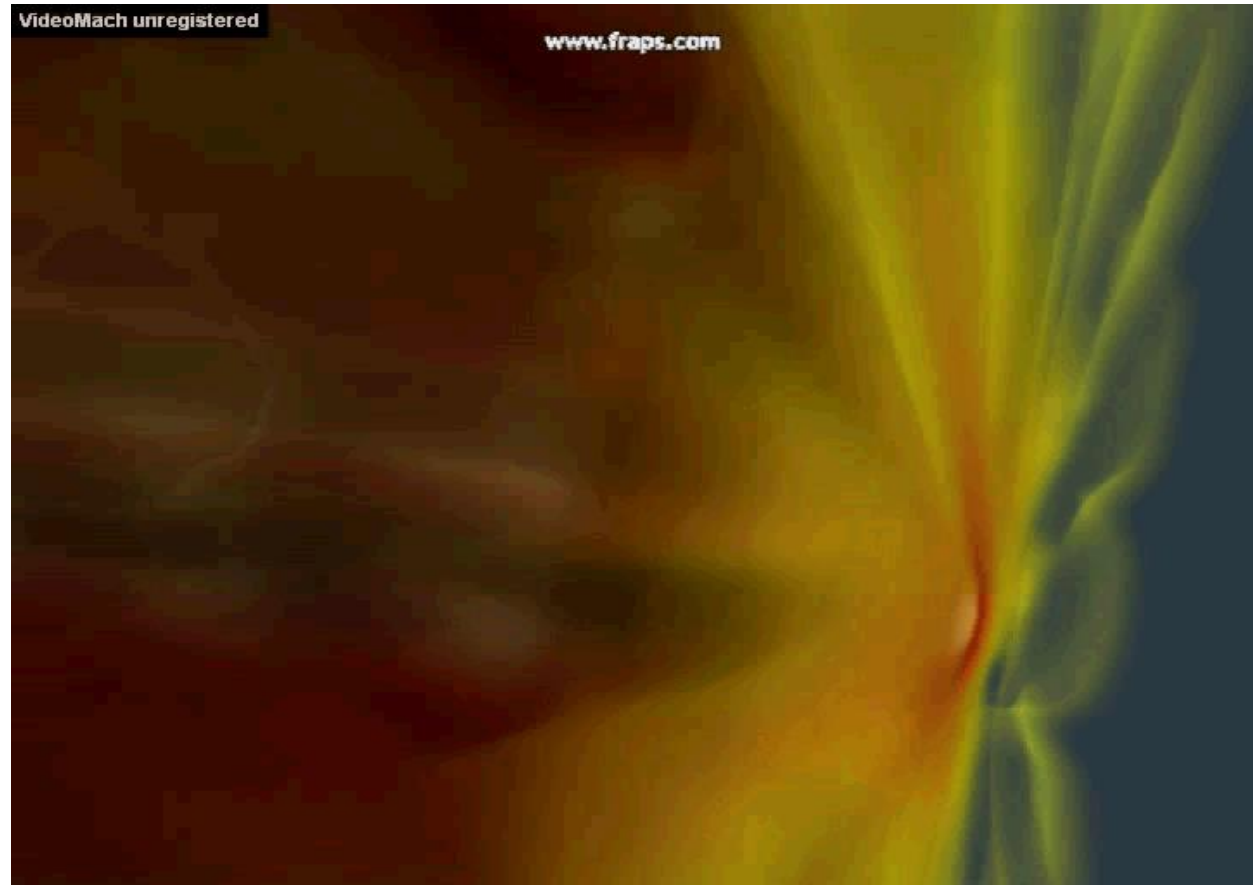
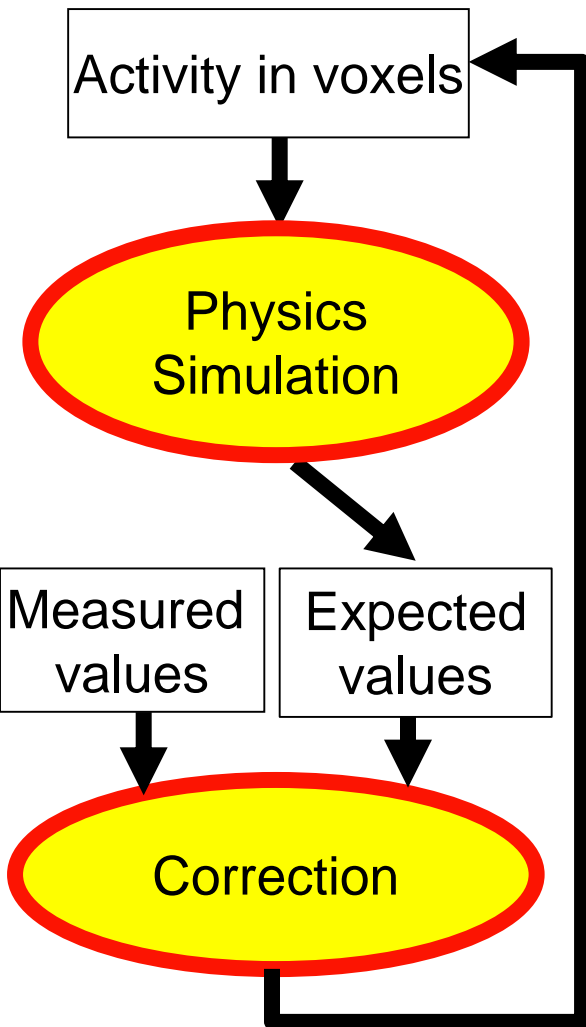
Expected number of hits:

$$\tilde{\mathbf{y}} = \mathbf{A} \cdot \mathbf{x}$$

Maximize the probability of the measurement data \mathbf{y} :

$$\mathbf{x} = \arg \max \Pr\{\mathbf{y} \mid \tilde{\mathbf{y}}(\mathbf{x})\}$$

Iterative solution

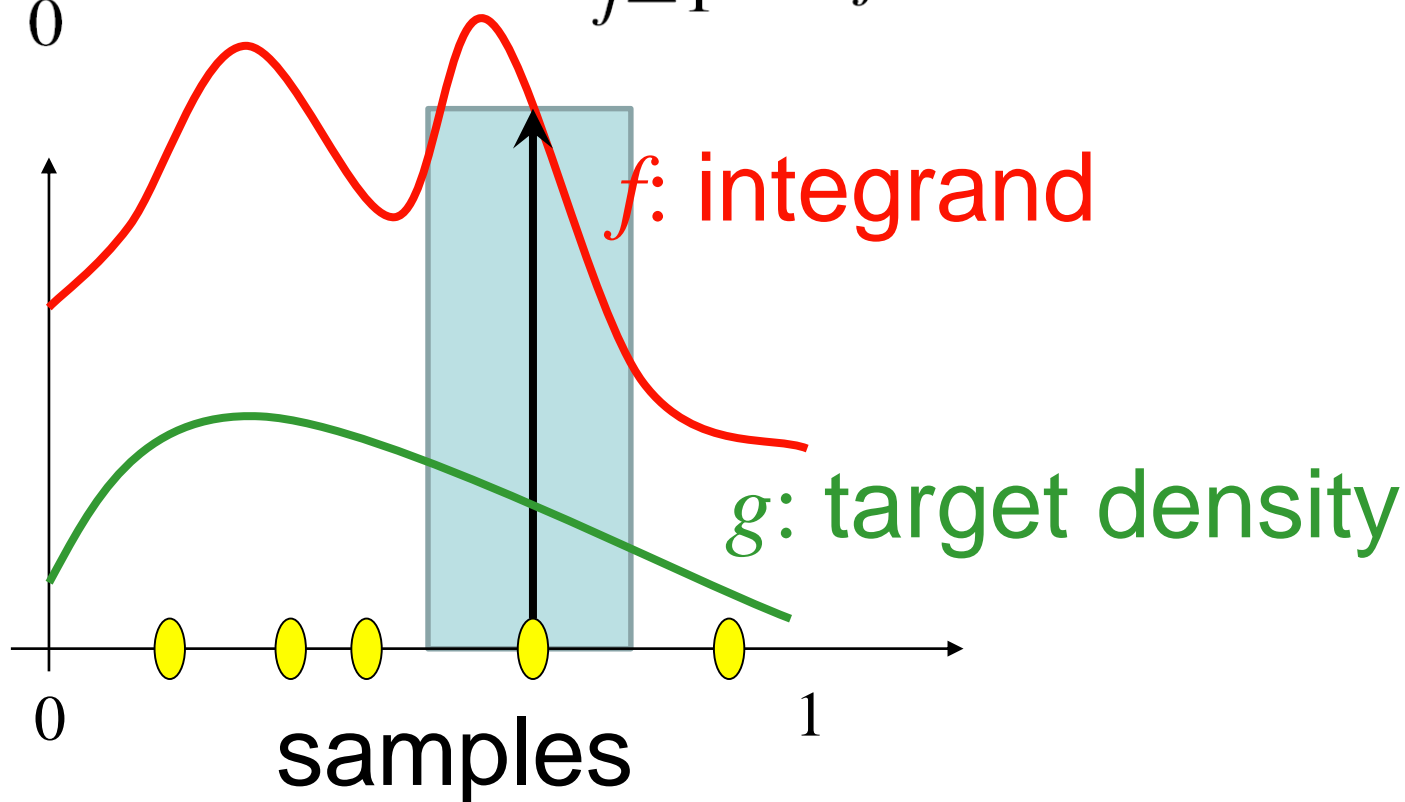


Computational challenges

- Numbers of LORs and voxels: hundred millions!
- System matrix \mathbf{A} : 10^{16} elements (PetaBytes)
 - Probability that a positron of a voxel is detected by a LOR
 - Patient dependent
 - Not sparse if accurate simulation is needed
 - Do not store, estimate on-the-fly
- Matrix elements are high dimensional integrals
 - Monte Carlo quadrature
 - Reuse of computation
 - High performance (parallel) computational platform
- Minimize the effect of estimation error

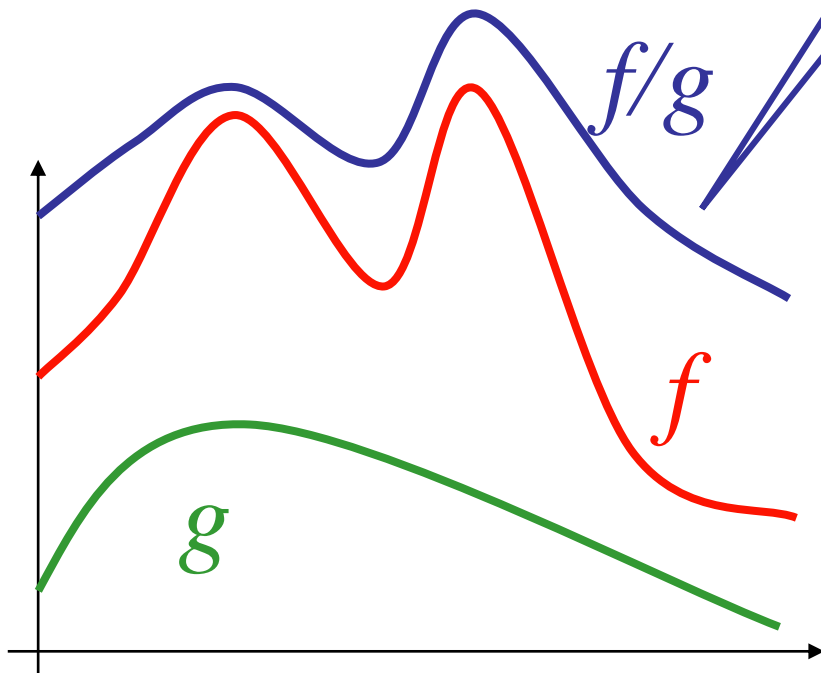
Numerical integration

$$\int_0^1 f(t) dt \approx \frac{1}{M} \sum_{j=1}^M \frac{f(t_j)}{g(t_j)}$$

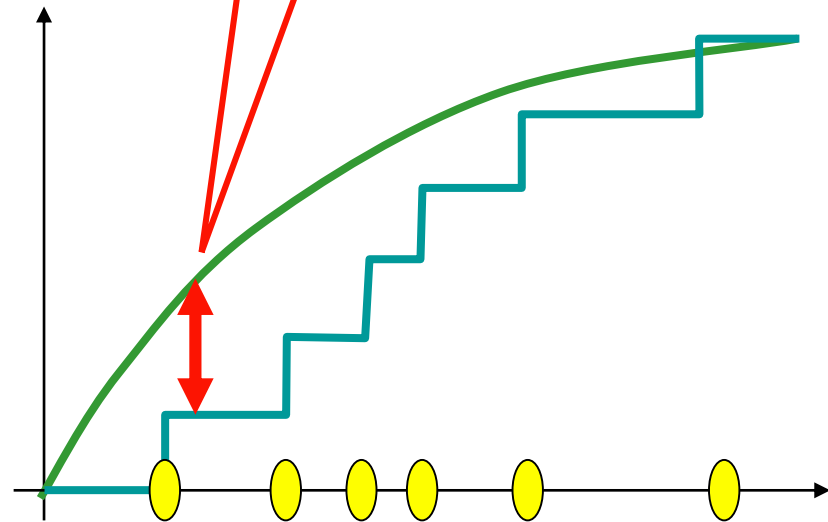


Quadrature error

$$\left| \int_0^1 f(t) dt - \frac{1}{M} \sum_{j=1}^M \frac{f(t_j)}{g(t_j)} \right| = \left| \int_0^1 \left(\frac{f}{g} \right)' \cdot \left(G(t) - \frac{m(t)}{M} \right) dt \right|$$

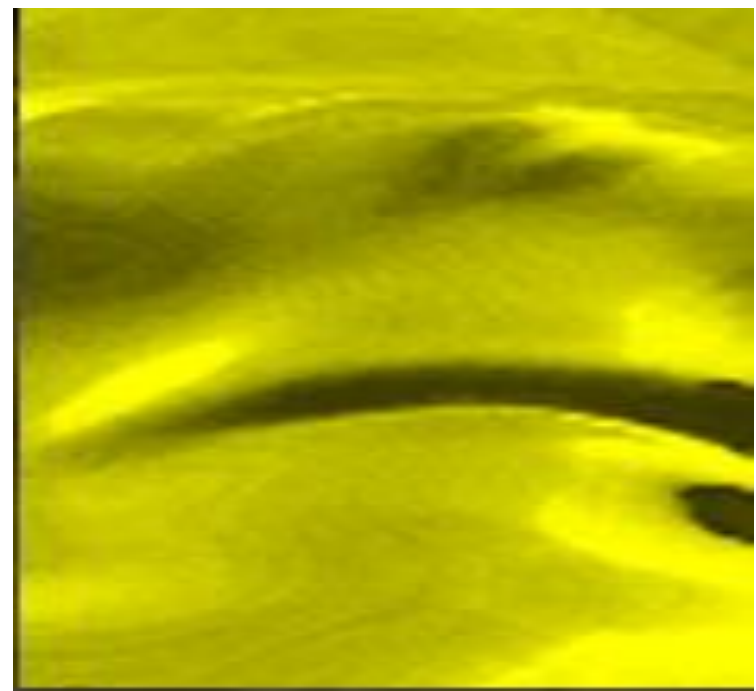
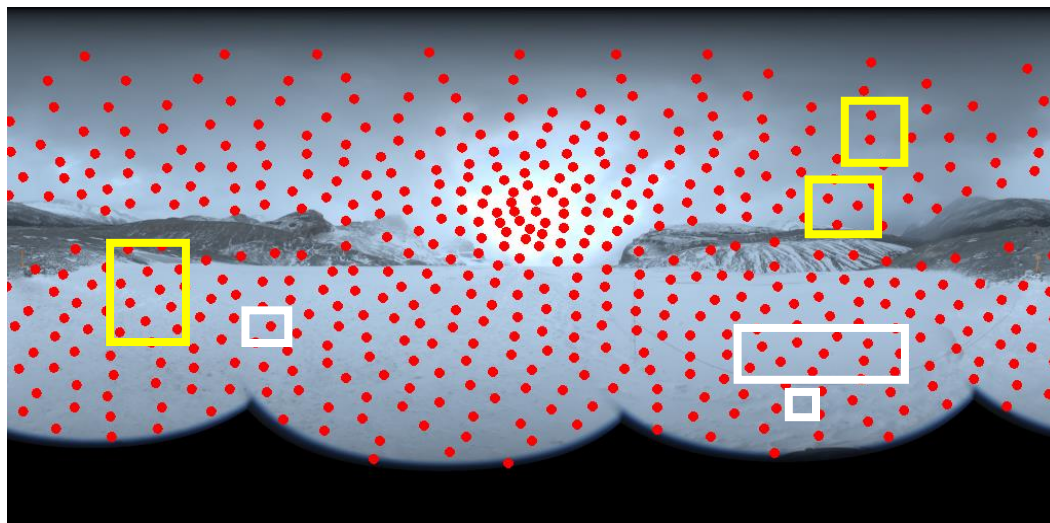
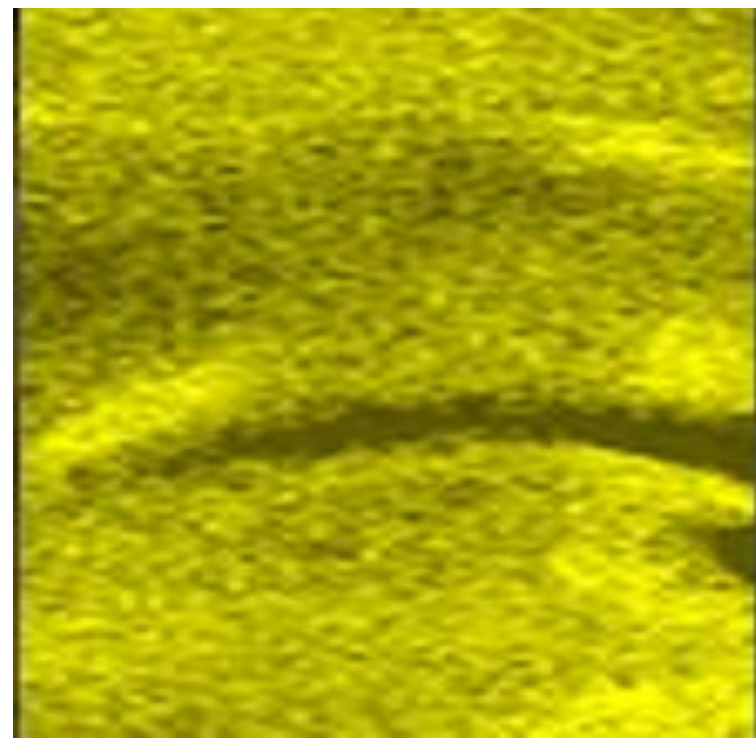
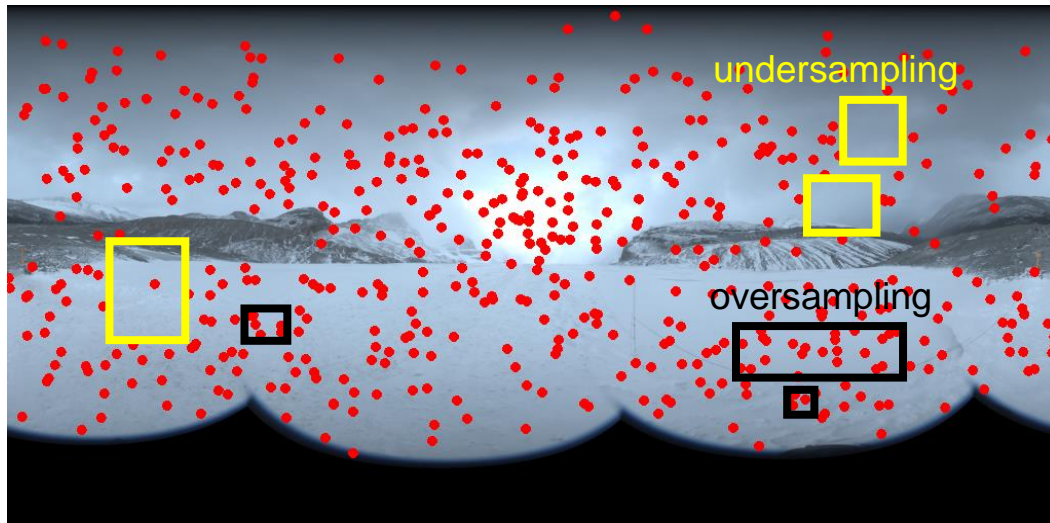


importance sampling



stratification

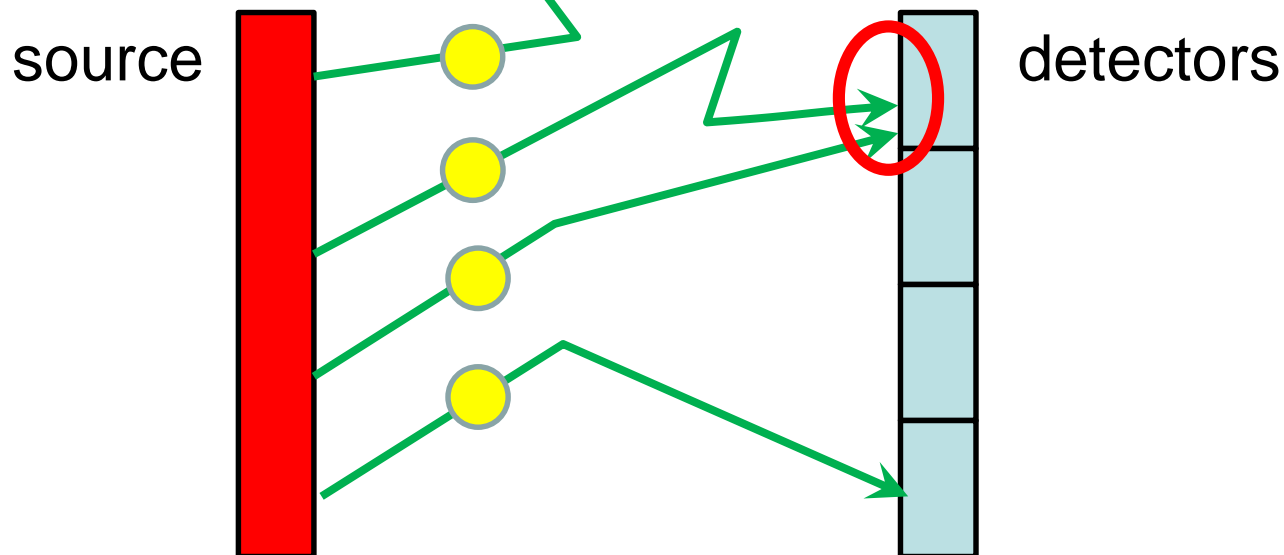
Effect of stratification



Direct physical simulation

Input driven, scattering type algorithm

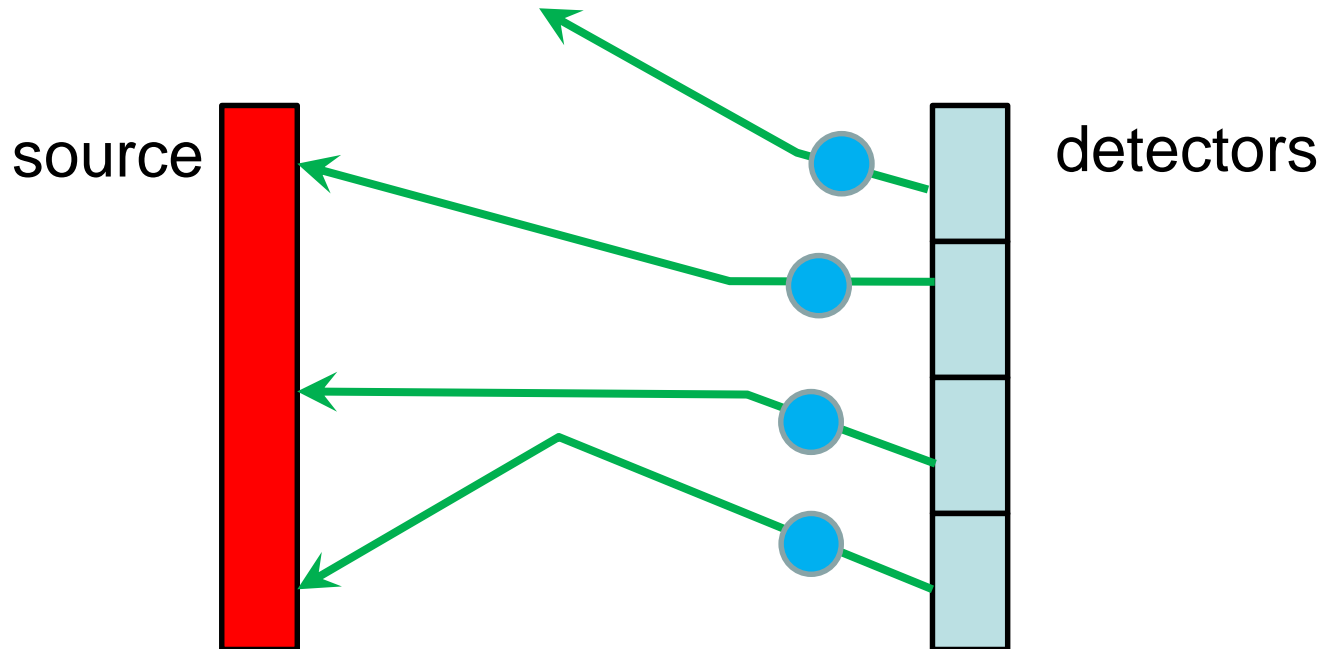
- **Thread = photon**
- Photons scatter different number of times
- The same detector is hit: write collision
- Random memory access
- Cannot mimic the detectors



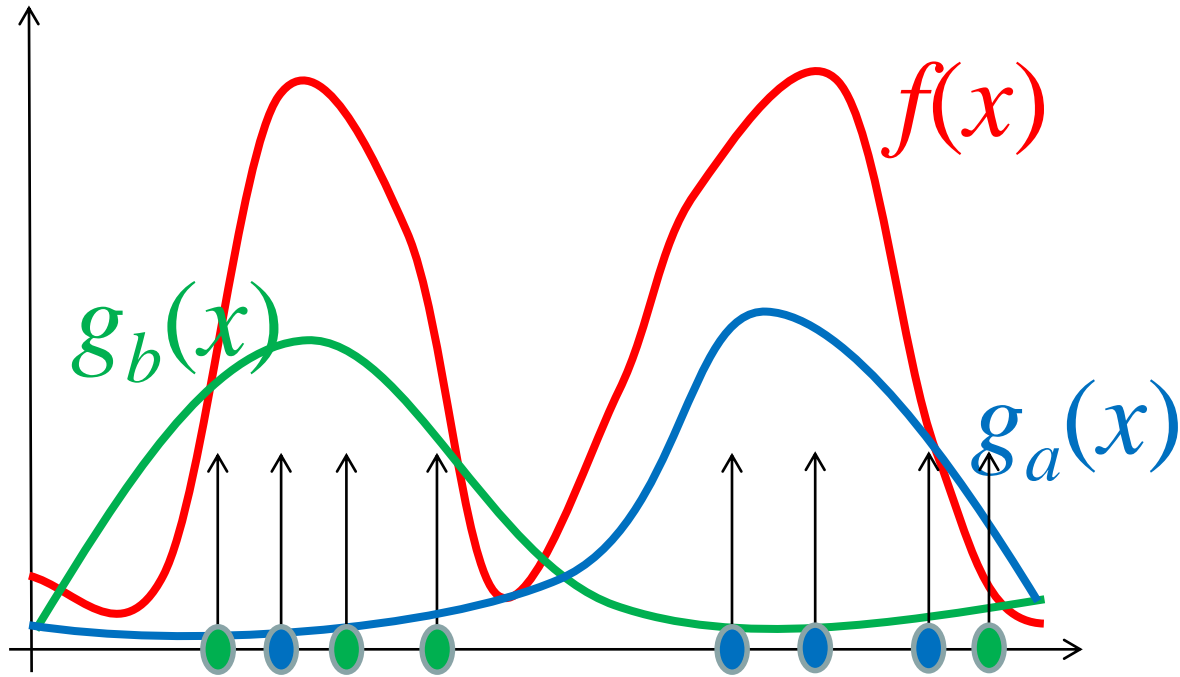
GPU friendly approach

Output driven, gathering type

- **Thread = importon**
- SIMD: grouping importons
- No write collision: LOR-driven
- Cannot mimic the source

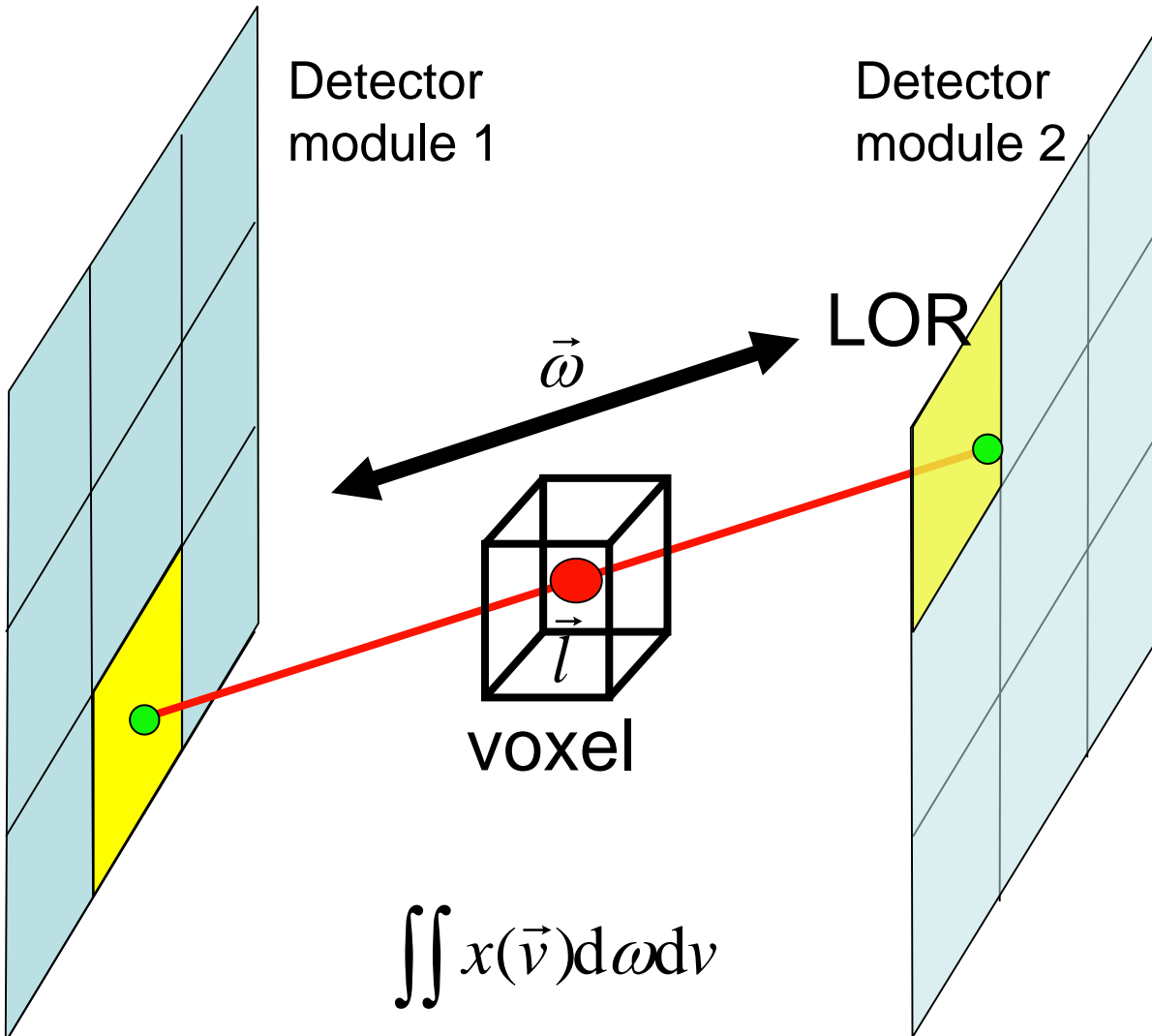


Multiple Importance Sampling



$$\frac{f(x_i)}{g_a(x_i)} \quad \frac{f(x_i)}{g_b(x_i)} \quad \frac{f(x_i)}{g_a(x_i) + g_b(x_i)}$$

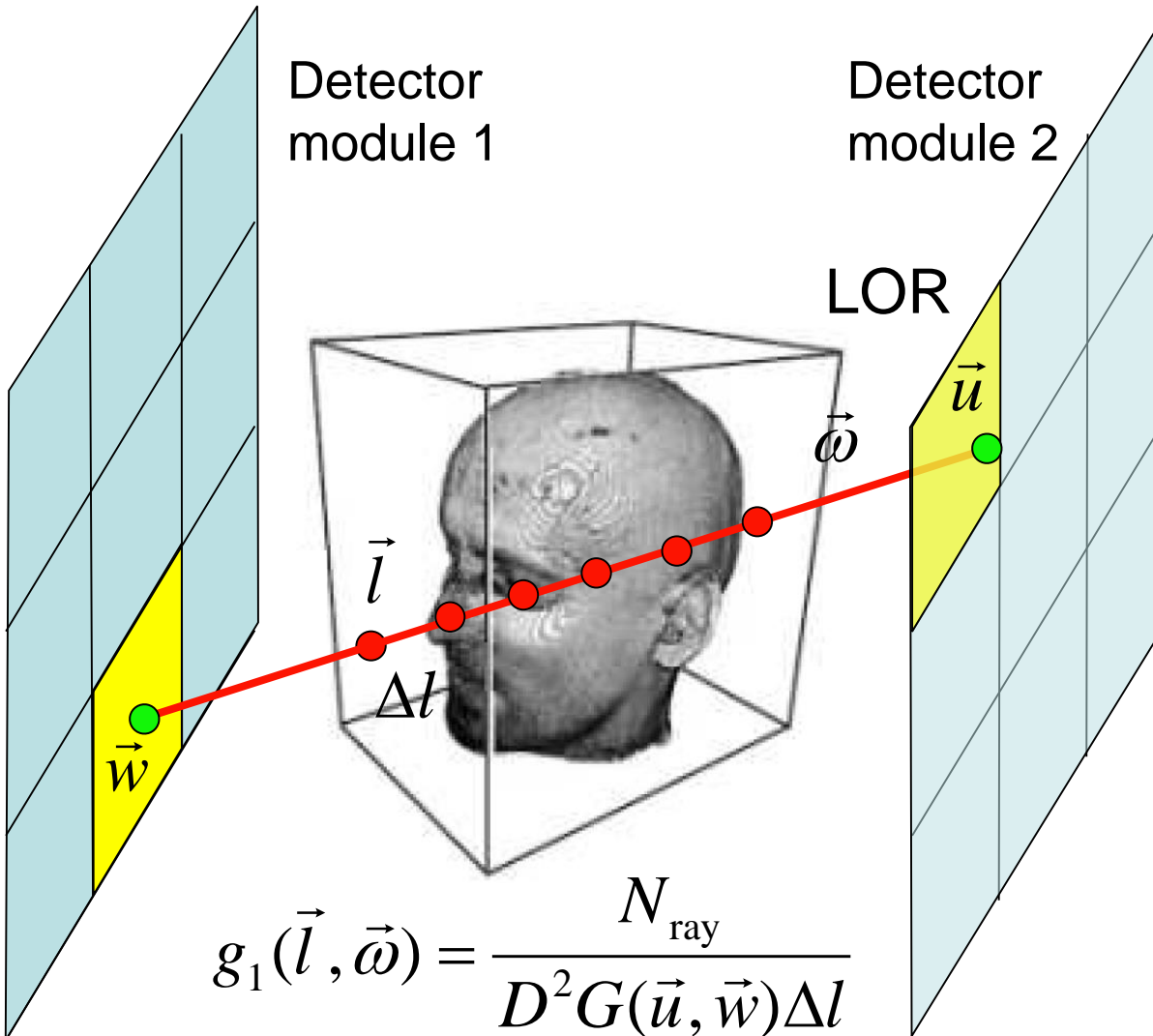
Direct gamma photon contribution



5D Integration

- Accuracy for given sample number
- Cost of a sample

Output- or LOR-driven sampling



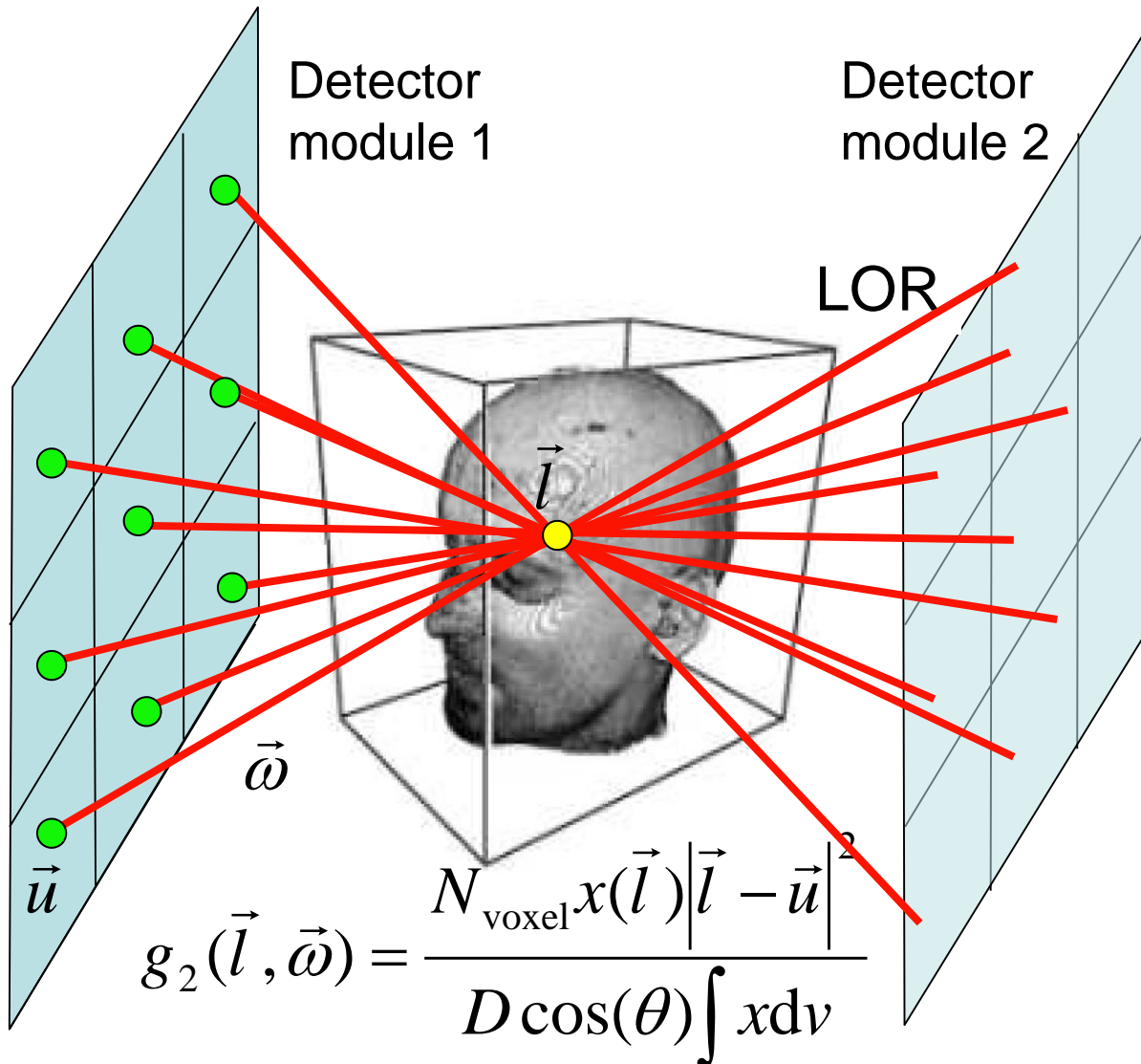
Pros:

- Gathering
- Thread coherence
- Texture coherence
- Uniform on detectors
- Low-cost samples due to reuse

Cons:

- Cannot mimick activity

Input or voxel-driven sampling



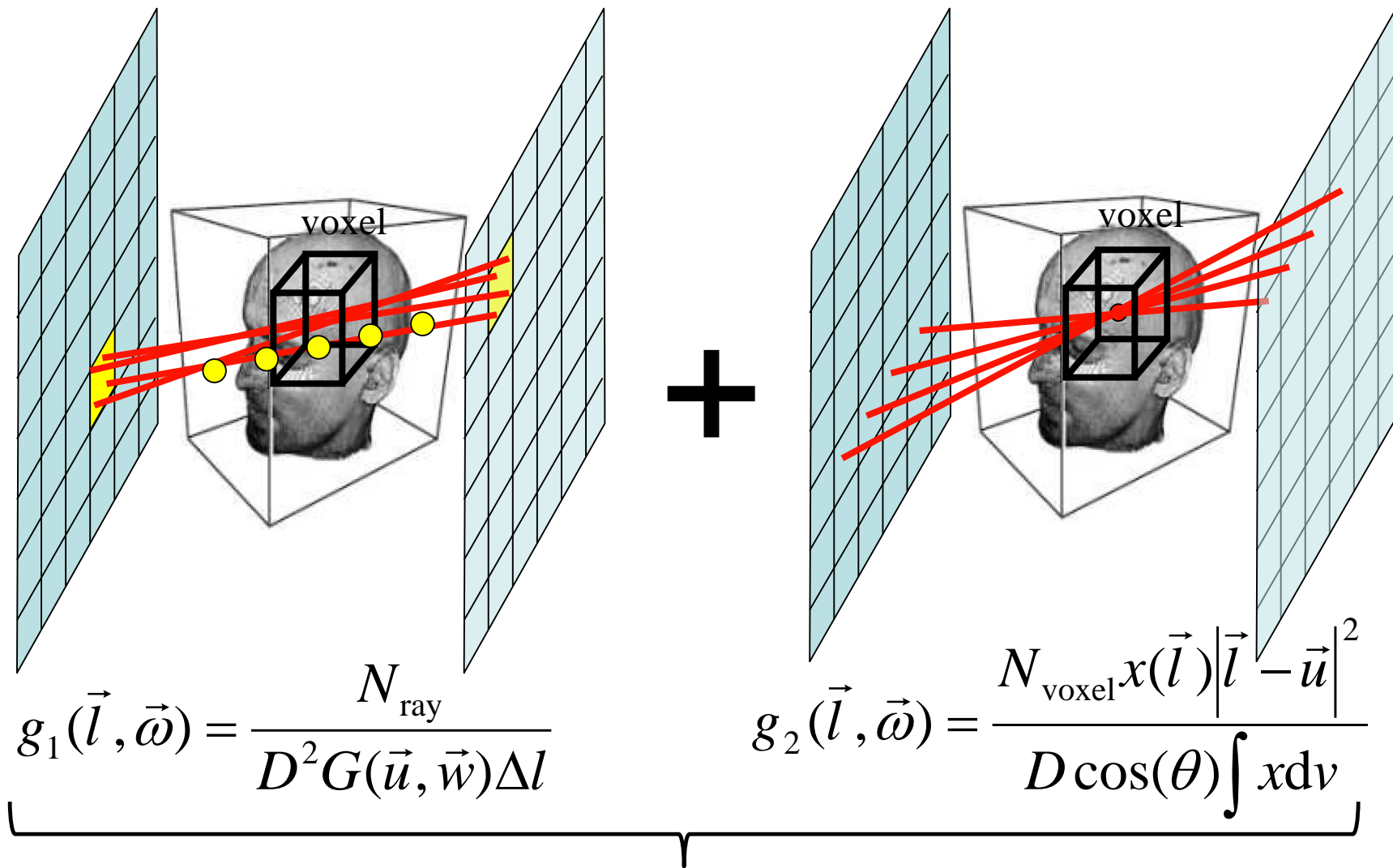
Pros:

- Can mimick activity

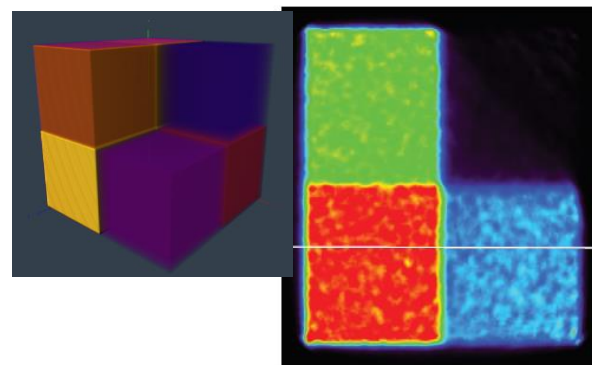
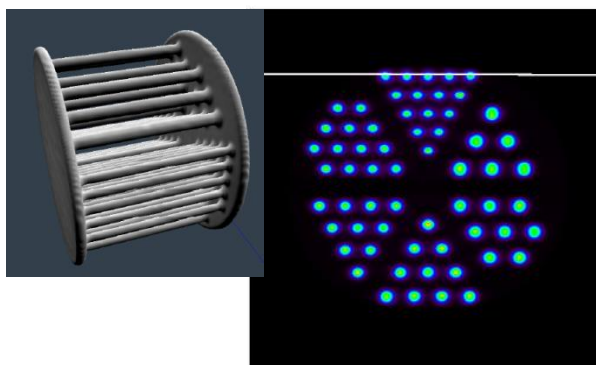
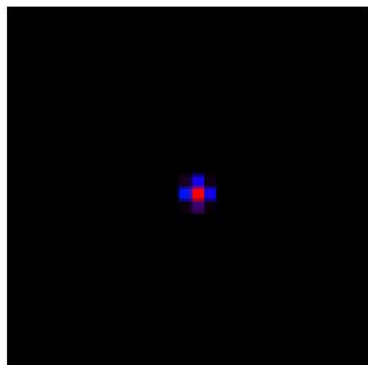
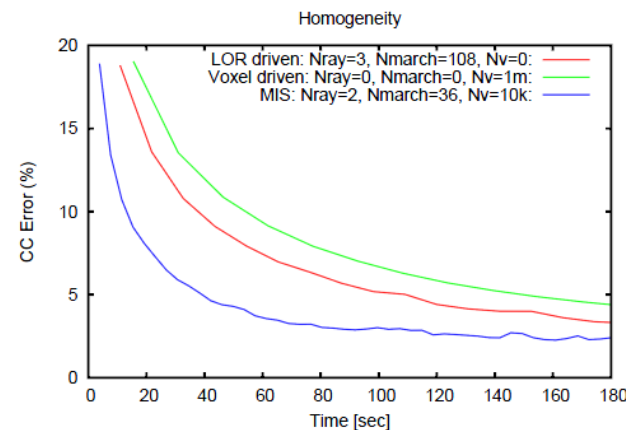
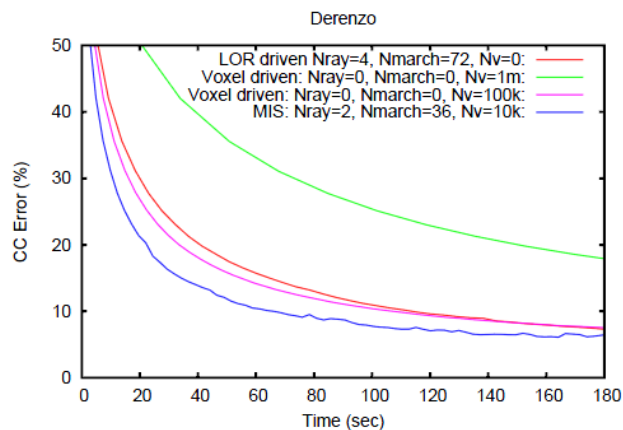
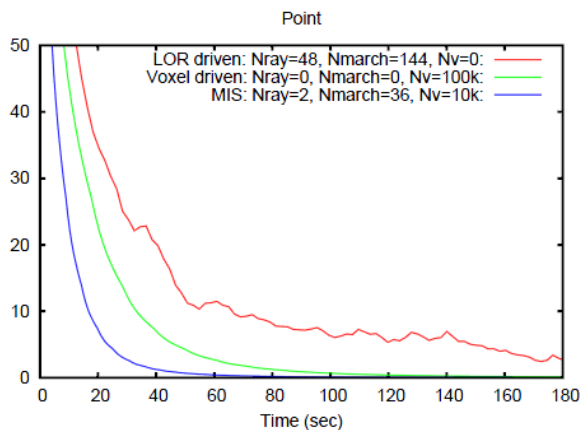
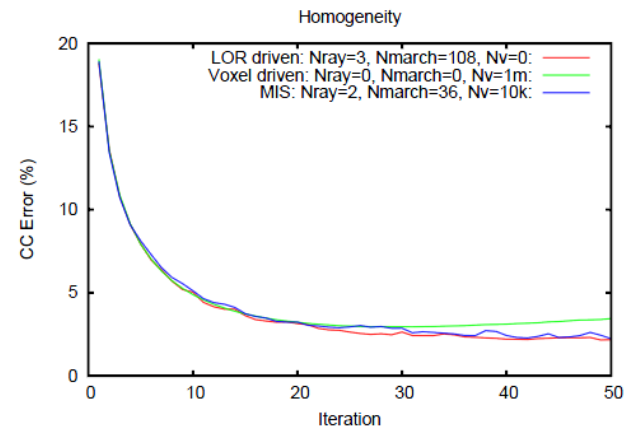
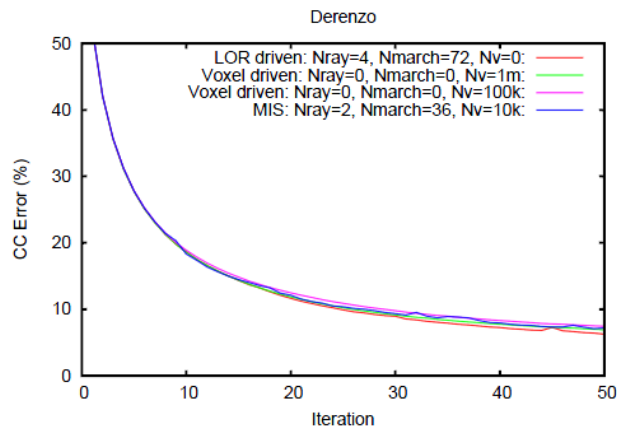
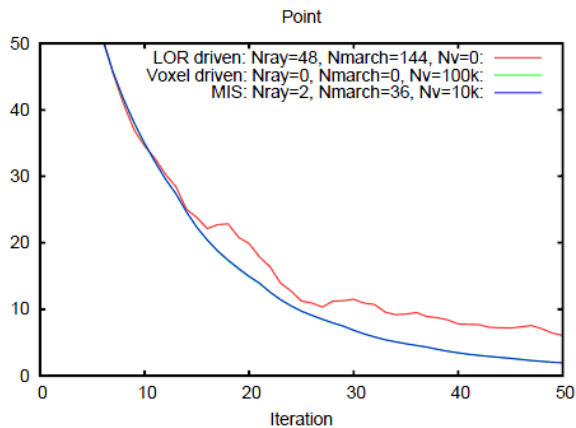
Cons:

- Write collisions
- Less coherence

Multiple Importance Sampling

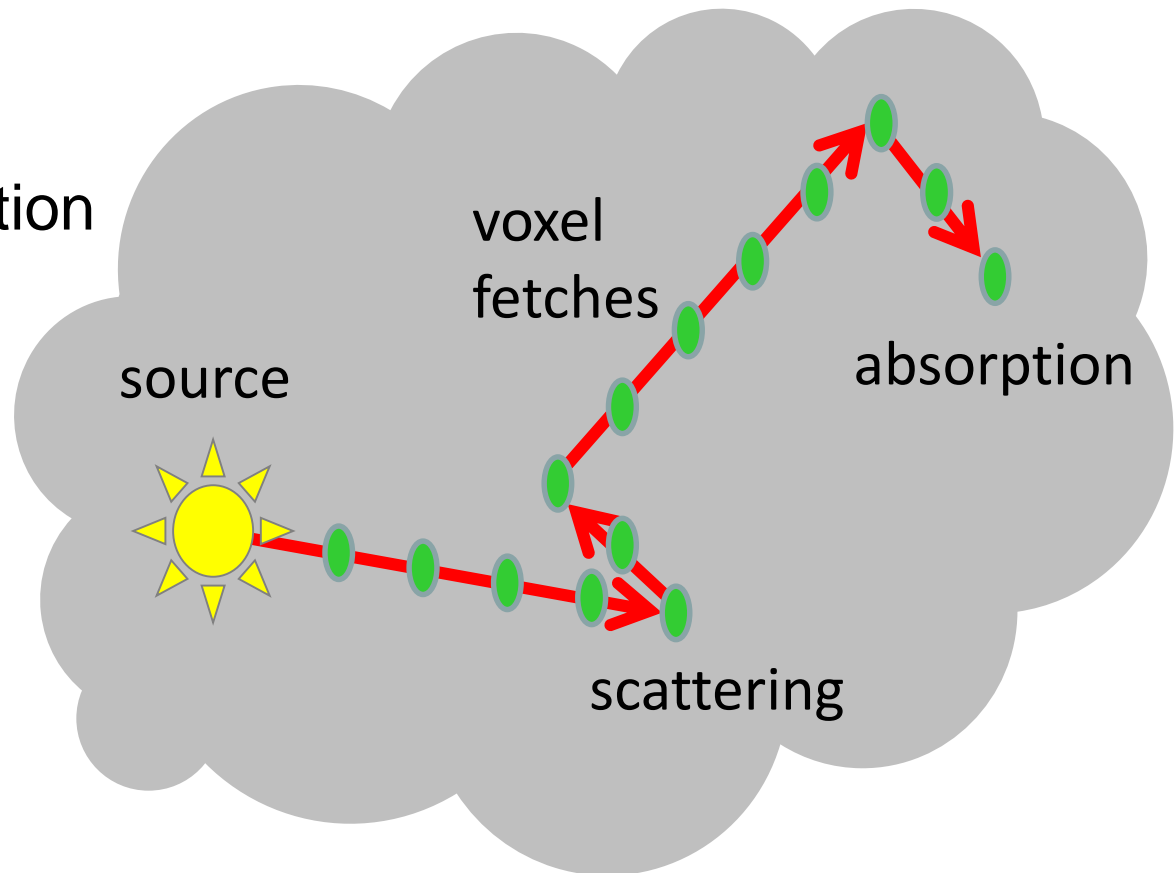


Multiple Importance Sampling

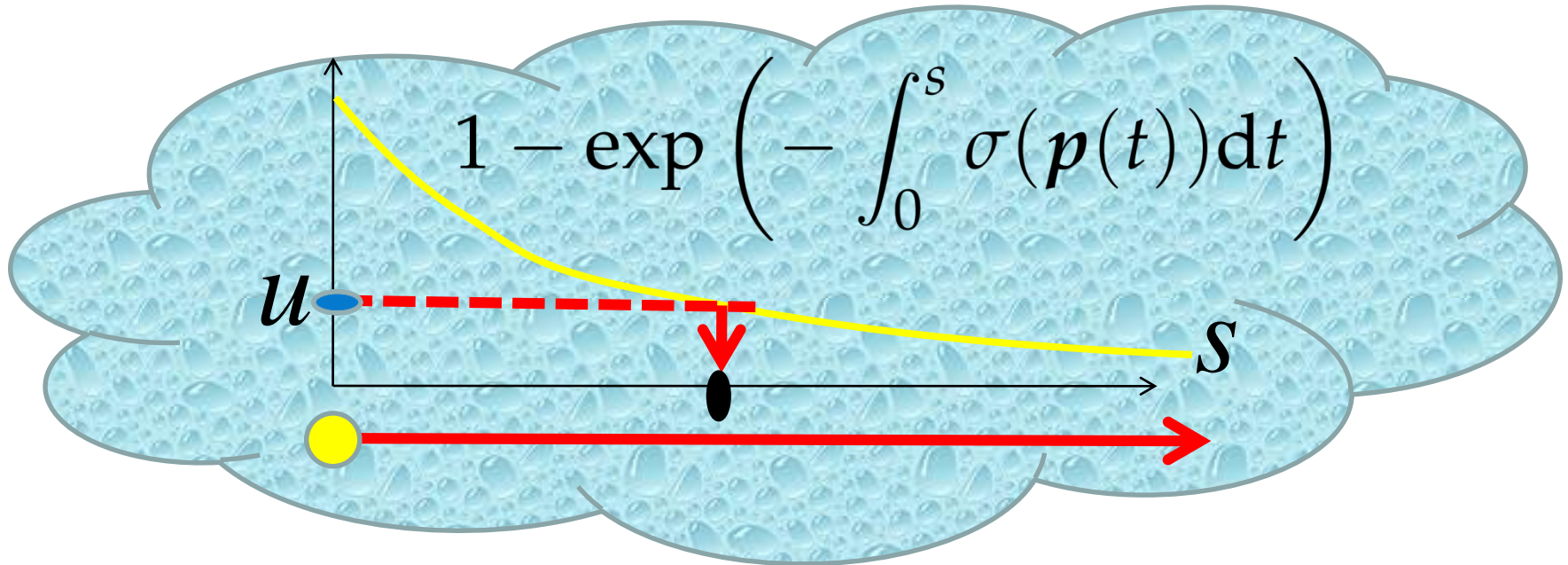


Input-driven scattered photon transport

- Monte Carlo simulation:
 - **Free path**
 - Absorption?
 - Scattering direction



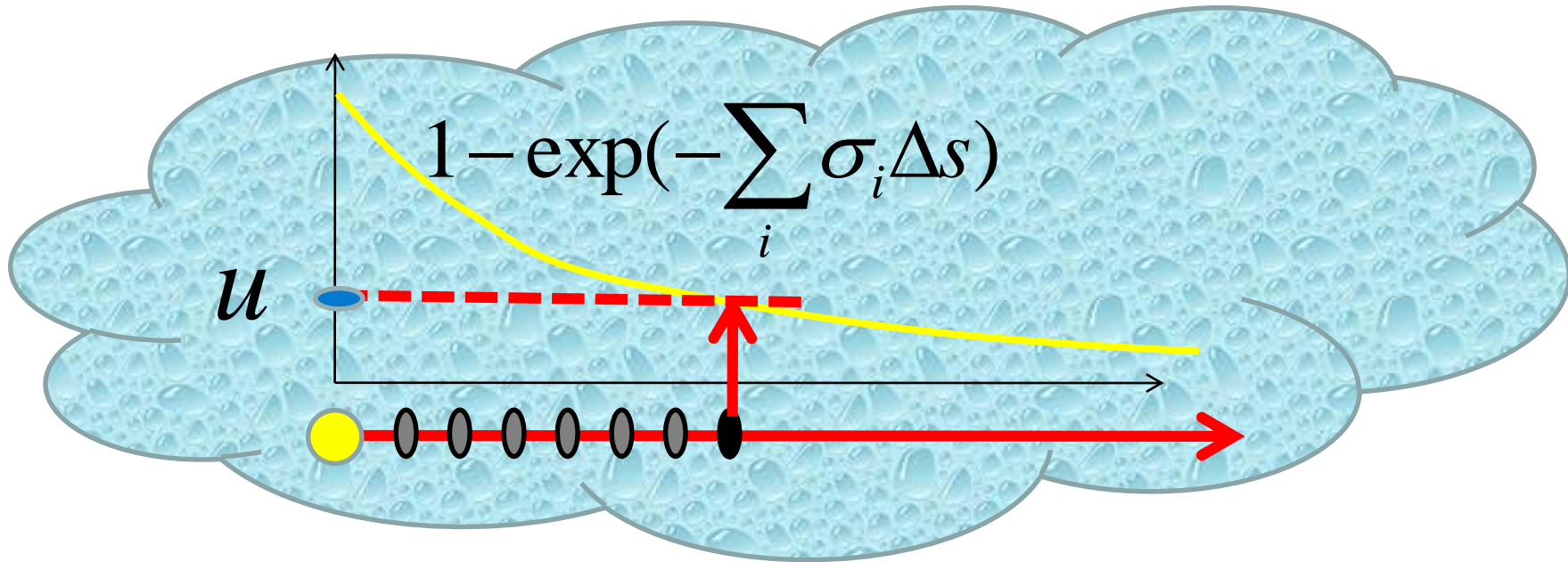
Free Path Sampling



$$u = 1 - \exp\left(-\int_0^s \sigma(\mathbf{p}(t)) dt\right) \longrightarrow s = ???$$

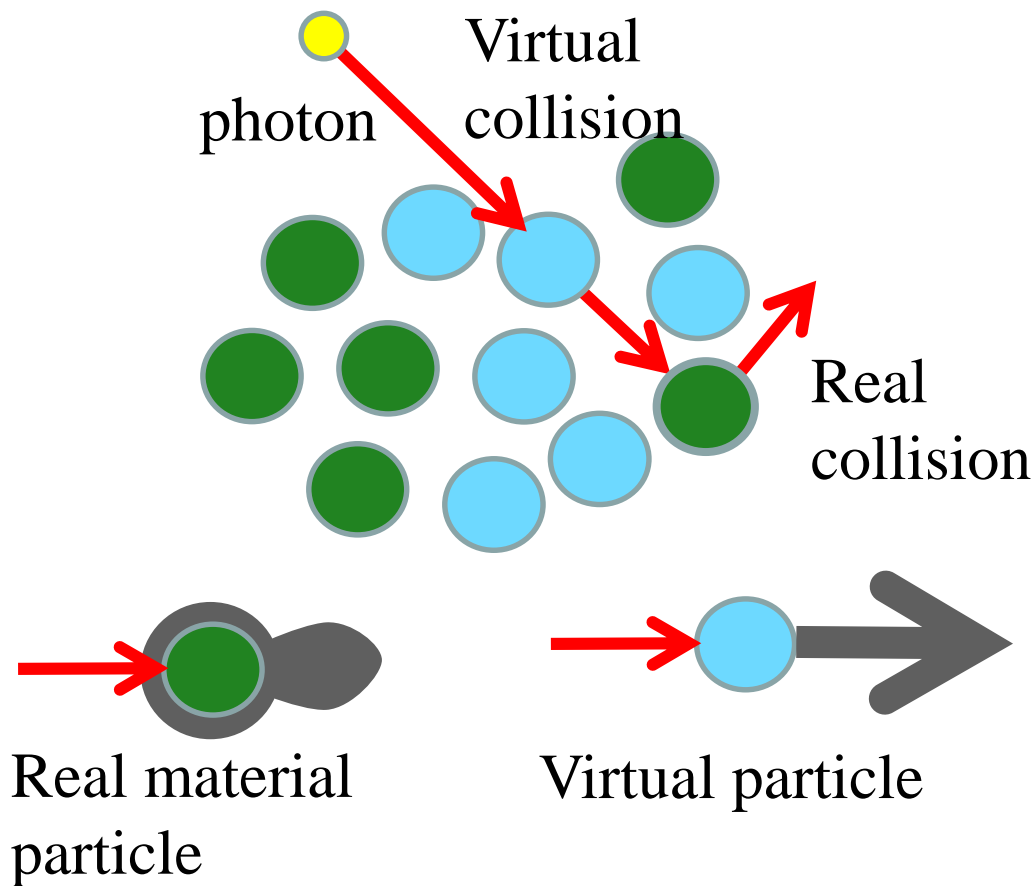
For special cross section functions $\sigma(t)$,
it can be solved analytically.

Ray marching



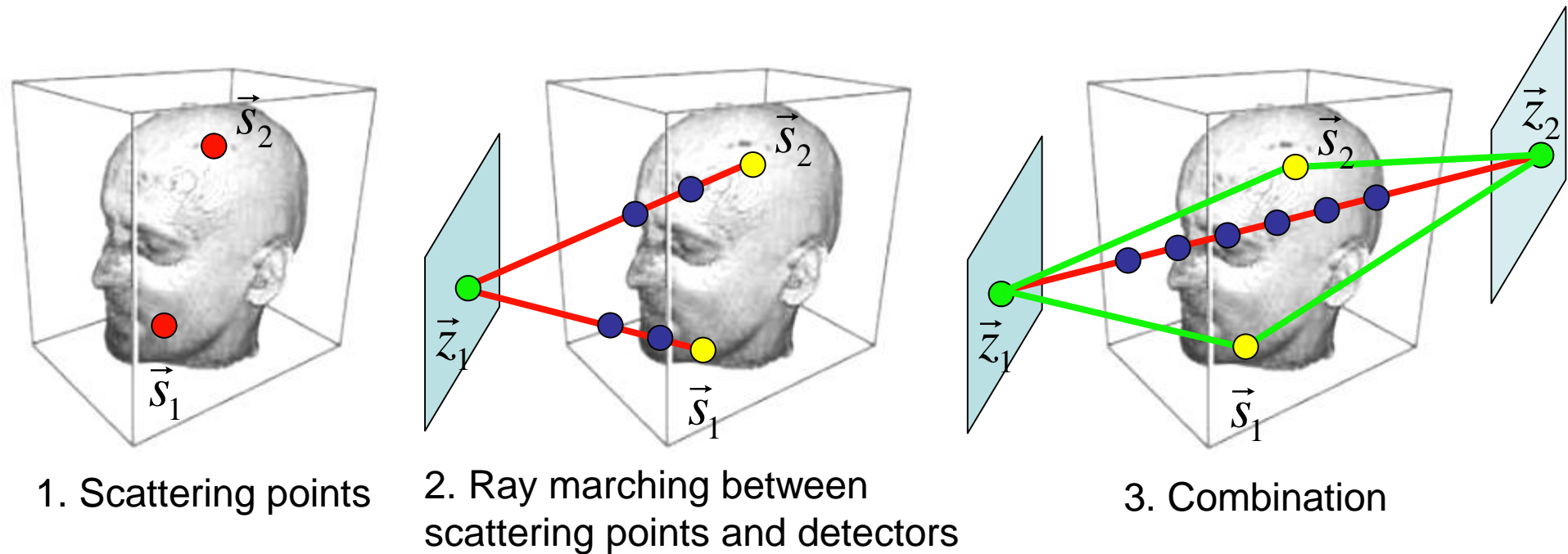
- Complexity grows with the resolution
- Slow in high resolution low density media

Mix virtual particles to obtain a density that can be solved analytically



4096³ effective resolution
64 billion sample points

Output-driven single-scattered photon transport with reuse



Multiple Importance Sampling

0, 1, 2, ...
scattering

0 scattering

0 scattering

1 scattering

LOR-driven
unscattered
contribution

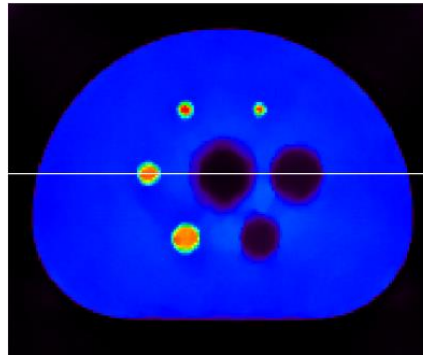
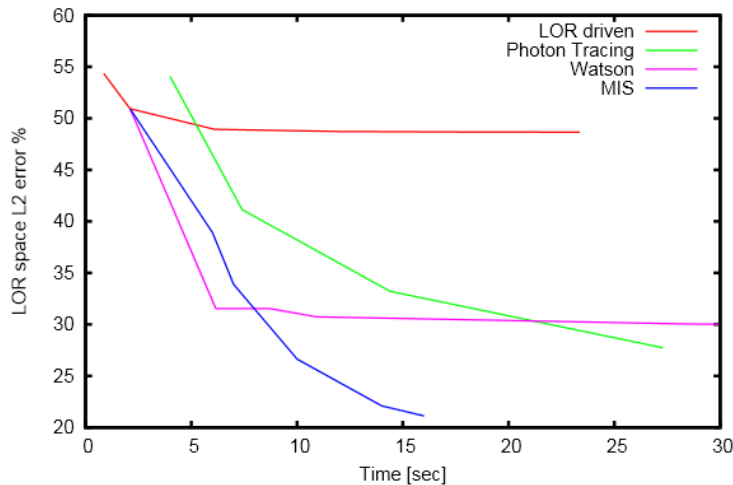
Voxel-driven
unscattered
contribution

Photon
transport

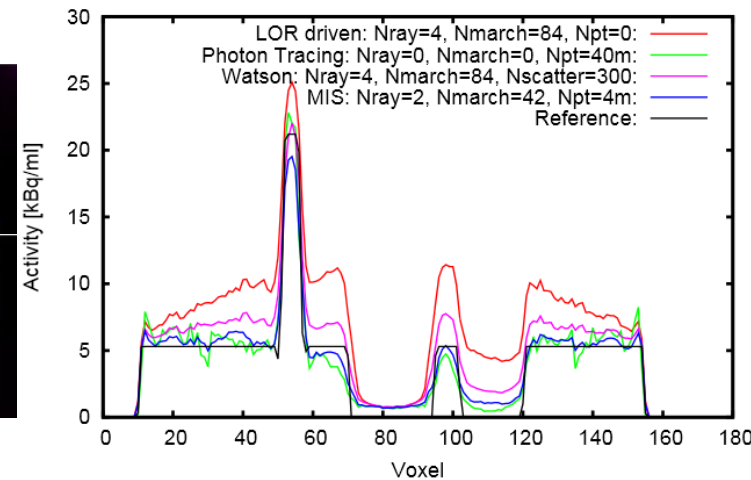
LOR-driven
single scatter

MIS

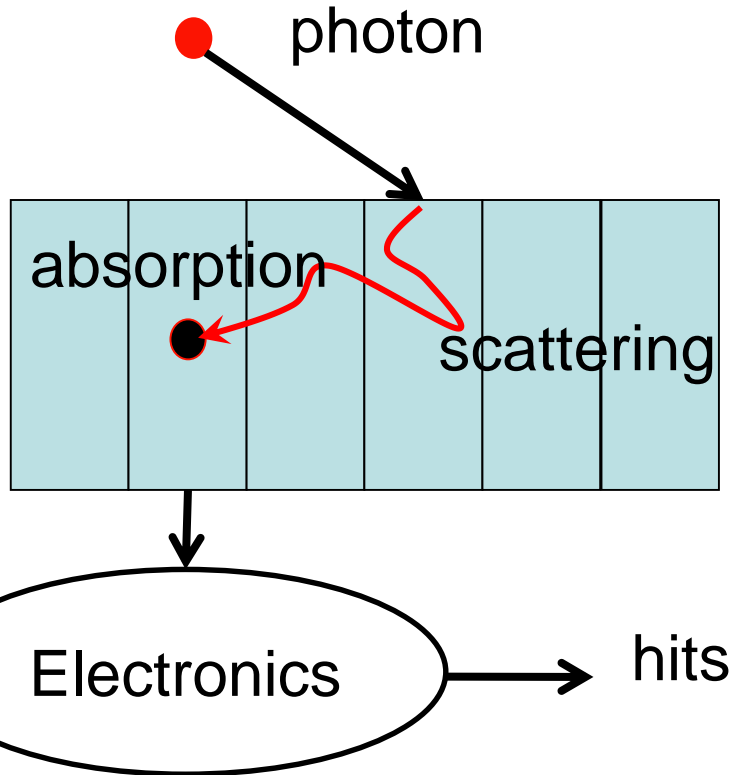
Human IQ



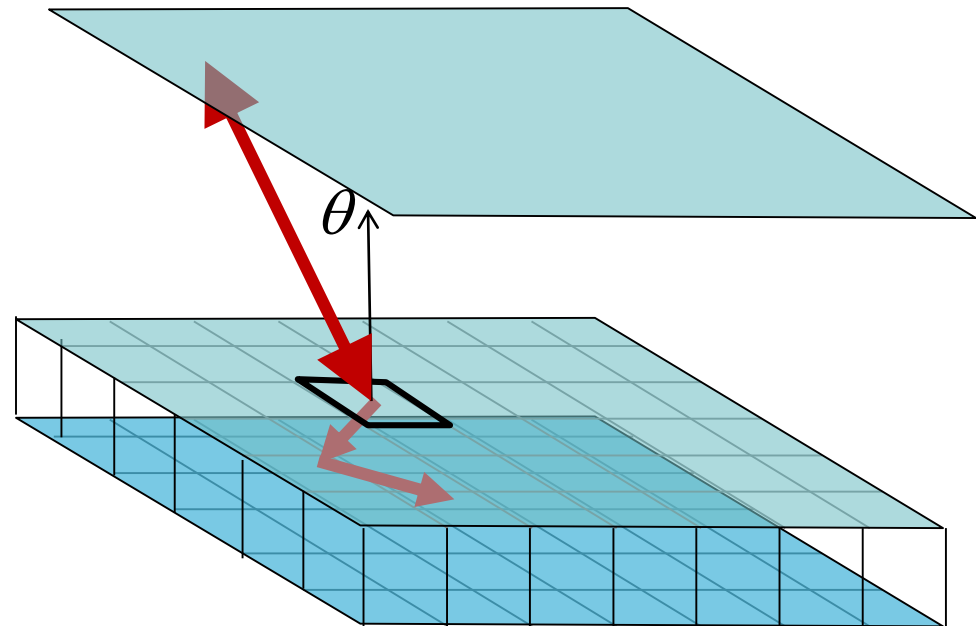
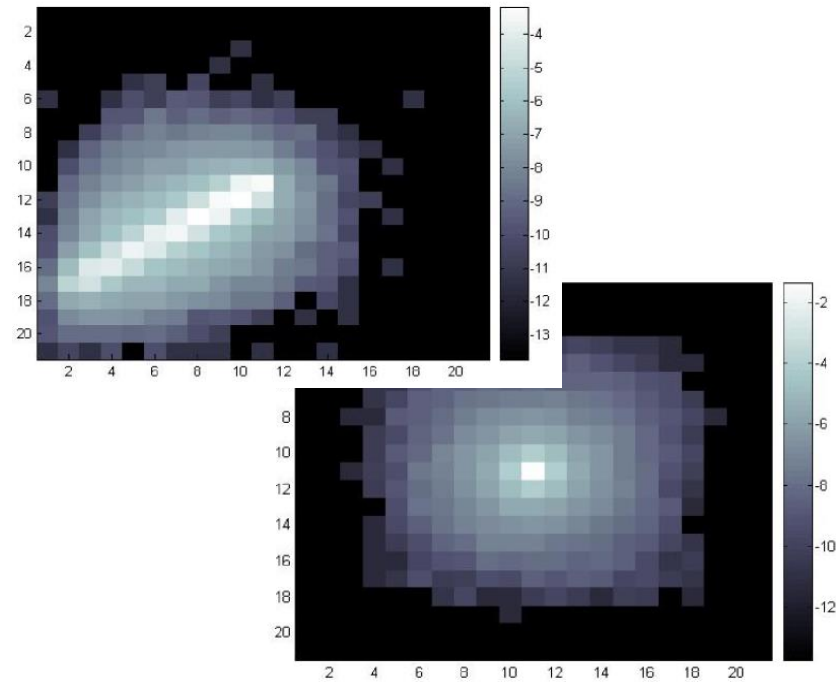
Human IQ



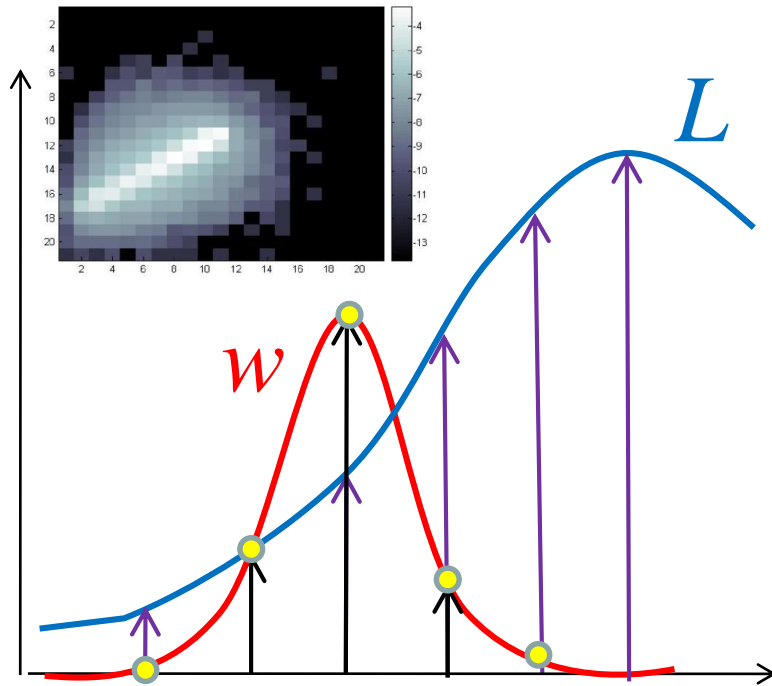
Detector response



Problem:
The domain is 4 dimensional.

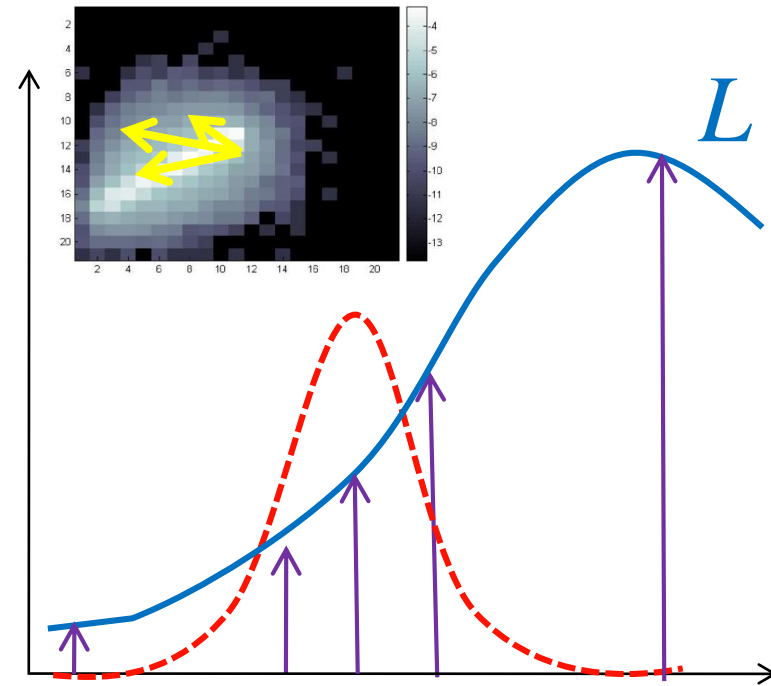


Quasi-Monte Carlo filtering



$$\int L(X - x)w(x)dx$$

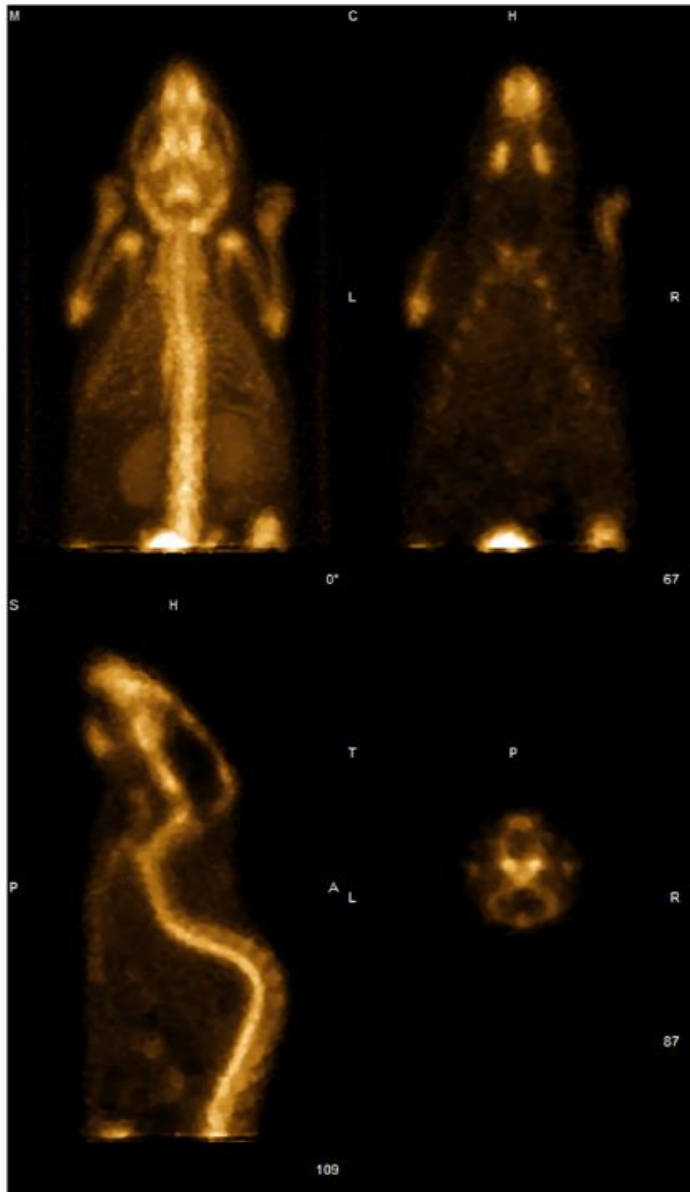
$=$



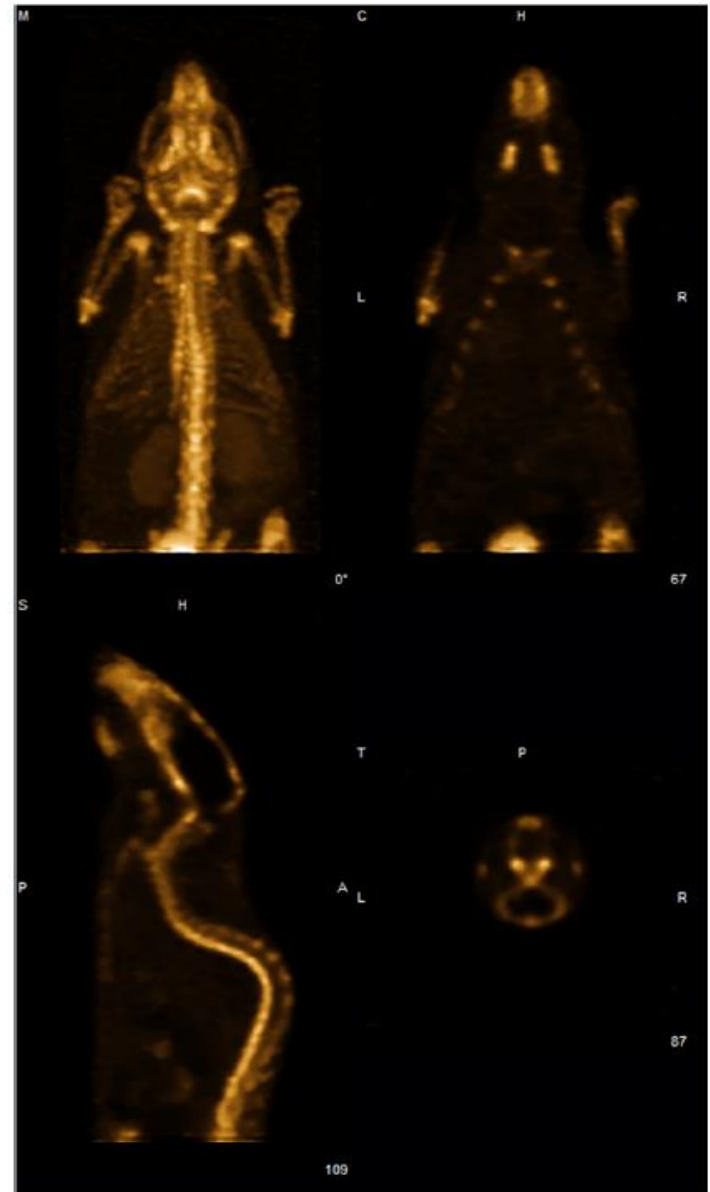
$$\int_0^1 L(X - x(t))dt$$

Detector Scattering Compensation

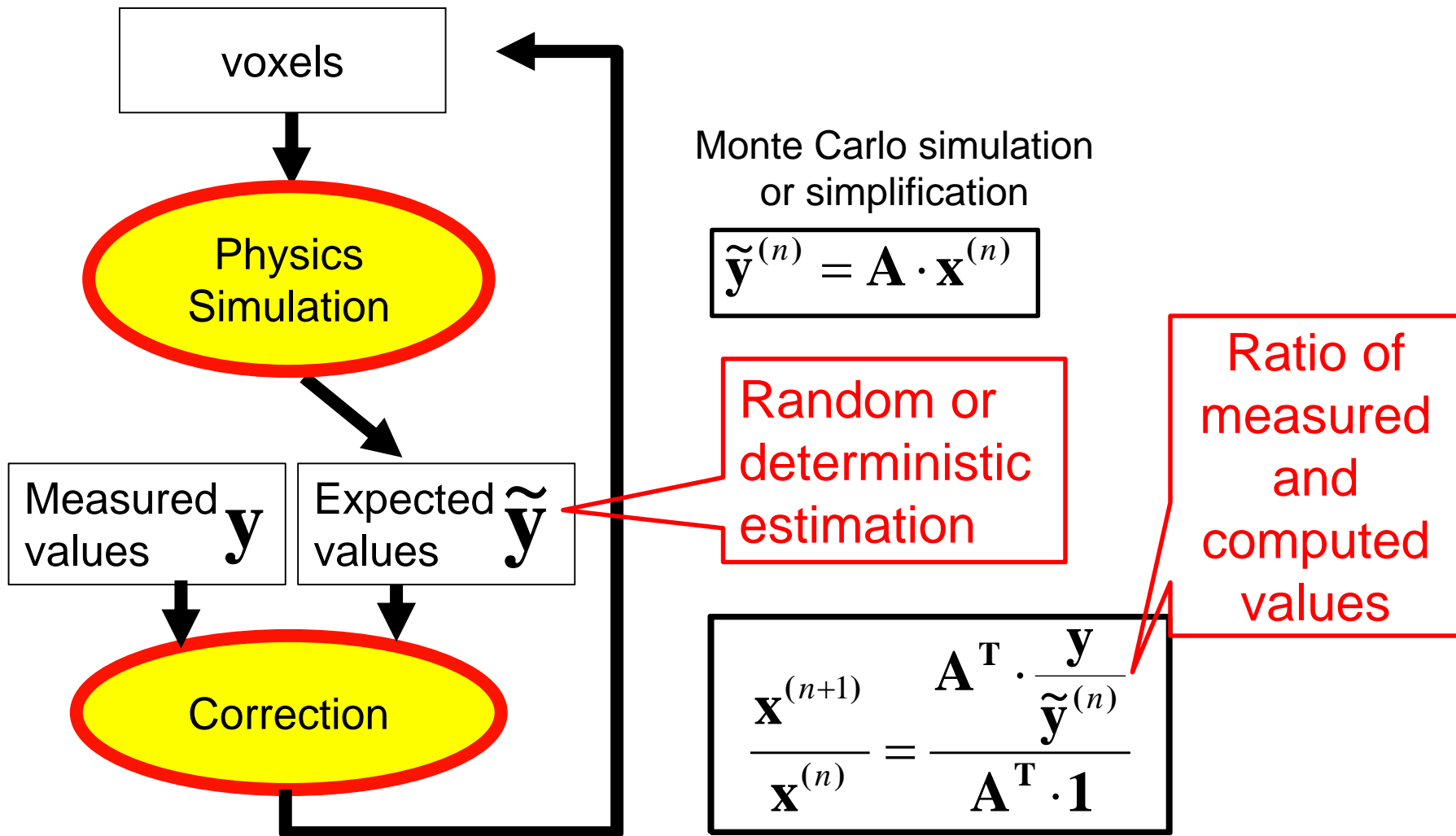
without



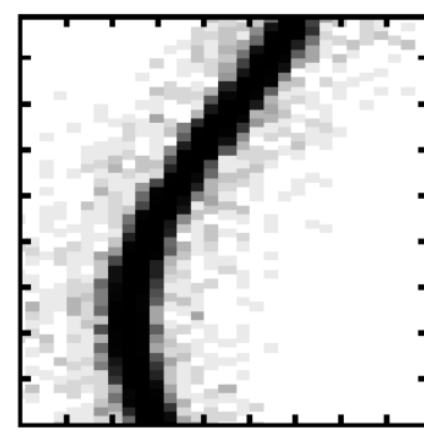
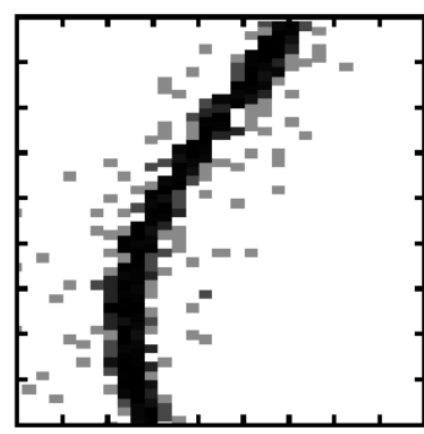
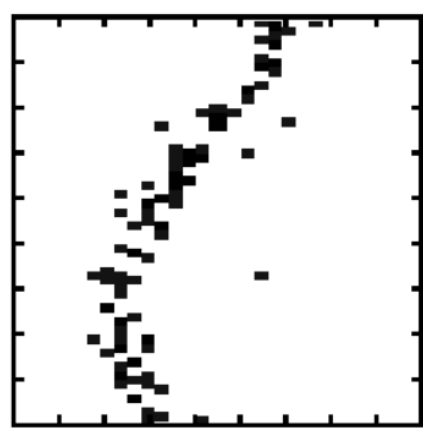
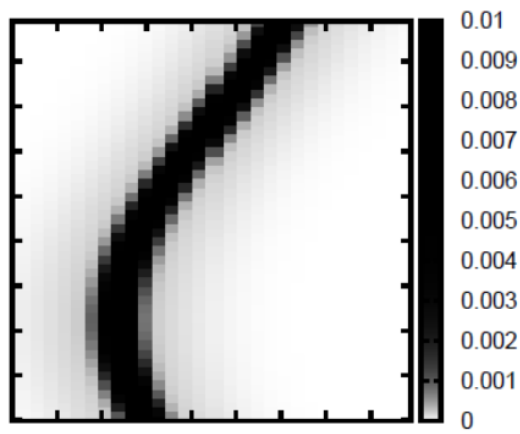
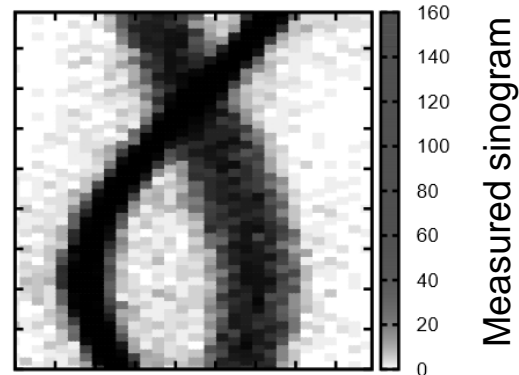
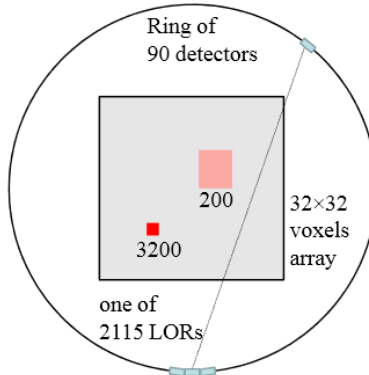
with



Back projection



2D study

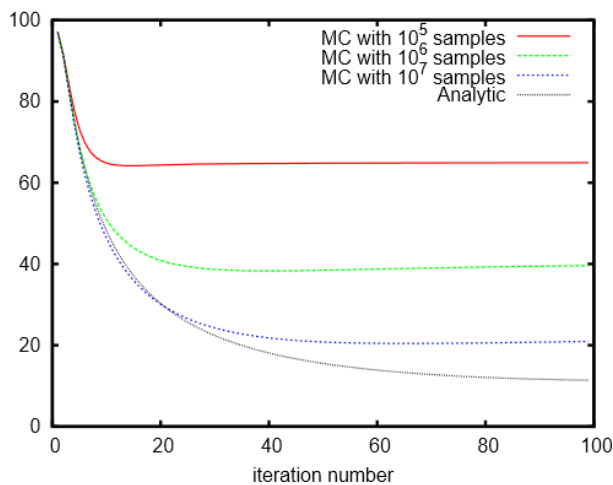


analytic SM

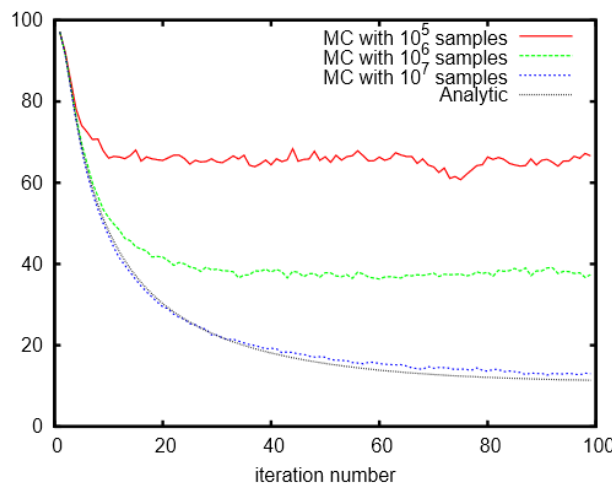
10^5 samples

10^6 samples

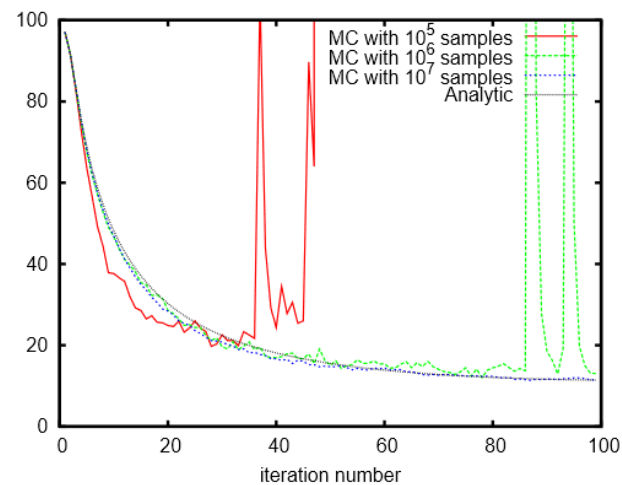
10^7 samples



Fixed

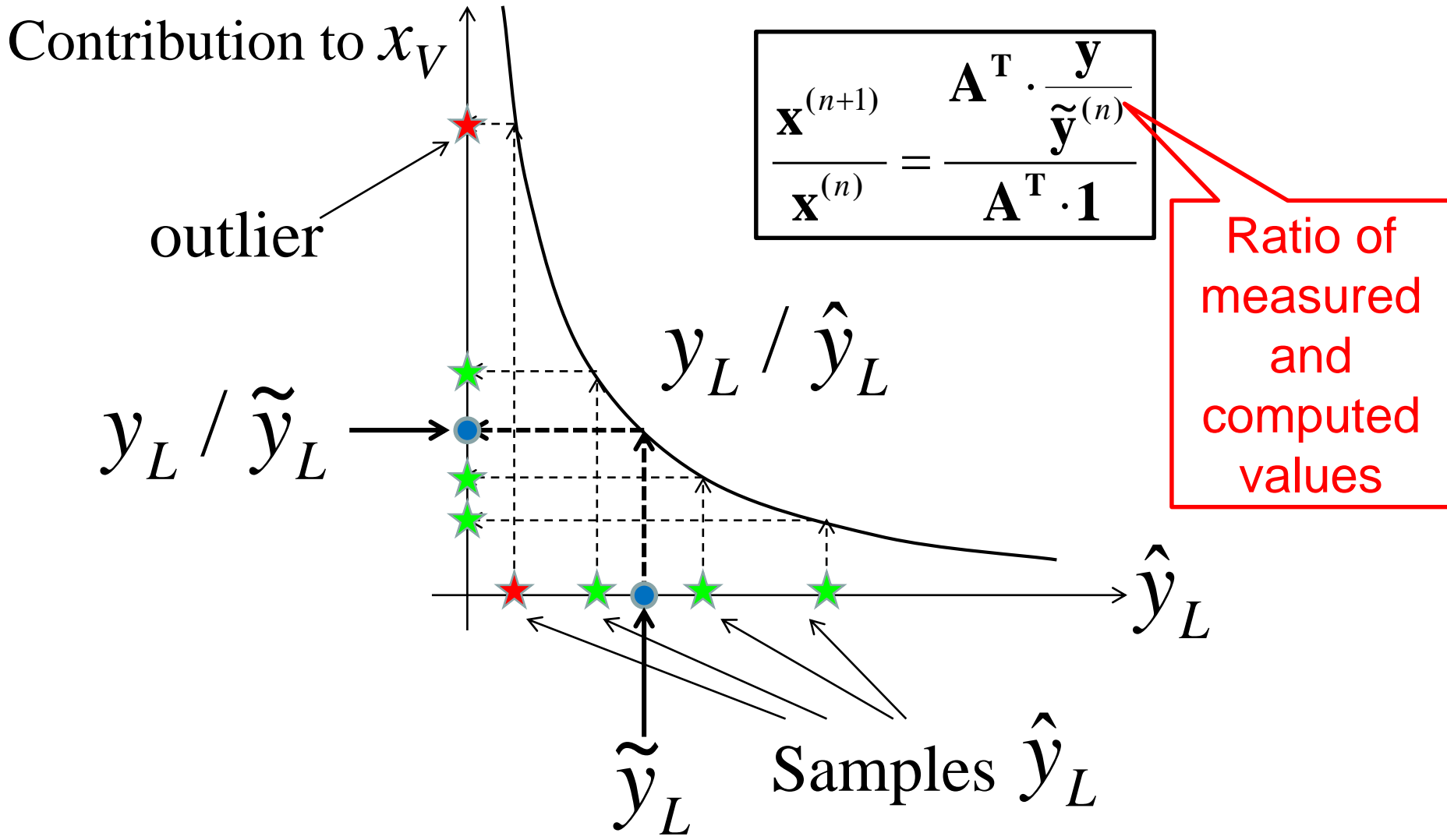


Deterministically matched



Statistically matched

Backprojection with unbiased forward projection



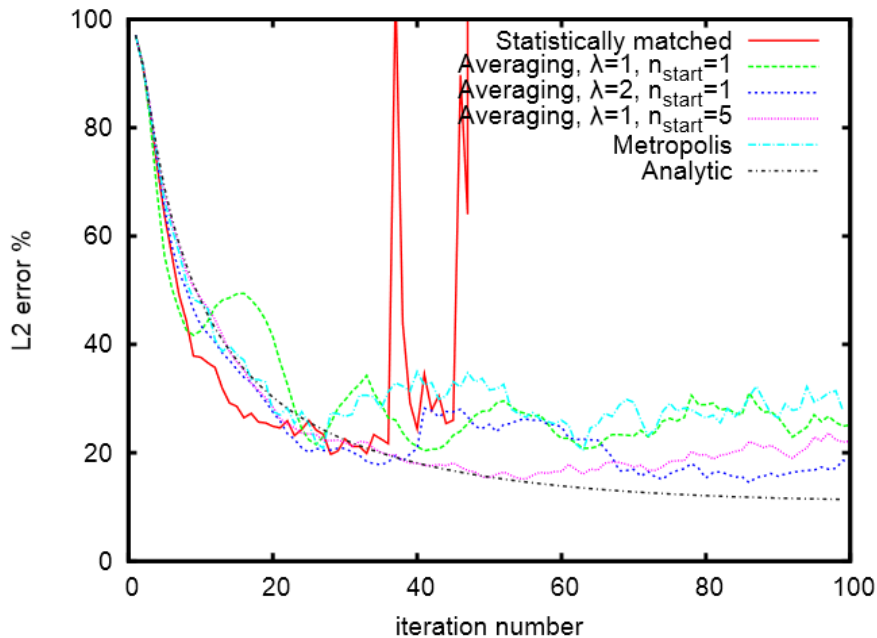
Reduce bias and outliers

- Averaging iteration:

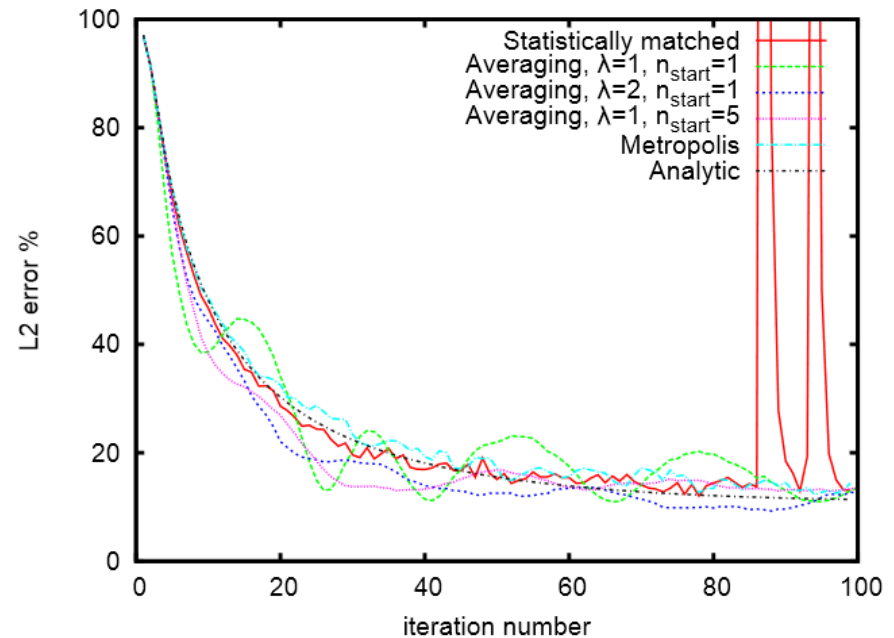
$$\tilde{y}_L^{(n)} = (1 - \tau_n) \tilde{y}_L^{(n-1)} + \tau_n \hat{y}_L \quad \tau_n = \lambda / n$$

- Metropolis iteration: Ignore outliers randomly

Acceptance with probability $a_L = \min\{\hat{y}_L / \tilde{y}_L^{(n)}, 1\}$

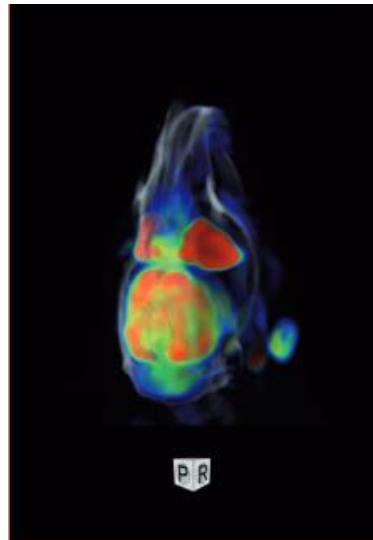
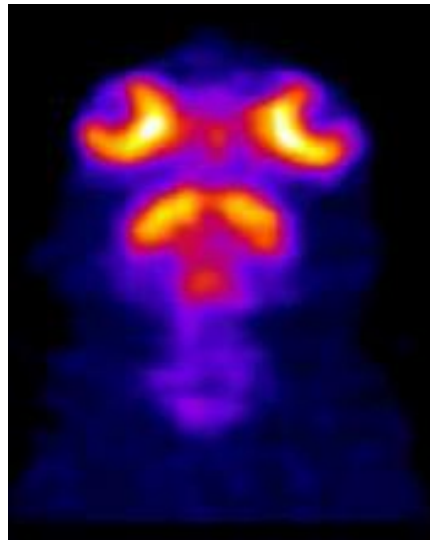


10^5 sample projections

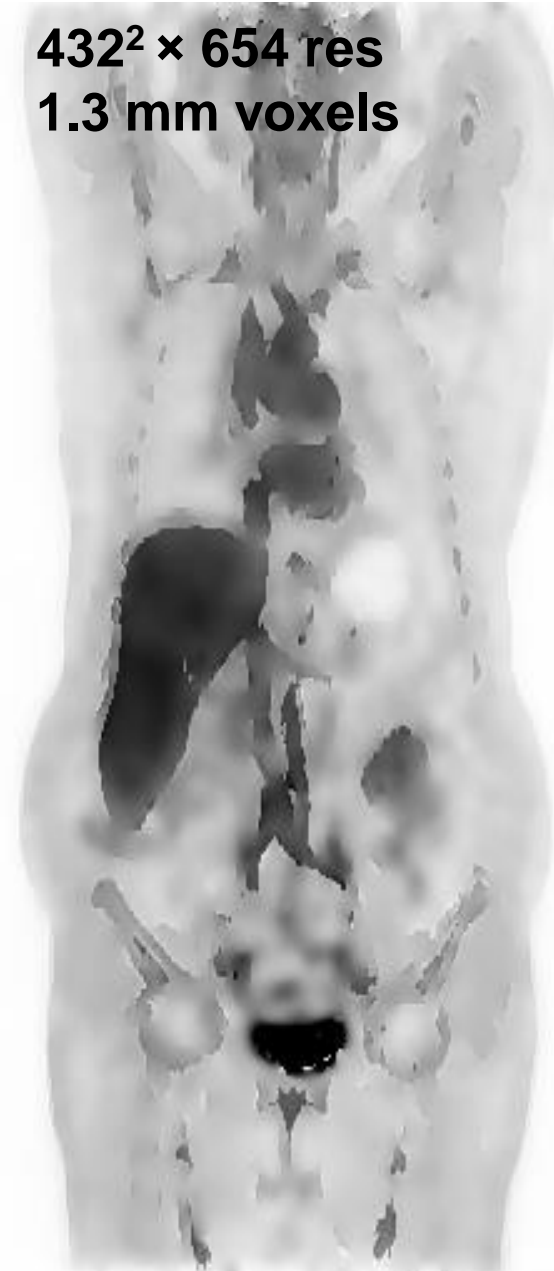


10^6 sample projections

Recons



$432^2 \times 654$ res
1.3 mm voxels



Conclusions

- GPU is an effective tool for computing tens of thousands of parallel threads having no conditionals and collisions.
- The problem must be interpreted and solved to keep this requirement in mind.
- Randomization (Monte Carlo) can help structure the problem in this way.