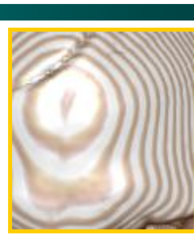
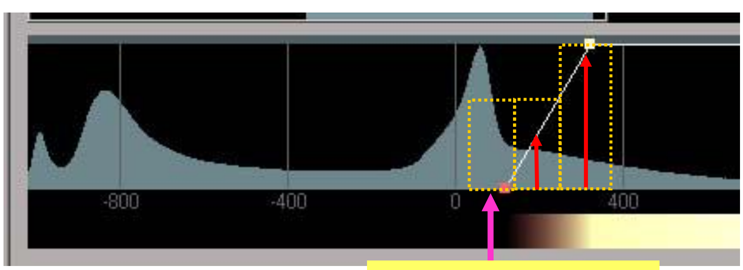
What we will cover

- Contour Tracking
- Surface Rendering
- Direct Volume Rendering
- Isosurface Rendering
- Optimizing DVR
- Pre-Integrated DVR

Image Quality Depends on

- Sampling Rate
- OTF variance
- OTF*Sampling Rate

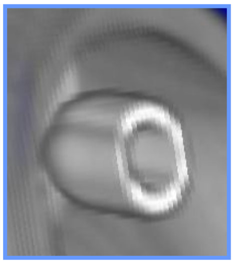


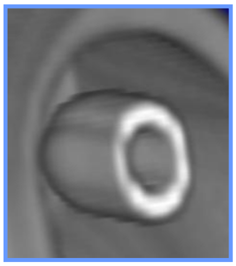


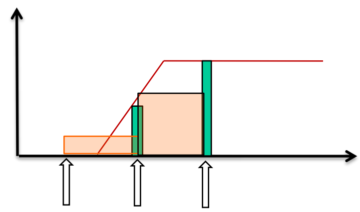
Really nothing?

Ray-Integration

Discrete approximation of volume rendering integral will converge against correct result



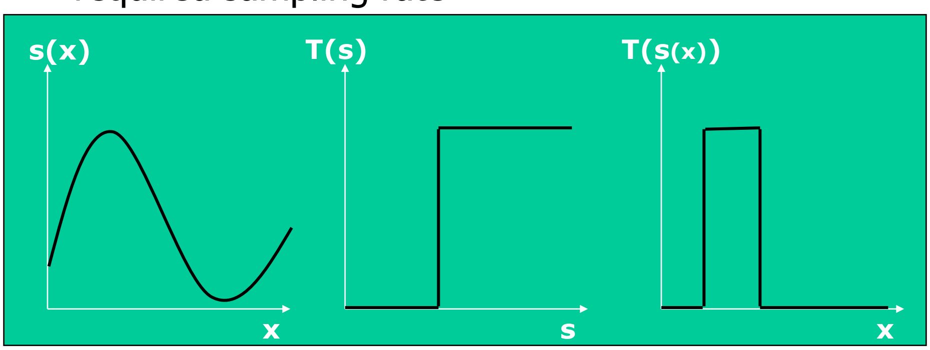




3

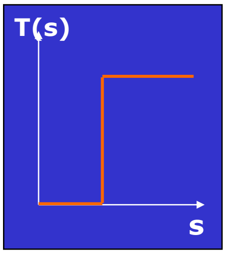
High-Frequency TFs

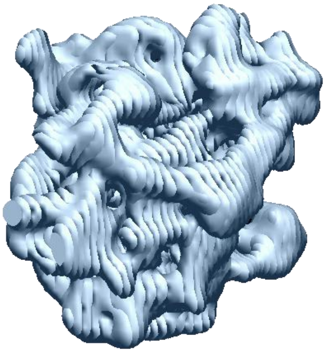
High frequencies in the transfer function T increase required sampling rate



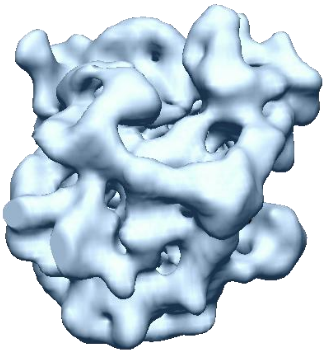
4

High-Frequency TFs





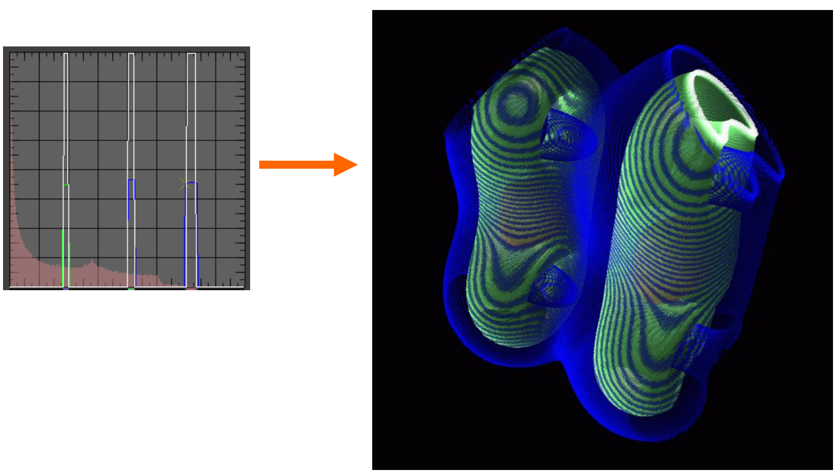
64 data slices



10 times more slices

5

High-Frequency TFs

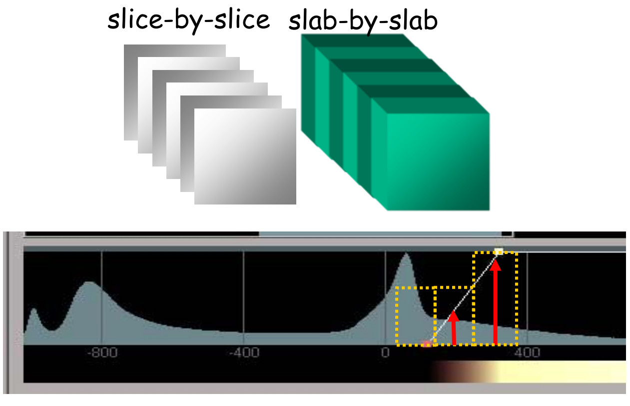


6

Pre-Integrated Volume Rendering

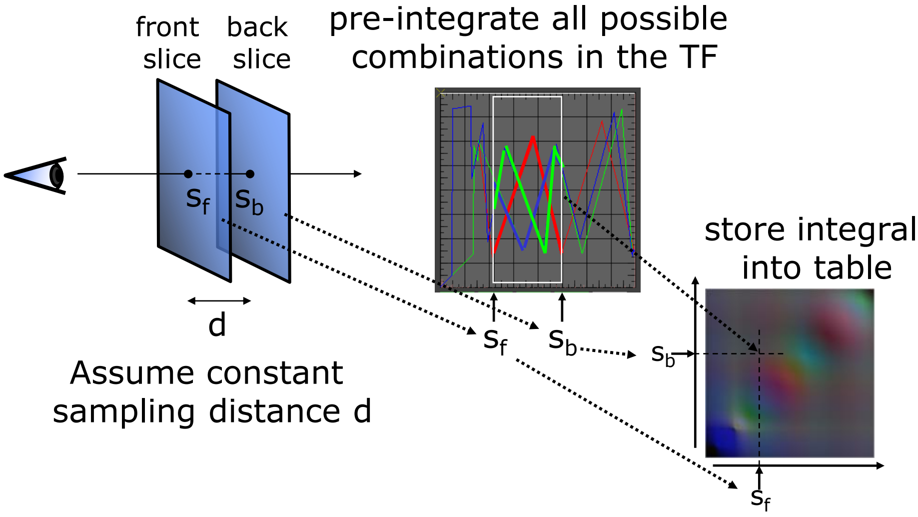
Idea: Pre-Integrated Classification

– slab-by-slab rendering



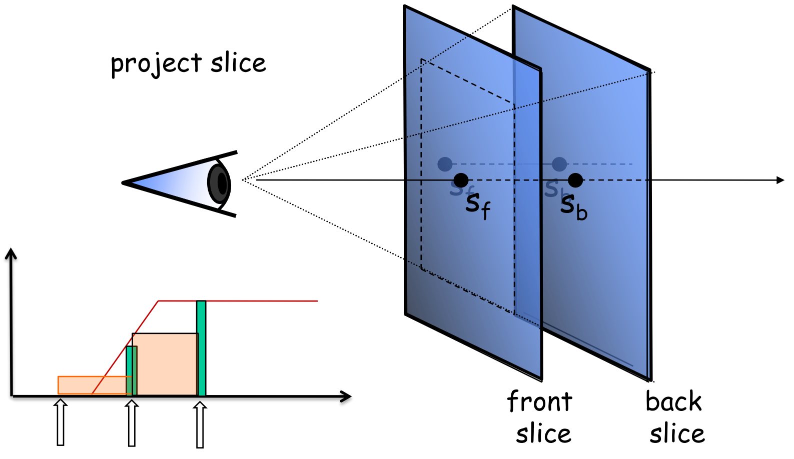
7

Pre-Integrated Classification

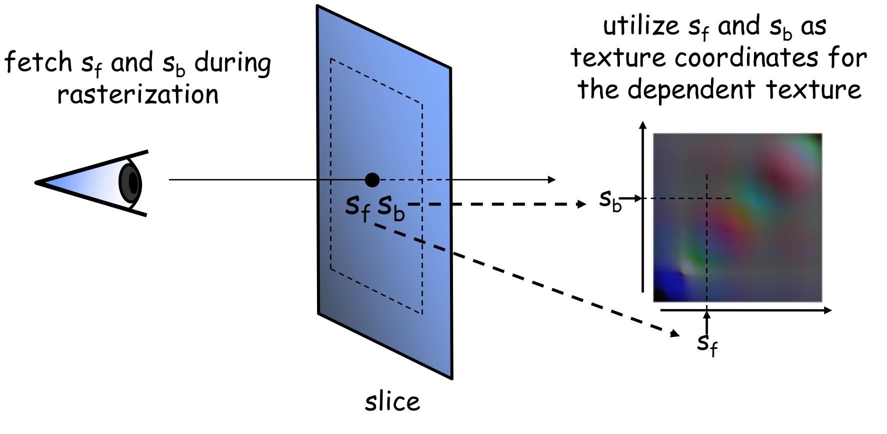


8

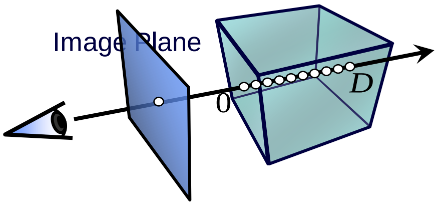
Pre-Integrated Volume Rendering

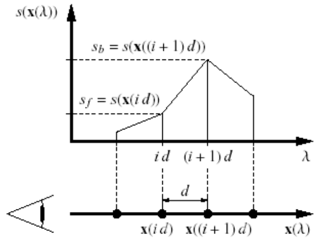


Pre-Integrated Volume Rendering



Pre-integrated Transfer Function





$$I(0,D)=\int_0^D color(x(\lambda))e^{-\tau(0,\lambda)}d\lambda$$

$$=\int_0^D \tilde{c}\left(s(x(\lambda))\right)\exp\left(-\int_0^\lambda \kappa(s(x(\lambda')))d\lambda'\right)d\lambda$$

sampled density

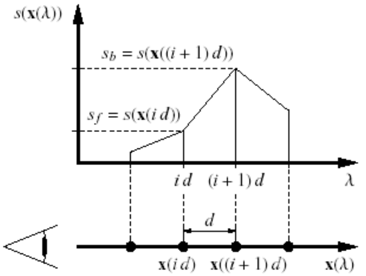
$$\tau(s_1,s_2) = \int_{s_1}^{s_2} \kappa(s)\,ds.$$

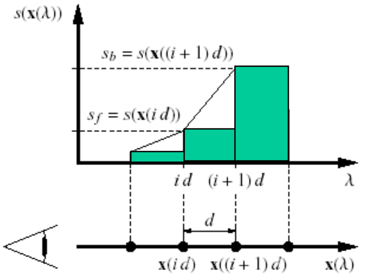
Pre-integrated Transfer Function

Approximate $\exp\left(-\int_0^\lambda \kappa\left(s\left(x\left(\lambda'\right)\right)\right)d\lambda'\right)$ by

$$\exp\left(-\sum_{i=1}^{\lambda/d}\kappa(s(x(i\times d)))d\right)=\prod_{i=1}^{\lambda/d}\exp\left(-\kappa\left(s(x(id))\right)d\right)=\prod_{i=1}^{\lambda/d}(1-\alpha_i)$$

where $\alpha_i \approx 1-\exp(-\kappa(s(x(i\,d)))d) \approx \kappa(s(x(i\,d)))d$





Pre-integrated Transfer Function

$$I(0,D)=\int_0^D \tilde{c}\left(s(x(\lambda))\right)\exp\left(-\int_0^\lambda \kappa\left(s(x(\lambda'))\right)d\lambda'\right)d\lambda$$

By approximating color $\tilde{C}\approx\tilde{c}\left(s(x(id))\right)d$

$$\rightarrow \quad I(0,n)\approx\sum_{i=0}^{n}\tilde{C}_i\prod_{j=0}^{i-1}(1-\alpha_j)$$

Back-to-front compositing

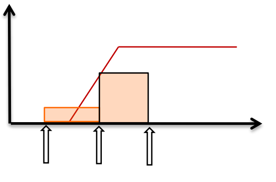
$$\tilde{C}_i'=\tilde{C}_i+(1-\alpha_i)\tilde{C}_{i+1}'$$

13

Pre-integrated Transfer Function

$$\alpha_i=\alpha(s_f,s_b,d)=1-\exp(-\int_{id}^{(i+1)d}\kappa(s(x(\lambda)))d\lambda)$$

$$\approx 1-\exp(-\int_0^1\kappa((1-\omega)s_f+\omega s_b)d\,d\omega)$$



$$\tilde{C}_i=\int_0^1\tilde{c}((1-\omega)s_f+\omega s_b)$$

$$\times\exp(-\int_0^\omega\kappa((1-\omega')s_f+\omega' s_b)d\,d\omega')d\,d\omega$$

We can get $I(0,n)\approx\sum_{i=0}^{n}\tilde{C}_i\prod_{j=0}^{i-1}(1-\alpha_j)$

with pre-computed color and opacity values

14

The approximated opacity of the *i-th* segment

$$\alpha_i = 1 - \exp(-\int_{id}^{(i+1)d} \kappa(s(x(\lambda)))d\lambda)$$

$$\approx 1 - \exp(-\int_0^1 \kappa((1-\omega)s_f + \omega s_b)d\, d\omega)$$

$$= 1 - \exp\left(-\frac{d}{s_b - s_f}\int_{s_f}^{s_b} \kappa(s)ds\right)$$

$$= 1 - \exp\left(-\frac{d}{s_b - s_f}\left(T(s_b) - T(s_f)\right)\right) \quad \text{where, } T(s) = \int_0^s \kappa(s)ds$$

15

The approximated color of the *i-th* segment

$$\tilde{C}_i = \int_0^1 \tilde{c}((1-\omega)s_f + \omega s_b)$$

$$\times \exp(-\int_0^{\omega} \kappa((1-\omega')s_f + \omega' s_b)d\, d\omega')d\, d\omega$$

Self-attenuation within segment

Neglect the self-attenuation

$$\tilde{C}_i \approx \int_0^1 \tilde{c}((1-\omega)s_f + \omega s_b)d\, d\omega$$

$$= \frac{d}{s_b - s_f}\int_{s_f}^{s_b} \tilde{c}(s)ds$$

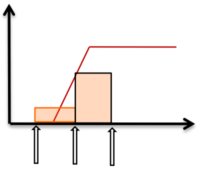
$$= \frac{d}{s_b - s_f}(K(s_b) - K(s_f)) \quad \text{, where } K(s) = \int_0^s \tilde{c}(s)ds$$

16

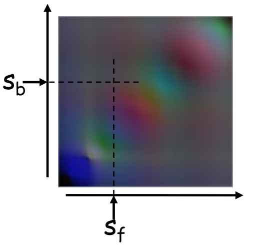
12-bit Image Handling

- 2D Pre-integration table with a fixed *d*
  - 12bits per s_f, s_b
  - Construct Pre-integrated Table requires too much time and space

$$\text{Lookup table}: 2^{12} \times 2^{12} \times (R, G, B, \alpha) \times 8bit = 64MByte$$



$$\alpha_i = \alpha\left(s_f, s_b, d\right) \approx 1 - \exp\left(-\frac{d}{s_b - s_f}\left(T(s_b) - T(s_f)\right)\right)$$



Pre-integrated classification

$$1 - \alpha_i = 1 - \alpha\left(s_f, s_b, d\right) \approx \exp\left(-\frac{d}{s_b - s_f}\left(T(s_b) - T(s_f)\right)\right)$$

$$= \exp\left(-\frac{d}{s_b - s_f}\sum_{s=s_f}^{s_b}\kappa(s)\right) = \sqrt[|s_b - s_f|]{\exp\left(-d\sum_{s=s_f}^{s_b}\kappa(s)\right)}$$

$$= \sqrt[|s_b - s_f|]{\prod_{s=s_f}^{s_b}\exp(-\kappa(s)d)} = \sqrt[|s_b - s_f|]{\prod_{s=s_f}^{s_b}\text{Transparency(s)}d}$$

Note that $\exp(-\kappa(s)d) \approx 1 - \kappa(s)d = \text{transparency}(s)d$

Opacity Computation

Transparency =	0.4 A	0.6 B	0.3 C	0.2 D

S=100   101   102   103   104

When d=1

$$\alpha_i = \alpha\left(s_f, s_b, d\right) = \alpha(100,104,1)\,?$$

$$\alpha = 1 - \sqrt[4]{0.4 \times 0.6 \times 0.3 \times 0.2}$$

$$\alpha_i = \alpha\left(s_f, s_b, d\right) = \alpha(s, s+1,\, 4)\,?$$

When d=4

$$\alpha = 1 - \sqrt[1]{0.4 \times 4}$$

19

1D Pre-Integration Table

- Instead of integral function, we use arithmetic average term for the fast generation of Table
- Use 1D table with extra operation but less memory

$$\text{Lookup table}: 2^{12} \times (R, G, B, \alpha) \times 8bit = 16KBytes$$

$$\alpha(s_f, s_b, d) = 1 - \sqrt[|s_b - s_f|]{\prod_{s=s_f}^{s_b} \exp(-\kappa(s)d)}$$

$$\approx 1 - \frac{1}{\left|s_b - s_f\right|} \sum_{s=s_f}^{s_b} \exp(-\tau(s)d)$$

T=0.4	0.6	0.3	0.2

$$\alpha = 1 - (0.4 + 0.6 + 0.3 + 0.2)/4$$

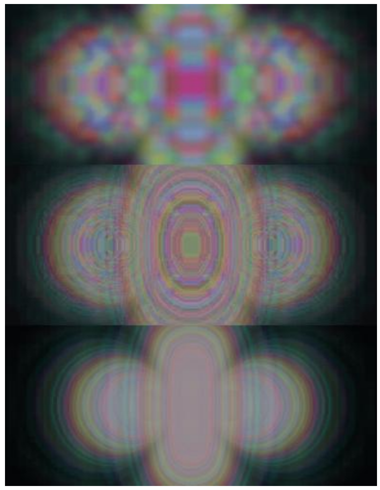
20

Pre-Integrated Volume Rendering

Pre-Classification

Post-Classification

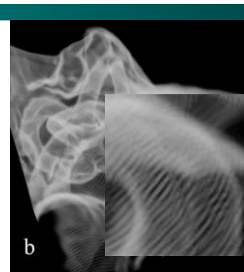
Pre-Integrated-
Classification



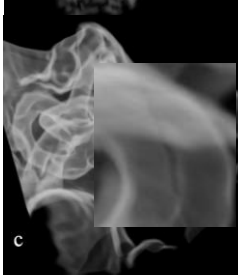
21

Pre-Integrated Volume Rendering

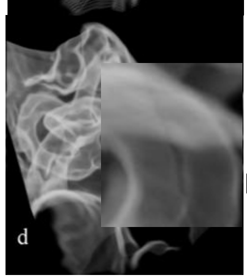
128 slices
pre-
classification



128 slices
post-
classification



284 slices
post-
classification



128 slices
pre-integrated

22

Pre-Integrated Volume Rendering

- Texture-based (2D/3D)
  - Pre-computed ray-segment lookup
  - Dependent texture
- Especially suited for:
  - Low resolution volume data
  - Non-linear transfer functions

Pre-Integrated Volume Rendering

- Reduce compositing time by larger re-sampling interval
- Need to re-compute the pre-integrated table whenever OTF is changed.
- Brings blurring effects
- More compositing voxels (20-30%)