GPU-based Volume Rendering Techniques

Michal Červeňanský
cervenansky@sccg.sk

Outline

- Volume Data
- Volume Rendering (VR)
- Graphics processor unit (GPU)
- VR + GPU
- Classification
- Focus and Context

Volume Data

- Describe interior structures
- Liquids, gases, fire, natural phenomena
- Voxel – intensity, color
- Measurement, simulations
- Medicine, geology, physics, computer sc.
- CT, MRI, confocal microscopy
- 512x512x2000

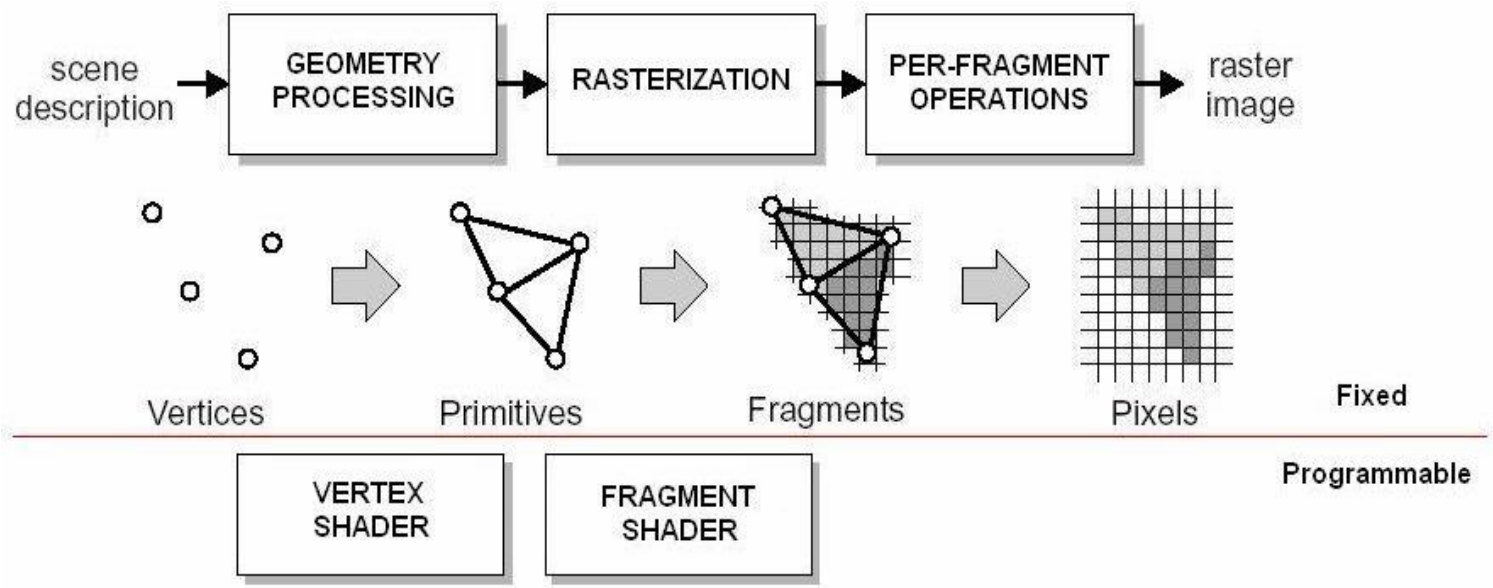
Volume Rendering

- Indirect methods – surface rendering
  - Contour detection
  - Isosurface extraction (marching cubes)
- Direct methods – direct volume rendering
  - Object oriented methods (shear warp)
  - Image oriented methods (ray casting)

- Scalar values → optical properties

GPU

- Fixed pipeline
- Programmable pipeline (vertex, fragment pr.)
- Parallelization – 48 shaders
- GPGPU - General-Purpose Computation Using GPU



GPU - API

- Application programming interface
- Communication: gpu ↔ program
- HLSL, glsl, Cg – shaders languages

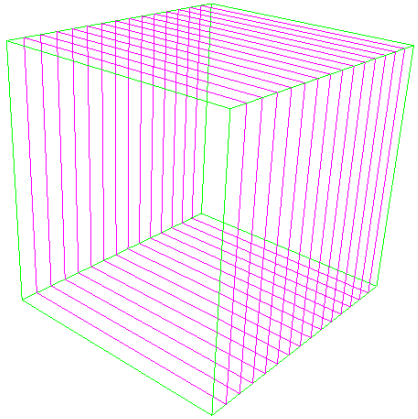
	OpenGL	**Direct3D**
Operating systems	UNIX, Linux, MacOs, OS/2, Windows 95/98/ME Windows NT/2000	Windows 95/98/ME Windows 2000
Programming languages	C, C++, Fortran, Ada Perl, Python, Java	all languages that support Microsoft's COM
Open standard	yes	no (Microsoft proprietary)
Extendable	yes (OpenGL extensions)	no
Pros and cons	⊕ platform independent ⊕ independent standard ⊕ open source reference ⊕ extensions ⊖ rather slow standardization	⊕ fast evolution ⊕ reference rasterizer ⊖ Microsoft proprietary ⊖ restricted to Windows

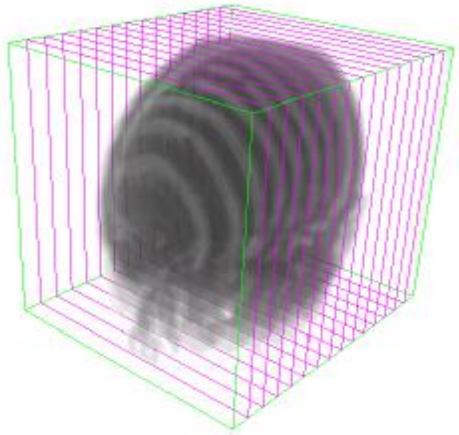
VR + GPU
Outline

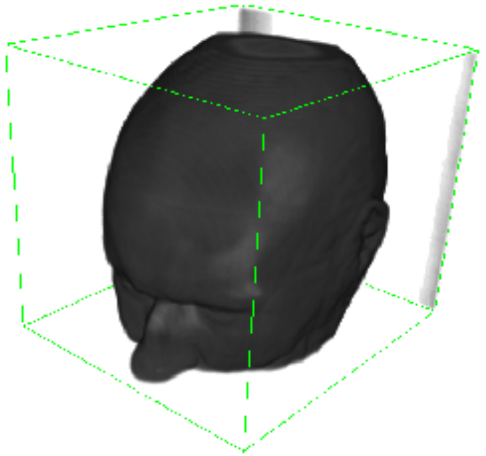
- 2D textures
- 3D textures
- Ray-casting
- Compositing, lighting
- Classification

2D textures

- Polygonal primitives
- Object aligned slices



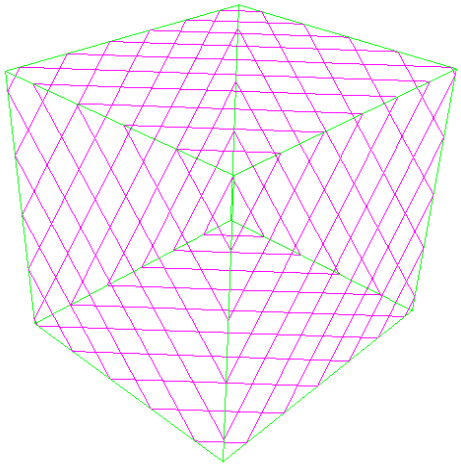


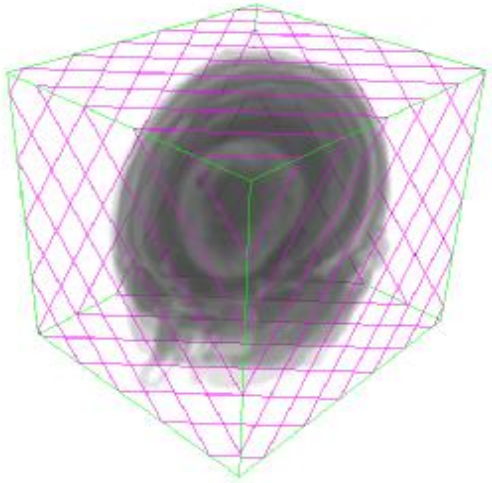


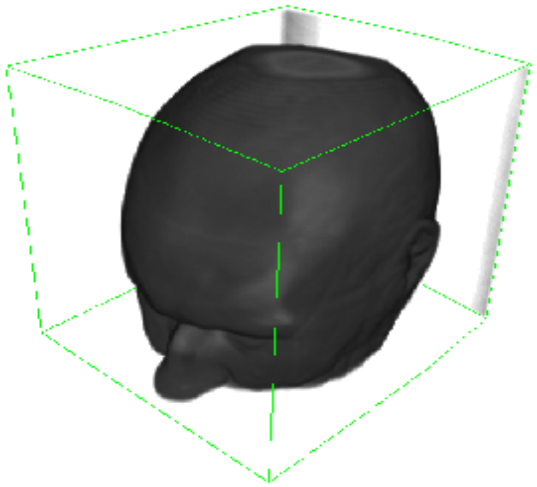
2D-Texture-Based Approach	
Pros	Cons
⊕ very high performance ⊕ high availability	⊖ high memory requirements ⊖ bilinear interpolation only ⊖ sampling artifacts ⊖ switching effects ⊖ inconsistent sampling rate

3D textures

- View aligned slices



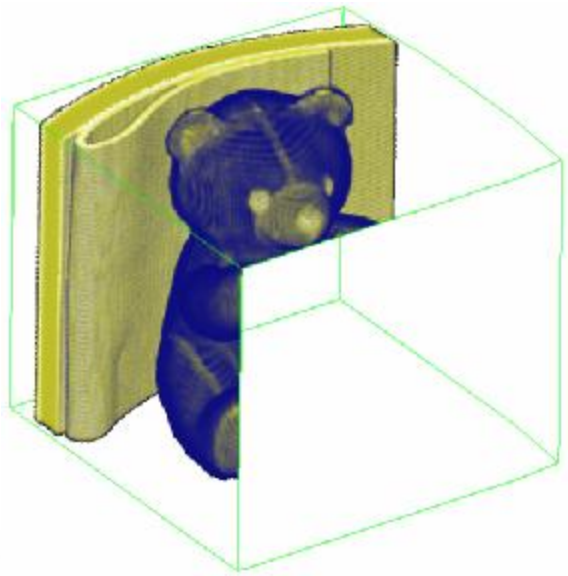


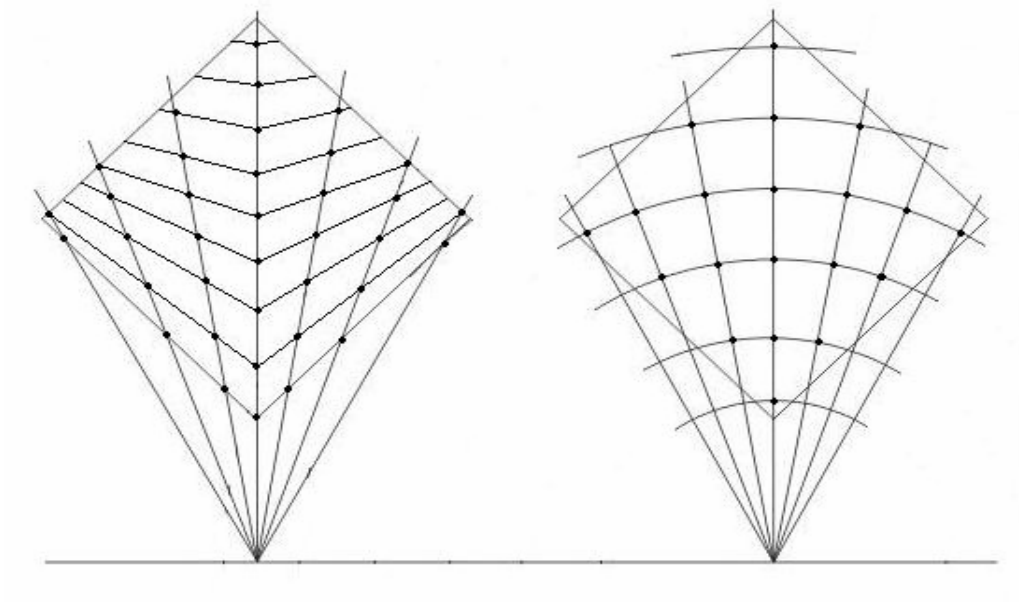


3D-Texture-Based Approach	
Pros	Cons
⊕ high performance ⊕ trilinear interpolation	⊖ availability still limited ⊖ inefficient memory management ⊖ inconsistent sampling rate for perspective projection

Ray-casting

- One pass rendering (sm 3.0)
- BBox faces rendered
- Early ray termination
- Empty space skipping
- Reflection, refraction
- Correct ray start positions





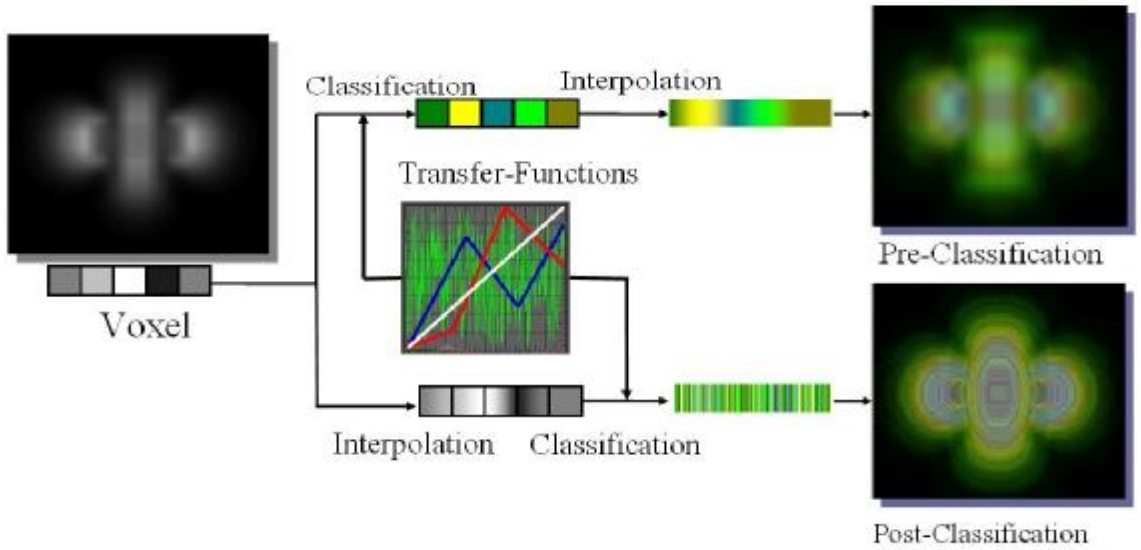
Compositing, Lighting

- Blending – back to front
- Blending equations – add, maxip, minip
- C' = (Cs * As) + (Cd *(1-As))
- RGBA – colour, transparency

- Lighting model in fp (Phong)
- Gradient - on the fly / pre-processing

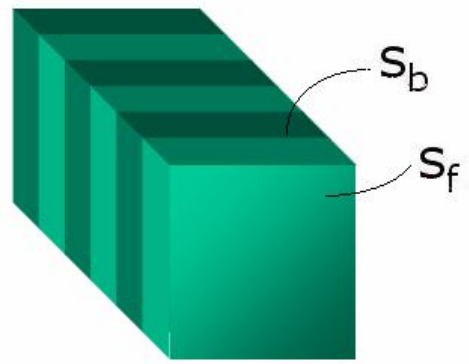
Classification

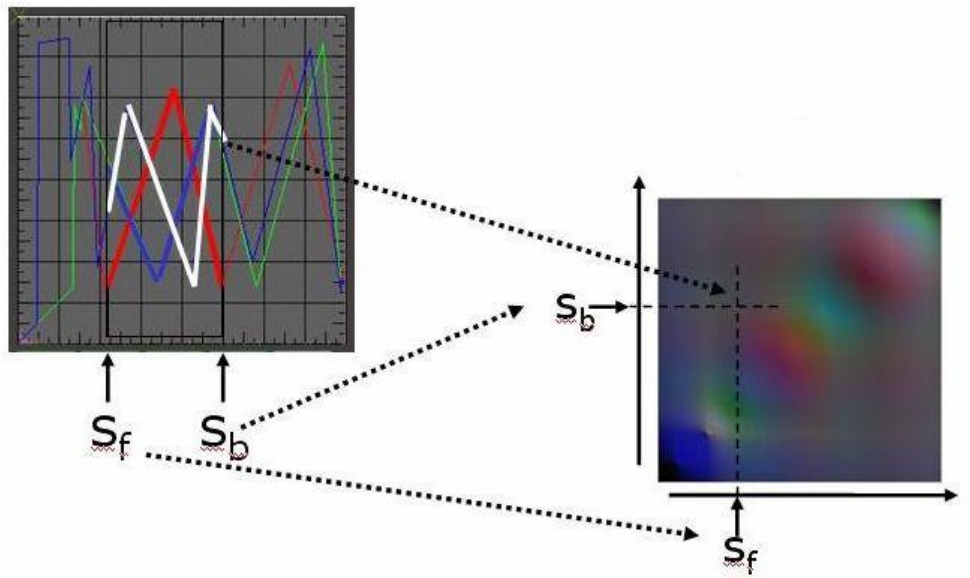
- Emission, absorption → colour values
- Transfer functions → modify colour, transp.
- Underline special properties in data
- One/Multi-dimensional (intensity, gradient, …)
- Pre-classification → before interpolation
- Post-classification → after interpolation



Classification pre-integration

- Maping after interpolation
- Slab by slab rendering
- Pre integration of all possible combinations
- Lookup table

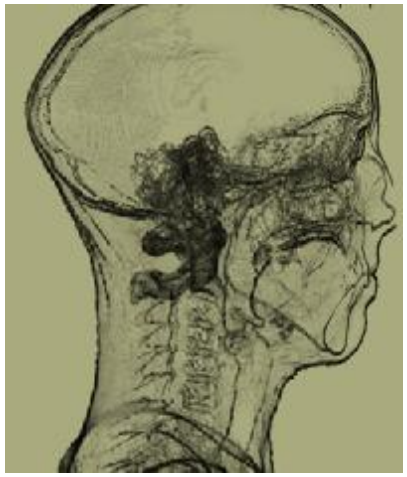




Focus and Context outline

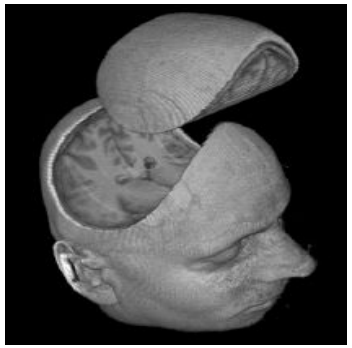
- Overview
- Main idea
- Footprint
- Filtering
- Rendering
- Future work

Overview



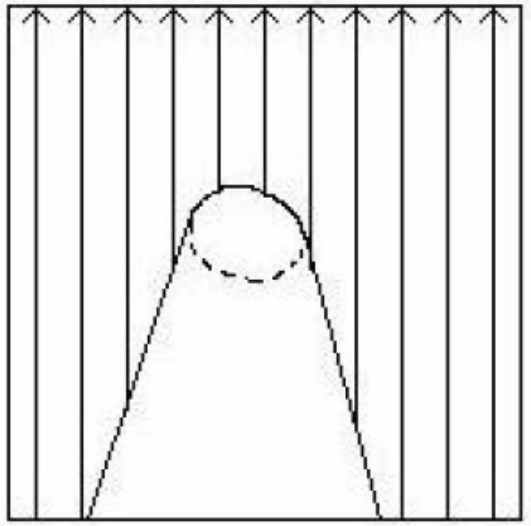
- Important parts – focus
- Dataset - context

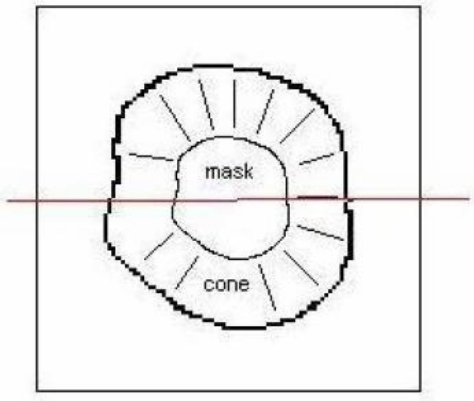
- Transfer functions → difficult design
- Illustrative VR, non-photorealistic VR
- Volume clipping

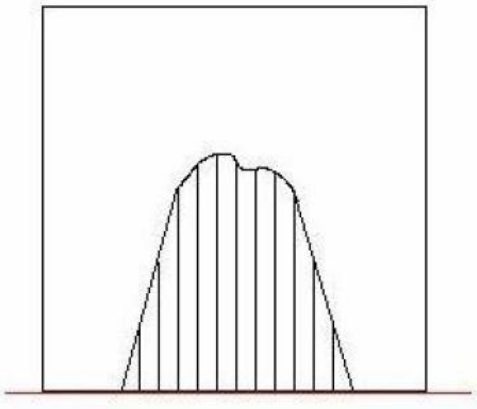


Main idea

- Segmented data (mask) – focus
- Clipping extended cone – hole
- Based on ray-casting
- Shift ray start point
- Height map

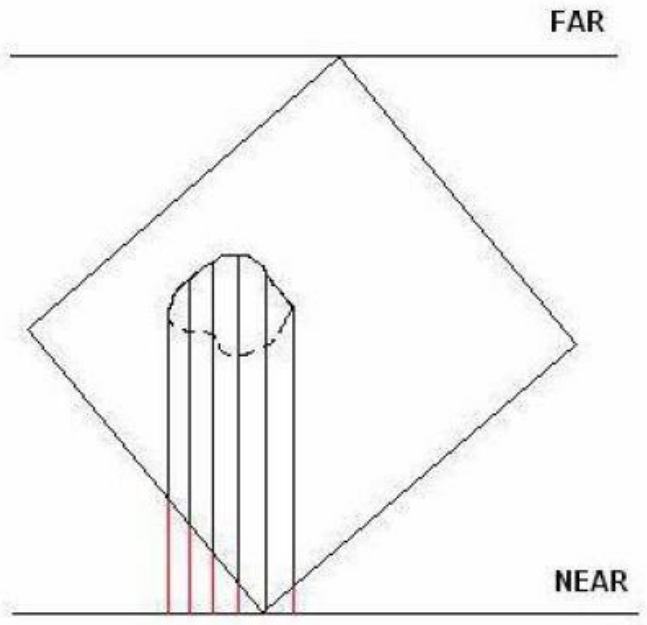


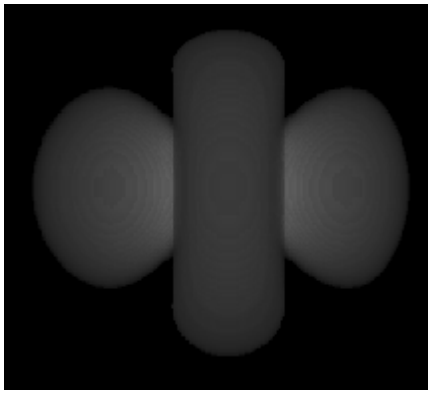


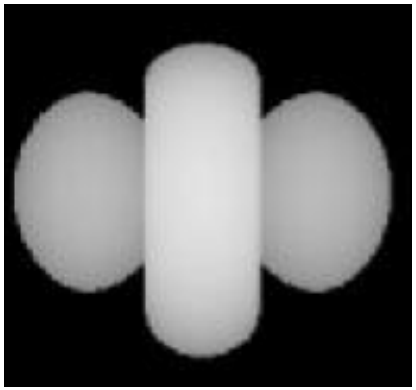


Footprint

- Based on ray-casting
- Last hit
- Length of ray
- Added |BB – near|
- Height map

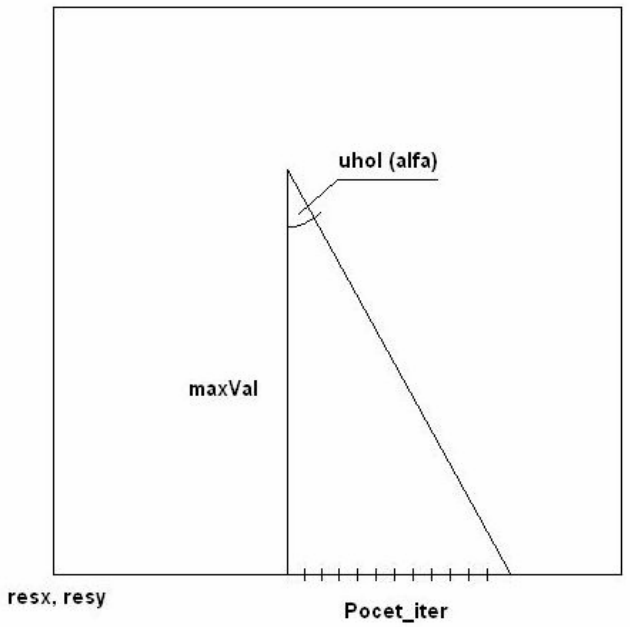






Filtering

- Clipping cone → filtering
- Filtering → footprint expansion, attenuation
- Angle:
  - Cone largeness
  - Number of iteration
- Decrement
  - shortening factor
- maxVal → glReadPixels()

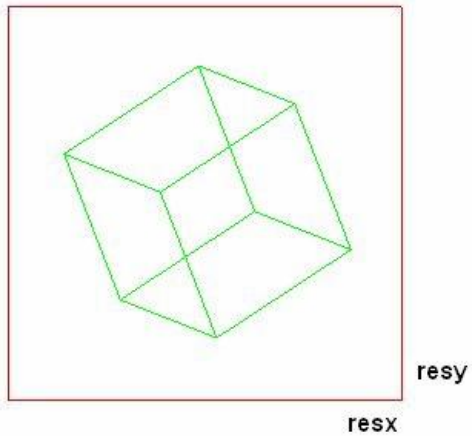


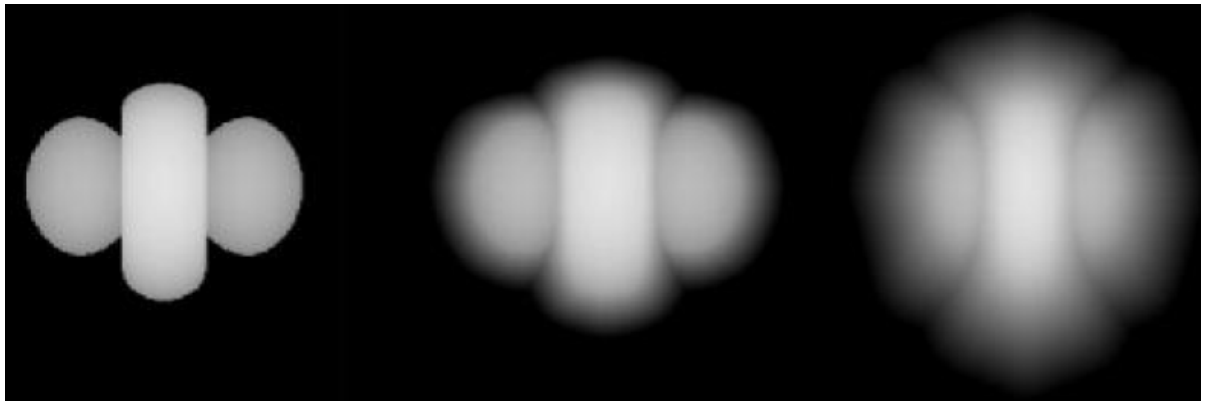
Filtering (2)

- numIter = maxVal*tan(alpha)*max(resx,resy)
- decrement = maxVal / numIter
- res, resy – POT
- Loop
  - filter height map
  - copy to texture
- Filter – footprint expansion to all directions

Dekrement *

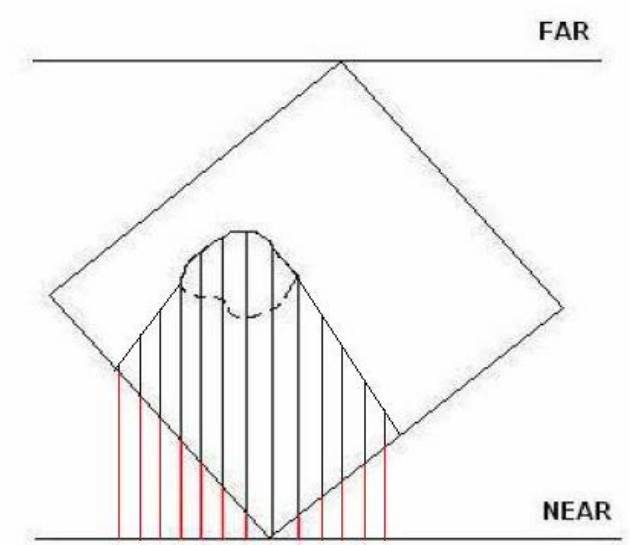
$-\sqrt{2}$	-1	$-\sqrt{2}$
-1	0	-1
$-\sqrt{2}$	-1	$-\sqrt{2}$

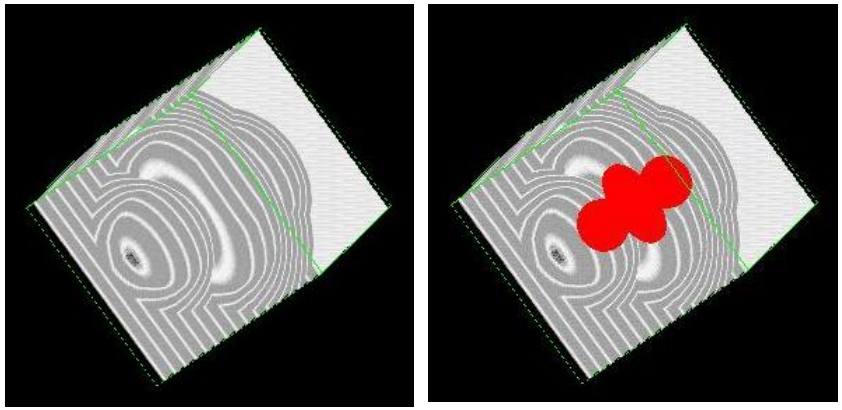




Rendering

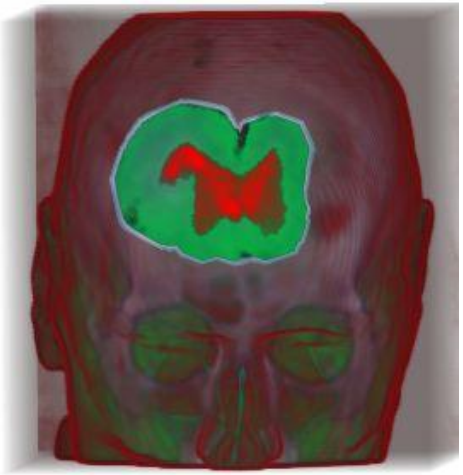
- Based on ray-casting
- Height map – clipping cone
- Subtract |BB – near|
- Shift starting position
- Data rendered without clipping cone
- Render mask

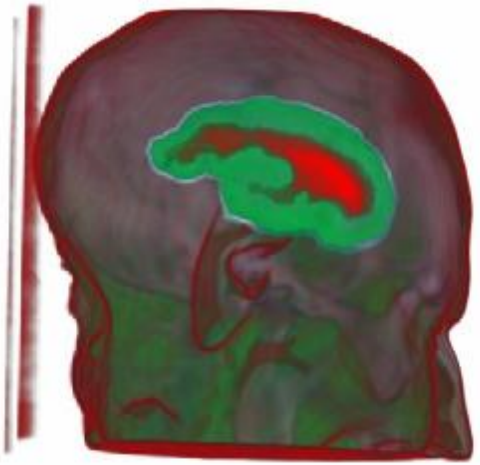




Future Work

- Ray-casting reverse order (depth buffer)
- Direct rendering to texture (FBO)
- Floating point textures
- Perspective projection
- Different transfer functions





Thank you for your attentions.