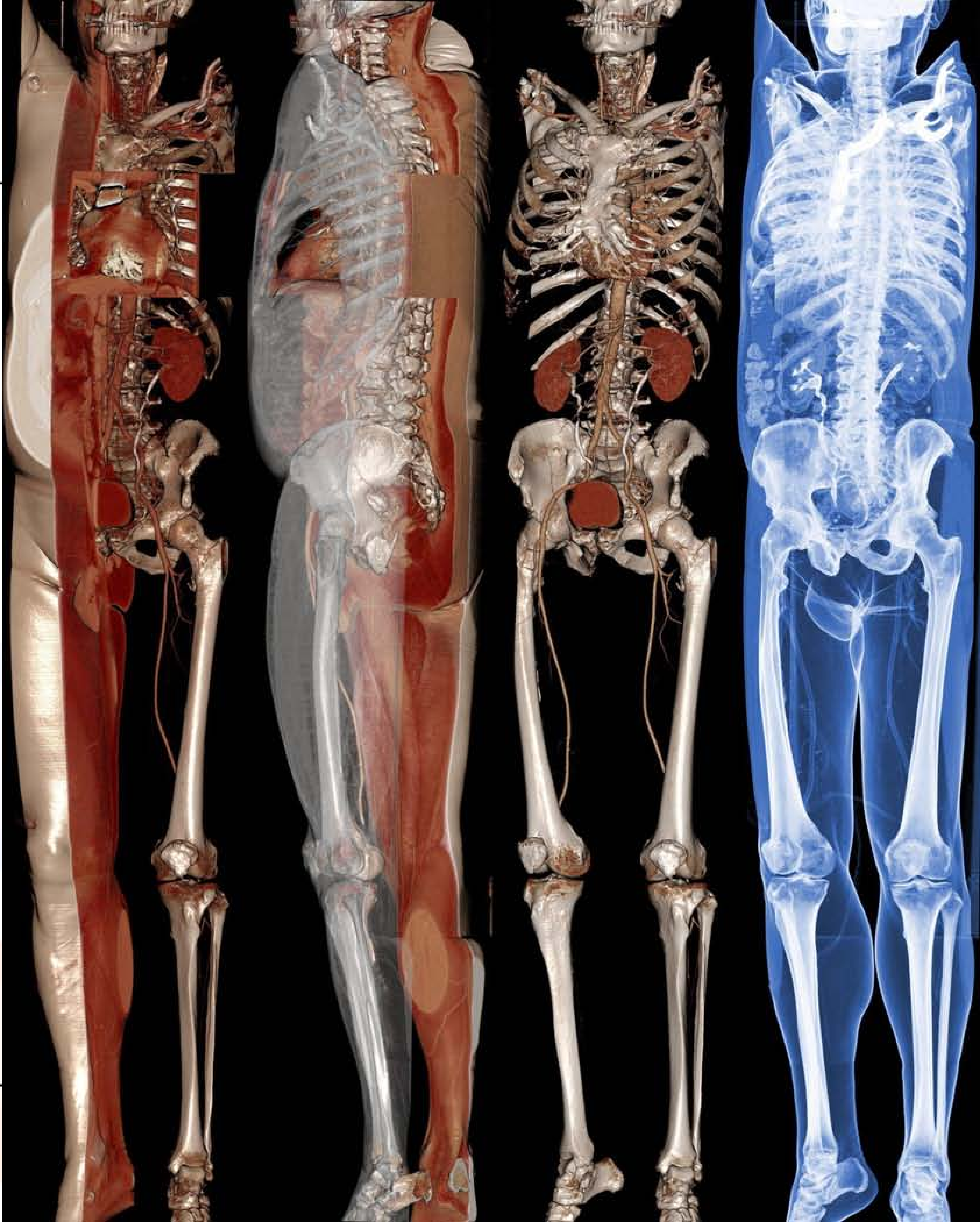
Real-Time Volume Graphics

[14] Large Volume Data

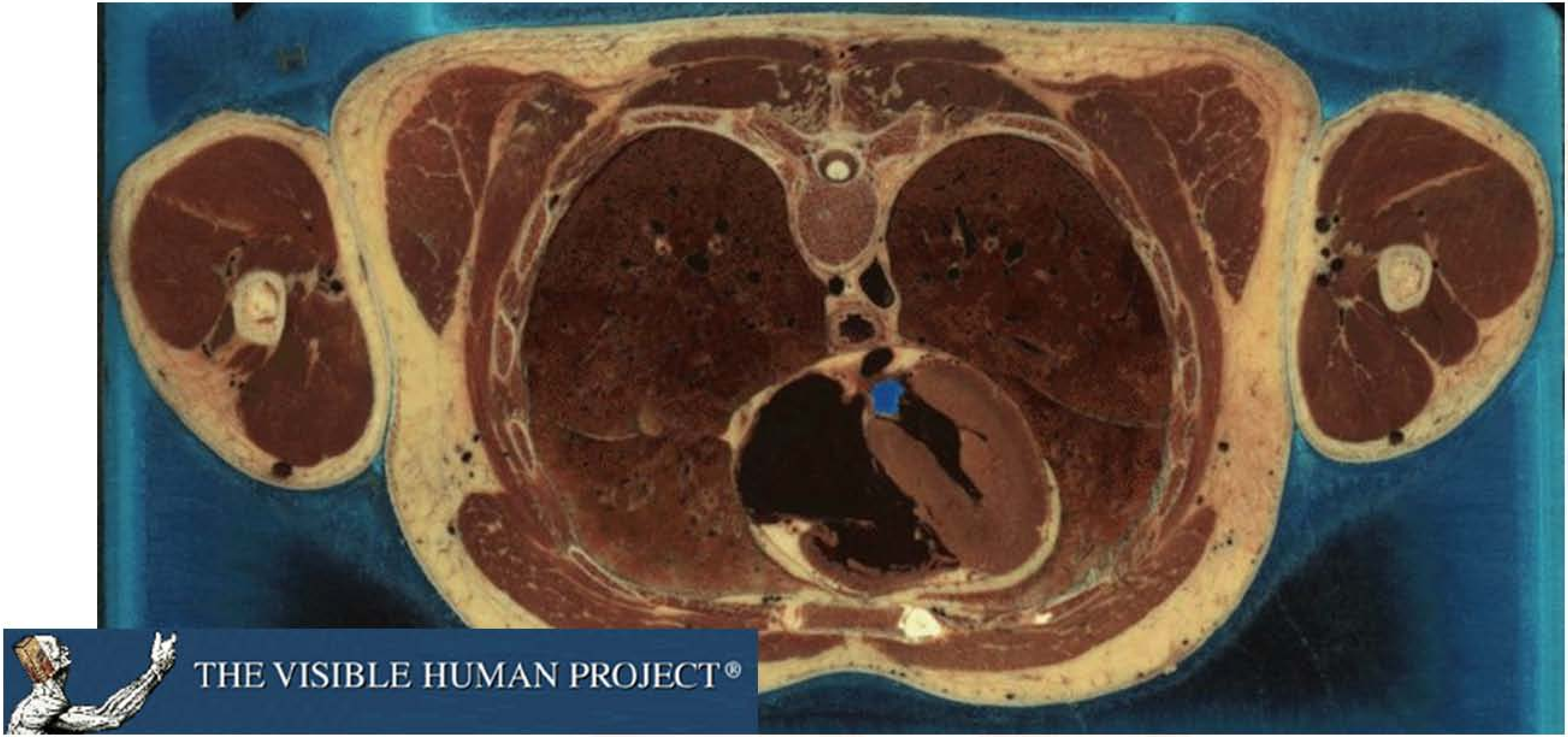
Motivation

- Long-leg study 512x512x3172 @16bit ~ 1.7GB



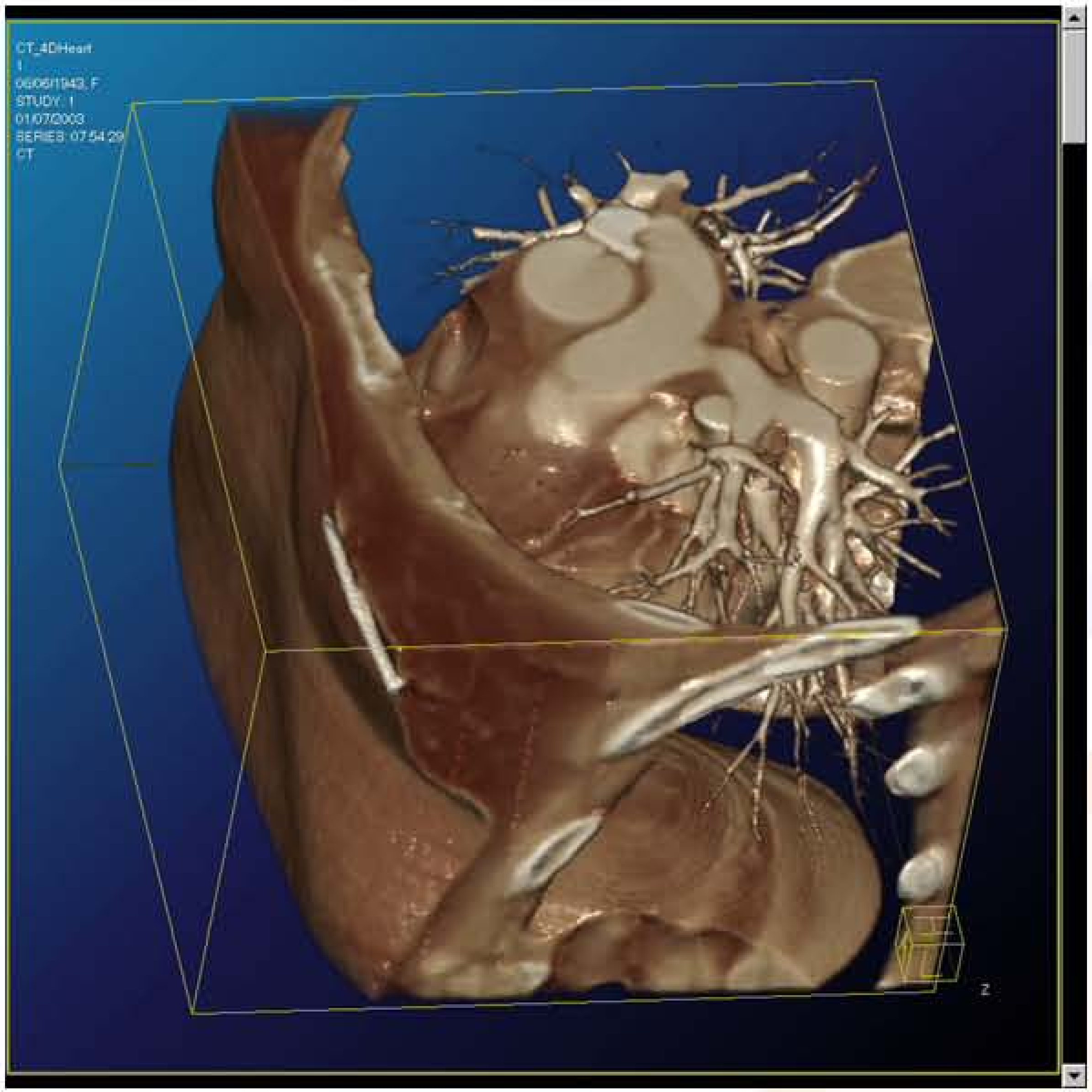
Large Volumes - *Motivation*

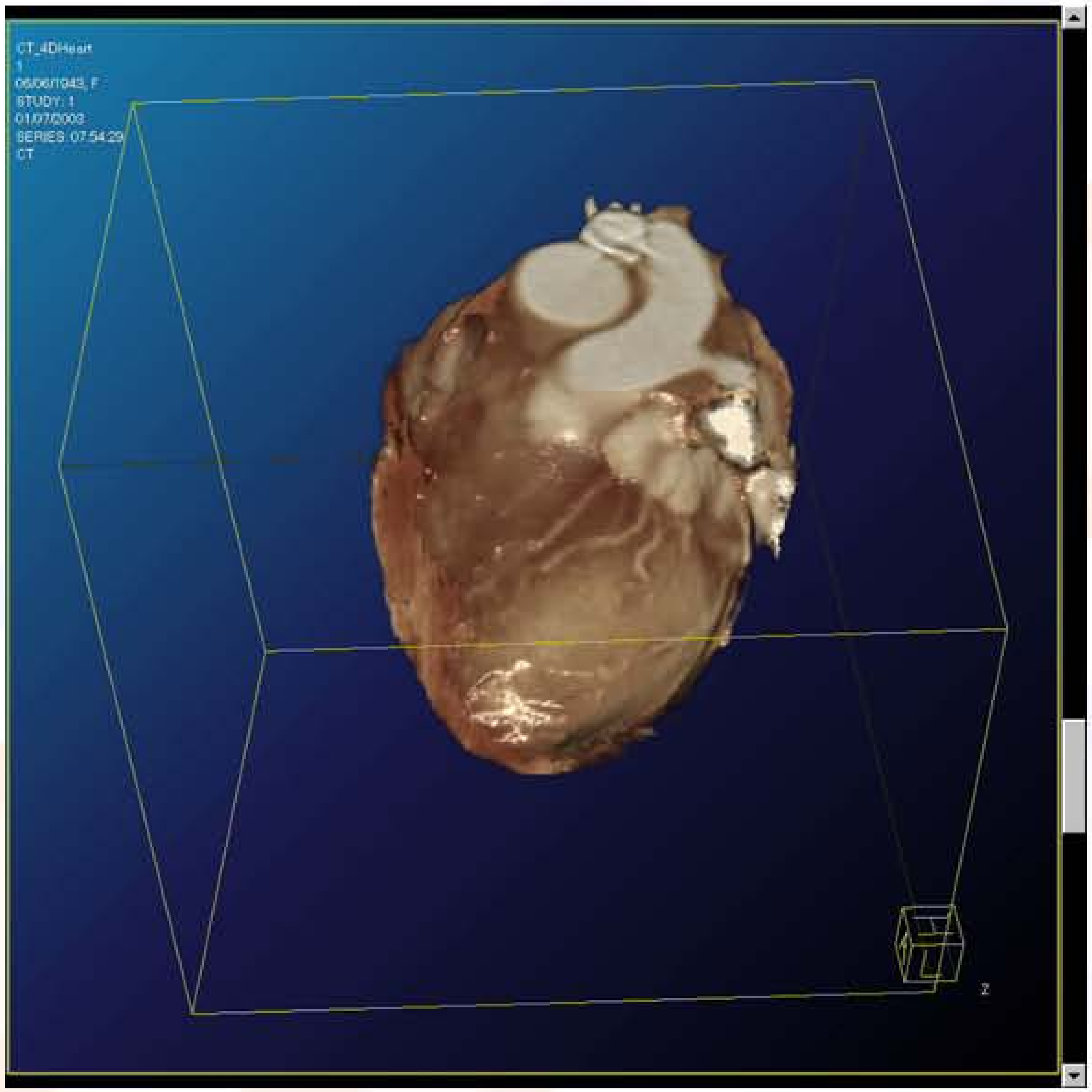
- Visible Male cryosection RGB data: 2048x1216x1877@24bit ~ 14GB



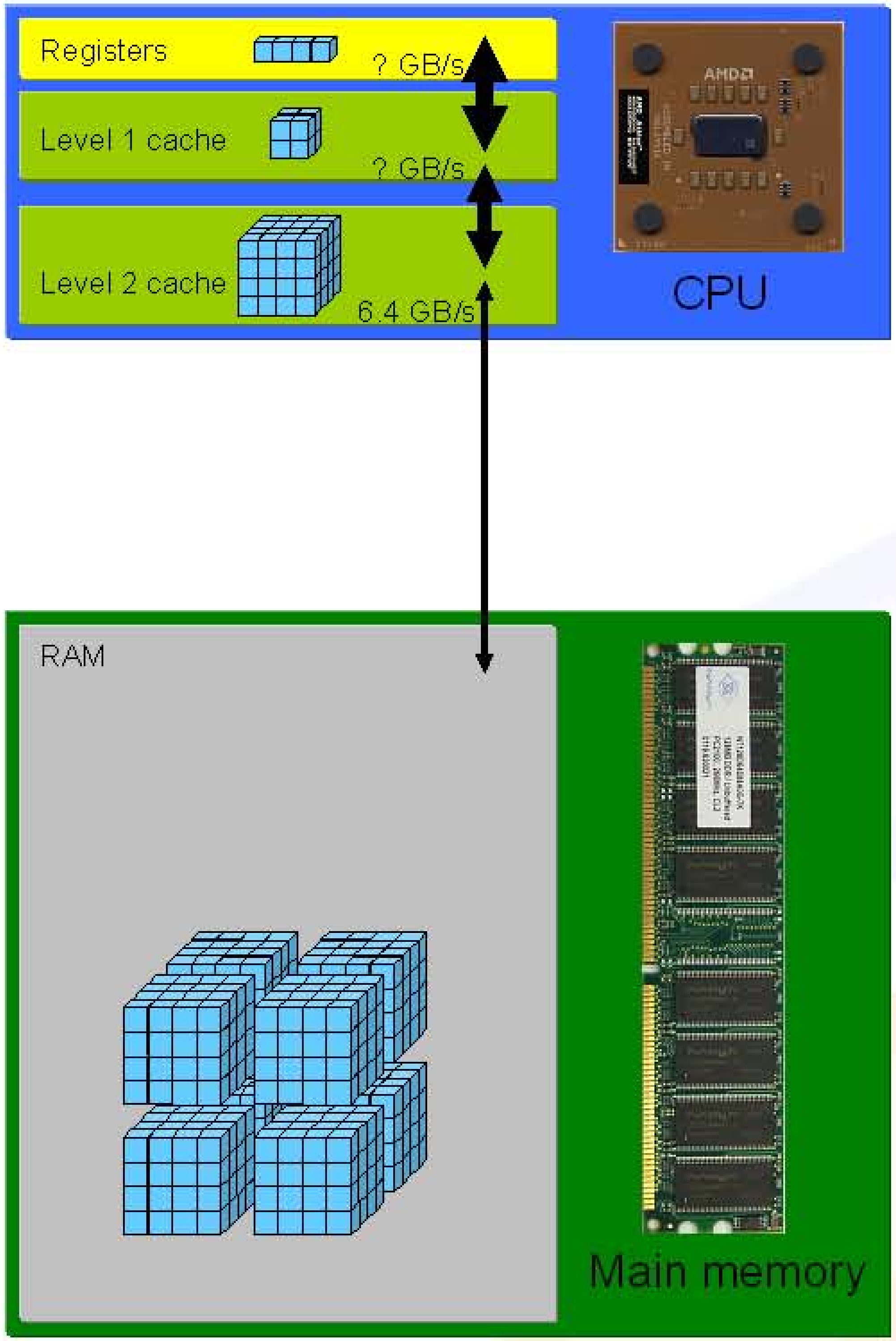
Large Volumes - Motivation

- 4D cardiac data:
  512x512x240@16 bit, 20 frames ~ 2.5GB

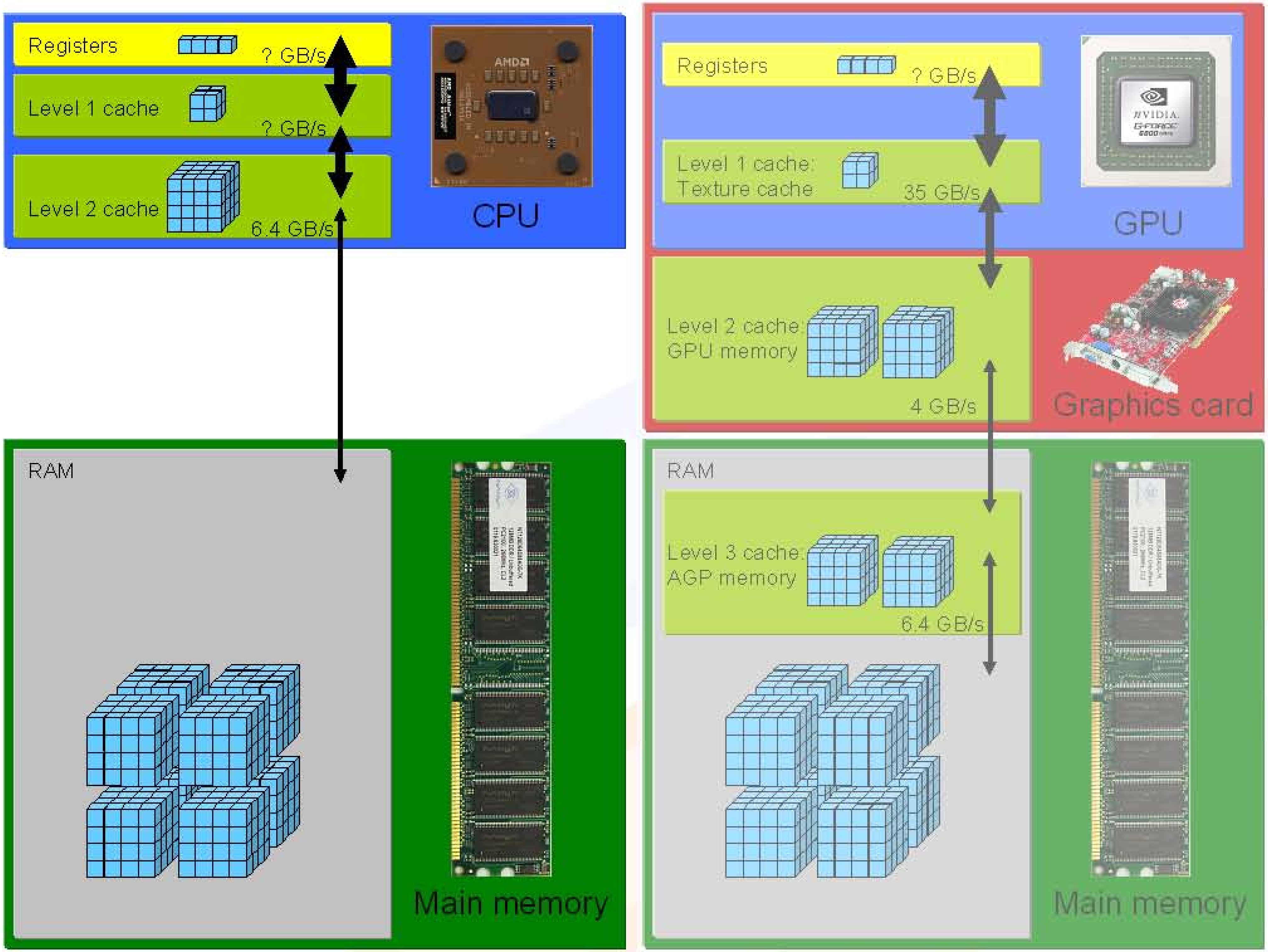




Large Volumes - Caches



Large Volumes - Caches



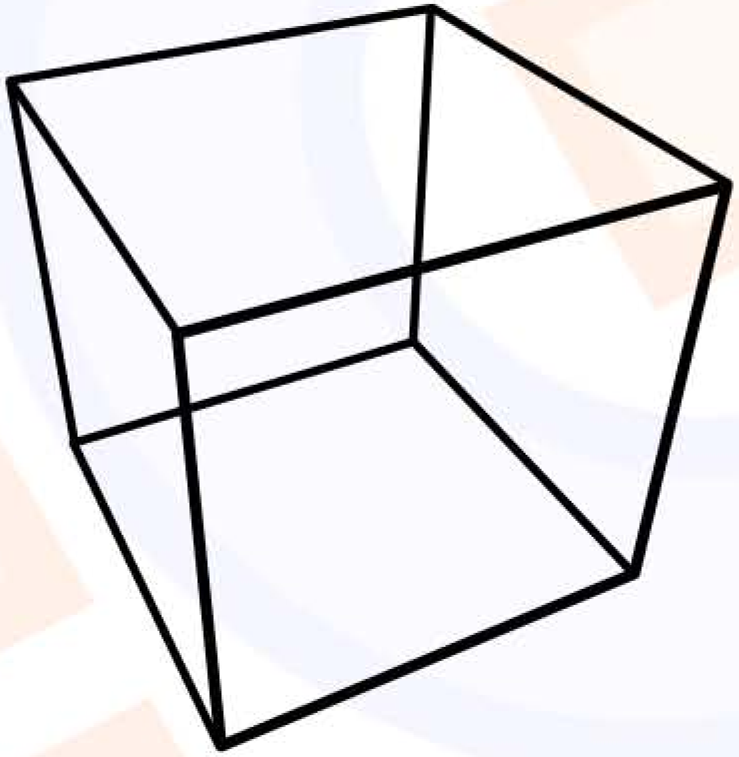
Large Volumes - Introduction

- Problems with large volumes on GPUs
  - Each Voxel is accessed multiple times
    - Filtering
    - Gradient Computation
    - Oversampling

  => Huge memory bandwidth required
  - Limited memory bandwidth:
    GPU: >30GB/s
    AGP8x: 2GB/s
  - Limited GPU memory: typically 256/512 MB

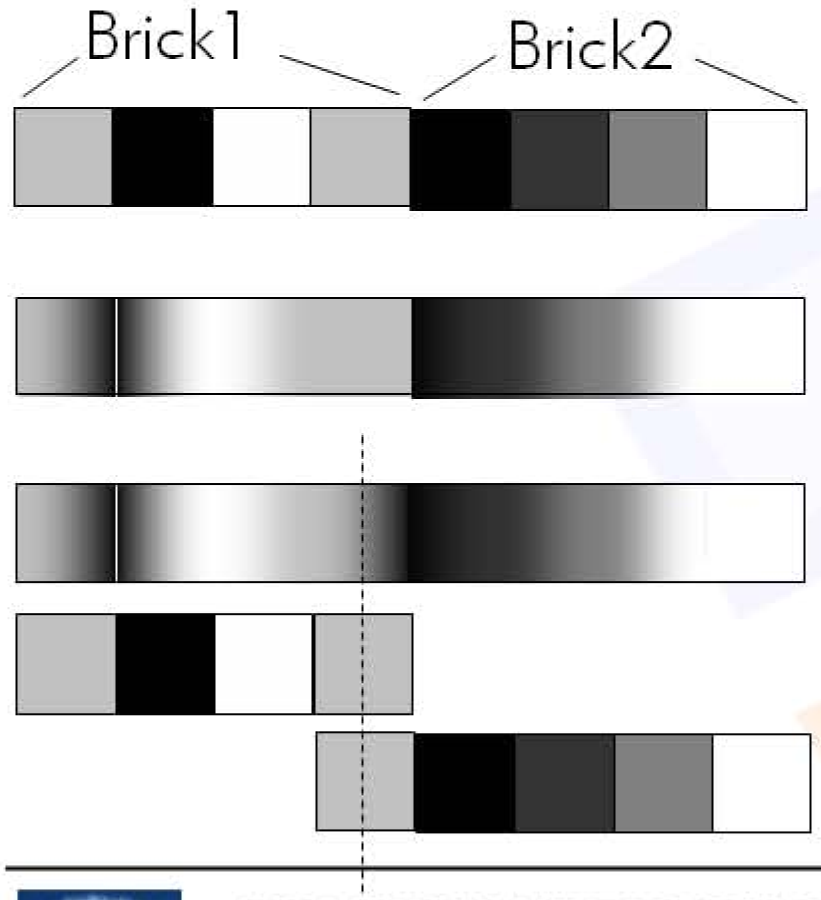
Large Volumes - Bricking

- Subdivide volume into smaller blocks
- Allocate memory for one block on GPU
- Copy in GPU mem. and render one block at a time
- One voxel overlap for contiguous interpolation



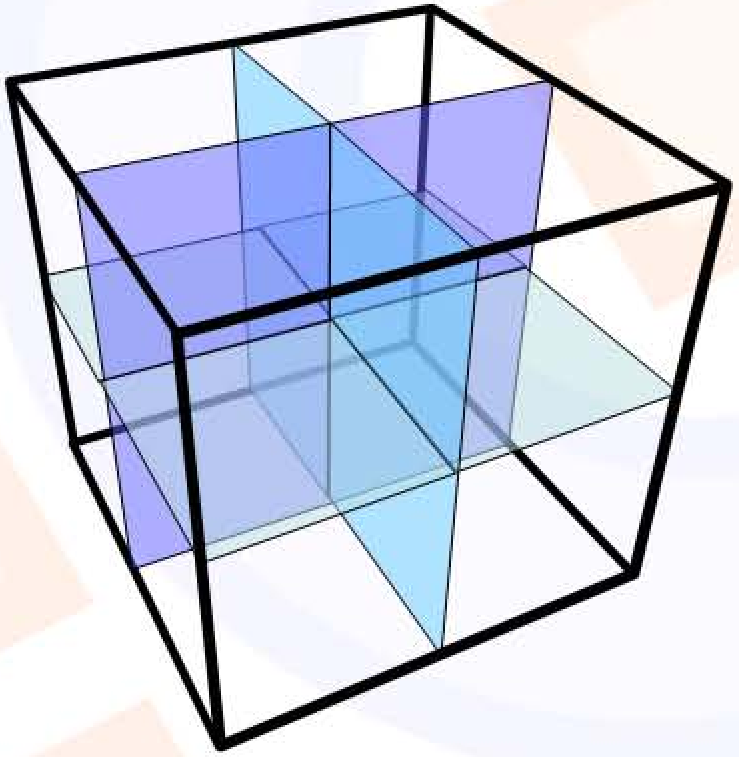
Large Volumes - Bricking

- Subdivide volume into smaller blocks
- Allocate memory for one block on GPU
- Copy in GPU mem. and render one block at a time
- One voxel overlap for contiguous interpolation



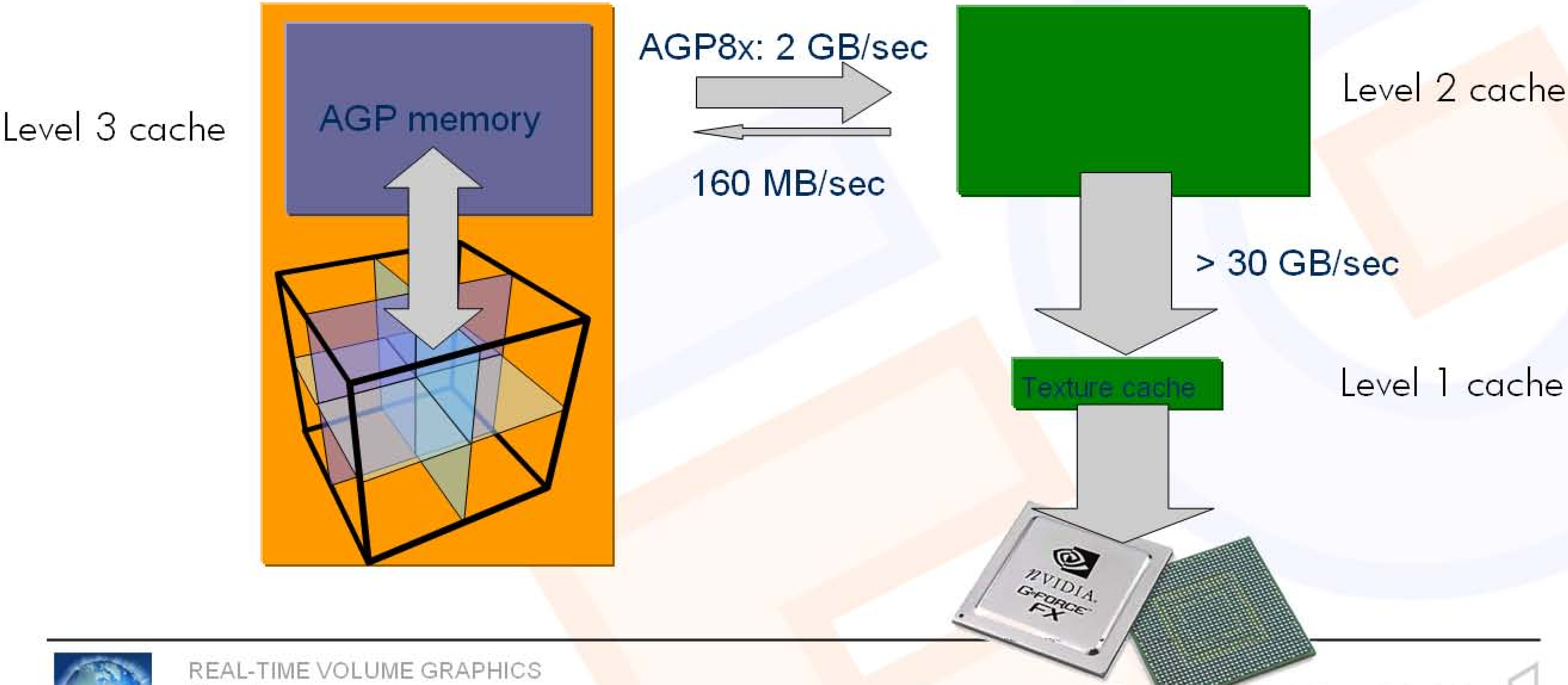
No overlap

One voxel overlap



Large Volumes - Bricking

- Performance mainly limited by AGP transfer
  - Subsampled copy of data in GPU mem. for interaction
  - Bricking only for final quality image



Level 3 cache

AGP memory

AGP8x: 2 GB/sec

160 MB/sec

Level 2 cache

> 30 GB/sec

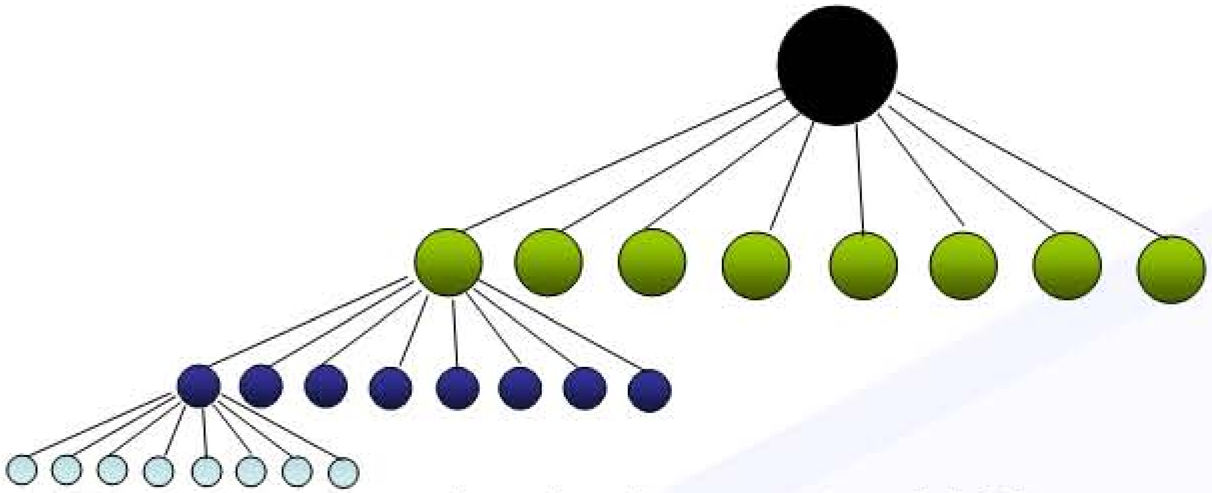
Texture cache

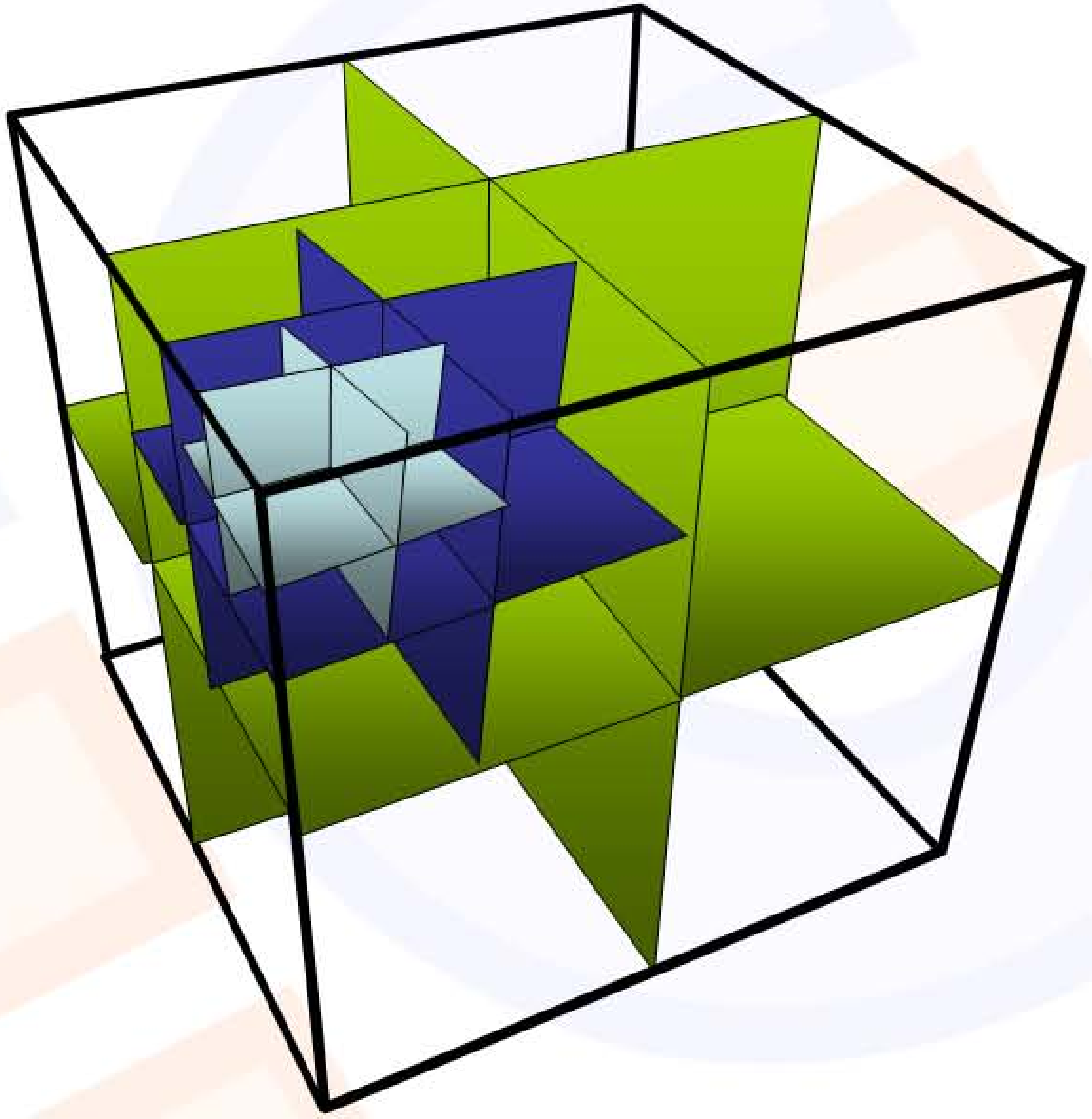
Level 1 cache

Large Volumes - *Multi-Resolution* VR

LaMar et al., *Multi-Resolution techniques for interactive texture-based volume visualization*, IEEE Visualization'99

- Octree-based decomposition of volume into bricks

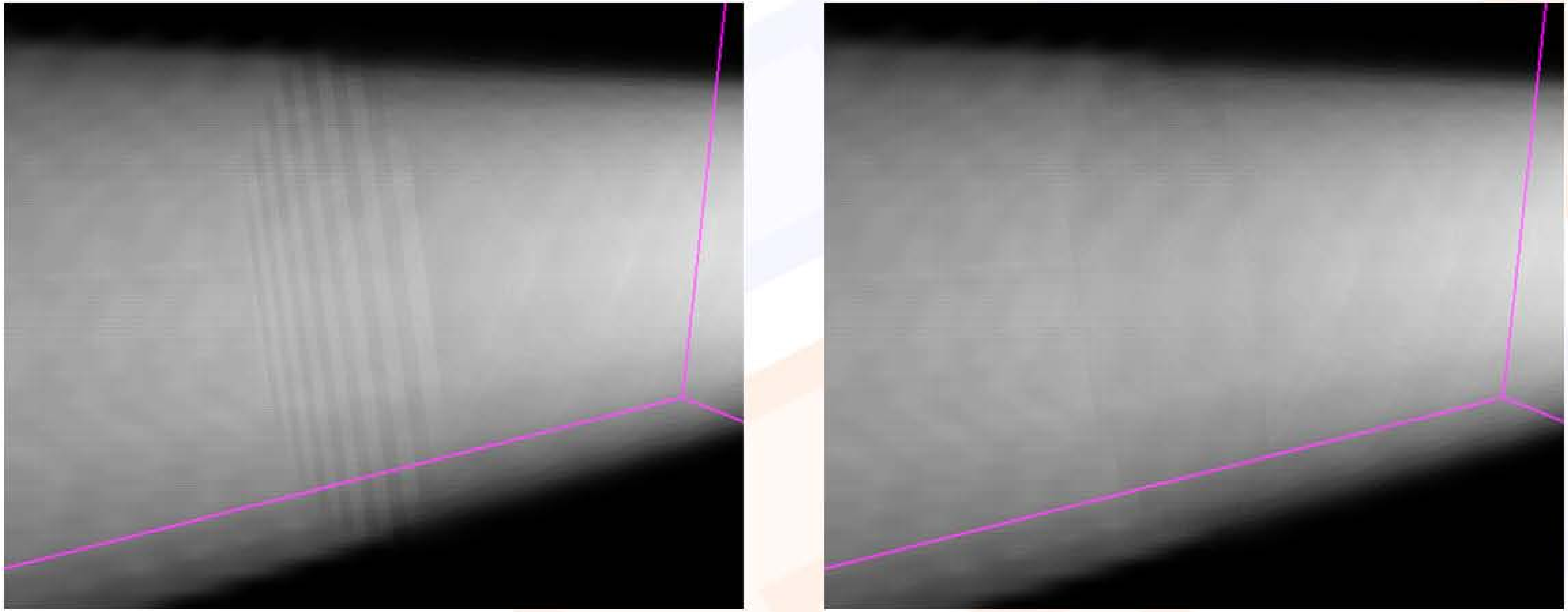




- Render bricks at different resolution:
  - Distance to viewer
  - Focus point

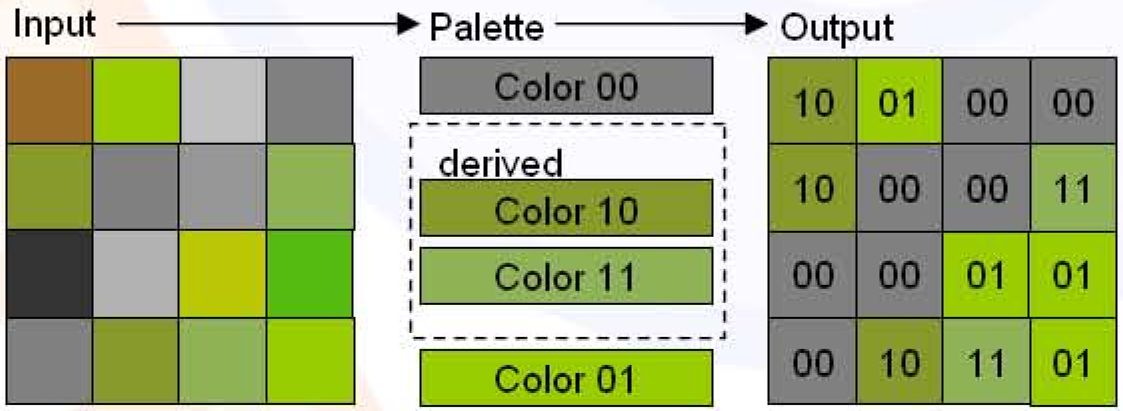
Large Volumes - Multi-Resolution VR

- Weiler et al., *Level-of-Detail volume rendering via 3D textures*, In *Volume Visualization and Graphics Sympsium 2000*
  - Extension to fix boundaries



Large Volumes - Compression

- Texture Compression OpenGL extensions:
  2D: EXT_texture_compression_s3tc.
  3D: NV_texture_compression_vtc
  - Hardware implementation in several graphics chips, e.g. NVIDIA GeForce series, ATI Radeon Series.
- Disadvantages of S3TC:
  - Moderate compression ratios,
  - Block-artifacts, inappropriate for non-smooth data
  - Fixed compression scheme
  - Only for RGB(A) data
- 3Dc: ATI only, normals



10	01	00	00
10	00	00	11
00	00	01	01
00	10	11	01

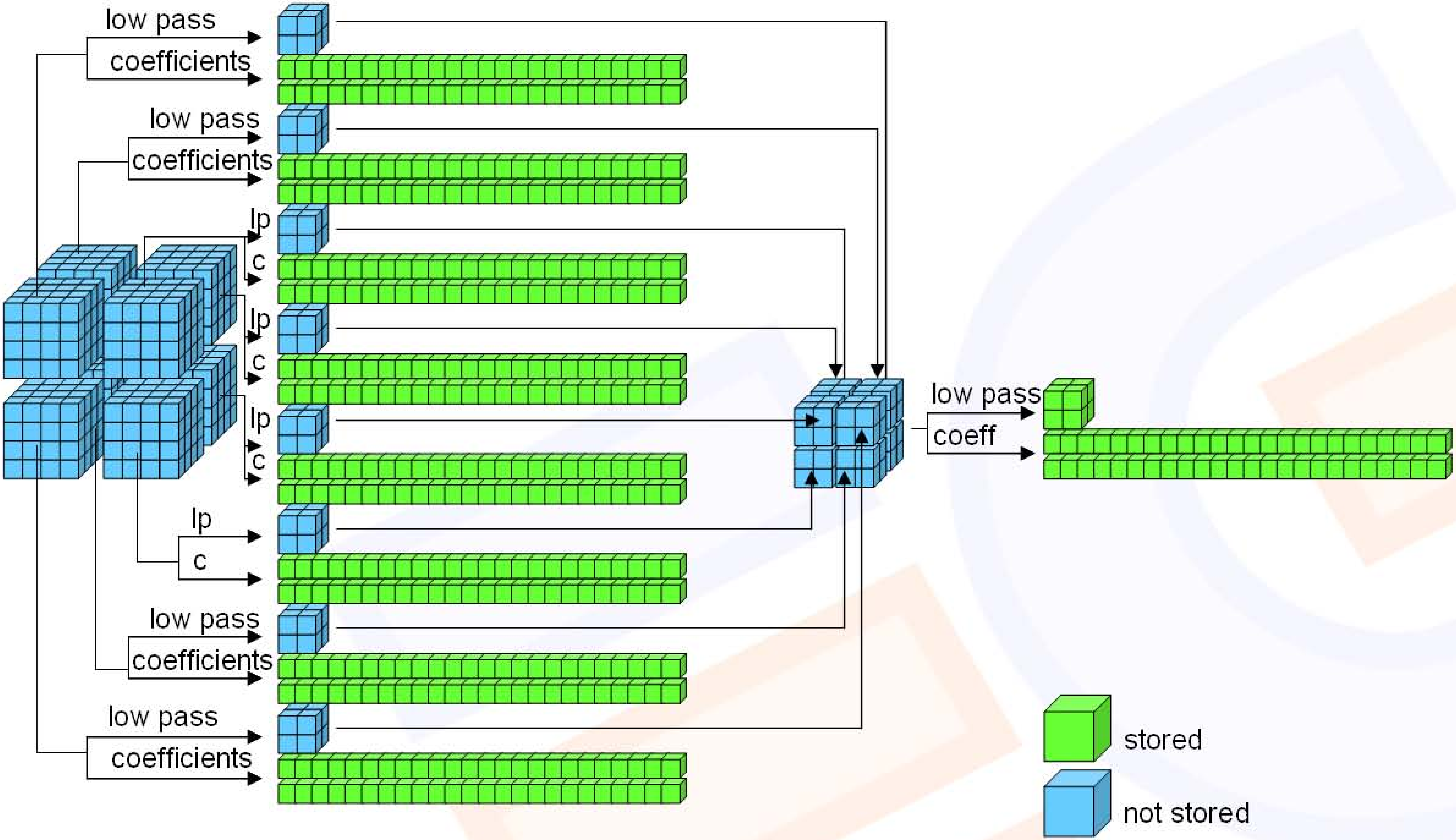
Large Volumes - Wavelets

- Volume data
  - Mostly smooth
  - Fine high-frequency detail in certain regions
- Wavelets
  - Analyze data at different resolutions and frequencies
  - Many coefficient very small => compression
  - Hierarchy of signals => Multi-res reconstruction

Large Volumes - Wavelets

- Guthe et al., *Interactive Rendering of Large Volume Data Sets*, Visualization'02
- Hierarchical Wavelet Representation
  - Divide data into blocks of $(2k)^3$ voxels (k=16)
  - Apply wavelet filters
    => lowpass filtered block
    => wavelet coefficients
  - Group 8 blocks
  - Repeat until 1 block left
- 2 encoding schemes

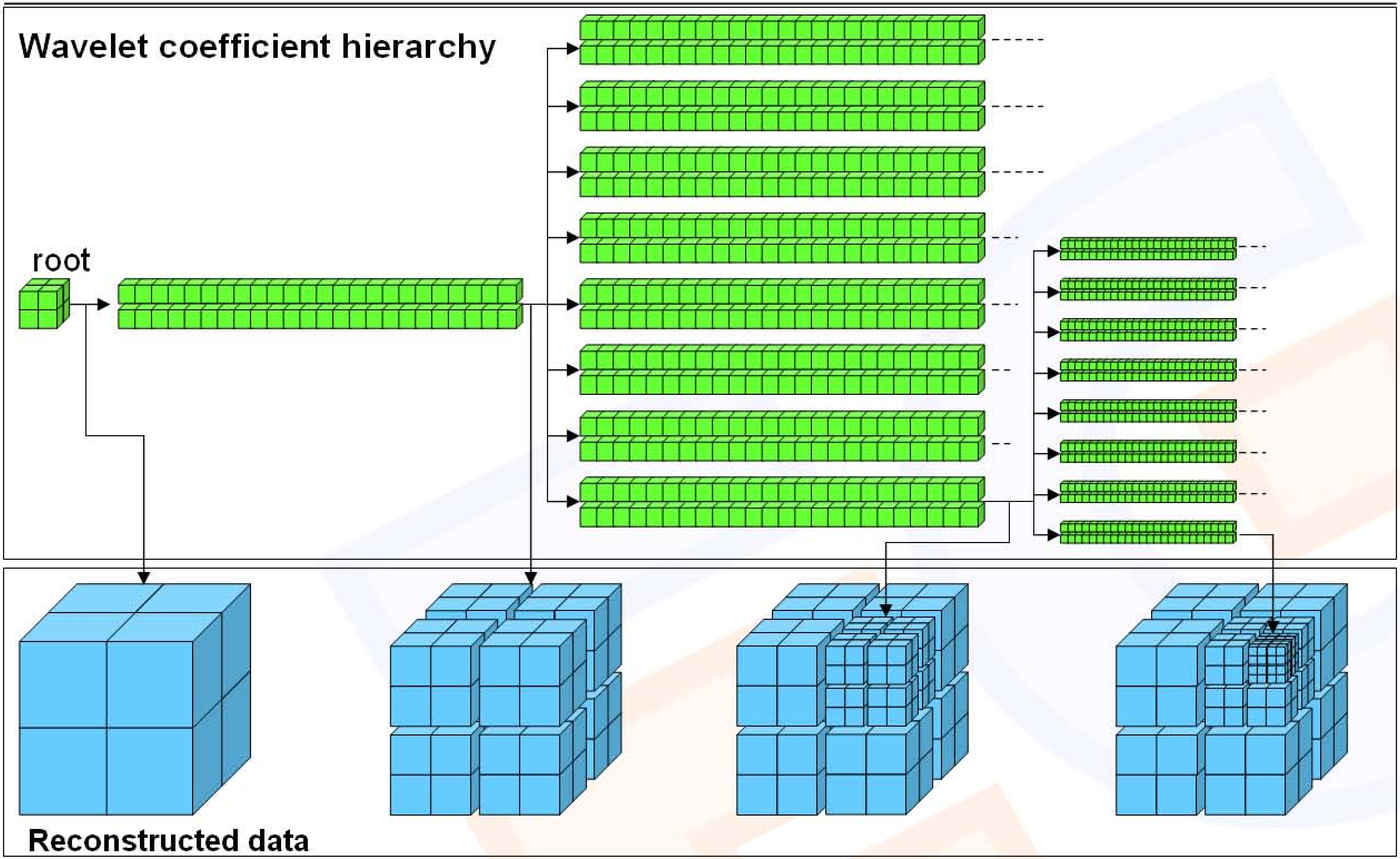
Large Volumes - Wavelets



Large Volumes - Wavelets

- Decompression of blocks during rendering on CPU
- Rendering of block on the GPU
- Caching Strategy

Large Volumes - Wavelets



Wavelet coefficient hierarchy

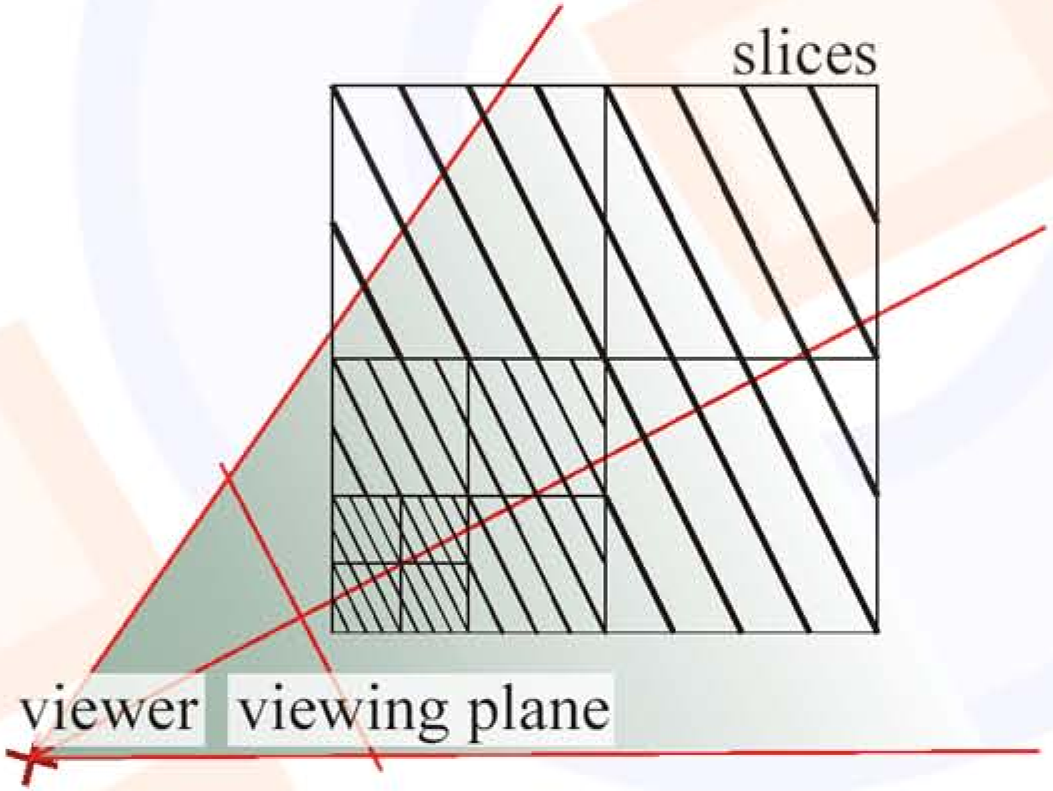
root

Reconstructed data

Large Volumes - Wavelets

- Adjust resolution of data to screen resolution
  - Project voxel-space to screen
  - Refine if above screen res.
- View-dependent priority schedule
- Interactive walkthrough of visible female/male
- Still AGP bandwidth bound => GPU decompression possible ?

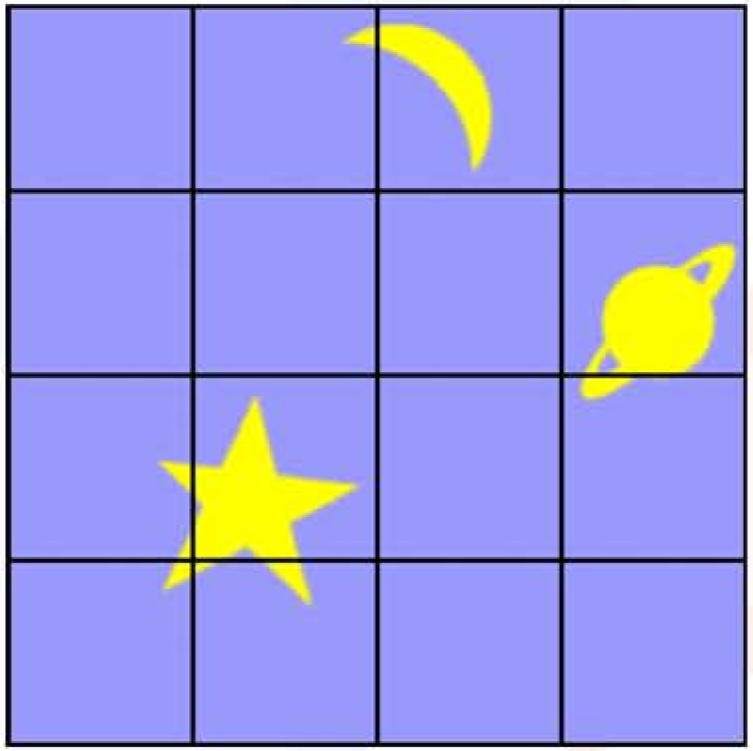
Google "dwtgpu"

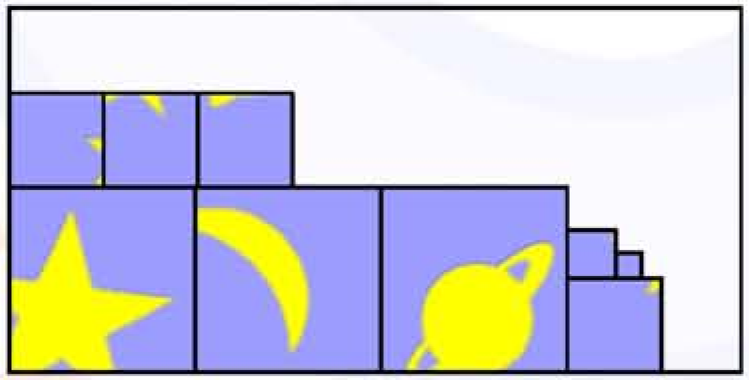


Large Volumes - Packing

TWO LEVELS OF THE DATA REPRESENTATION:

- Index data (upper level):
  - Each cell/texel of a coarse grid corresponds to one data block.
  - Each cell/texel specifies coordinates and scaling factors of the corresponding data block.
- Packed data (lower level):
  - All data blocks packed into one uniform grid/texture.



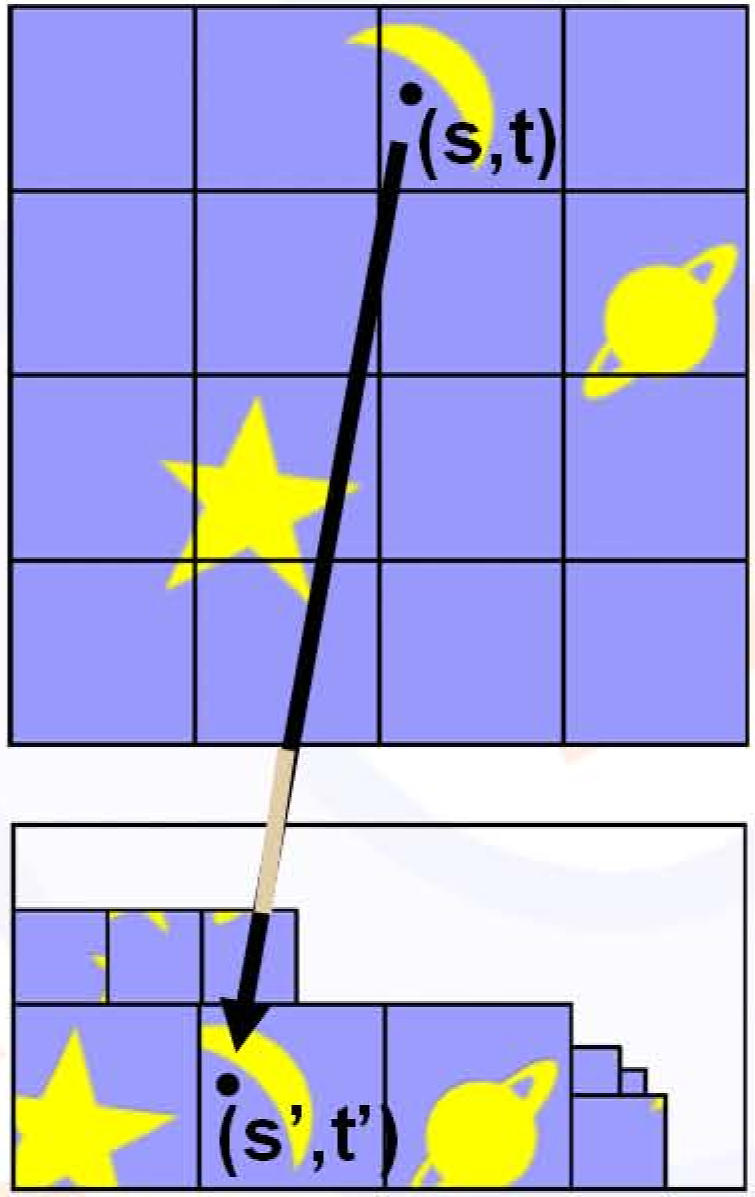


Kraus et al., *Adaptive Texture Maps*, Graphics Hardware Workshop'02

Large Volumes - Packing

TWO STEPS OF SAMPLING ADAPTIVE TEXTURES:

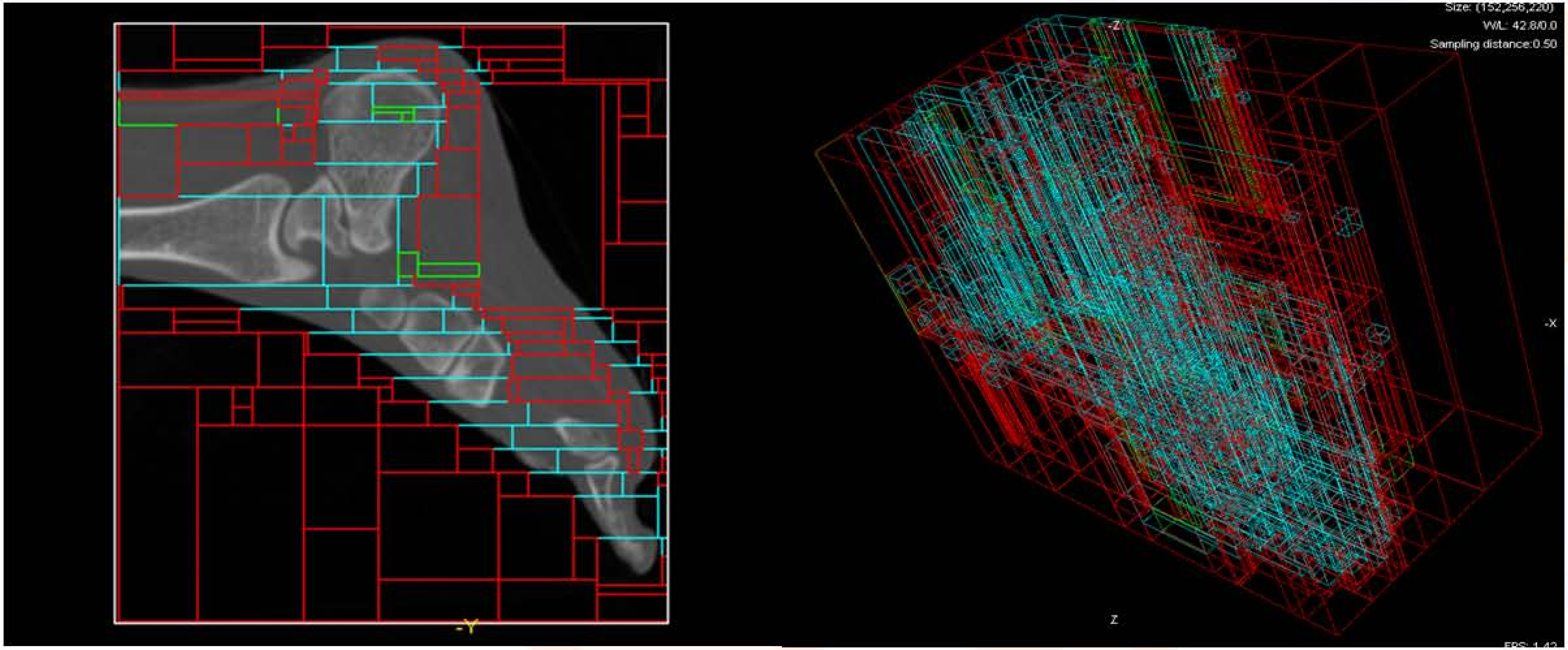
- Read index data and calculate coordinates for the second step.
- Read and interpolate actual texture data from packed data.
- Decoding in fragment stage



Large Volumes - Packing

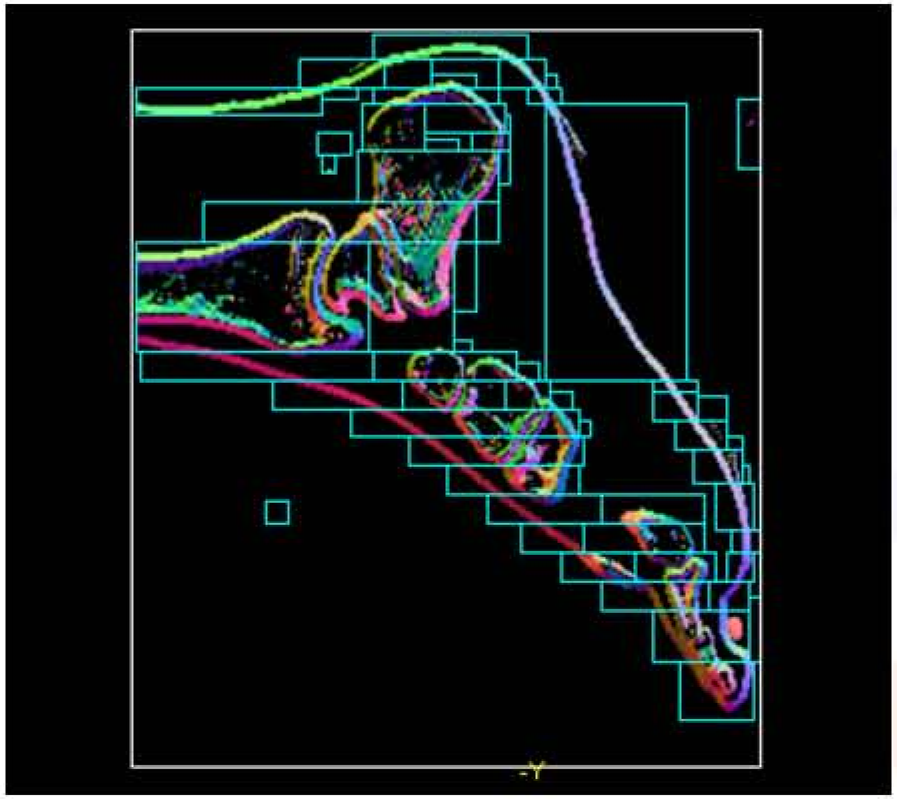
Wei Li et al. – *Texture Partitioning and Packing for Accelerating Texture-based Volume Rendering*, GI 2003

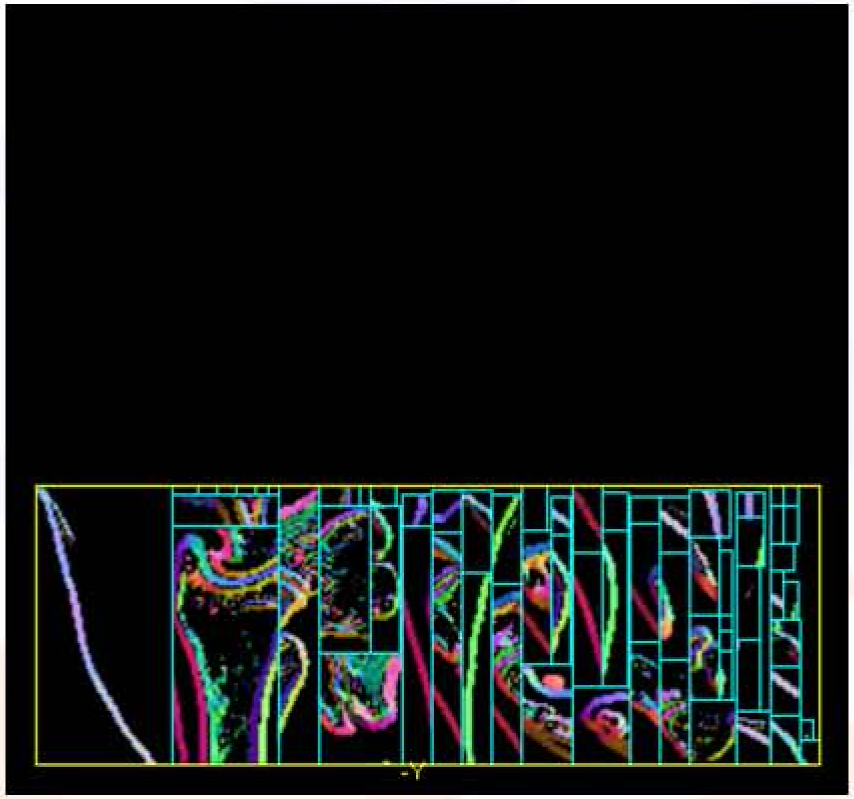
- Partition texture space with box-growing algorithm
- Based on similar densities and non-zero gradient magnitudes

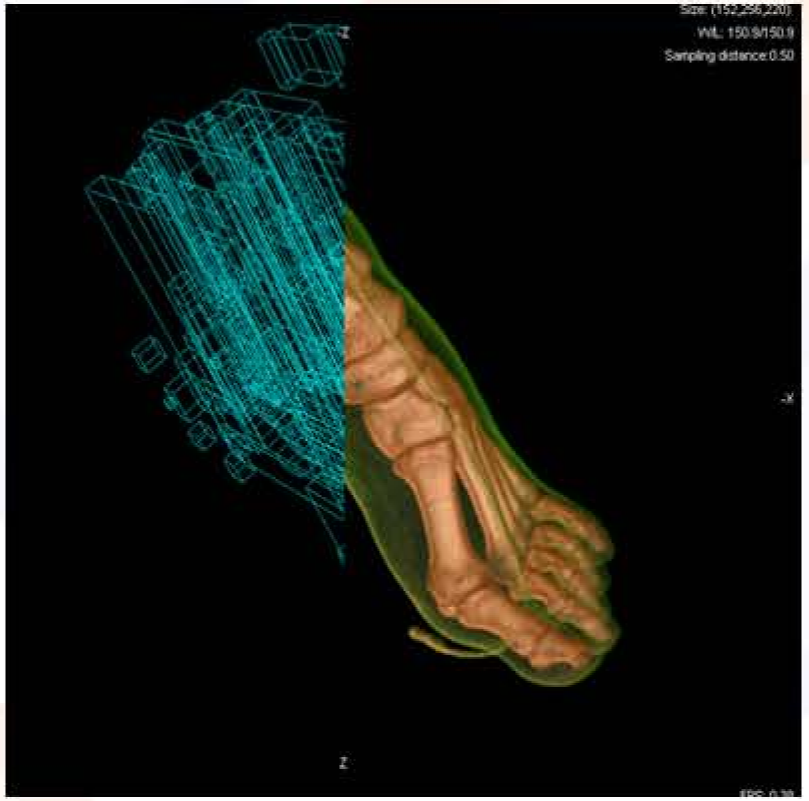


Large Volumes - Packing

- Determine gradient sub-textures with non-zero magnitude
- Pack sub-textures into a single smaller texture
- Decoding in vertex stage using texture coords.

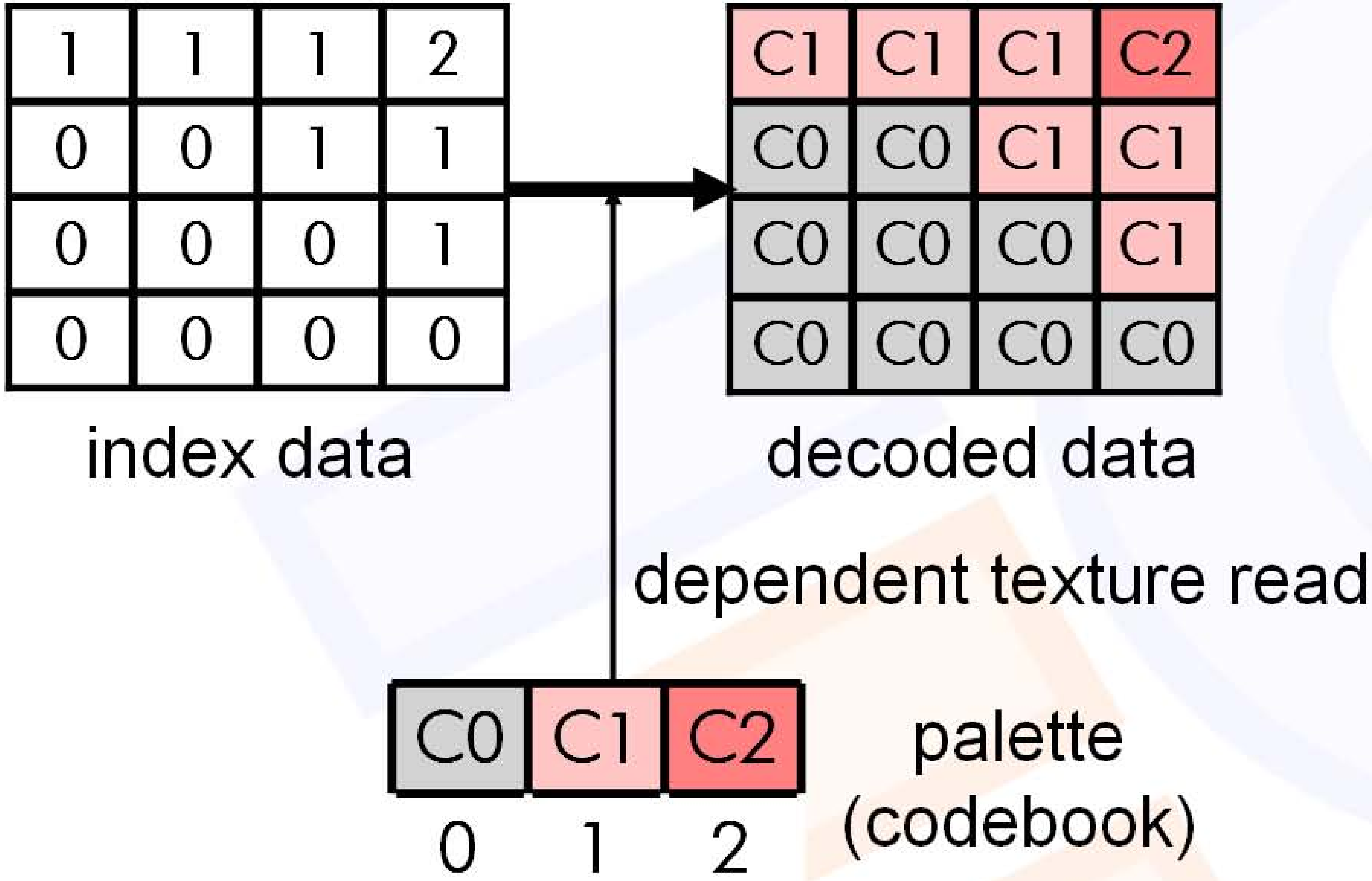






Large Volumes - VQ

- Image formats with palettes specify for each pixel one index into a color palette (= codebook).



1	1	1	2
0	0	1	1
0	0	0	1
0	0	0	0

index data

C1	C1	C1	C2
C0	C0	C1	C1
C0	C0	C0	C1
C0	C0	C0	C0

decoded data

dependent texture read

C0	C1	C2
0	1	2

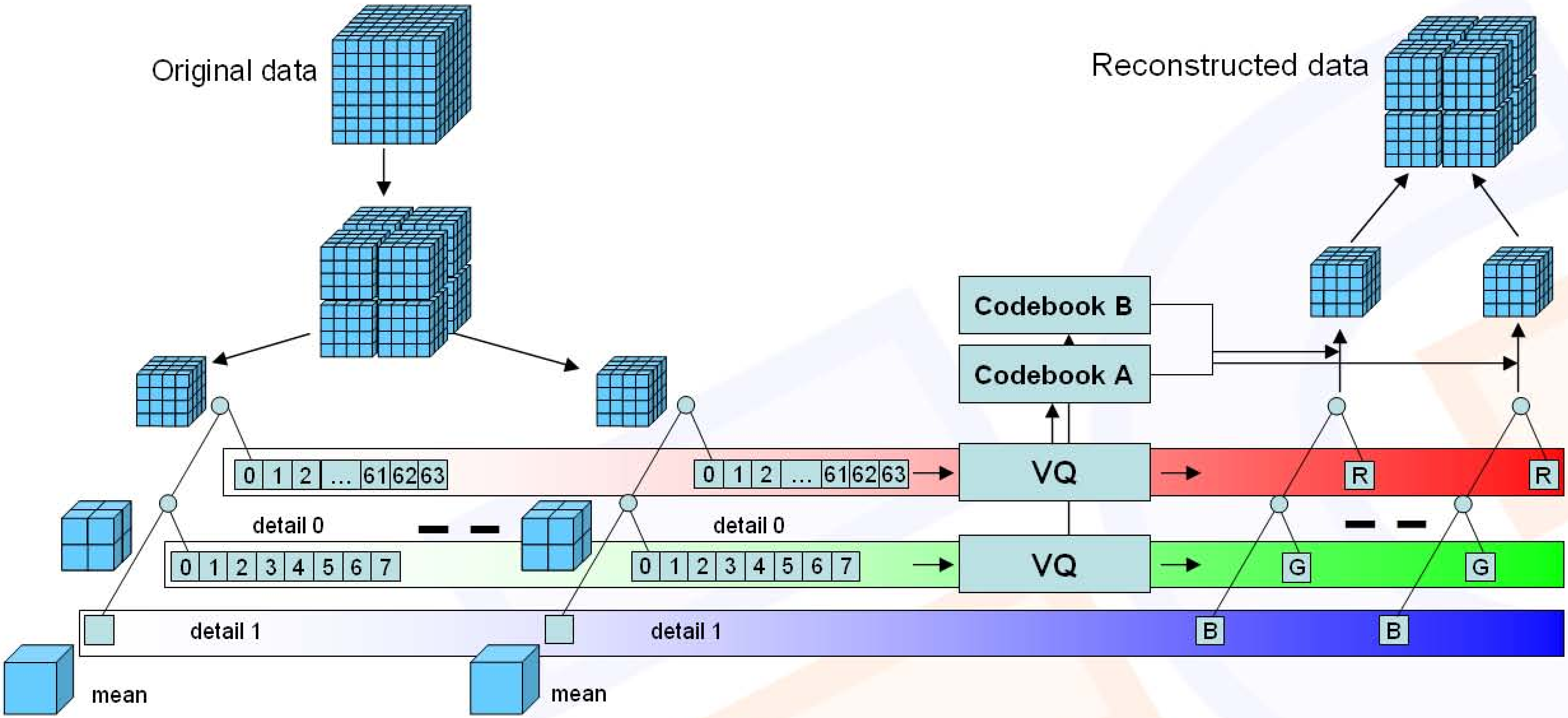
palette (codebook)

Large Volumes - VQ

Schneider/Westermann – *Compression Domain Volume Rendering*, IEEE Visualization 2003

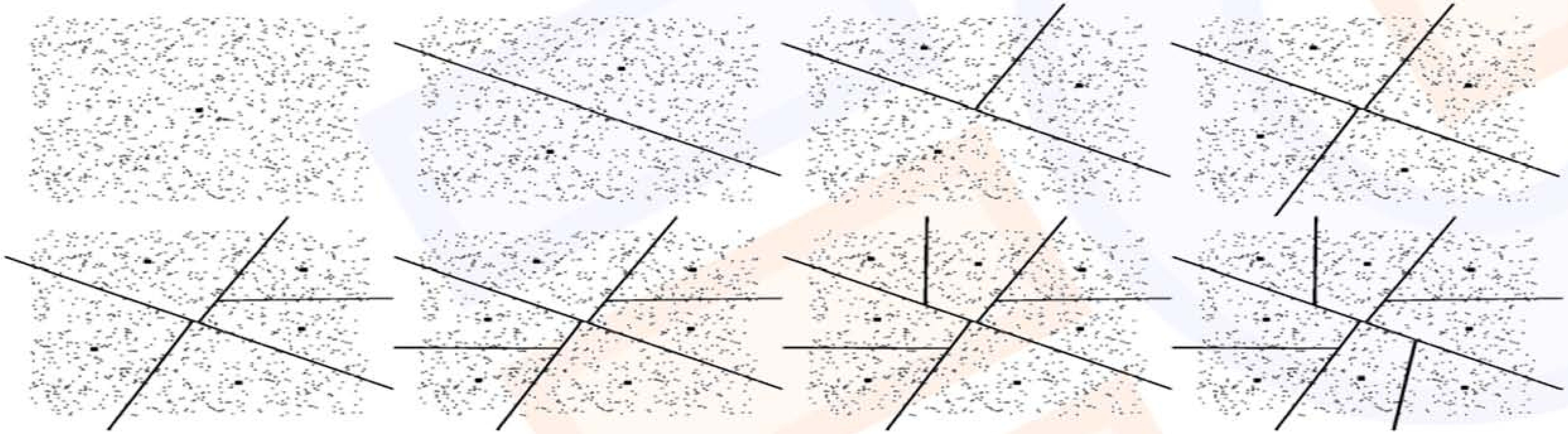
- 3 Level Hierarchical decomposition:
  - Partition data into blocks of size 4^3
  - Downsample to 2^3, store difference vector (64 vector)
  - Downsample to 1, store difference vector (8 vector)

Large Volumes - VQ



Large Volumes - VQ

- Codebooks: 256x64 + 256x8 images, dependent texture lookup
- Codebook generation:
  modified LGB-Algorithm (Linde, Buzo and Gray)
- Series of PCA-Splits (Principle component analysis) determine initial codebook



Large Volumes - VQ

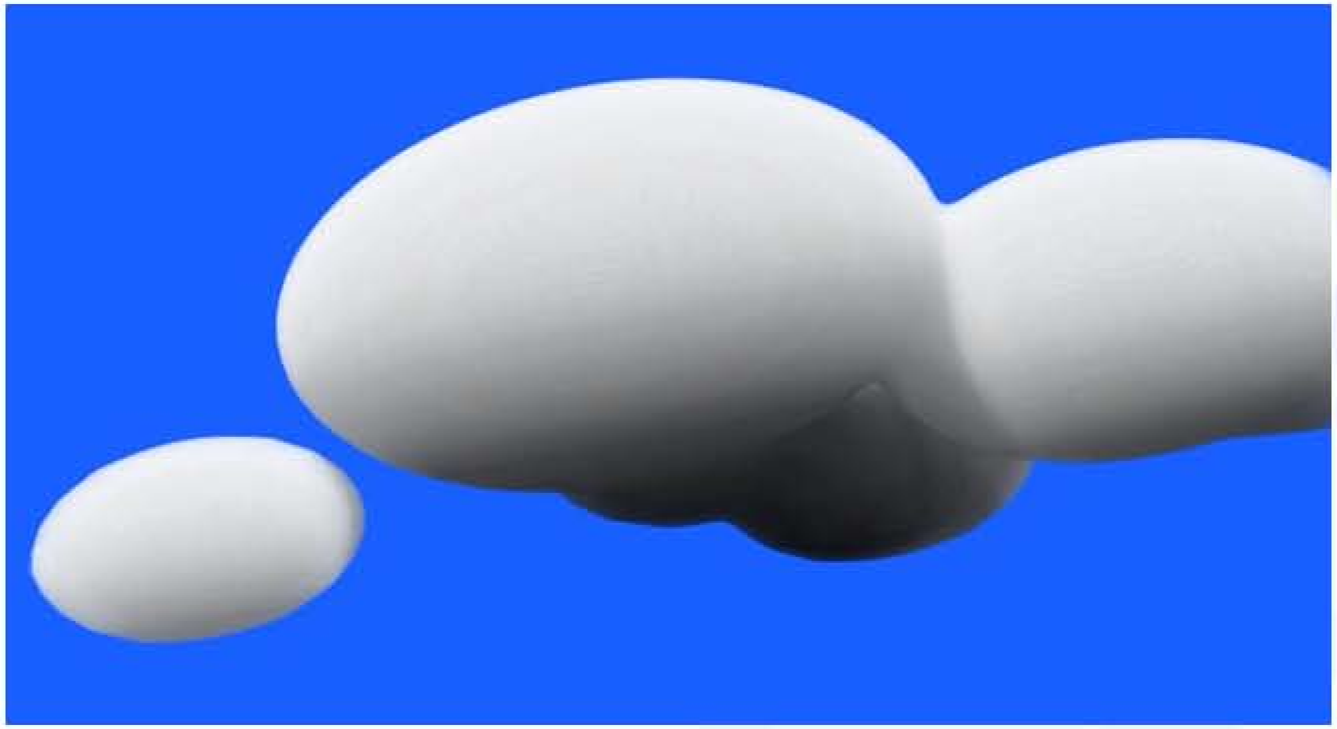
- With a codebook of length 256, a 1024^3 volume is reduced to $3*256^3$ bytes = 48 MBytes, i.e. it fits easily into texture memory.
- Compression of 4D sequences: store complete sequence in GPU memory (shockwave sequence - original: 1.5 GB, compressed: 70 MB).
- currently: only nearest-neighbor filtering
  => decouple decompression and rendering

Really Huge Volume Data

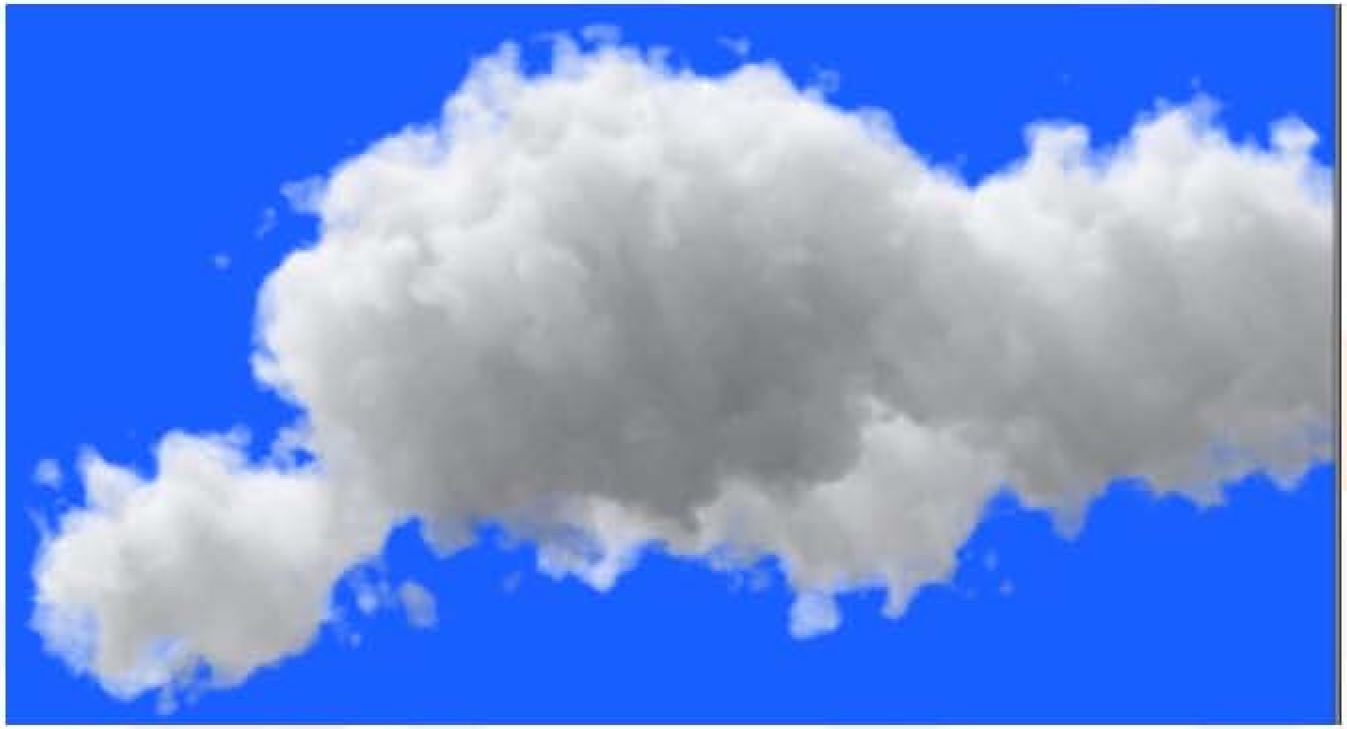
- Data/compressed Data larger than main memory (e.g. Geological data)
  - Out-of-core techniques
  - Keep data on disk
  - Use main memory as another cache level
  - Multi-resolution techniques
  - Rendering Clusters

Circumvent Large Volumes

Ebert et al.: Texturing and Modeling:
A Procedural Approach, Academic Press, 1998



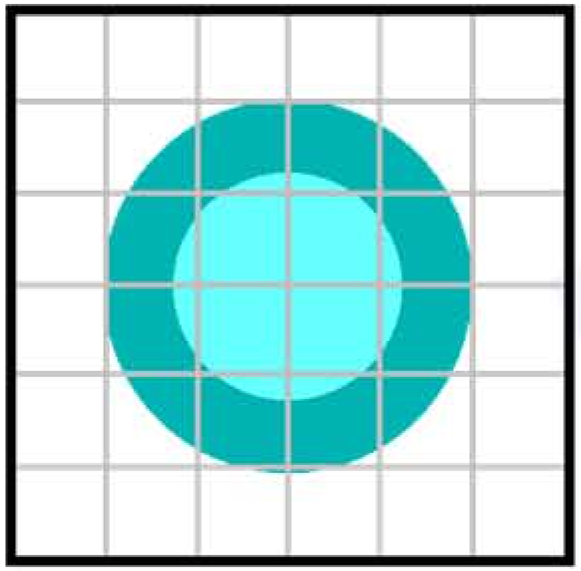
coarse volume
for macrostructure



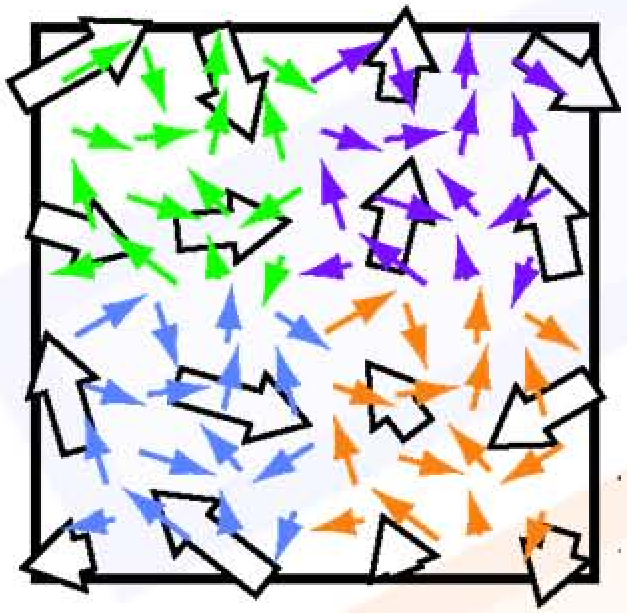
procedural noise for
microstructure

Circumvent Large Volumes

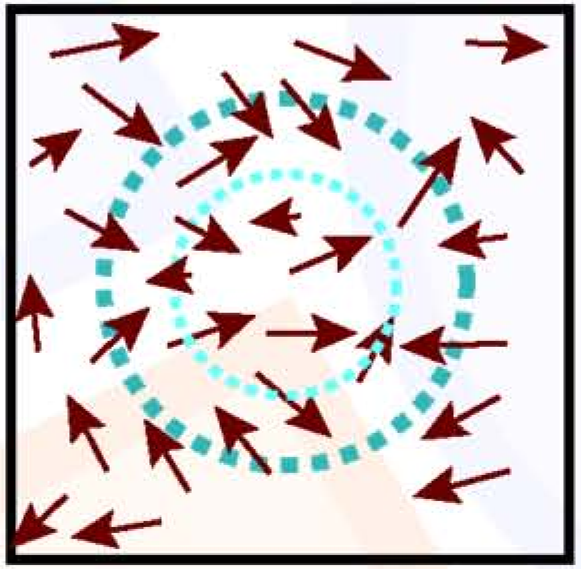
- Kniss et al., *Interactive Translucent Volume Rendering and Procedural Modeling*, Visulization'02:
  - perturb data access (instead of data)
  - good for distortion of boundaries
  - implemented using offset-textures



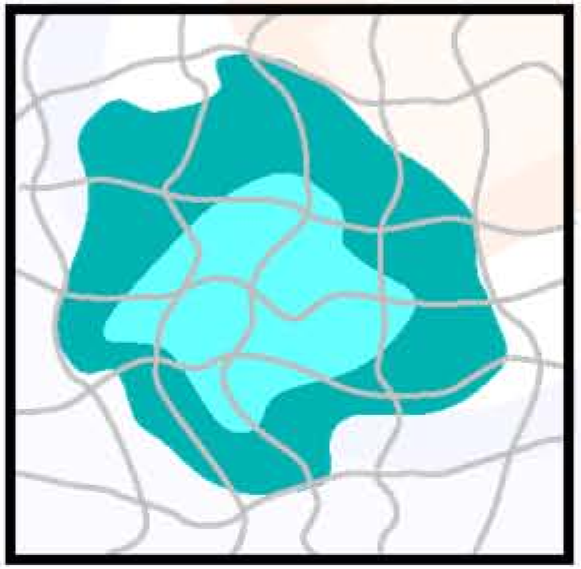
original texture



multiple scaled versions of perturbation texture



summed offset vectors



result of dependent fetch

Conclusions

- It's possible to render Large Volumes with GPUs (larger than texture memory)
- Compression and adaptive Algorithms required for interactive Performance
- Lots of Optimizations required
  - Early-Z/Stencil
  - Multi-Resolution
  - Compression Domain
  - …