Wavelet-Based 3D Compression Scheme for Very Large Volume Data

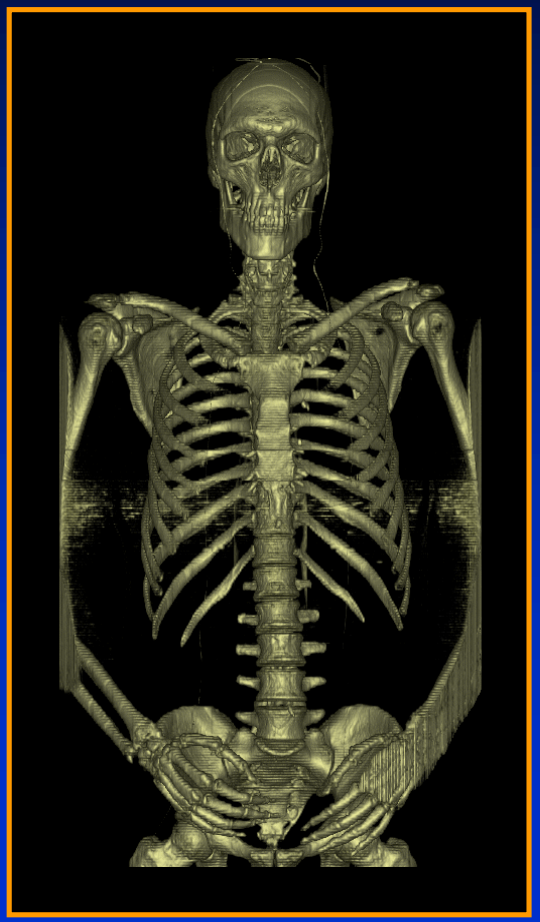
Insung Ihm and Sanghun Park
Dept. of Computer Science
Sogang University
Seoul, Korea

Background

- Volume data are often very large.
  - Several hundred MB ~ several dozen GB
- Huge volume data place considerable demands on run-time memory space.
- Such data sets need special treatment for effective manipulation.

☞ *Can we interactively visualize a 2GB volume data set on my computer with 128MB of main memory?*

Volume Rendered Visible Man



Visible Human Dataset from NLM

- CT, MRI and RGB cryosection images
- Visible Man
  - Axial scans : 1mm interval, over 1870 cross-sections
  - Cross-sectional images : 512 x 512
  - Ex) Frozen CT 512 x 512 x 1871 x 2 bytes = 935.5MB
- Visible Woman
  - Axial scans : 1/3mm interval

☞ The whole data sets amount to 15 GB to 40 GB.

Research Goal

Design a compression scheme
for very large volume data
suitable for developing interactive applications!

- Load the whole very large volume data, say 2GB, into a main memory of moderate size, say 128MB
- Allows users feel as if they have a main memory of larger sizes
- Helps develop interactive applications for very large volume data on PCs or workstations with limited memory

Two Important Design Factors

- High compression ratio
  - Achieve the best compression rate with minimal distortion in the reconstructed images?
  - Often impose some constraints on random access ability
- Fast run-time random access ability
  - The access patterns in interactive applications change in somewhat complicated ways.

☞ Compromise between these two factors!

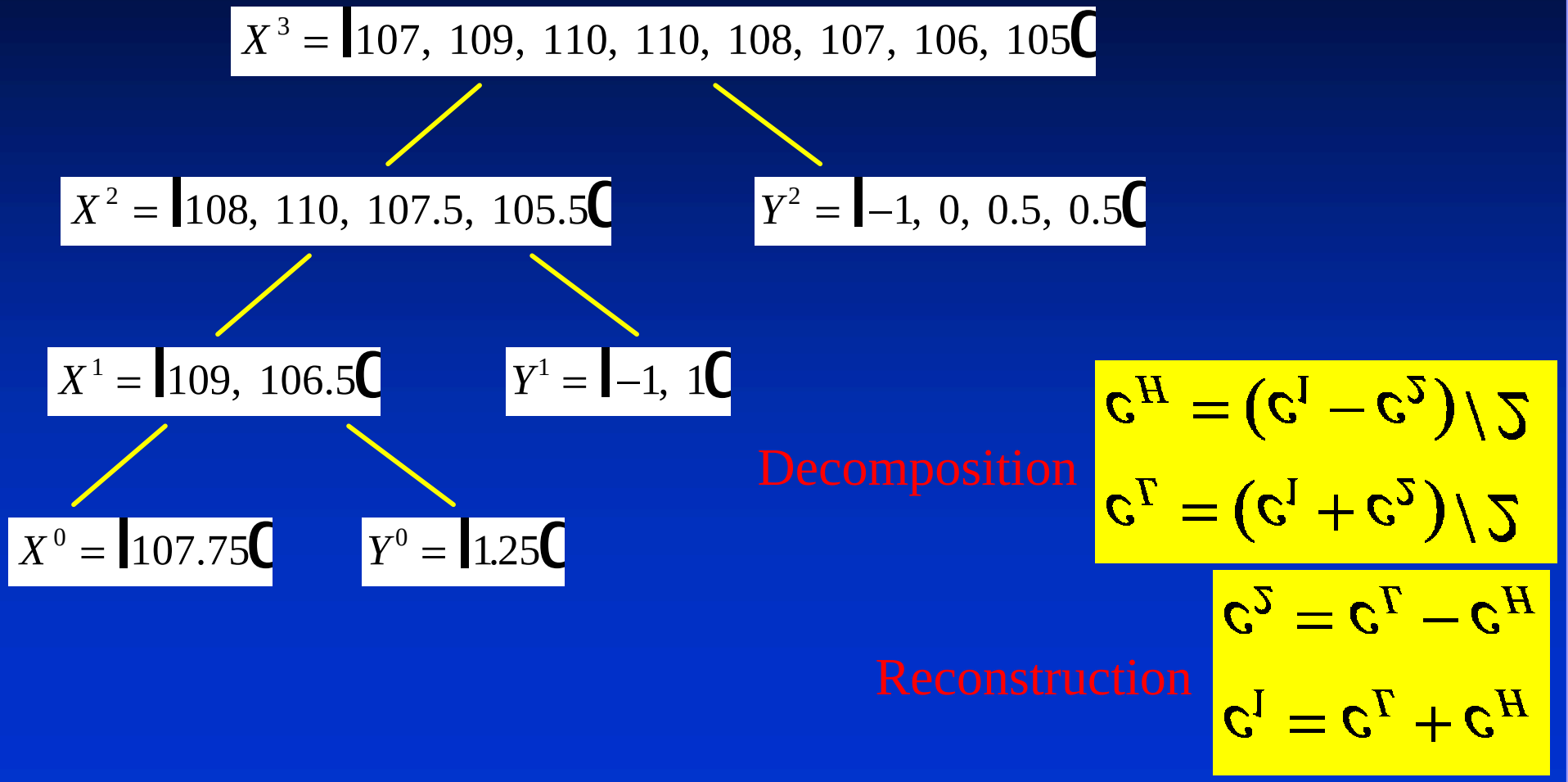
Related Work

- Ning et al. [Vis. 93]
  - Vector quantization
- Muraki [Vis. 92, CG&A 93]
  - 3D Wavelet compression
- Ghavamnia et al. [Vis. 95]
  - Laplacian pyramid
- Yeo et al. [IEEE TVCG 95]
  - DCT-based 3D compression
- Thoma et al. [IEEE Mult. 97]
  - Experimental results on 2D compression of VH

Wavelets

- A mathematical tool for representing functions hierarchically
  - ✓ Allow functions to be represented in multi-resolution forms
  - ✓ Offer a mathematical basis for data compression
  - ✓ Have recently had a great impact in computer graphics

1D Haar Wavelet Transforms

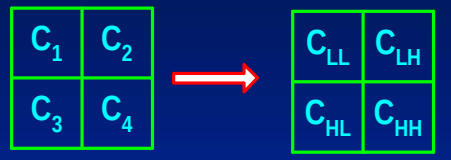


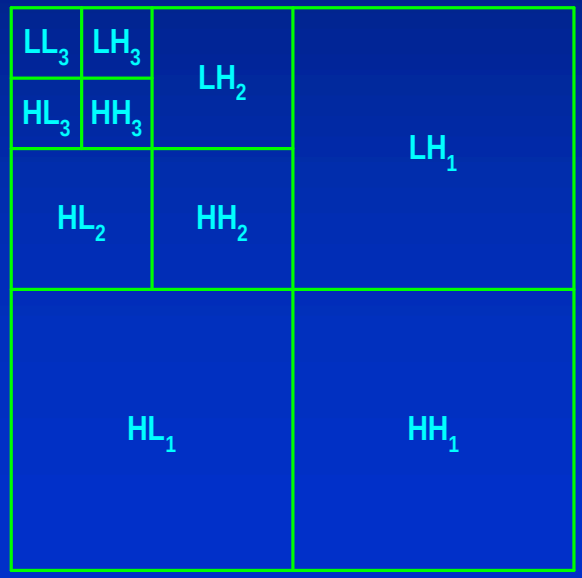
2D Haar Wavelet Transforms

Decomposition

$$c_{LL} = \frac{c_1 + c_2 + c_3 + c_4}{4} \qquad c_{HL} = \frac{c_1 - c_2 + c_3 - c_4}{4}$$

$$c_{LH} = \frac{c_1 + c_2 - c_3 - c_4}{4} \qquad c_{HH} = \frac{c_1 - c_2 - c_3 + c_4}{4}$$





Reconstruction

$$c_1 = c_{LL} + c_{HL} + c_{LH} + c_{HH}$$
$$c_2 = c_{LL} - c_{HL} + c_{LH} - c_{HH}$$
$$c_3 = c_{LL} + c_{HL} - c_{LH} - c_{HH}$$
$$c_4 = c_{LL} - c_{HL} - c_{LH} + c_{HH}$$

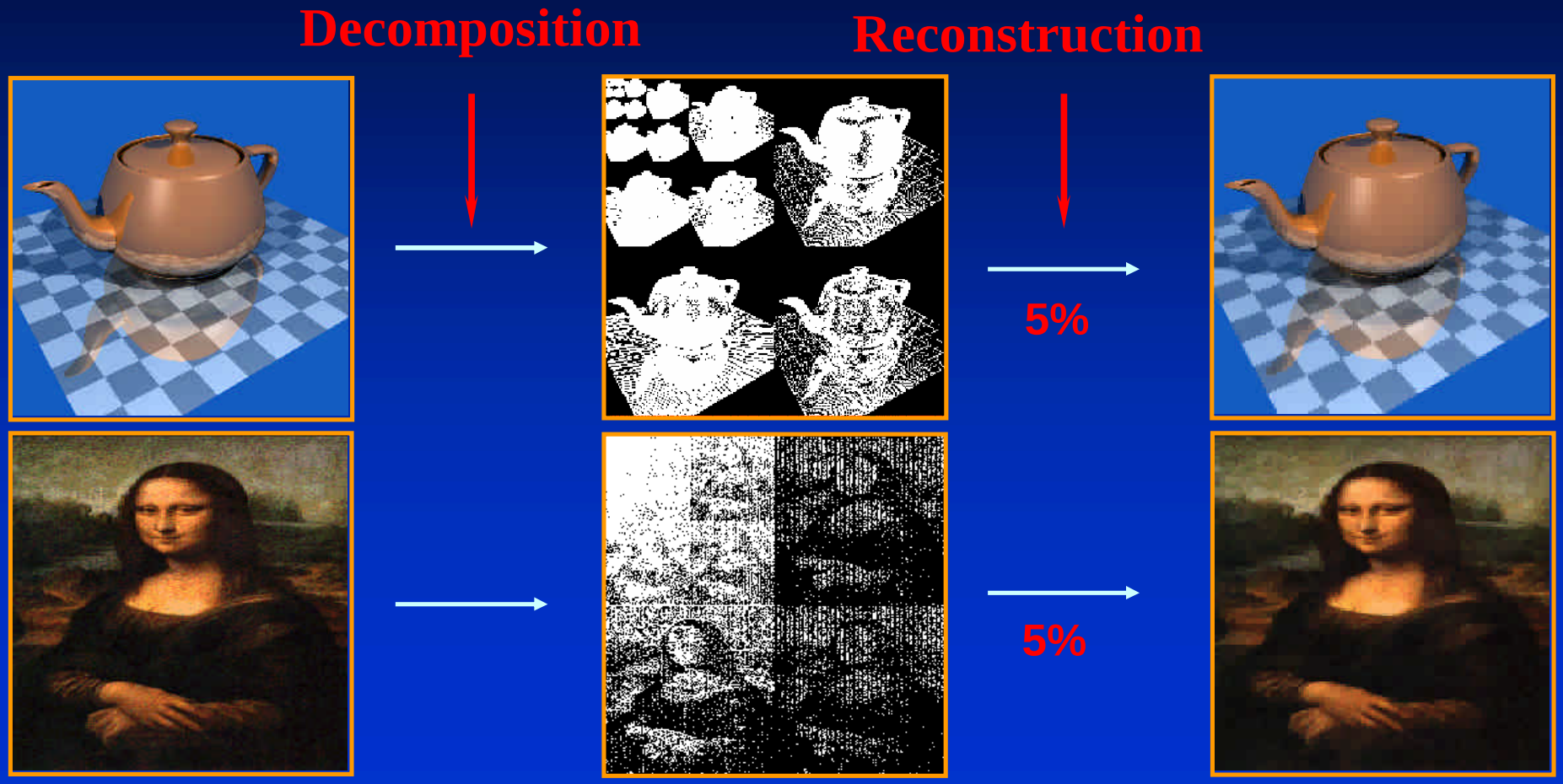
Wavelet-Based Compression

- Idea
  - Decompose an input data set through wavelet transforms.
  - Sort the wavelet coefficients in order of decreasing magnitude.
  - Given an error measure, delete as many coefficients with smaller magnitude as possible.

$$f(x) = \sum_{i=1}^{m} c_i \cdot u_i(x) \quad \Rightarrow \quad \overline{f}(x) = \sum_{i=1}^{\overline{m}} \overline{c}_i \cdot \overline{u}_i(x) \quad (\overline{m} \le m)$$

☞ This is the best choice for orthonormal bases under the L2 norm.

Ex: 2D Image Compression

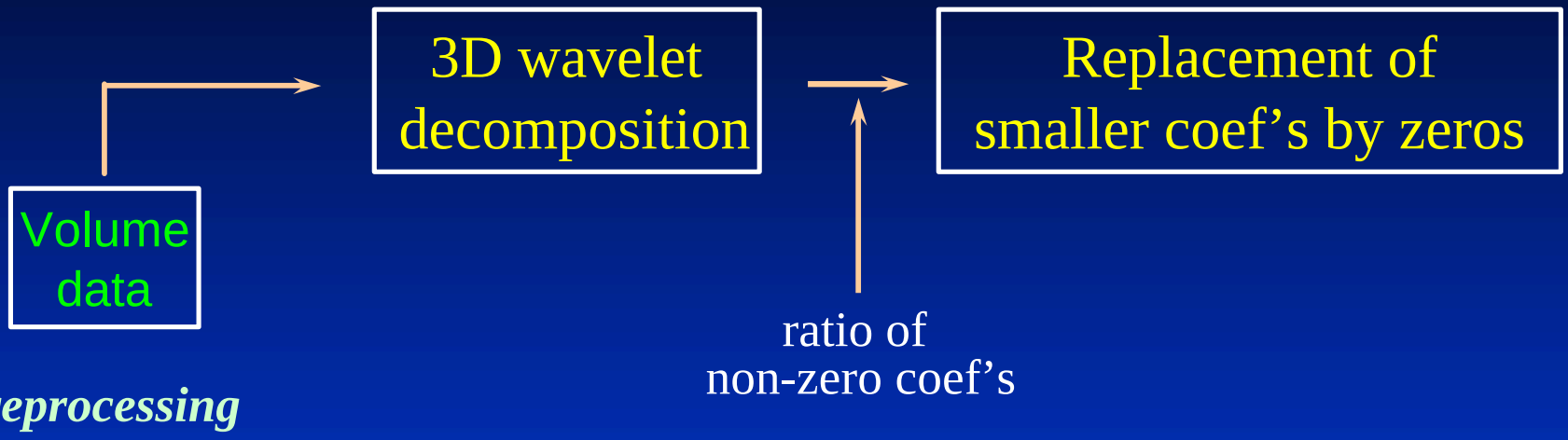


3D Haar Wavelet Transforms

Decomposition

$$C_{LLL} = (C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_7 + C_8)/8$$
$$C_{LLH} = (C_1 + C_2 + C_3 + C_4 - C_5 - C_6 - C_7 - C_8)/8$$
$$C_{LHL} = (C_1 + C_2 - C_3 - C_4 + C_5 + C_6 - C_7 - C_8)/8$$
$$C_{LHH} = (C_1 + C_2 - C_3 - C_4 - C_5 - C_6 + C_7 + C_8)/8$$
$$C_{HLL} = (C_1 - C_2 + C_3 - C_4 + C_5 - C_6 + C_7 - C_8)/8$$
$$C_{HLH} = (C_1 - C_2 + C_3 - C_4 - C_5 + C_6 - C_7 + C_8)/8$$
$$C_{HHL} = (C_1 - C_2 - C_3 + C_4 + C_5 - C_6 - C_7 + C_8)/8$$
$$C_{HHH} = (C_1 - C_2 - C_3 + C_4 - C_5 + C_6 + C_7 - C_8)/8$$

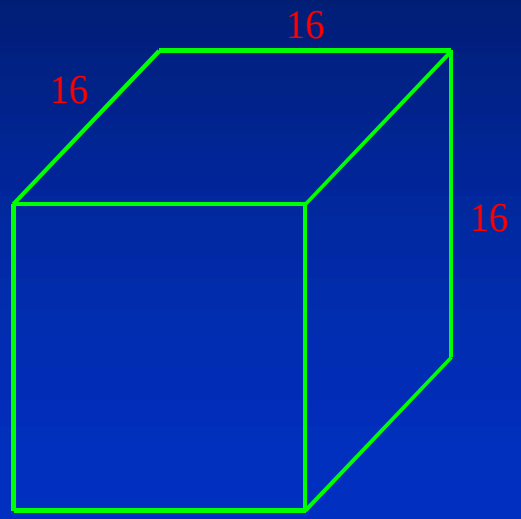
3D Wavelet Compression



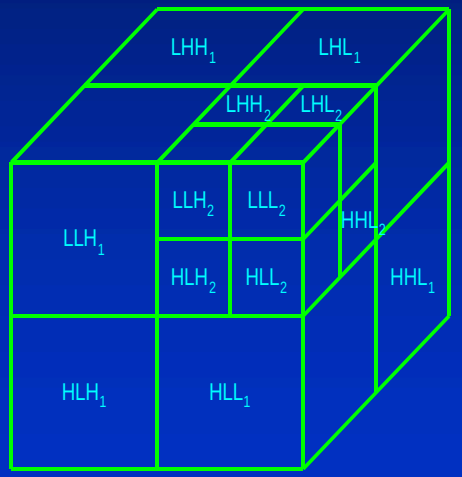
- How do we encode and store the necessary information in a smaller number of bits?
  - Zerotree [Shapiro IEEE TSP 93, Said et al. IEEE Symp. Cir. & Sys. 93, …]

3D Haar Wavelet Decomposition

- Apply the 3D wavelet transform twice.

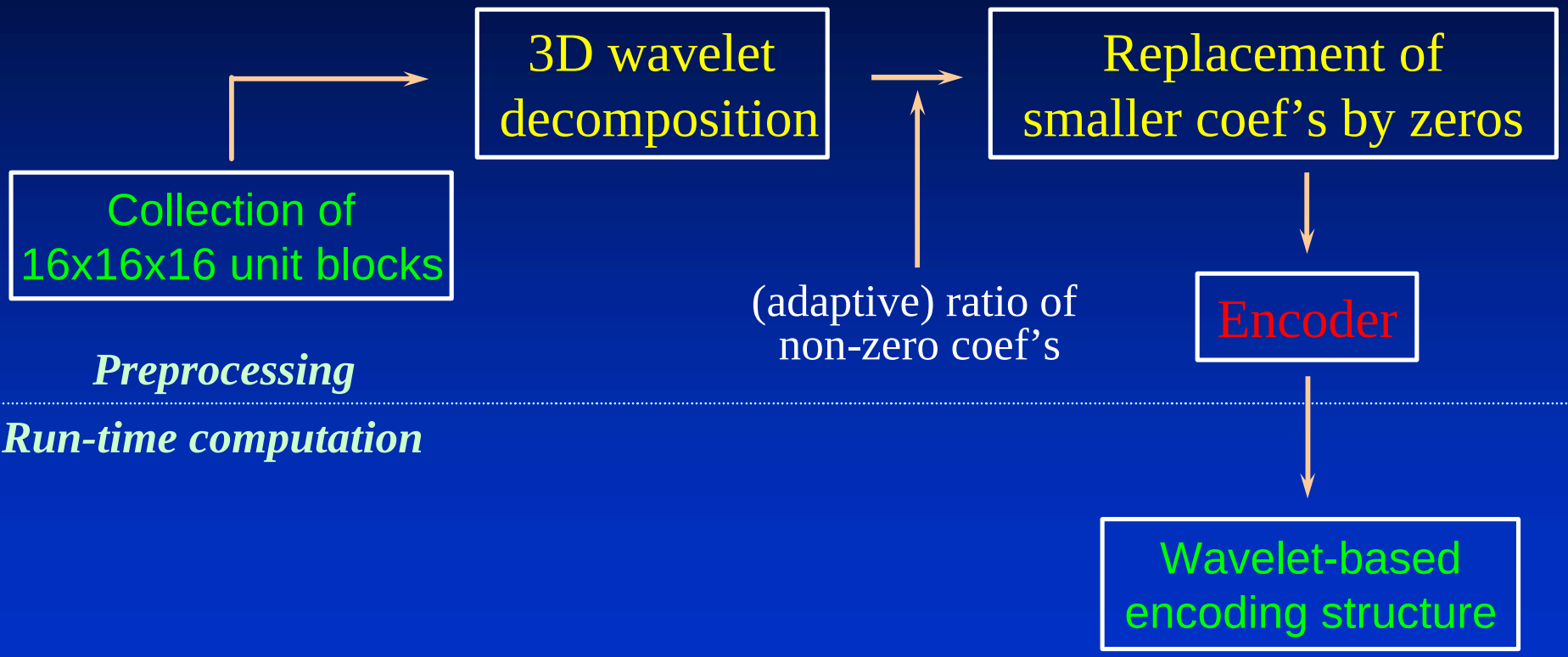


A Unit Block

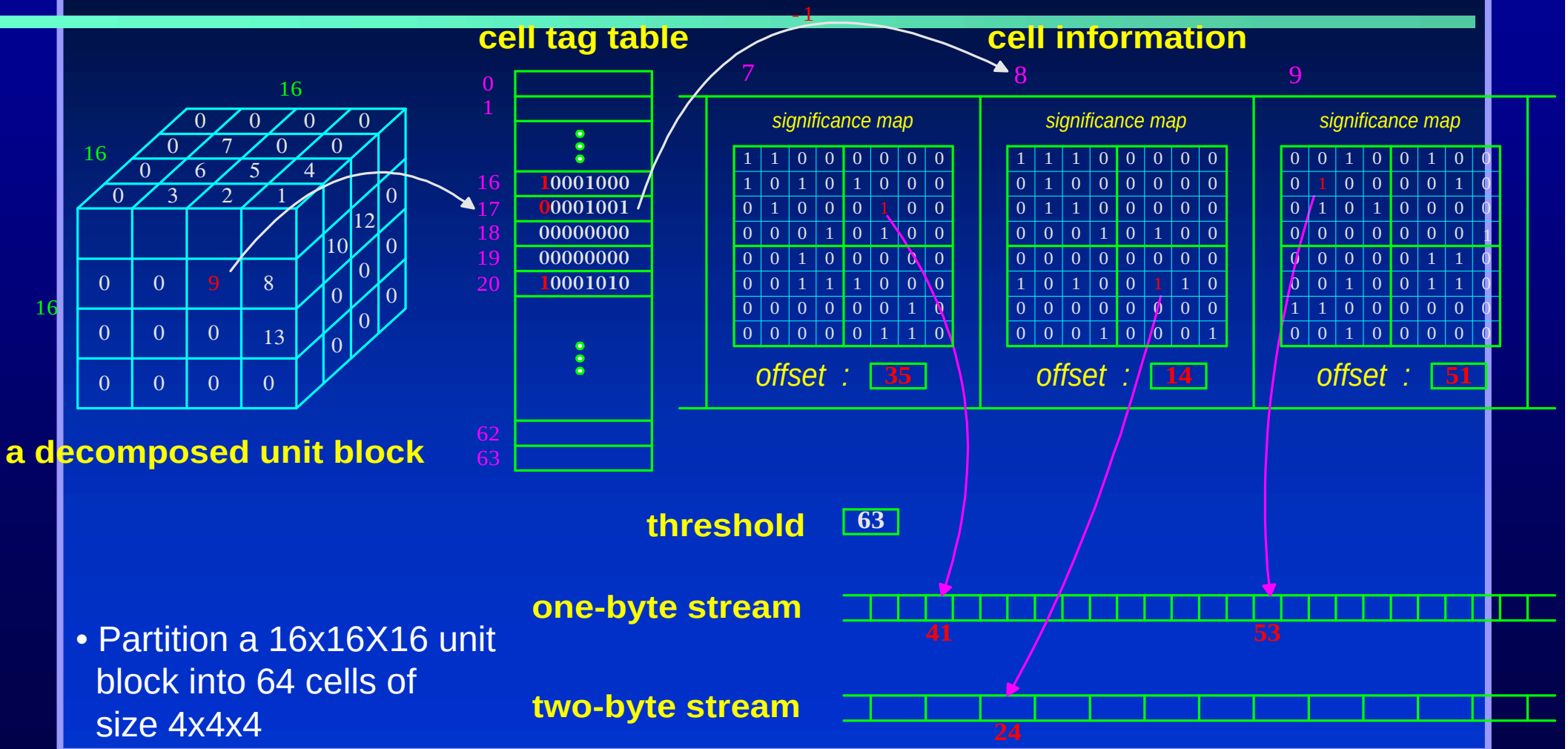


A Decomposed Unit Block

New Compression Scheme

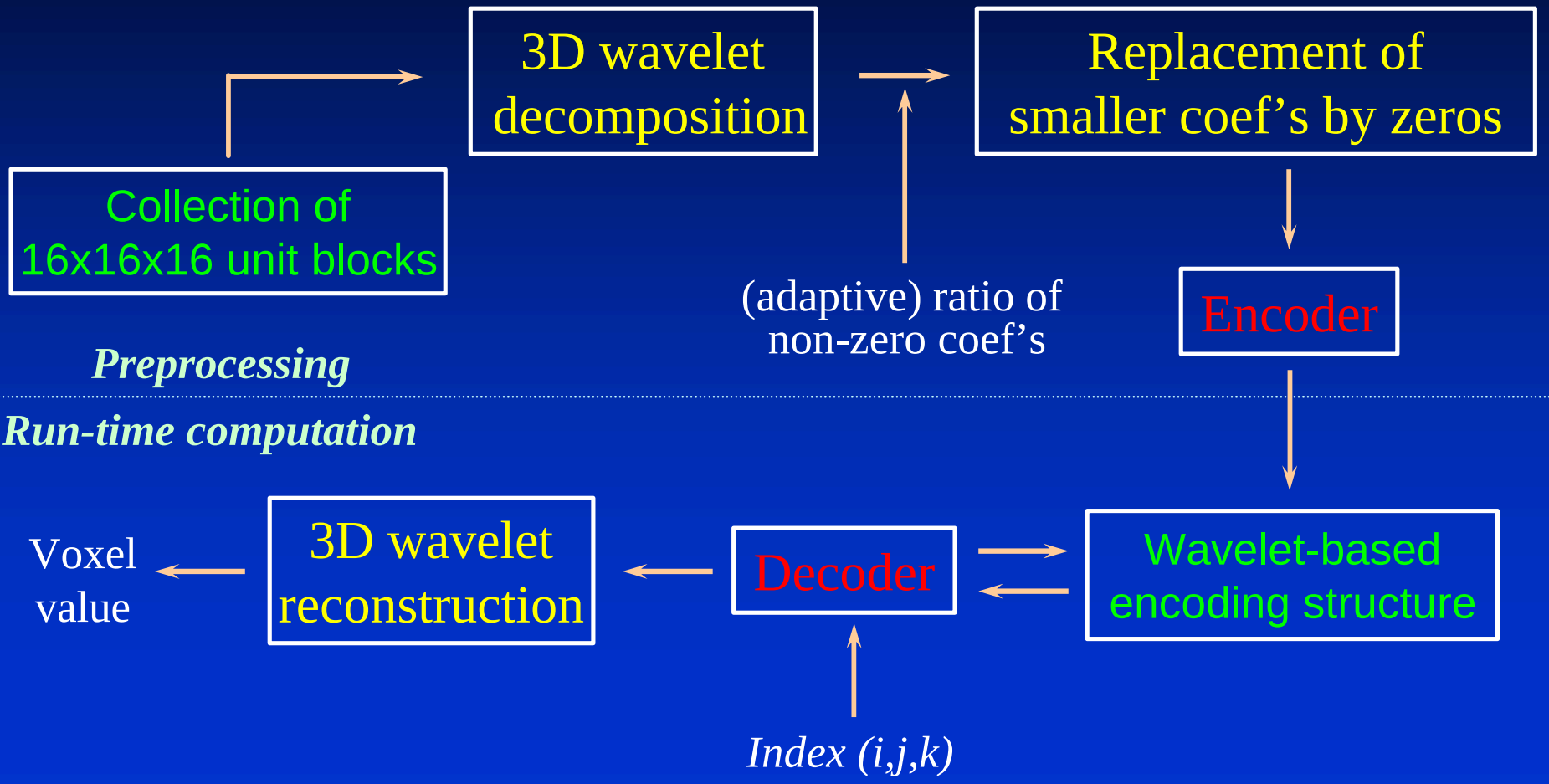


Encoding Scheme



- Partition a 16x16X16 unit block into 64 cells of size 4x4x4

New Compression Scheme



Reconstruction

$$c_1 = c_{LLL} + c_{LLH} + c_{LHL} + c_{LHH} + c_{HLL} + c_{HLH} + c_{HHL} + c_{HHH}$$

$$c_2 = c_{LLL} + c_{LLH} + c_{LHL} + c_{LHH} - c_{HLL} - c_{HLH} - c_{HHL} - c_{HHH}$$

$$c_3 = c_{LLL} + c_{LLH} - c_{LHL} - c_{LHH} + c_{HLL} + c_{HLH} - c_{HHL} - c_{HHH}$$

$$c_4 = c_{LLL} + c_{LLH} - c_{LHL} - c_{LHH} - c_{HLL} - c_{HLH} + c_{HHL} + c_{HHH}$$

$$c_5 = c_{LLL} - c_{LLH} + c_{LHL} - c_{LHH} + c_{HLL} - c_{HLH} + c_{HHL} - c_{HHH}$$

$$c_6 = c_{LLL} - c_{LLH} + c_{LHL} - c_{LHH} - c_{HLL} + c_{HLH} - c_{HHL} + c_{HHH}$$

$$c_7 = c_{LLL} - c_{LLH} - c_{LHL} + c_{LHH} + c_{HLL} - c_{HLH} - c_{HHL} + c_{HHH}$$

$$c_8 = c_{LLL} - c_{LLH} - c_{LHL} + c_{LHH} - c_{HLL} + c_{HLH} + c_{HHL} - c_{HHH}$$

Reconstruction Scheme

⓪ All the necessary wavelet coefficients are first decoded from encoded unit block.

① The reconstruction formula is applied twice.

- ☞ When a specific voxel is reconstructed,
  - – 15 decoding operations
  - – 14 +/- operations for reconstruction
- ☞ When 64 voxels in a cell are reconstructed simultaneously,
  - – 64 decoding operations: one per voxel
  - – 24 +/- operations for reconstruction : 3 and 3/4 per voxel

Decoding Algorithm

- Input : a 16x16x16 encoded unit block and an index *(i, j, k)*
- Output : the decoded value of the voxel with index *(i, j, k)*

⓪ Find the cell C that contains the index *(i, j, k)*.
① If the cell tag for C is 0, return 0. [case1]
② Compute the relative index *(i', j', k')* in C.
③ If the bit-flag for *(i', j', k')* is 0, return 0. [case2]
④ Count the number of preceding non-zero coefficients by table access.
⑤ Add the displacement to the offset to compute the correct address.
⑥ Access the appropriate data stream, and return the value. [case3]

☞ Most cases of decoding belong in either [case1] or [case2] → Decoding is cheap !

Analysis of Compression Ratio

$$\frac{1}{r} = \frac{2 + 64 + (8+2)\cdot 64 \cdot a + 16^3 \cdot g \cdot (b + 2\cdot(1-b))}{16^3 \cdot 2}$$

$$\approx 0.008057 + 0.078125 \cdot a + g \cdot (1 - 0.5 \cdot b)$$

r = **compression ratio**
a = **# of non-null cells / 64**
b = **# of coef's in one-byte stream / # of all non-zero coef's used**
g = **ratio of non-zero wavelet coef's**

Experiments

- H/W
  - SGI Octane with 175 MHz R10000 CPU
- Test data
  - Preprocessed fresh CT data of Visible Man
  - 512 x 512 x 1440 x 2 bytes = 720 MB
  - 92160 (= 32 x 32 x 90) 16 x 16 x 16 unit blocks

Adaptive Distribution of Nonzero Coef's

- $\bar{g}$: a desired ratio of nonzero coef's to be used

⓪ Apply the wavelet transforms to each unit block i, and compute the ratio r_i of nonzero coefficients.

① For the unit block i,

allocate $n_i = \frac{r_i}{\sum_j r_j} \cdot 512^2 \cdot 1440 \cdot \bar{g}$ coefficients.

② Compute the corresponding threshold value for each unit block.

Compression Ratio

	3%	5%	7%	10%	15%
Compressed Data Size (MB)	26.31	38.27	48.98	63.69	85.99
Compression Ratio	27.4 : 1	18.8 : 1	14.7 : 1	11.3 : 1	8.4 : 1
a	0.1156	0.1901	0.2481	0.3165	0.4014
b	0.6559	0.7327	0.7771	0.8185	0.8600
g	0.0293	0.0481	0.0667	0.0946	0.1407

α = # of non-null cells / 64

β = # of coef's in one-byte stream / # of all non-zero coef's used

γ = ratio of non-zero wavelet coef's

Compression Error

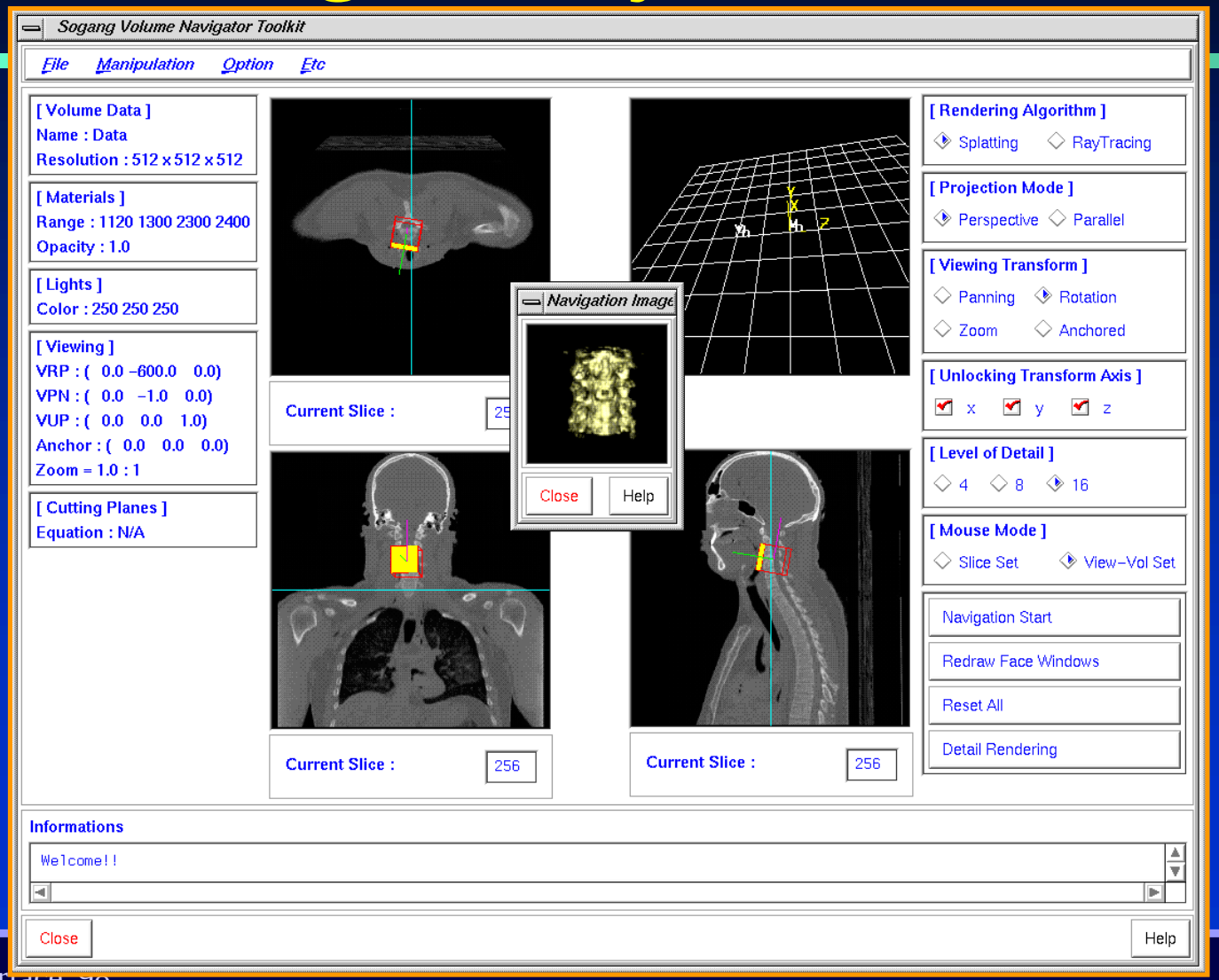
		3%	5%	7%	10%	15%
Errors in Voxel Values	SNR (dB)	22.6	25.9	28.6	32.0	36.4
	PSNR (dB)	44.5	47.8	50.4	53.8	58.2
Errors in Normals	Skin (deg)	15.4	11.4	8.6	6.0	3.9
	Bone (deg)	24.7	19.0	15.0	10.8	6.7

Random Access Time

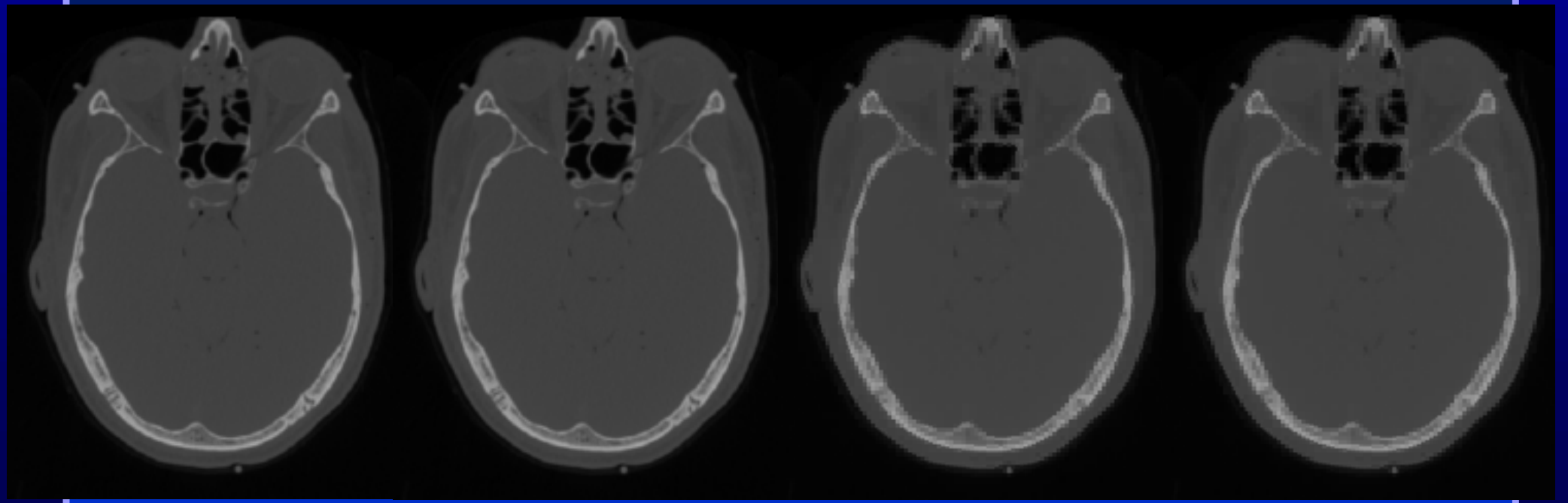
		Uncompressed	3%	5%	7%	10%	15%
Pure Random		2.18	7.75	8.20	8.62	9.15	9.84
Cell-Wise	All	19.95	30.31	31.58	32.52	33.62	35.05
	Skin	6.70	10.13	10.75	11.32	12.10	13.02

Test data : 512x512x512
Cell-wise (all) : all cells
Pure random: 1M random accesses
Cell-wise (skin) : cells classified as skin
43M voxels

Volume Navigation System

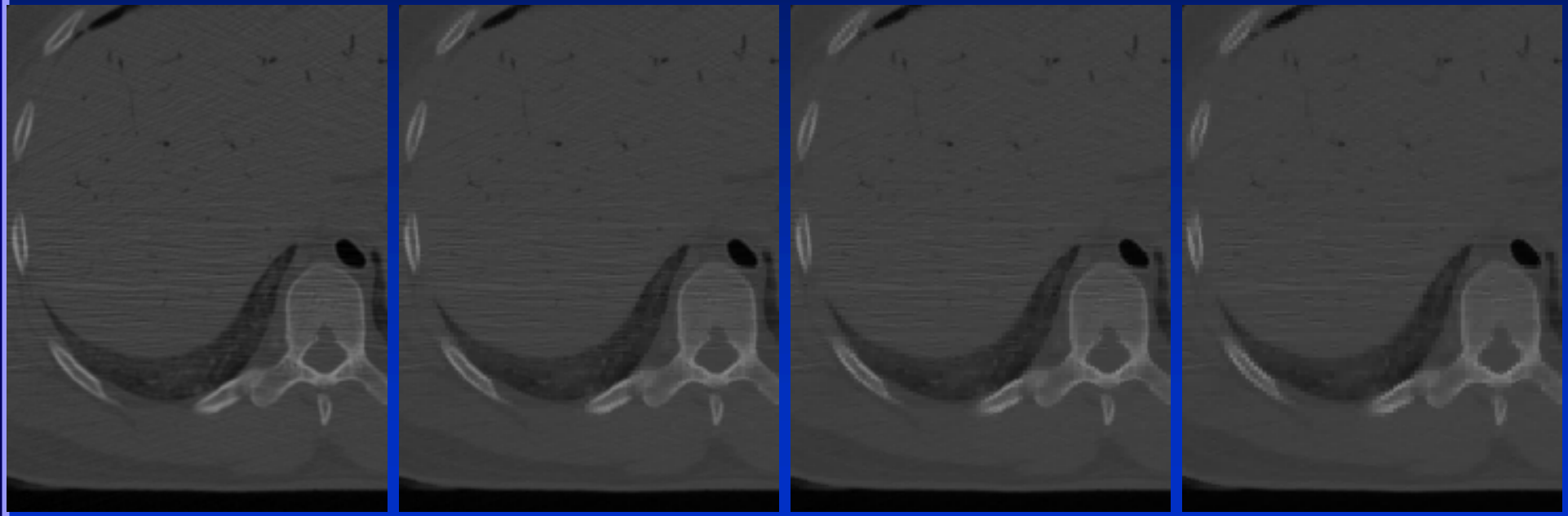


Sample Slices (I)



Uncompressed	14.7 : 1	18.8 : 1	27.4 : 1
720MB	48.98MB	38.27MB	26.31MB

Sample Slices (II)



Uncompressed	14.7 : 1	18.8 : 1	27.4 : 1
720MB	48.98MB	38.27MB	26.31MB

Ray-Cast Images (I)

Uncompressed
720MB

14.7 : 1
48.98MB

Ray-Cast Images (II)

Uncompressed
720MB

18.8 : 1
38.27MB

Ray-Cast Images (III)

Uncompressed
720MB

27.4 : 1
26.31MB

Ray-Cast Images (IV)

11.3 : 1
63.69MB

14.7 : 1
48.98MB

Ray-Cast Images (V)

18.8 : 1
38.27MB

27.4 : 1
26.31MB

Conclusions

- An effective 3D compression scheme for very large volume data
  - 3D wavelet-based compression (multi-resolution)
  - Fairly fast random access and good compression ratio
  - Visualization on computer with limited memory

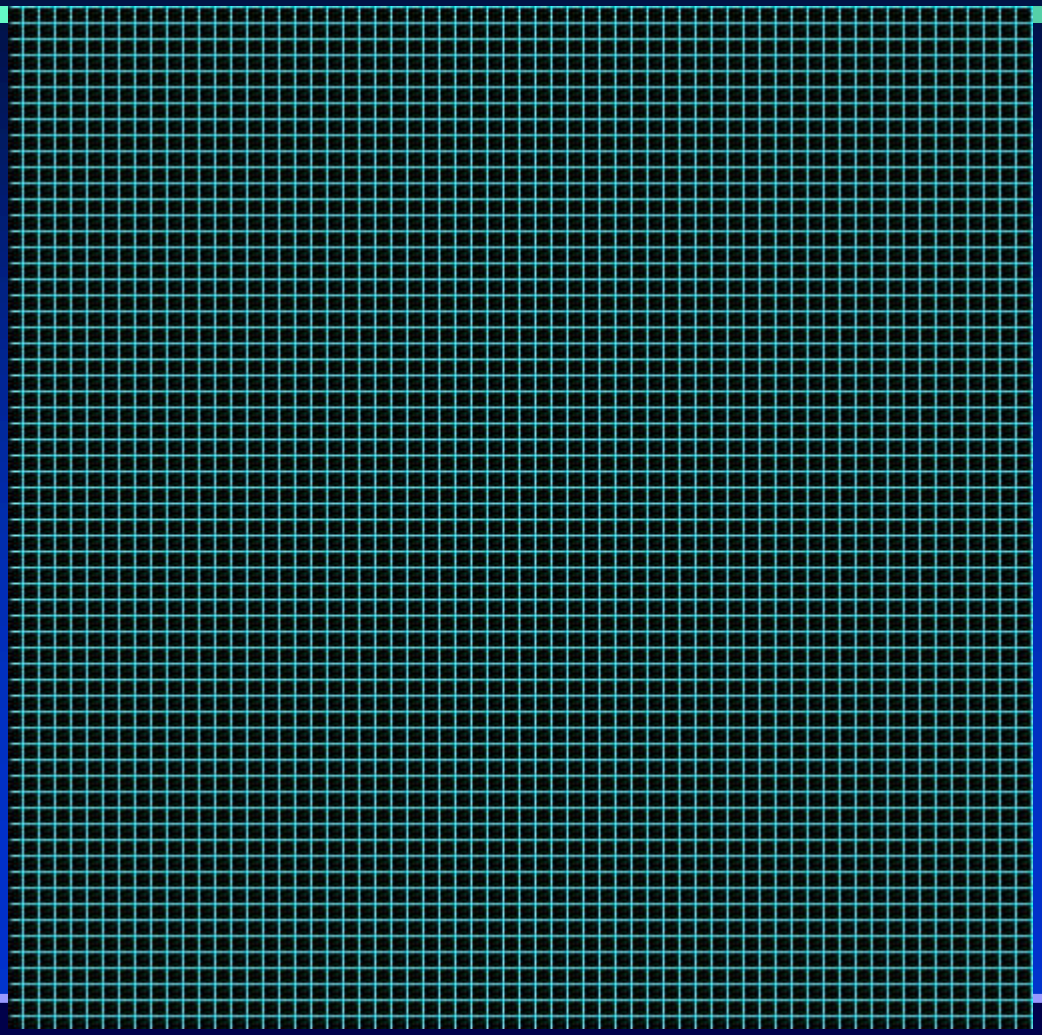
Future Works

- Performance improvement
- Application to cryosection RGB images
- Test with other filters
- Application development
  - Interactive virtual navigation for human body
  - Parallel volume rendering
  - 3D fusion of CT/MRI and RGB data
  - Etc.

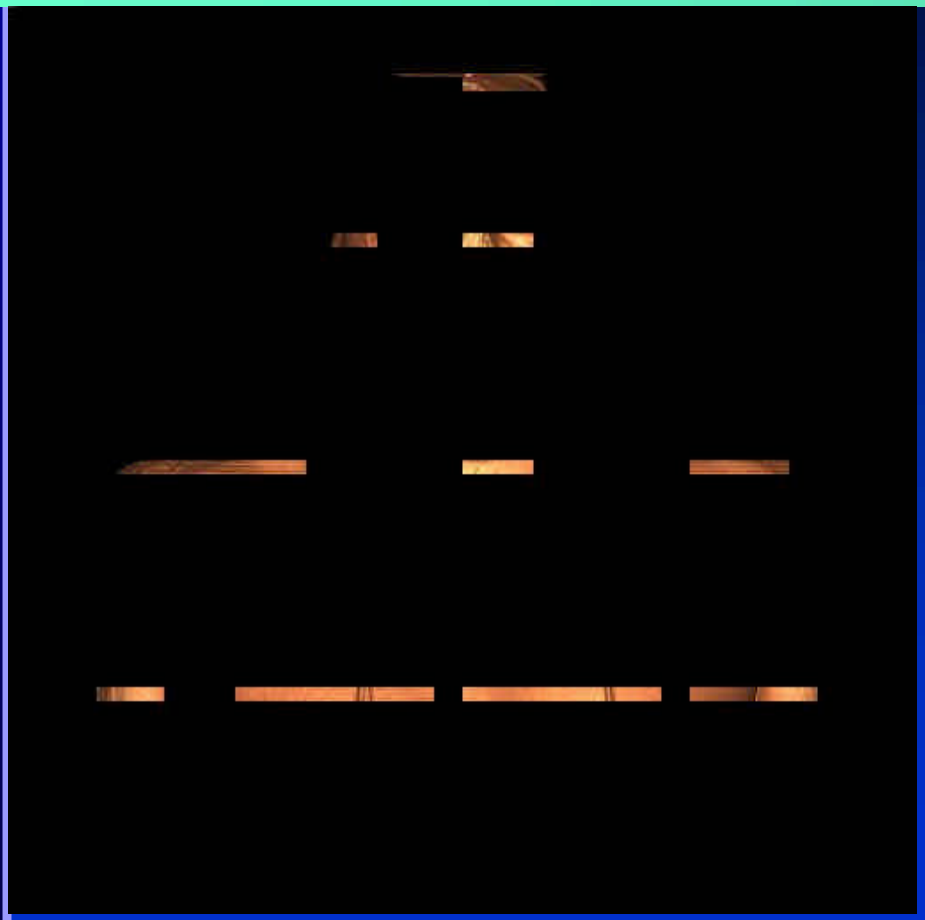
Parallel Volume Rendering on CRAY T3E

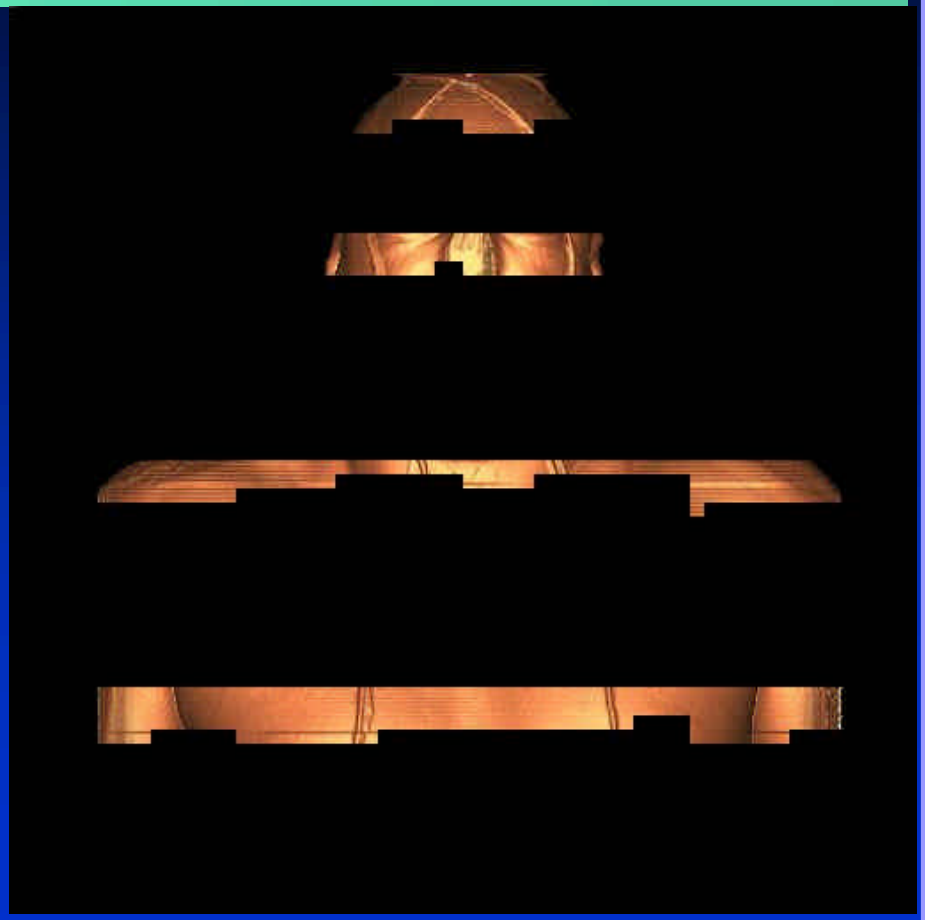
- Distributed memory parallel computer
- SHMEM (Shared Memroy Access routines)
- Based on our 3D encoding scheme
- Ray casting
- Image space partition
- No need for volume data redistribution

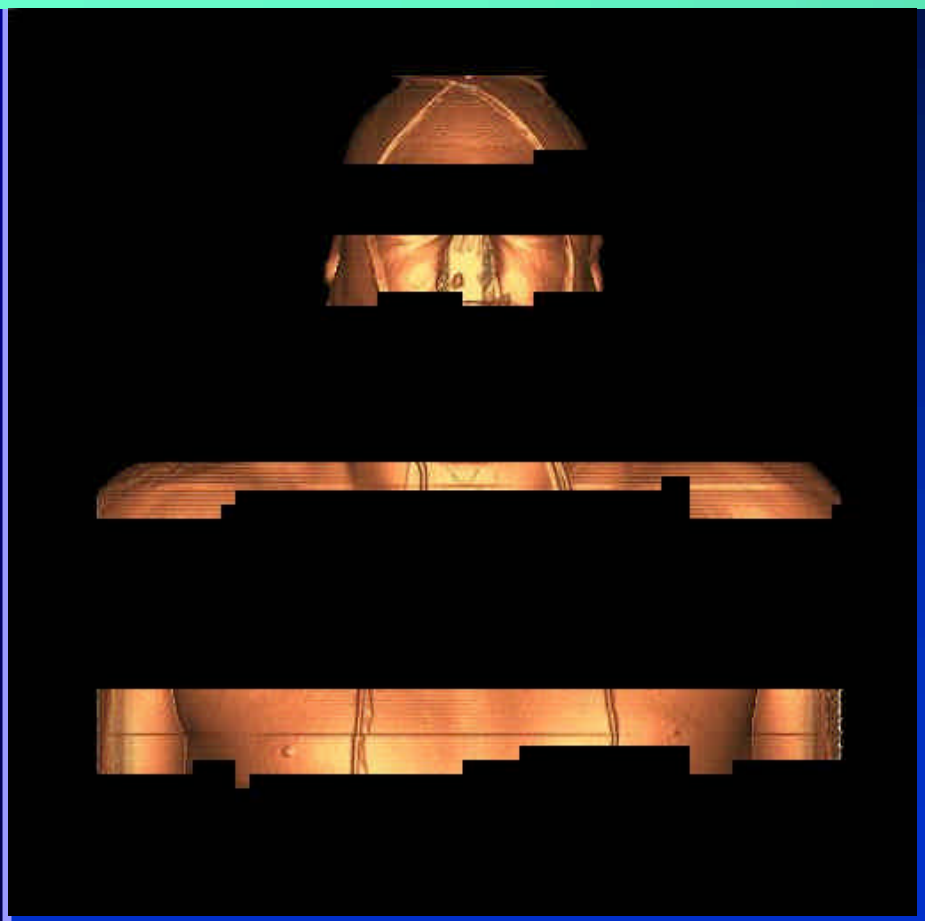
Task Partition on Image Plane

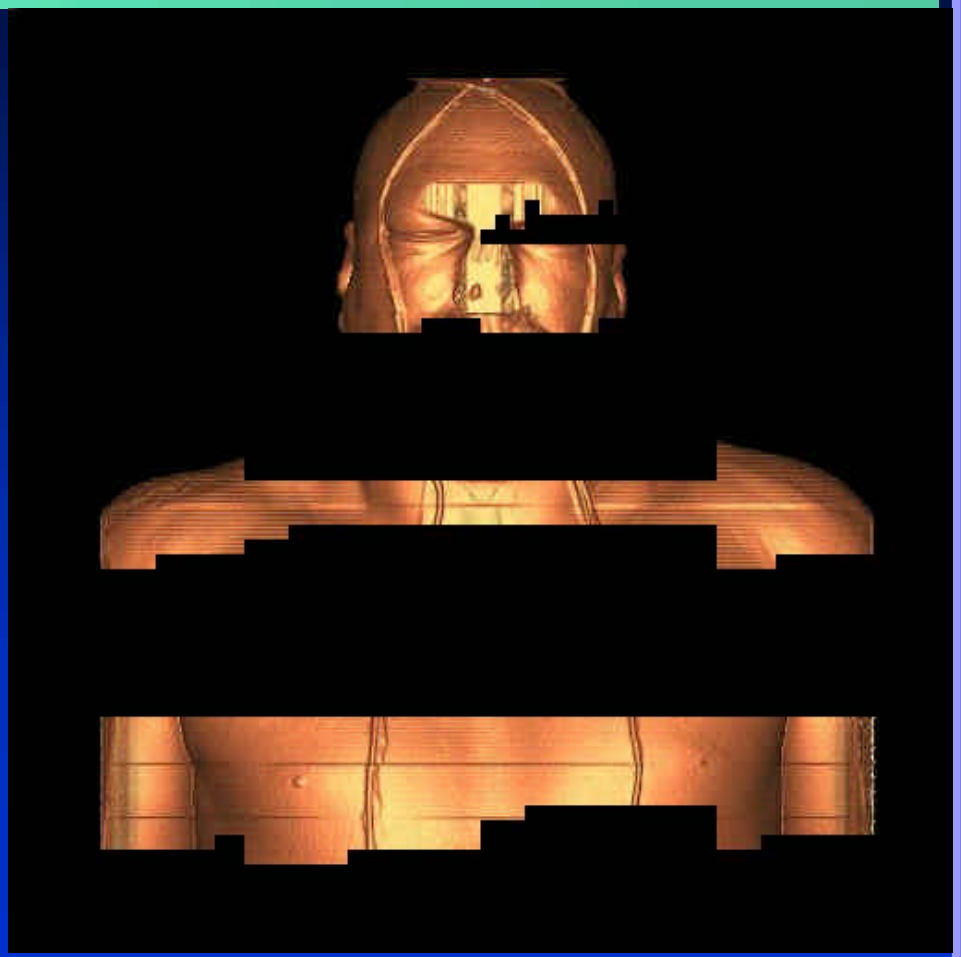


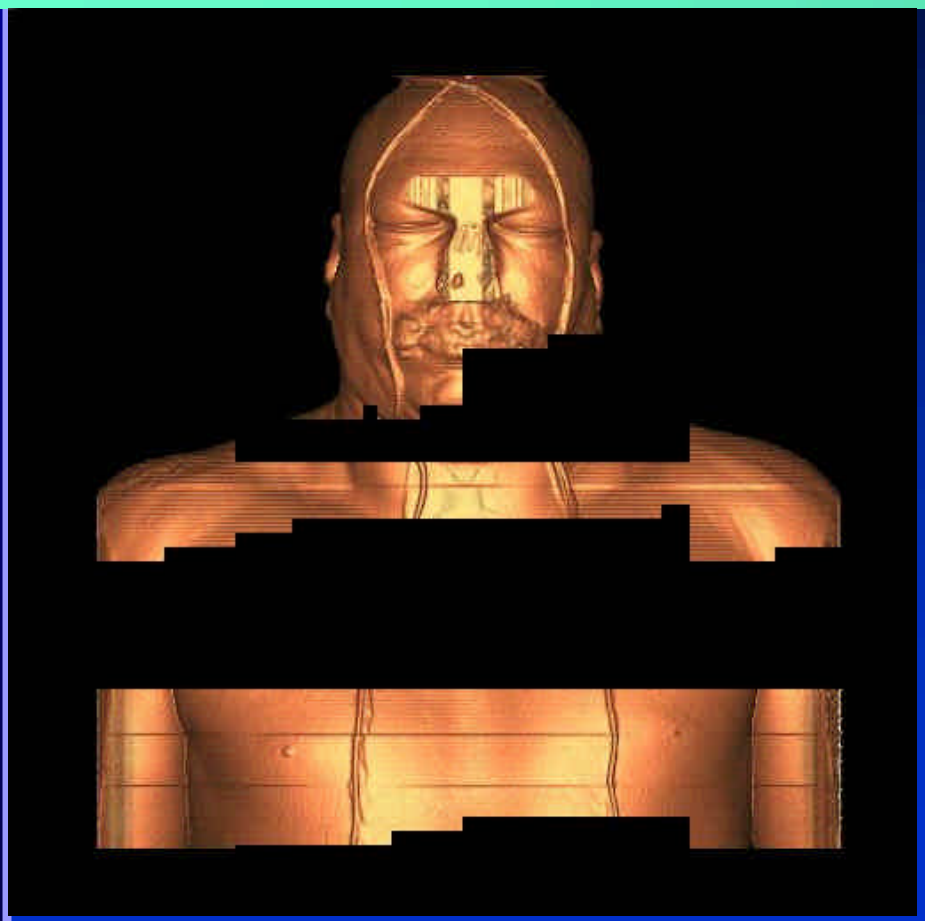
Sample Run

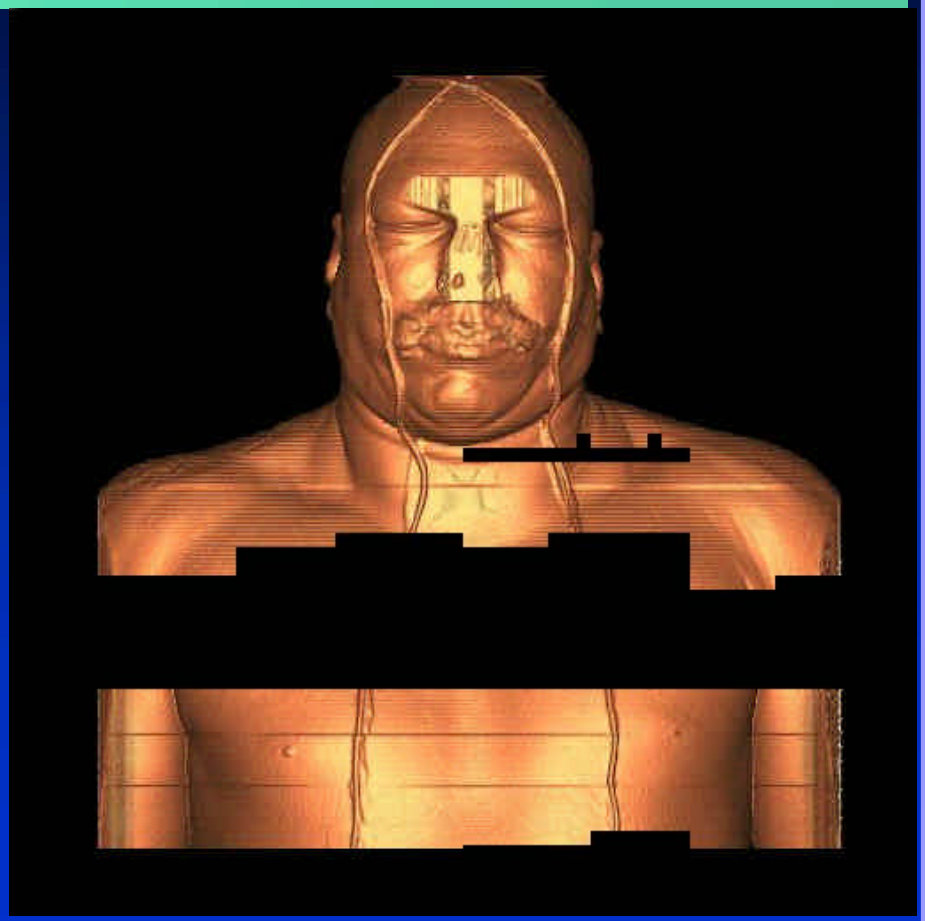


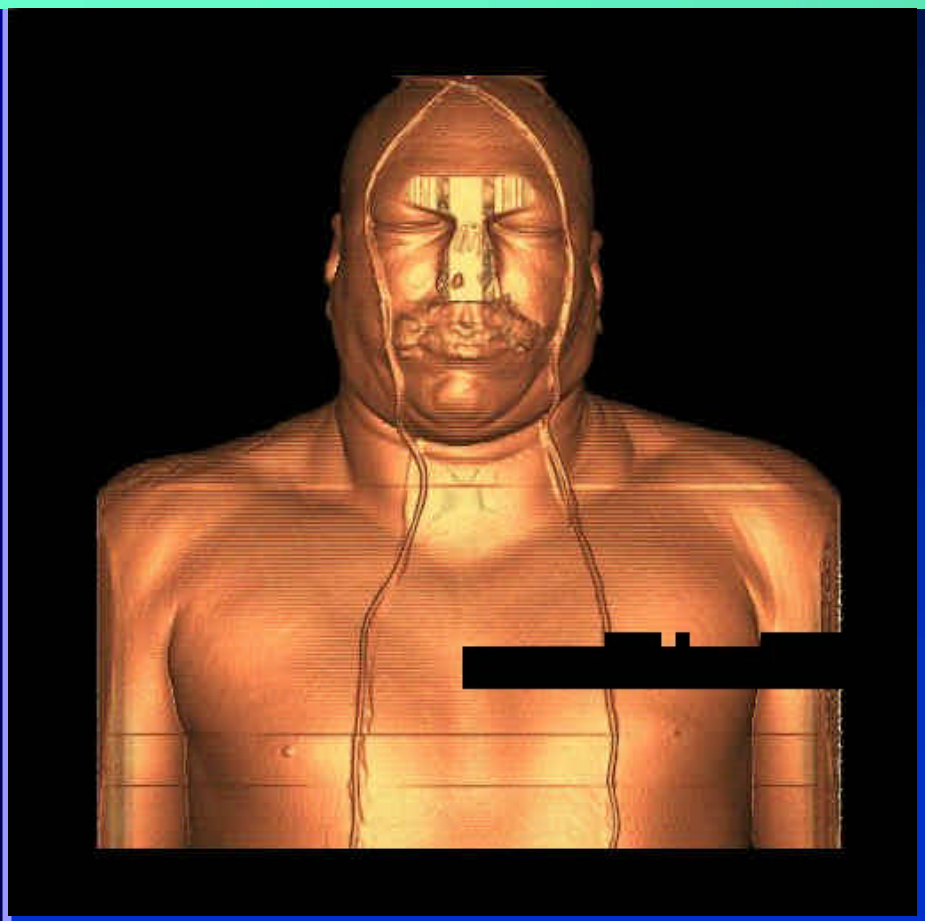


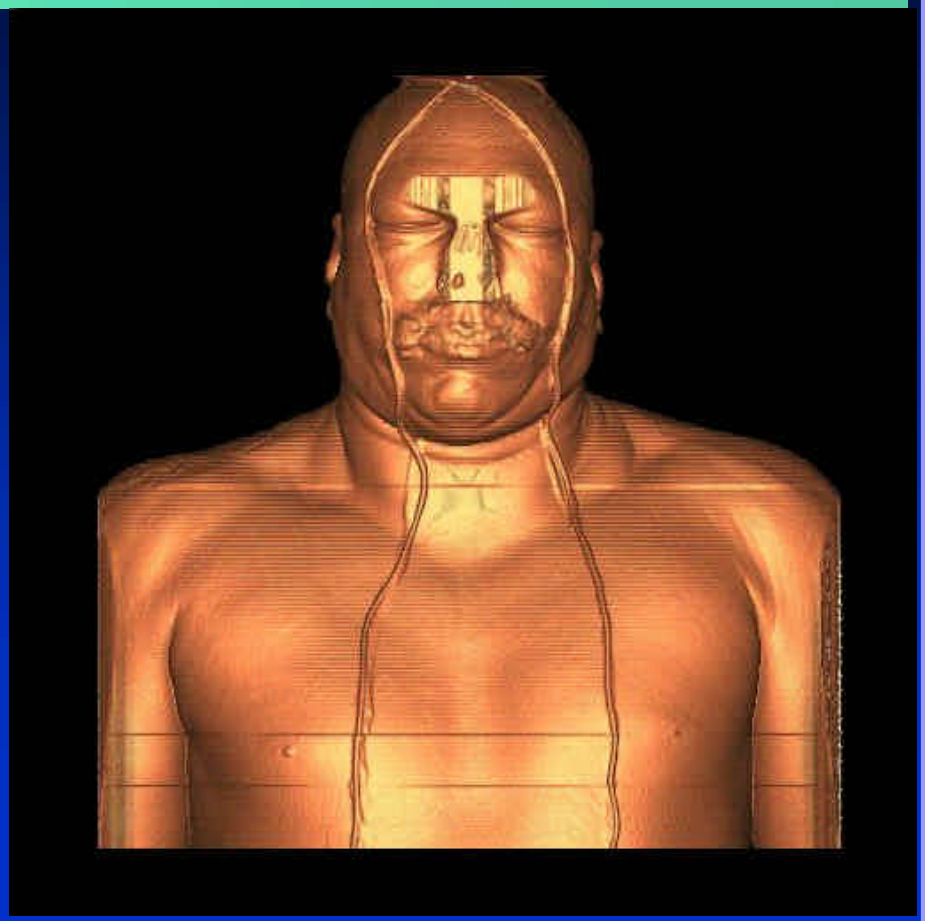


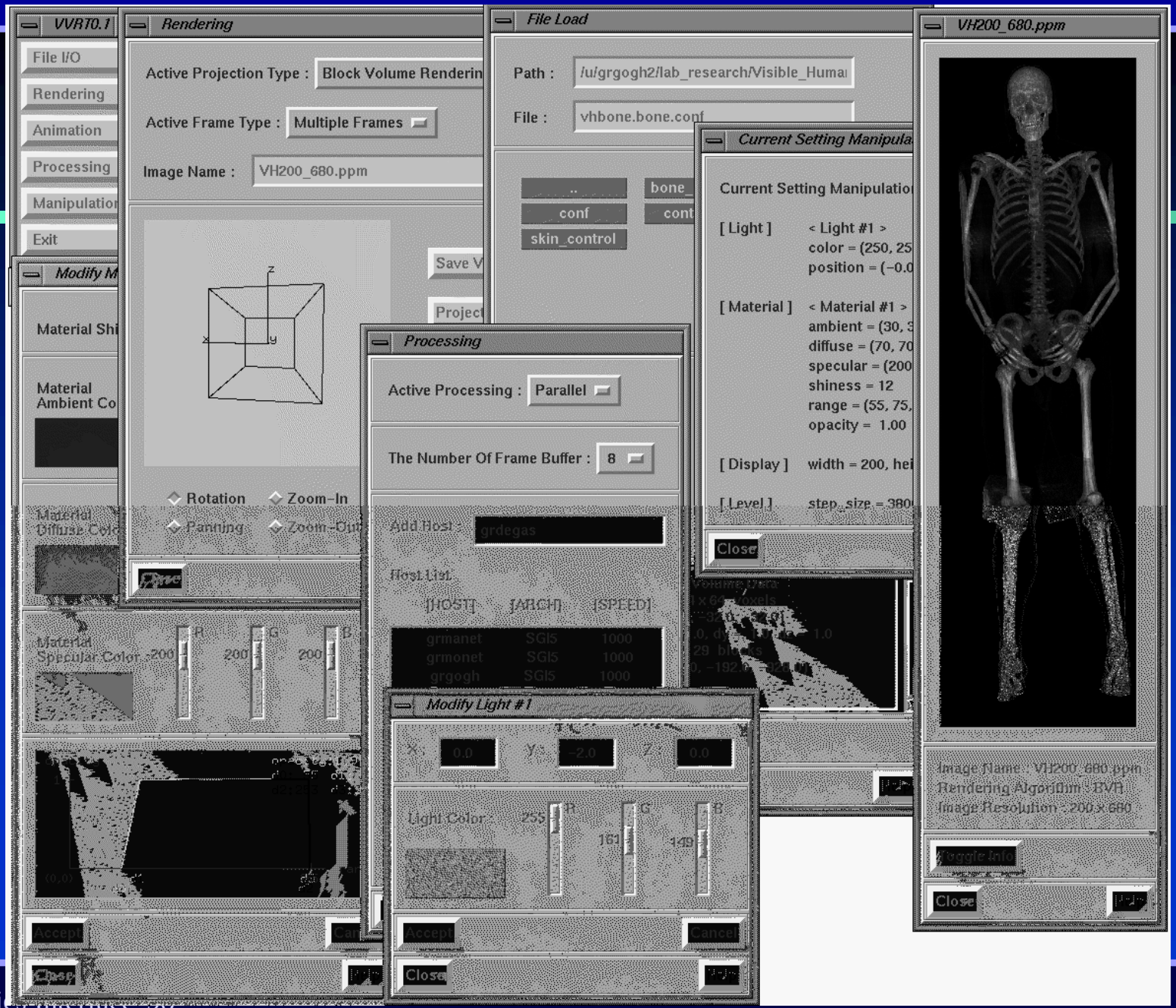











VVRT0.1
File I/O
Rendering
Animation
Processing
Manipulation
Exit
Rendering
Active Projection Type :
Block Volume Renderin
Active Frame Type :
Multiple Frames
Image Name :
VH200_680.ppm
Save V
Project
Rotation
Zoom-In
Panning
Zoom-Out
File Load
Path :
/u/grgogh2/lab_research/Visible_Huma
File :
vhbone.bone.conf
..
conf
skin_control
bone_
cont
Current Setting Manipula
Current Setting Manipulatio
[Light]
< Light #1 >
color = (250, 25
position = (-0.0
[Material]
< Material #1 >
ambient = (30, 3
diffuse = (70, 70
specular = (200
shiness = 12
range = (55, 75,
opacity = 1.00
[Display]
width = 200, hei
[Level]
step_size = 380
Close
VH200_680.ppm
Image Name : VH200_680.ppm
Rendering Algorithm : BVH
Image Resolution : 200 x 680
Toggle Info
Close
Modify M
Material Shi
Material Ambient Co
Material Diffuse Colo
Material Specular Color
R
G
B
200
200
200
Processing
Active Processing :
Parallel
The Number Of Frame Buffer :
8
Add Host :
grdegas
Host List
[HOST]
[ARCH]
[SPEED]
grmanet
SGI5
1000
grmonet
SGI5
1000
grgogh
SGI5
1000
Modify Light #1
X :
0.0
Y :
-2.0
Z :
0.0
Light Color :
255
161
149
Accept
Cancel
Close