

EUROVIS
2007

Scientific Visualization &
Computer Graphics



MIP rendering using adjunction pyramids accelerated on graphics hardware

Wladimir J. van der Laan & Andrei C. Jalba & Jos B.T.M. Roerdink

Institute for Mathematics and Computing Science

University of Groningen

P.O. Box 800, 9700 AV Groningen, The Netherlands

May 2007



- **Maximum Intensity Projection**: Compute maximum along the line of sight
- Scanner precision increases, resulting in **larger datasets**
- **Interactivity** is important for depth perception in MIP
- Efficient **incremental** multiresolution method using morphological pyramids

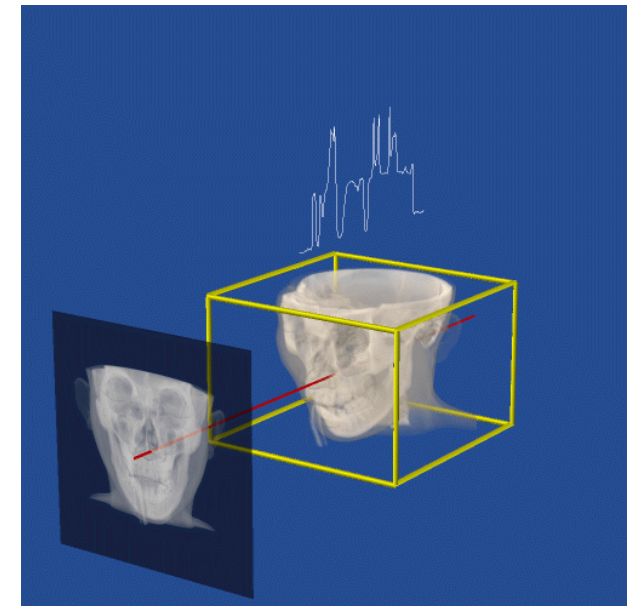
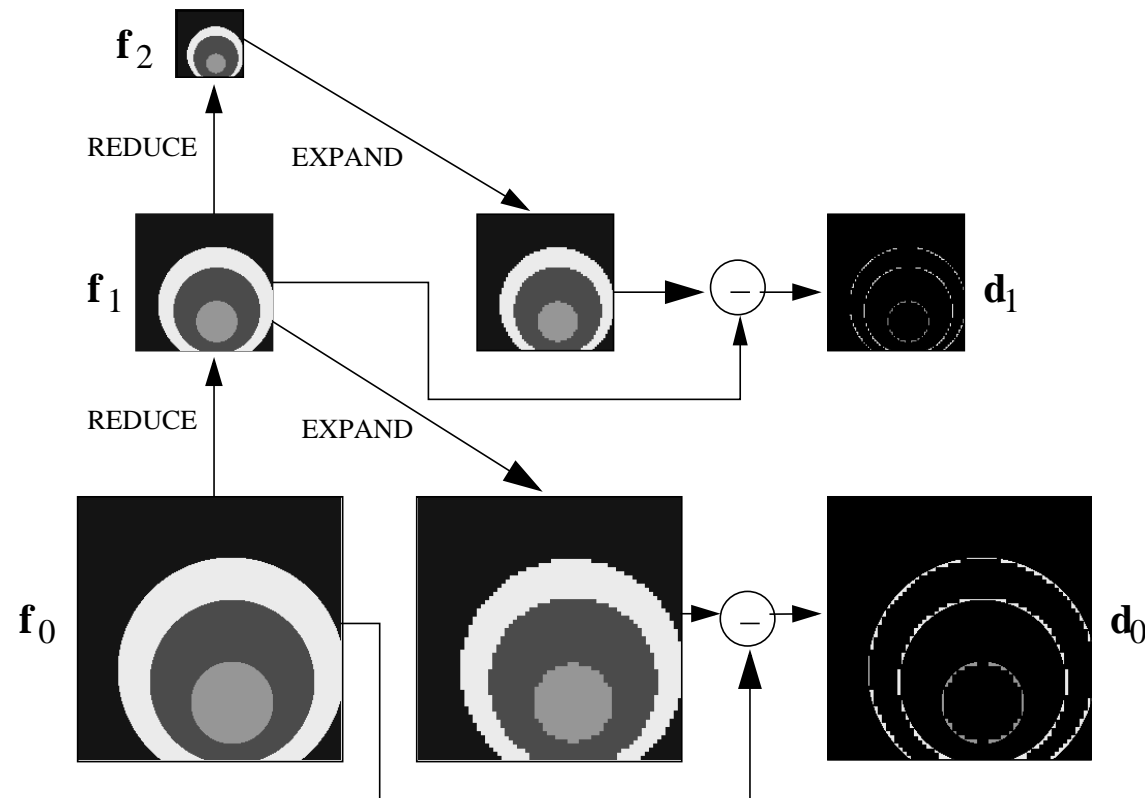




Image pyramids

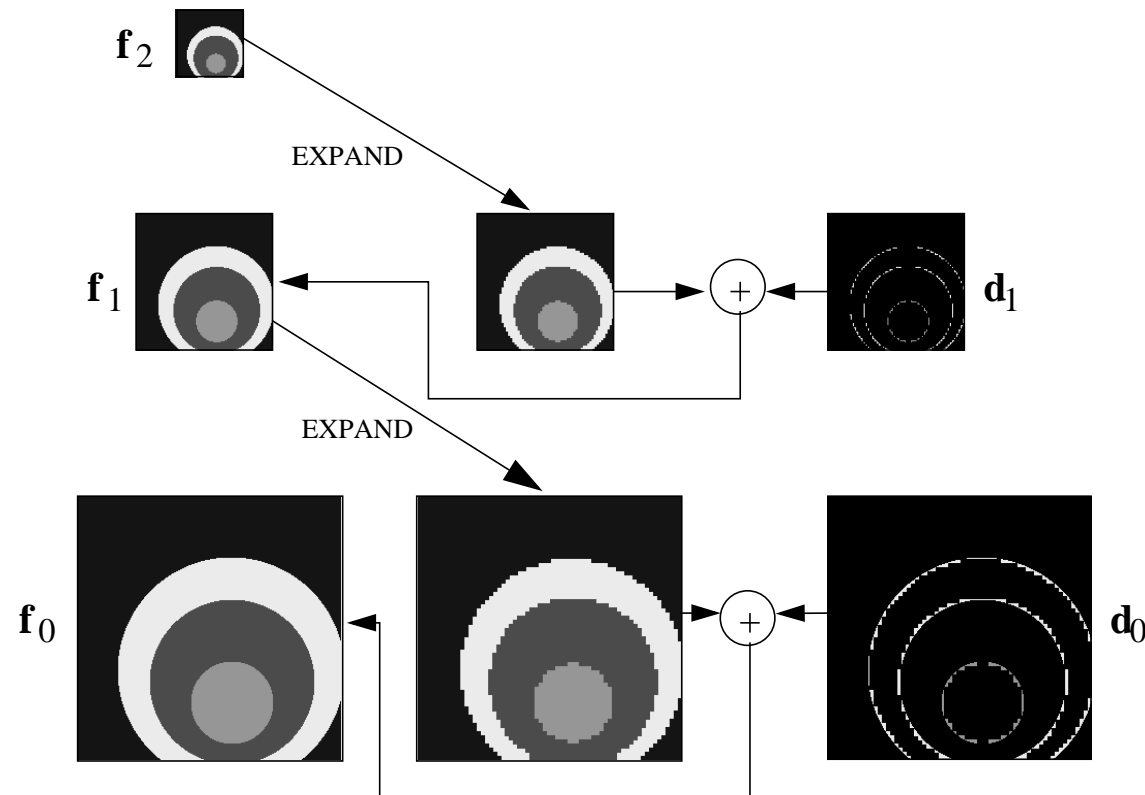
- Multiresolution signal decomposition scheme



- REDUCE is an analysis operator which reduces information
- EXPAND is a synthesis operator that maps back lost information



- Reverse order



- Original image (f_0) can be reconstructed from detail pyramid and last approximation

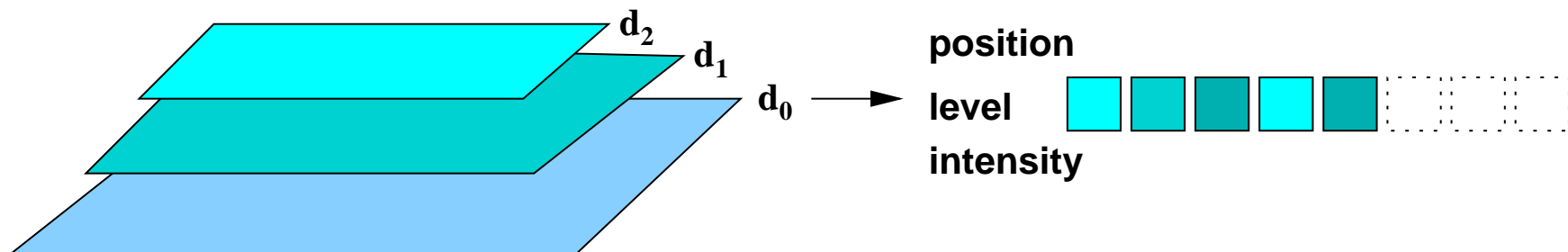


- Pyramid representation allows interchange of MIP projection with pyramidal synthesis operator EXPAND
- MIP can be performed on a coarse level, performing a 2-D EXPAND operation to a finer level, leading to an efficient algorithm
- Further property is that the elements of the detail pyramid can be projected in any order during reconstruction



Detail pyramid to detail stream

- Streaming MIP projects the coefficients individually instead of level-by-level
- The construction of the coefficient stream
 - ★ Merge the detail coefficients of all levels
 - ★ Assign all of these some error measure
 - ★ Sort with decreasing magnitude according to this error
 - ★ Result is ordered list of detail coefficients with attributes (position, level, intensity)



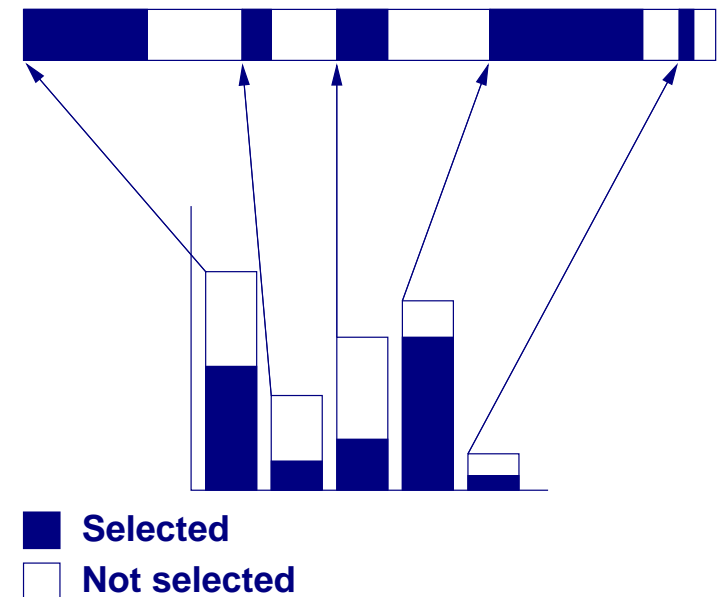


- Sorted coefficients are projected successively until
 - ★ an a-priori chosen maximum number of coefficients has been projected
 - ★ or a desired accuracy of the resulting MIP image is obtained
- As a coefficient is projected, its value is compared to the value at the point of projection, which is overwritten when smaller
 - ★ Do a local dilation with size depending on originating level
- Not very efficient, a lot of overhead per coefficient
- Method allows for a rendering order that is best in terms of efficiency



Efficient reordering

- Use an alternative representation
 - ★ Sorted array of detail coefficients is separated into (level,intensity) bins
 - ★ Ordering by decreasing error is maintained within these bins
 - ★ Coefficients have only position attribute as the rest is implied by the bin, conserving memory
 - ★ Keep sorted order around to select coefficients to render given a choice of error





Rendering on GPU

- Detail coefficients are stored in OpenGL vertex buffer, each coefficient a vertex
- Render each level of the pyramid separately
 - ★ By rendering in low-to-high intensity order no compare is needed on write
 - ★ Each coefficient is rendered at its originating level, thus render points
 - ★ vertex program decodes positions and projects them to their position in the image

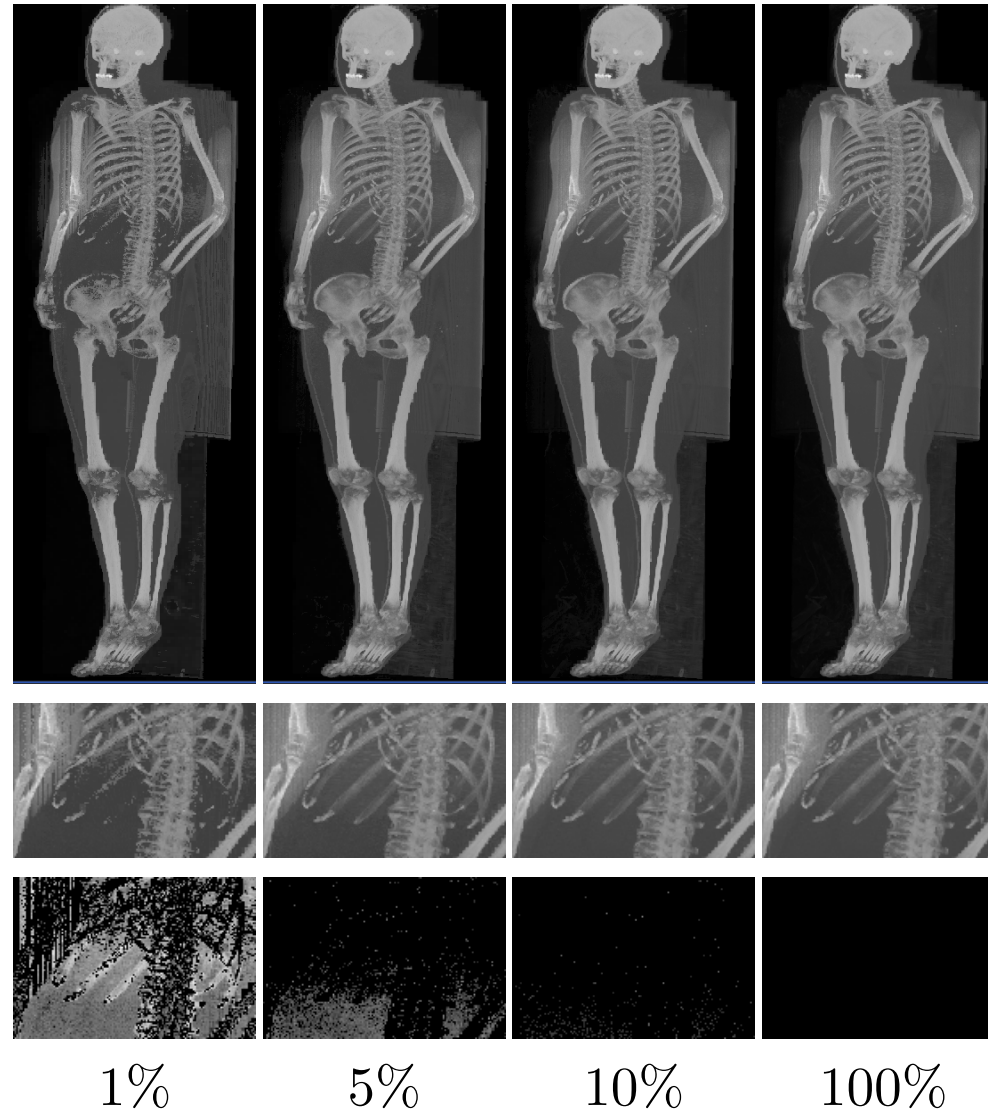


Rendering on GPU

- Each level is thus rendered to a texture (using the Frame Buffer Object extension)
- EXPAND operation was implemented in a fragment program
- Results are combined using frame buffer blending with the maximum operator



- Streaming MIP rendering of Visible Woman dataset at a few quality settings





Results

- Performance in frames per second on (FPS) VisibleWoman dataset, which has dimensions $512 \times 512 \times 1734$ and a total of 210 million detail coefficients

Method	Error (L_1)	FPS
3-D Texture-based	-	1.0
GPU ray-casting	-	2.0
Streaming MIP (software)	0.0	0.2
Streaming MIP (software)	0.07 (median 4)	3.5
Streaming MIP (GPU)	0.0	1.1
Streaming MIP (GPU)	0.07 (median 4)	18.4
Streaming MIP (2x GPU)	0.0	2.0
Streaming MIP (2x GPU)	0.07 (median 4)	30.2

All performance measurements were carried out on a machine with dual AMD Opteron 280 processor and two GeForce 7900GTX graphics cards. Unless mentioned otherwise, only one of the cards (and one of the CPUs) is active.



- Performance and error against % of coefficients

Coeffs. (%)	Error			FPS
	<i>max</i>	relative L_1	<i>median</i>	
100	0	0	0	1.14
50	3	9.6×10^{-5}	1	2.27
25	8	2.0×10^{-3}	3	4.41
13	13	1.2×10^{-2}	4	8.40
6	21	2.6×10^{-2}	4	15.4
1	63	1.0×10^{-1}	7	51.2



- Our streaming MIP GPU-method outperforms both its software implementation and existing ray-casting and 3-D texture-based methods
- The performance/quality ratio can be adjusted as desired
- Empty space skipping is implicit by rendering only non-empty voxels
- Load and dataset can be divided over multiple GPUs to achieve a near-optimal speedup

Questions?

