What we will cover

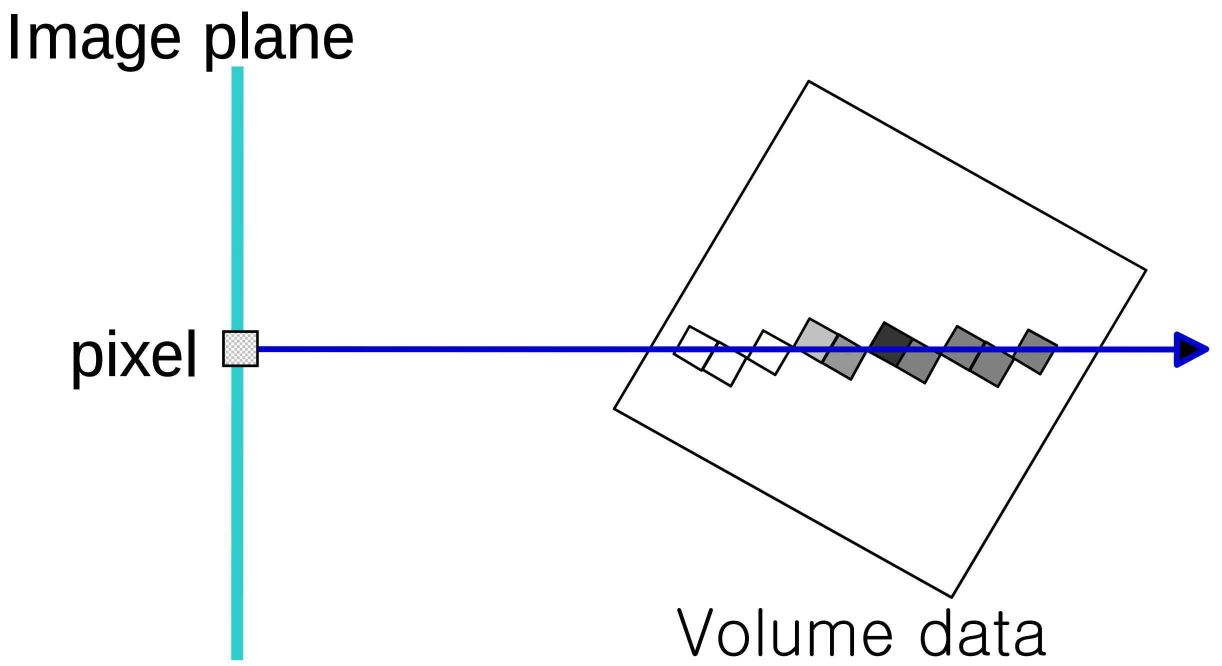
- Contour Tracking
- Surface Rendering
- Direct Volume Rendering
- Isosurface Rendering
- Optimizing DVR
- Pre-Integrated DVR

Optimizing DVR

- When we stop re-sampling along the ray?
- What voxels we can skip during rendering?
- How to improve re-sampling process?
- How to find the nearest object boundary?
- How to improve image quality?

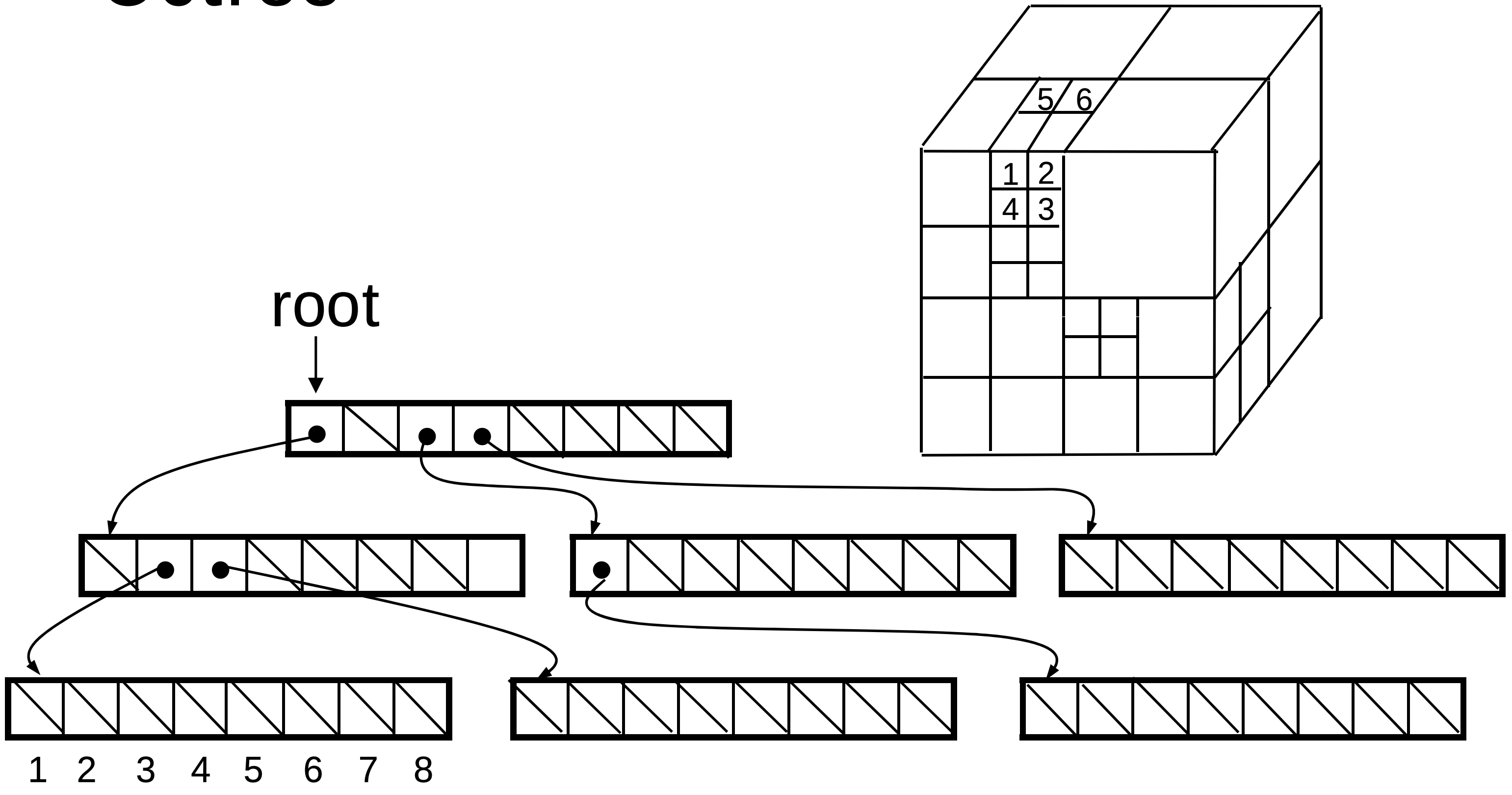
Early ray termination

- Terminate resampling when the accumulated opacity reaches the threshold value.



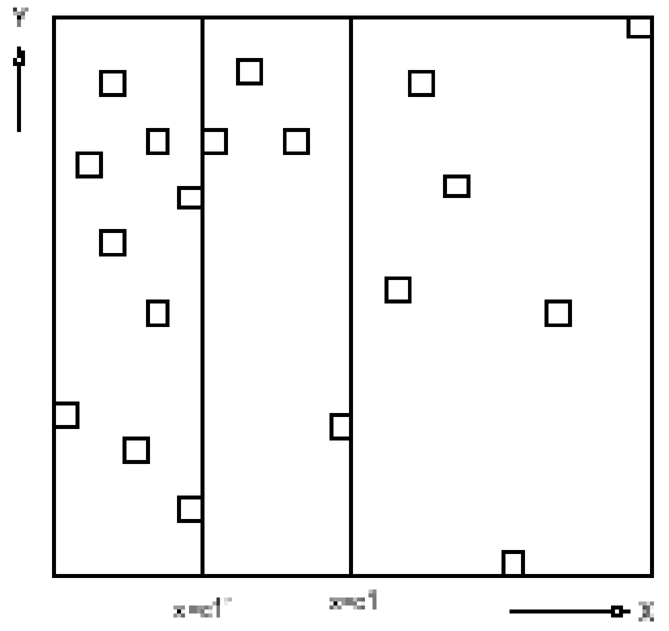
Using data coherency

- Good for empty space skipping
- Octree



Using data coherency

- K-d tree
  - Recursively subdivide the volume along x,y, and z-axis aligned planes.
- Run length encoding
  - encoded by first location of run and length of run



Discrete ray casting

- Traverse a discrete representation of the ray
- 3-D line scan-conversion or voxelization algorithm
- Three types of connected paths

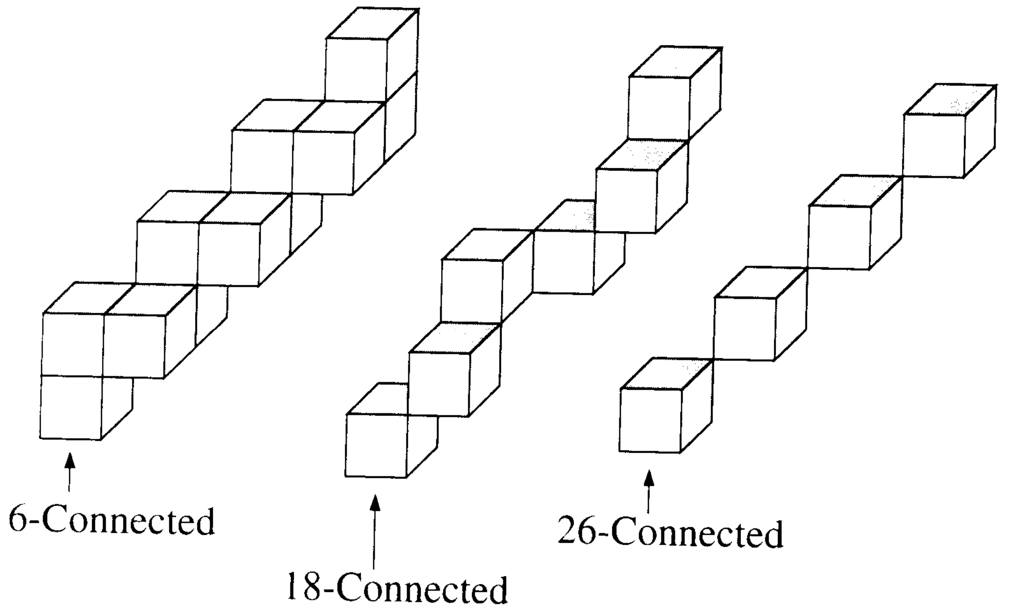
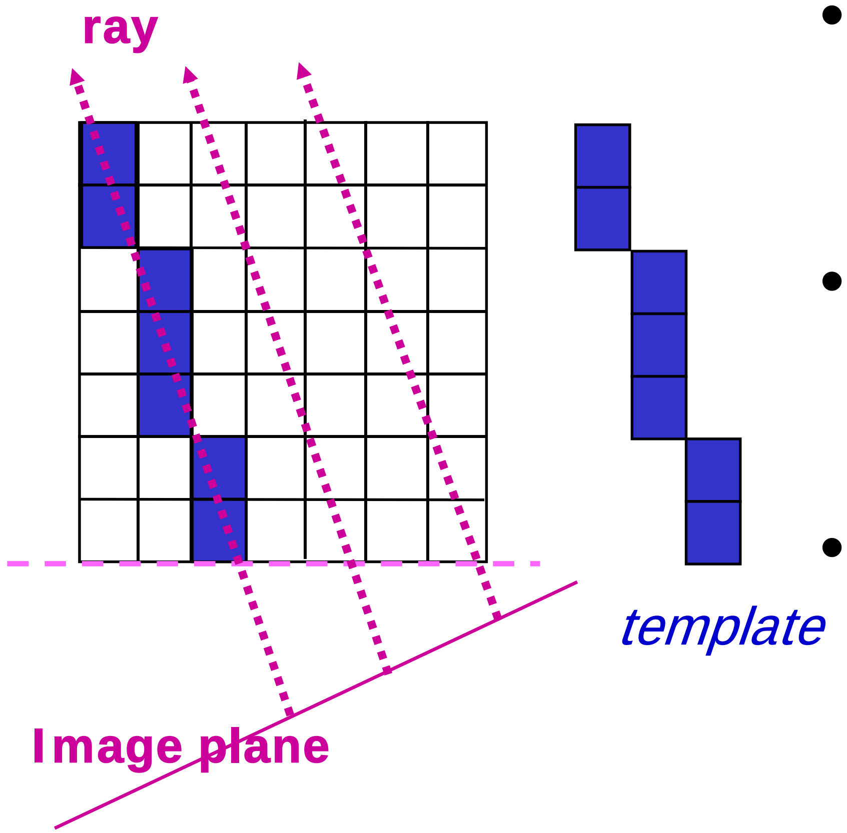


Figure 6.11 6-, 18-, and 26-connected paths

Template based rendering



- Accelerating ray casting by minimizing resampling time – use inter ray coherence
- Need a different template per individual displacement of a ray in a image pixel
- No interactive speed because of image order processing

[Yagel and Kaufmann 92]

Polygon assisted ray casting

- Efficient empty space skipping
- But the more twisted the object is, the more polygons are needed

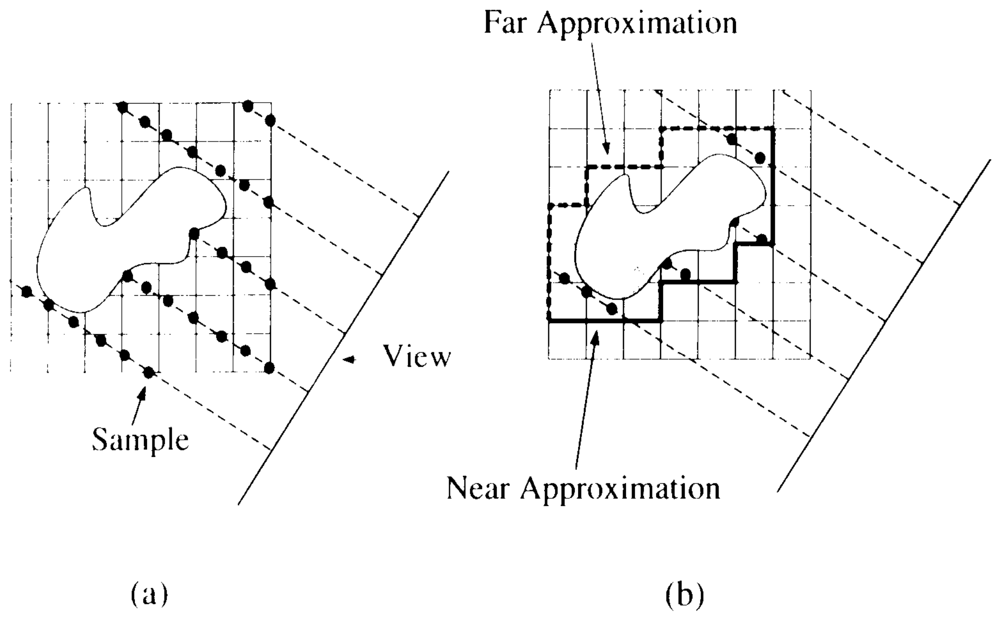


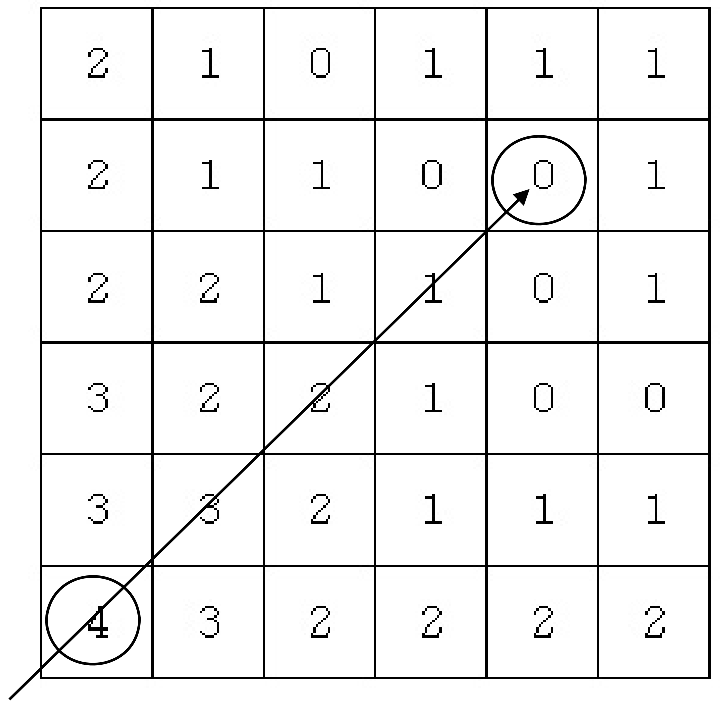
Figure 6.17: (a) Brute-force ray casting (b) Polygon assisted ray casting (PARC)

Distance map

- Good for empty space skipping
- Good for pre-classified volume
- Each voxel contains the shortest distance to the boundary for visualization
- Need extensive pre-processing time
- When the distance is too big, set the maximum value for reducing memory space for distance.

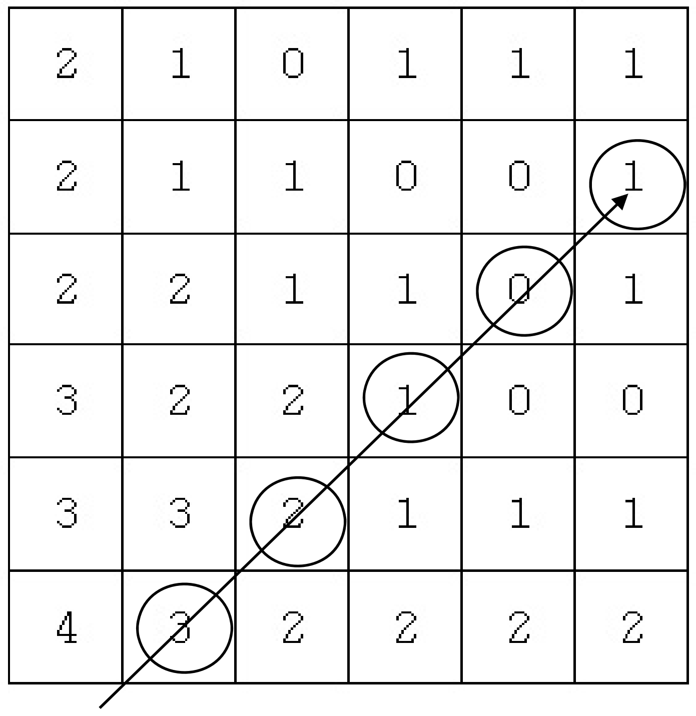
Distance map – example

Traversal time with Distance map : 2 times



2	1	0	1	1	1
2	1	1	0	0	1
2	2	1	1	0	1
3	2	2	1	0	0
3	3	2	1	1	1
4	3	2	2	2	2

Without Distance map : 5 times



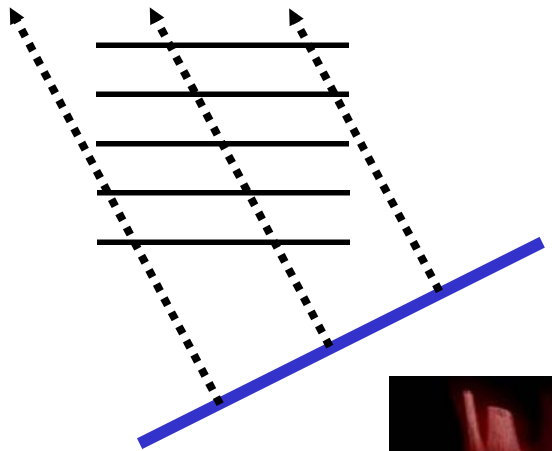
2	1	0	1	1	1
2	1	1	0	0	1
2	2	1	1	0	1
3	2	2	1	0	0
3	3	2	1	1	1
4	3	2	2	2	2

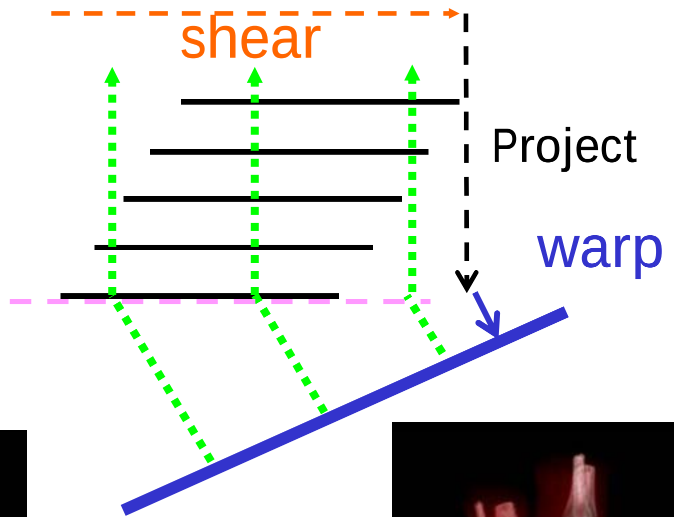
Shear Warp

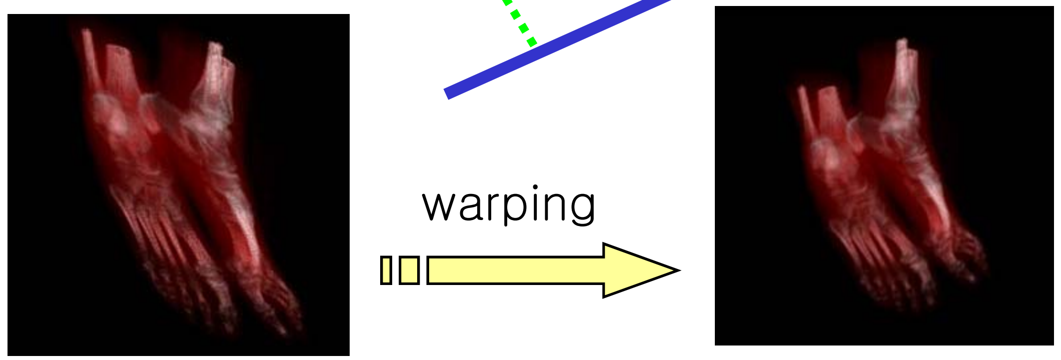
- P. Lacroute and Marc Levoy[94]
- Image and object space method
- Very fast s/w based algorithm
- Need preprocessing step for encoding
- Pre-classification with opacity-weighted colors are common

Shear-warp algorithm

1. Transform the volume data to sheared object space by translating and resampling each slice (also *scaling* for perspective transformation)







Shear warp rendering

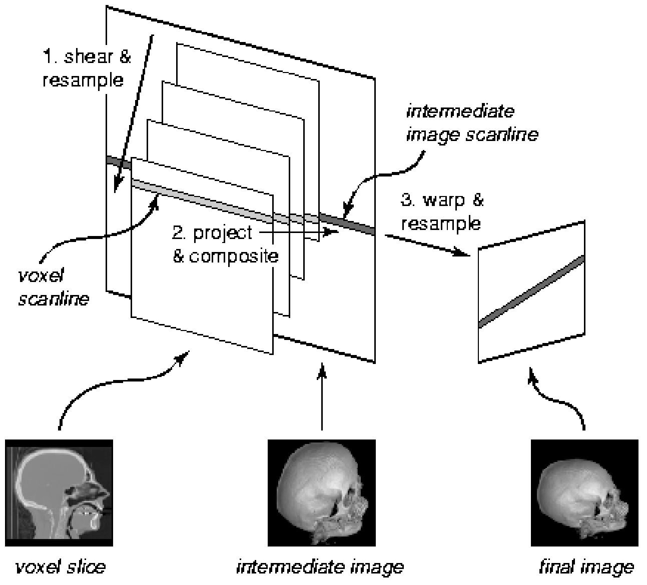
2. Composite the resampled slices together
 in front-to-back order using the “over” operato

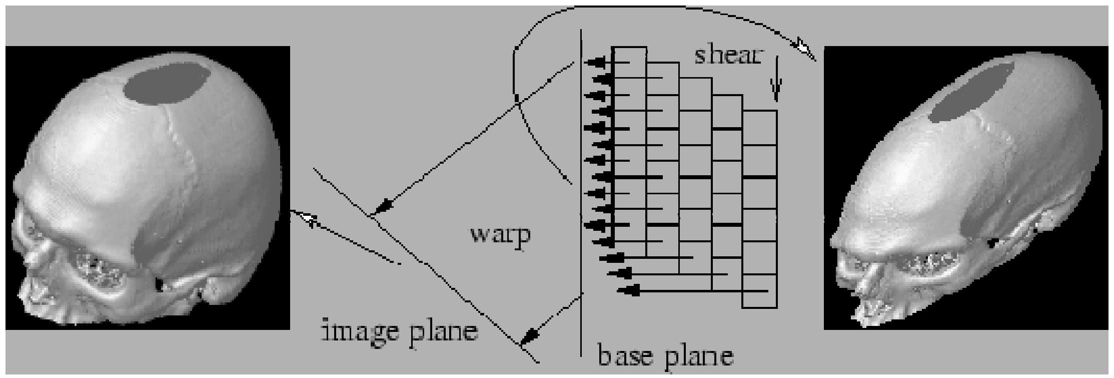
3. Transform the intermediate image to image
 space by warping it according to M_{warp}

$$M_{view} = P \cdot S \cdot M_{warp}$$

$$M_{warp} = S^{-1} \cdot P^{-1} \cdot M_{view}$$

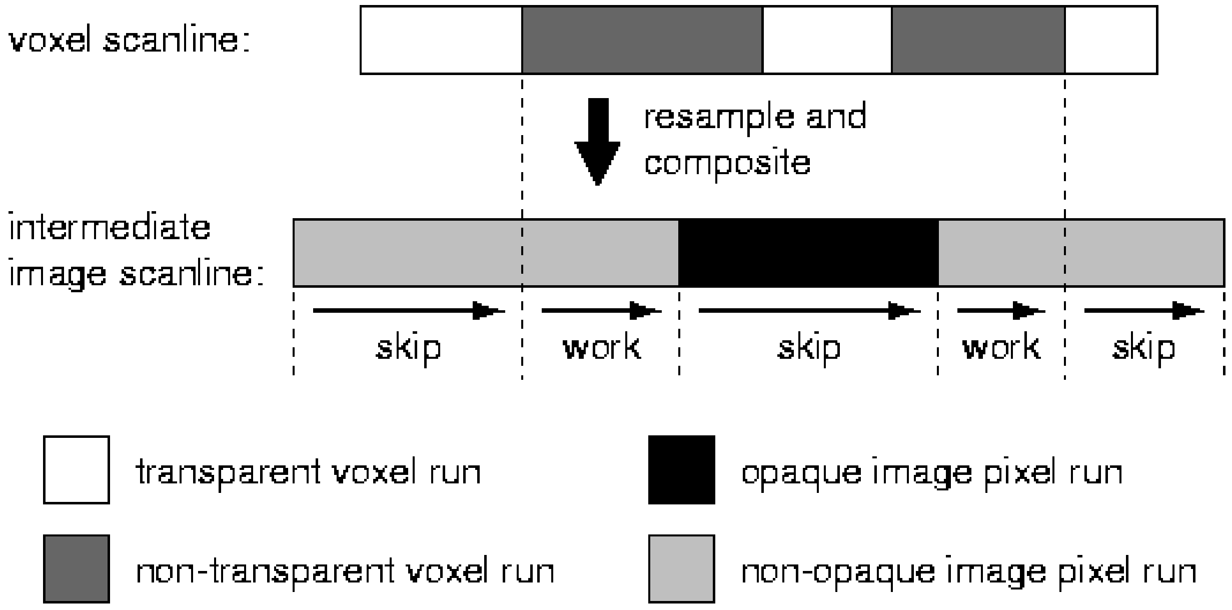
Shear warp rendering





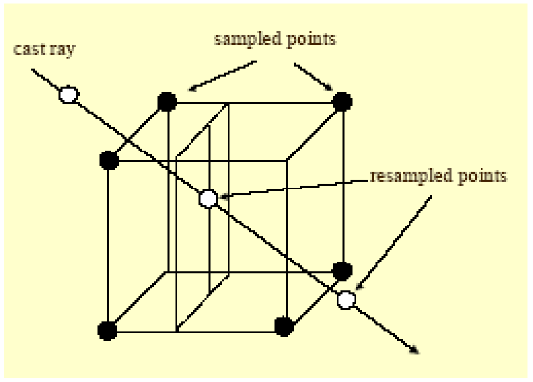
Shear warp rendering

- Run-length encoding



Shear-warp rendering

- Pros
  - Fast : image & object space algorithm
  - Simple
  - perspective projections possible
  - hardware acceleration possible
- Cons:
  - Not good image quality
    - bi-linear interpolation & warping distortion
    - voxel/pixel=1: problems for zooming
  - Require three sets of encoded volume

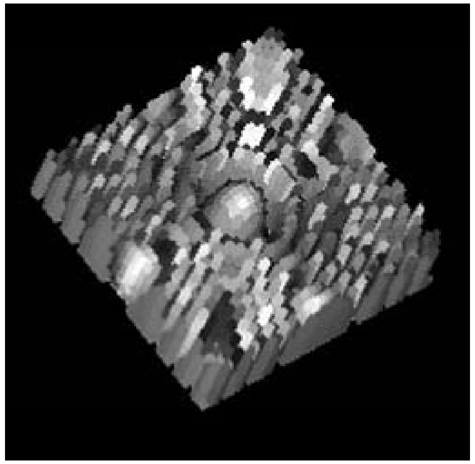


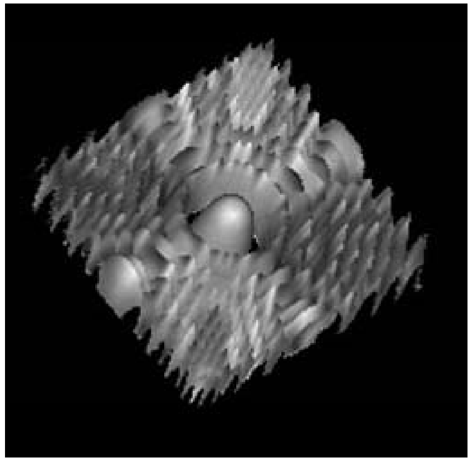
Shear-warp vs. Ray-tracing

nearest neighbor interpolation

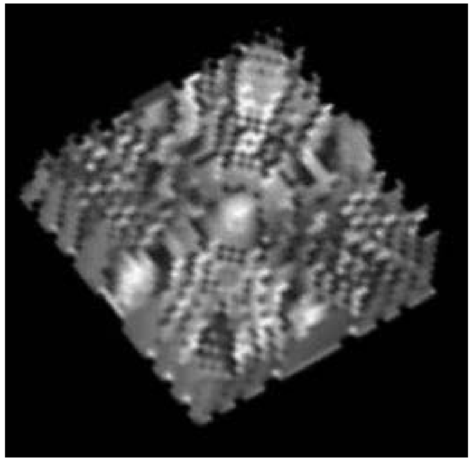
linear interpolation

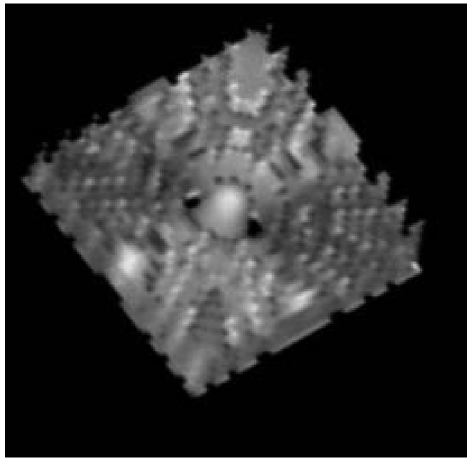
Ray tracing





Shear–warp





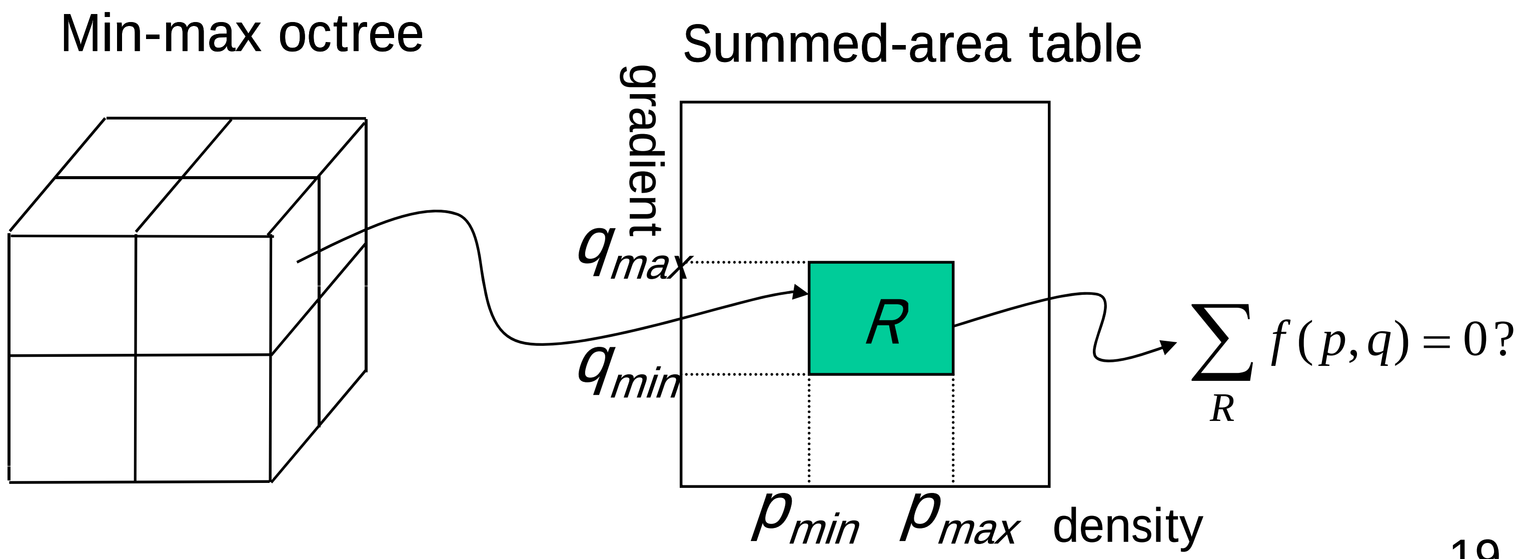
Interactive Classification

What is the best method for rendering dynamically classified volumes.

- Can I use run-length encoding?
- Can I use Octree?
- Can I use space leaping?
- How to modify shear-warp algorithm?

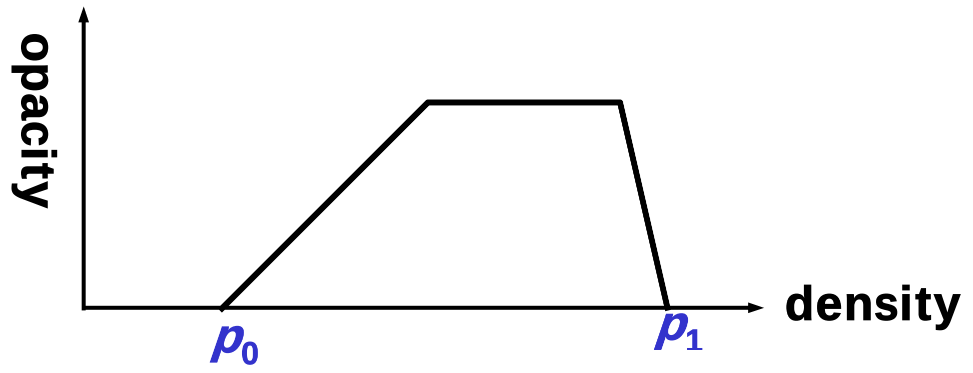
Fast classification [Lacroute94]

- Classify voxels in non-transparent portions of each scanline during *rendering* using a precomputed min-max octree and a summed-area table
- SAT tells whether the block is transparent or not.



Summed-area table

1D OTF using density values



000001111111111111111110000000

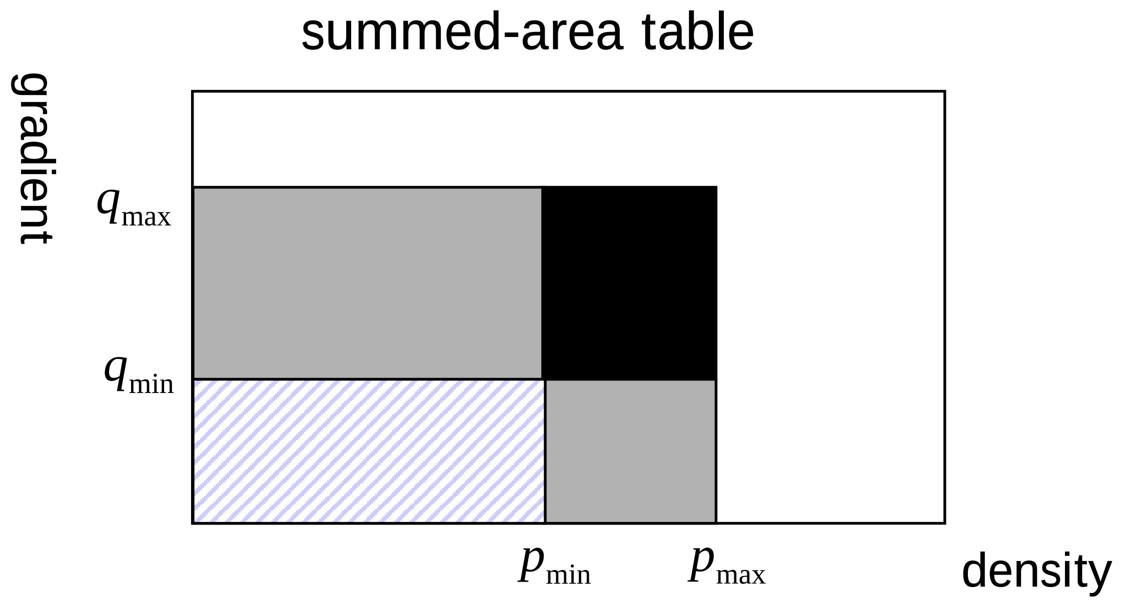
SAT 0,..,0 | 0,1,2,3,4,5⋯..,20 | 20,20,20,20,20,..

min max → nontransparent block

transparent

Summed-area table

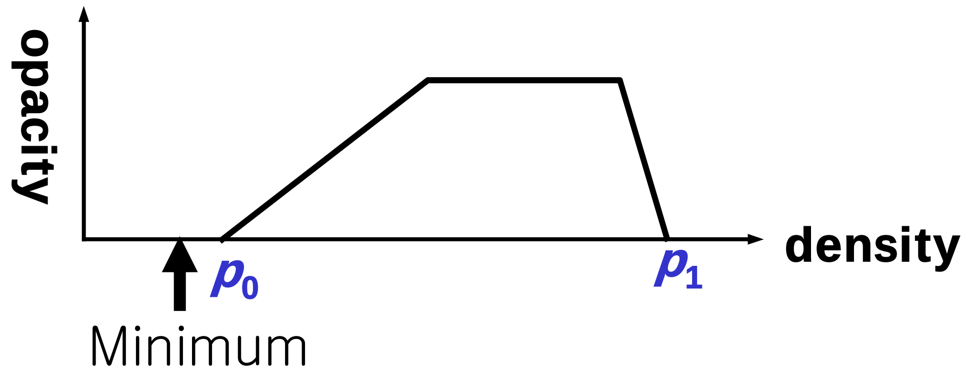
2D OTF using Density & Gradient



$$\sum_{p=p_{\min}}^{p_{\max}} \sum_{q=q_{\min}}^{q_{\max}} f(p,q) = S(p_{\max}, q_{\max}) - S(p_{\max}, q_{\min}-1) - S(p_{\min}-1, q_{\max}) + S(p_{\min}-1, q_{\min}-1)$$

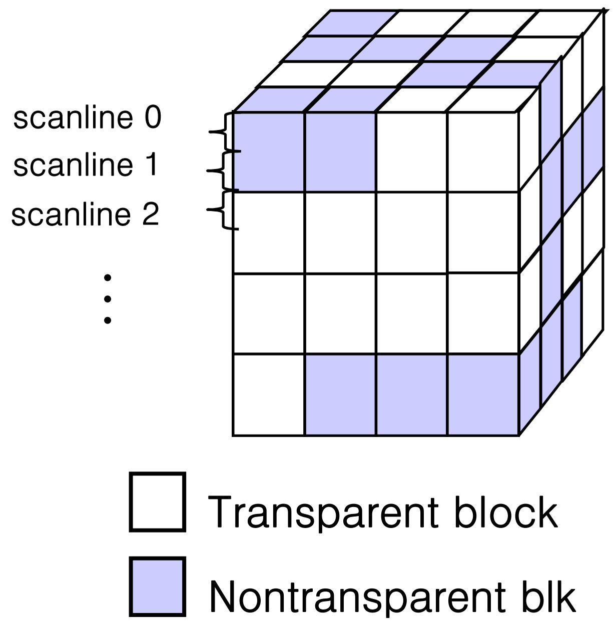
Interactive Classification [Wan99]

- Opacity threshold
  - minimal scalar field value with a non-zero opacity
- Classifying a volume with the lowest opacity threshold
- Adjust opacity transfer function with the higher opacity threshold



Interactive Classification (cont'd)

- Limitations of Wan's Approach
  - longer projection time
  - longer ray traversal time
- Interactive classification[Kim00]
  - Construct a block-based run-length array using block min-max table and summed-area table
  - progressive refinement of visible non-transparent blocks in run-length array

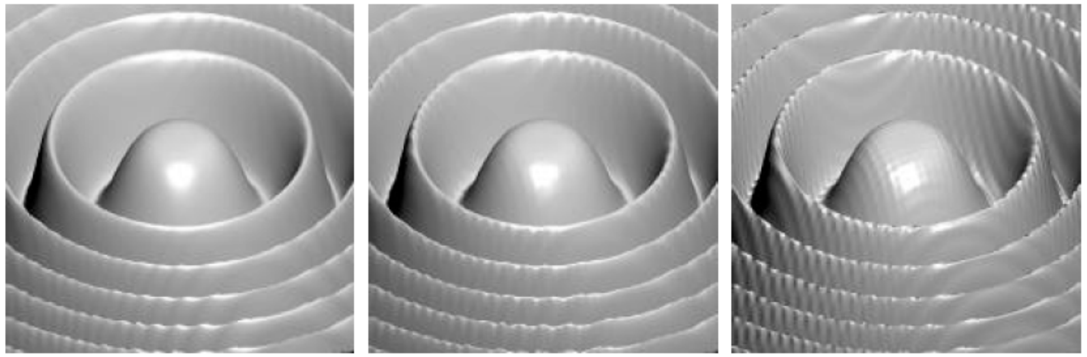


Determination of image quality

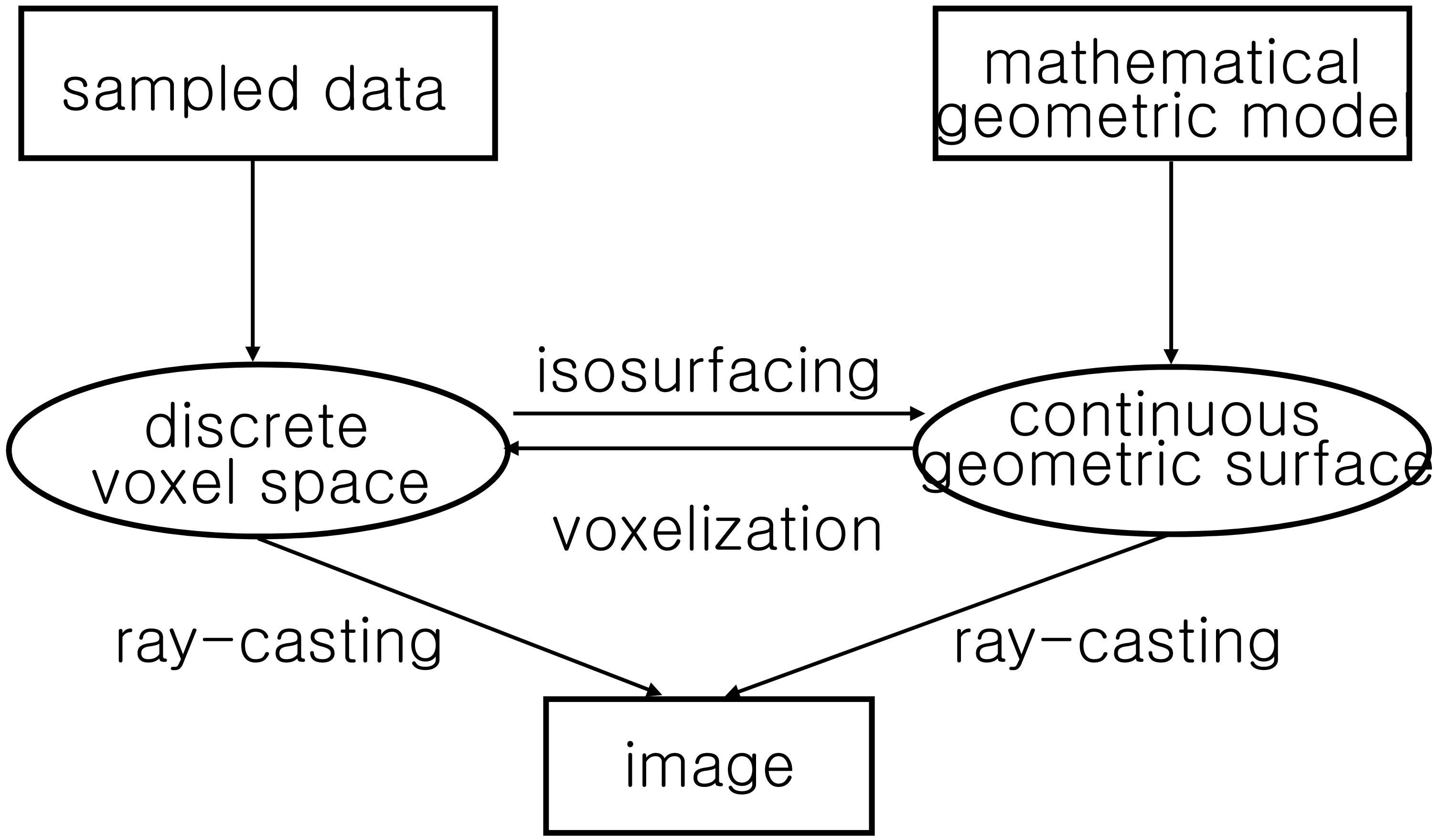
- The precision of the surface normals
- The rendering *quality* depends on viewpoint, magnification, and image size
- Clinical functionality

How to test image quality?

- the use of real phantoms
- human visual system – what you see

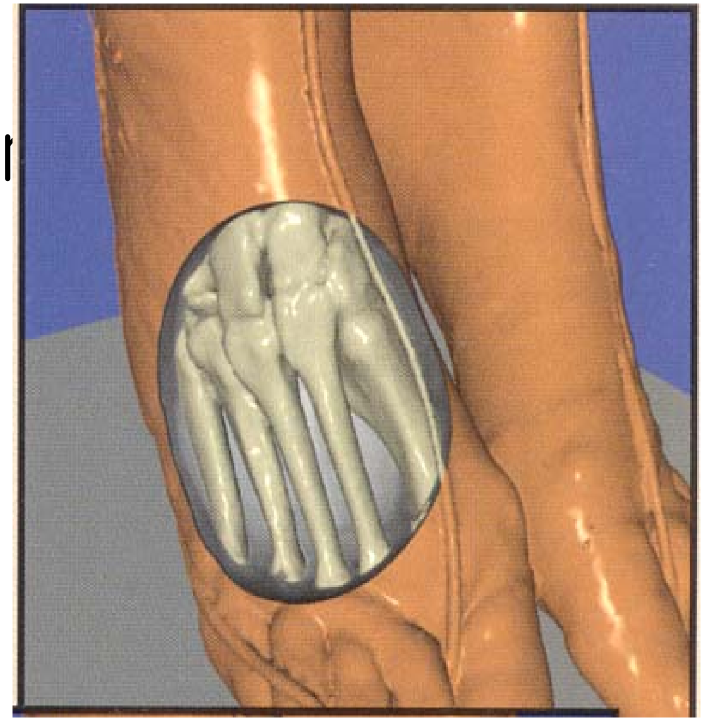


Intermixing analytically-defined surfaces and volumetric data



Volume + Surface

- Surface reconstruction algorithm
  - volumes ⇒ surfaces
- Scan conversion algorithm
  - surfaces ⇒ volumes



Volume + Surface

- Point-based approach
  - Surfaces, volumes ⇒ points
  - Extend the surface-based approach by adding a point primitive to the set of geometric objects. The primitive consists of a 3D location, a normal vector, and some additional values (e.g., color, opacity, density).

Volume + Surface

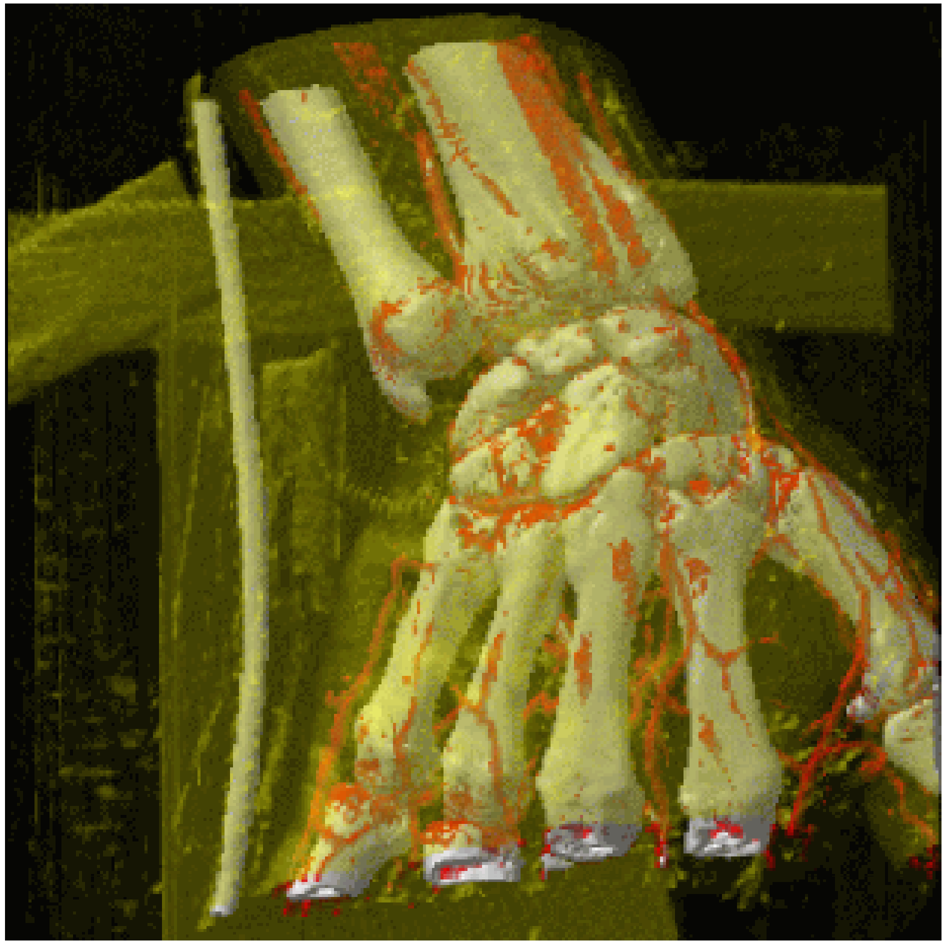
- Hybrid approach

support the hybrid data model by rendering each part of it separately and then combining the surface rendered and the volume rendered images into a final 2D image

Z-merging algorithm – use two z-buffer

ray-merging algorithm – use two rays

Two-level Volume Rendering

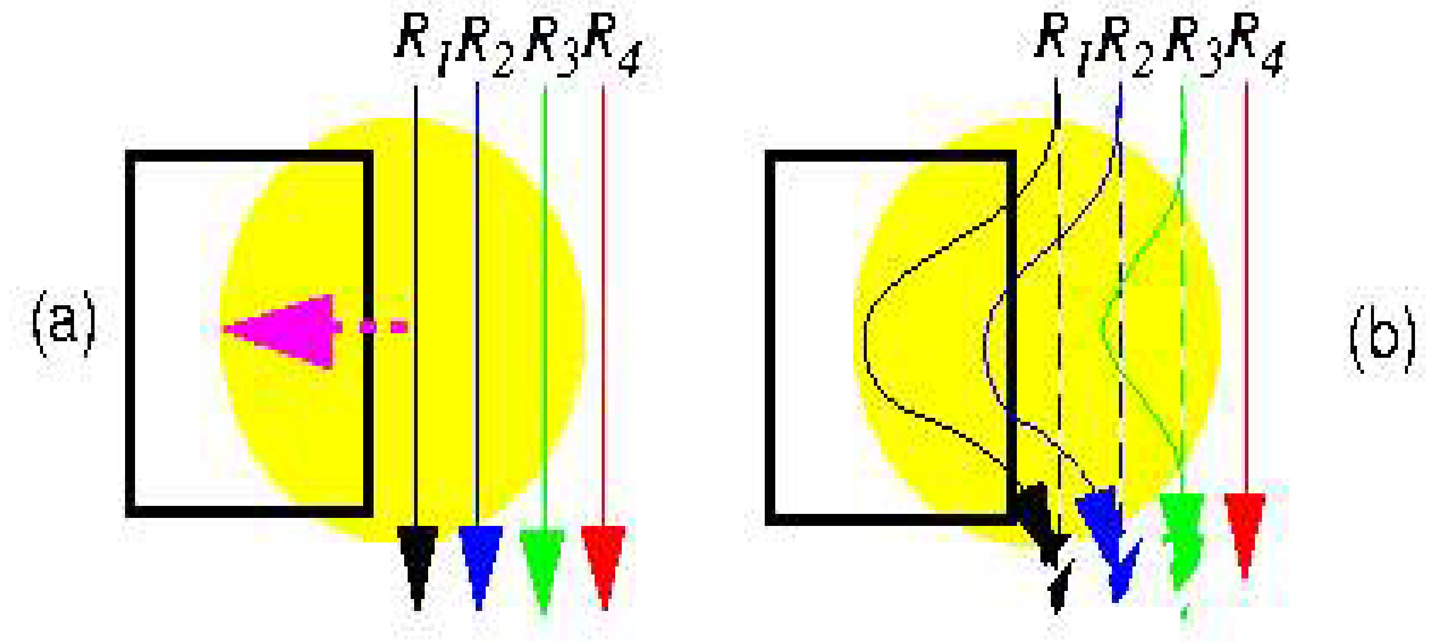


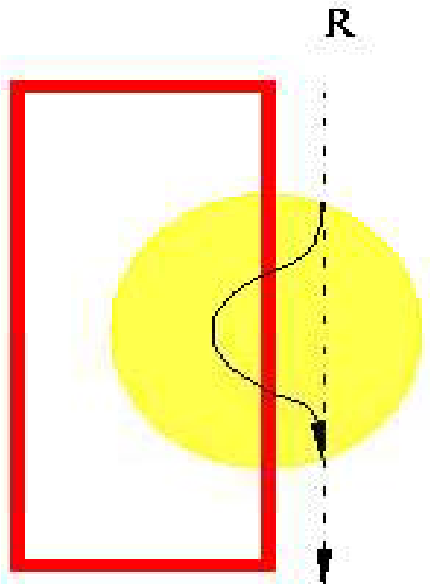
joining MIP and DVR

(bones and vessels: DVR; skin: MIP).

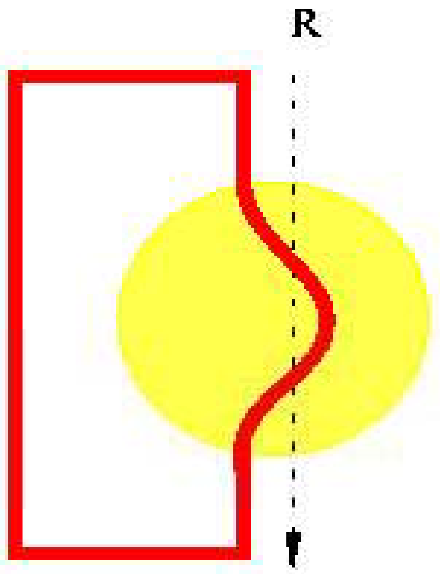
Volume Deformation by deforming rays

- Space Deformation with Deflectors

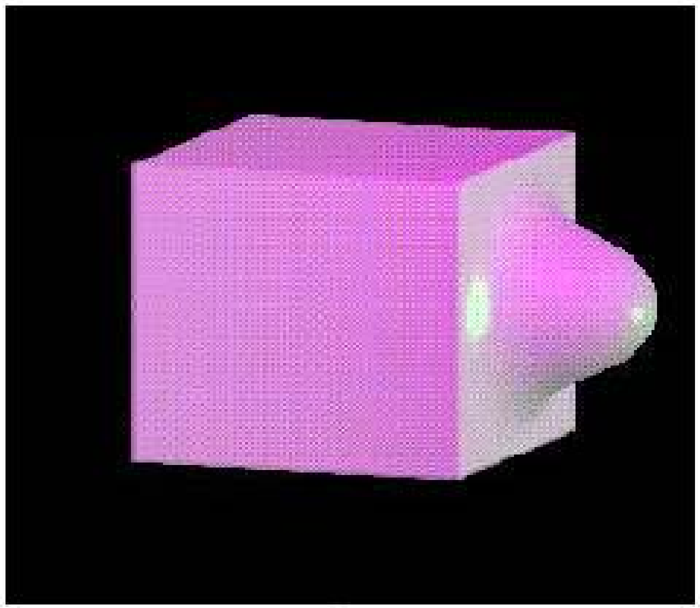




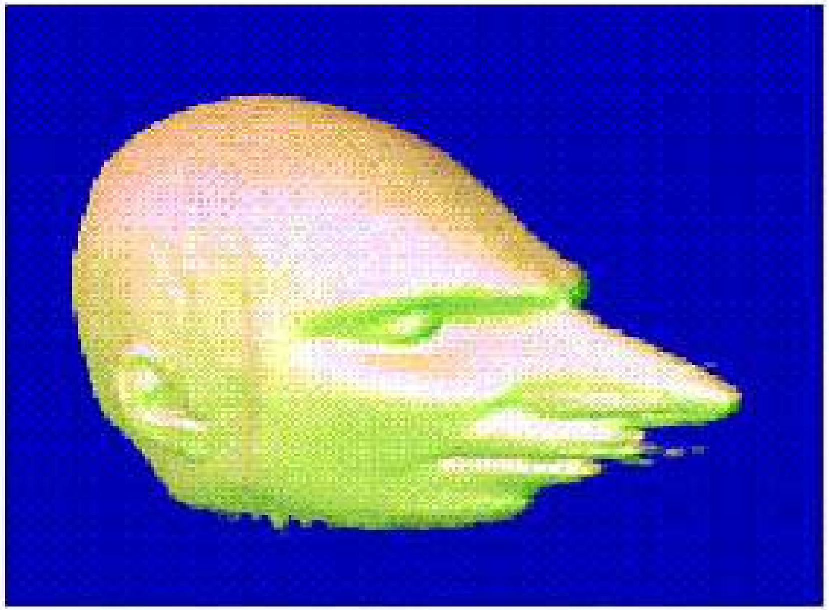
(a)



(b)

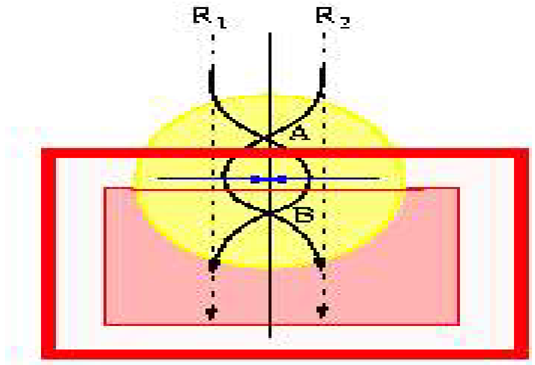


(c)

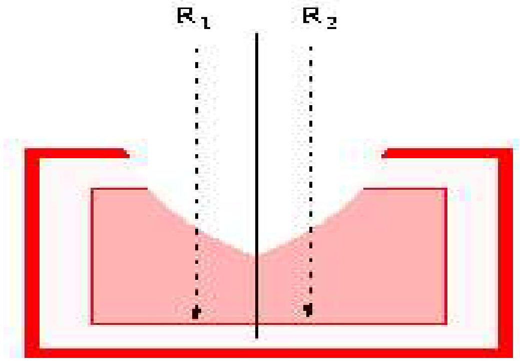


(d)

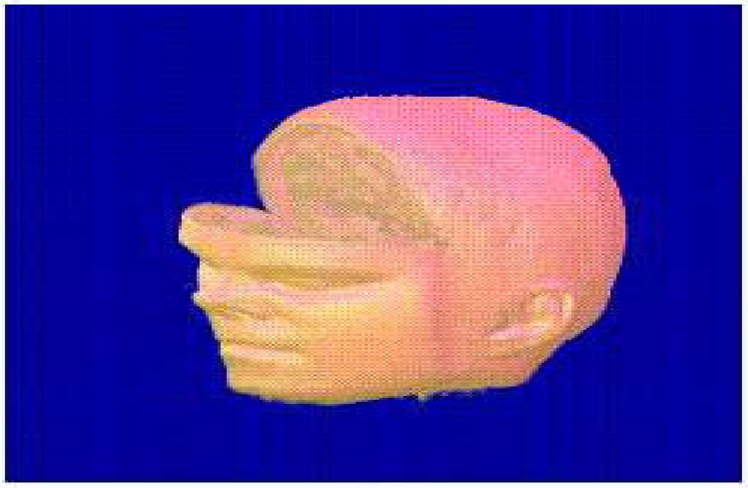
The Discontinuous Deflector



(a)

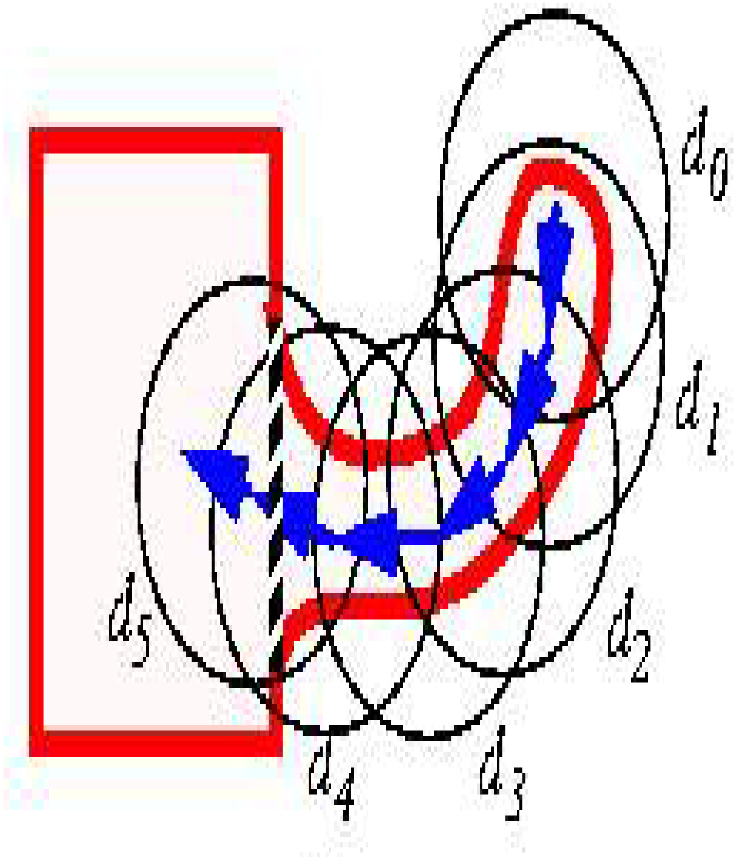


(b)

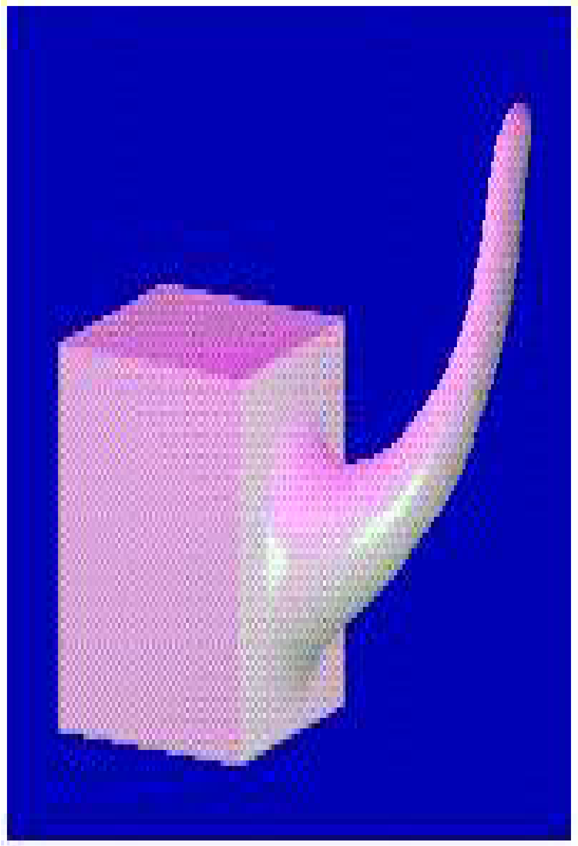


(c)

Modeling with Multiple Deflectors



(a)



(b)