Big Data Visual Analytics (CS 661)

Instructor: Soumya Dutta

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur (IITK)

email: soumyad@cse.iitk.ac.in

Acknowledgements

- Some of the following slides are adapted from the excellent course materials and tutorials made available by:
  - Prof. Han-Wei Shen (The Ohio State University)
  - Prof. Klaus Mueller (State University of New York at Stony Brook)
  - Engel, Hadwiger, Salama; Real time volume graphics tutorial, EuroGraphics 2006

Study Materials for Lecture 6

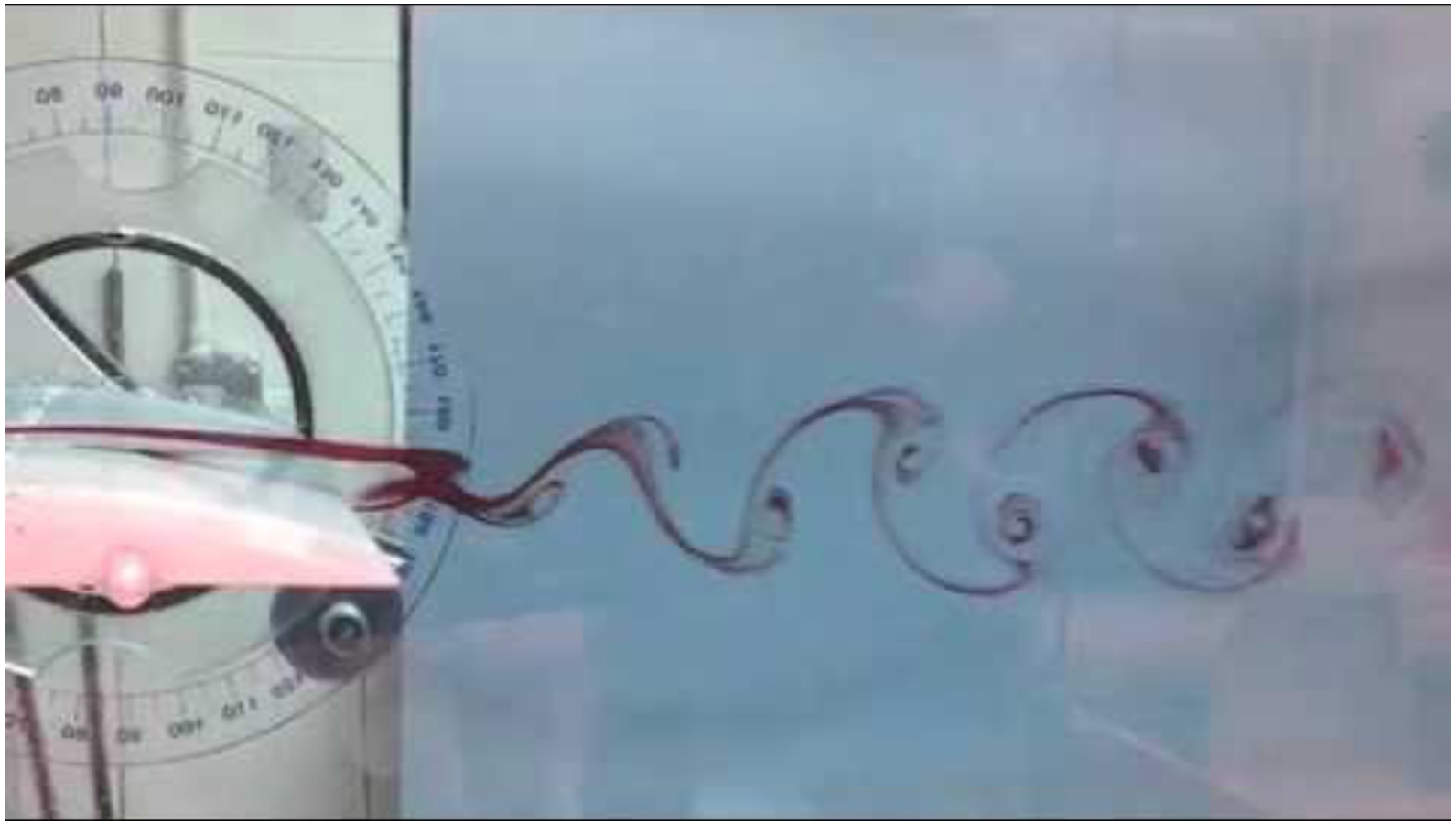
- The Visualization Toolkit by Will Schroeder, Ken Martin, Bill Lorensen
  - Chapter 7 (Volume Rendering section)
- GPU Gems Volume Rendering:
  - https://developer.nvidia.com/gpugems/gpugems/part-vi-beyond-triangles/chapter-39-volume-rendering-techniques
- For detailed mathematical modeling and algorithm
  - Optical Models for Direct Volume Rendering by Nelson Max

Form Assignment Groups

- Form groups of 2 students and update details in the following spreadsheet
  - https://docs.google.com/spreadsheets/d/1T4FC8I20l6fmM5zBxaTXNUercVJOEp4y2TrcIQJLgoc/edit?usp=sharing
  - Deadline: **Jan 24th, 11:59pm**

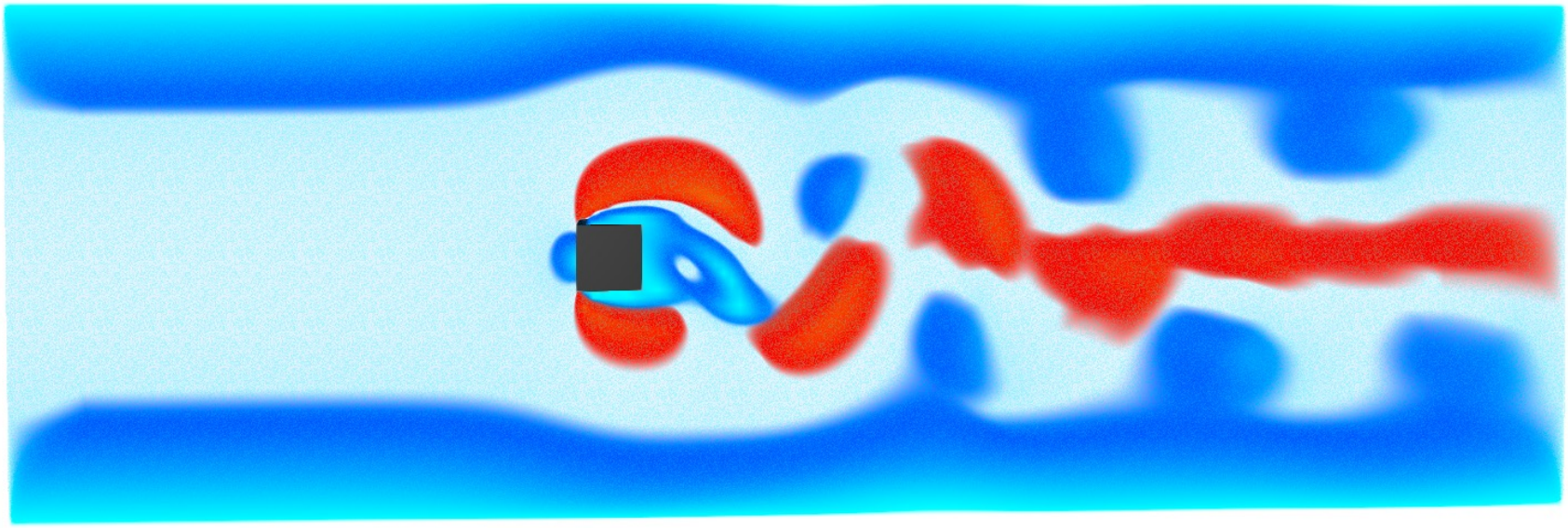
Volume Rendering Demo in ParaView

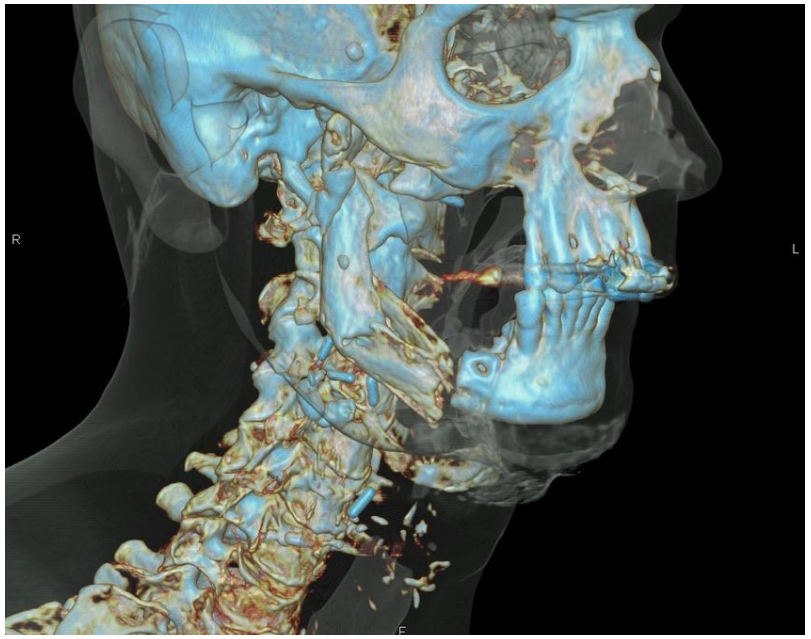
-Von Karman Vortex Street Visualization

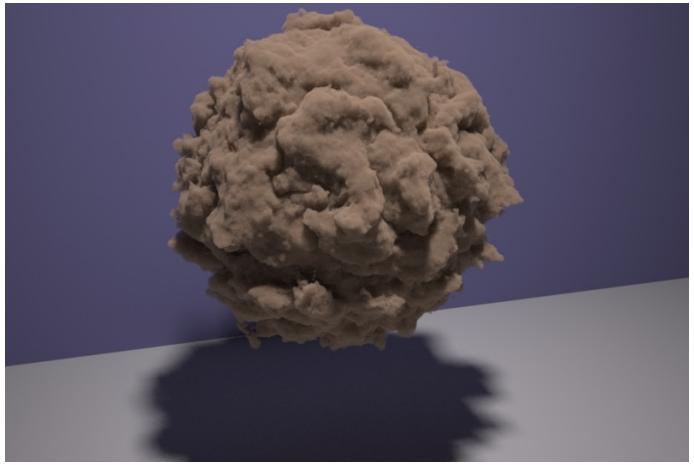


Volume Rendering Demo in ParaView

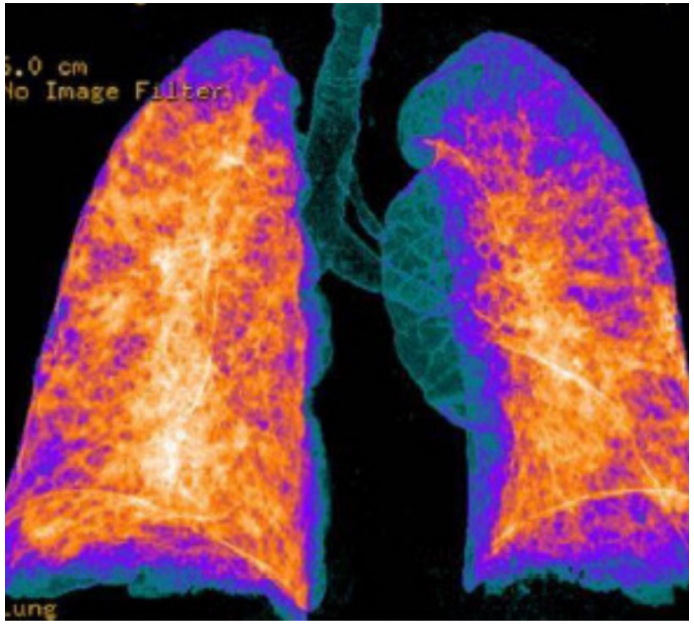
-Von Karman Vortex Street Visualization



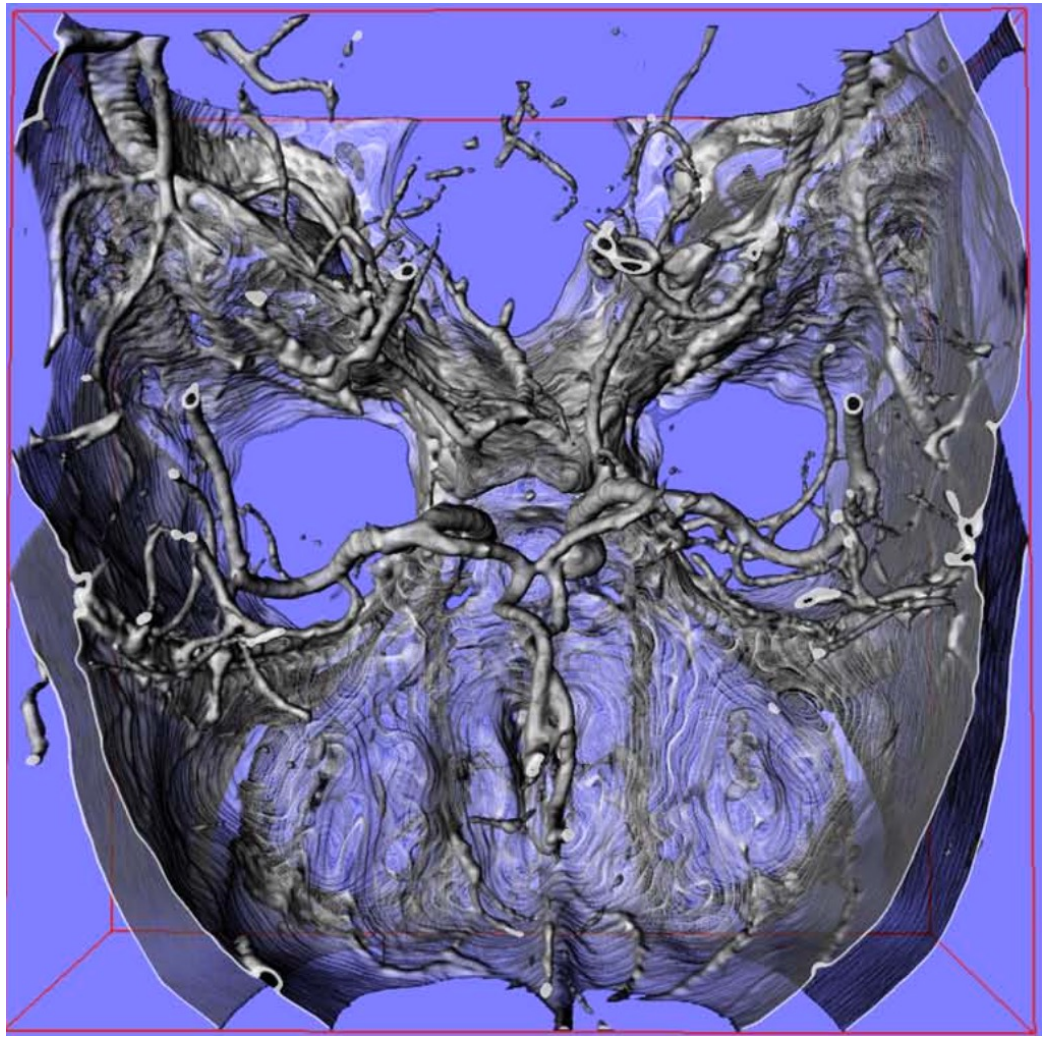




Volume Rendering



Applications: Medical Science

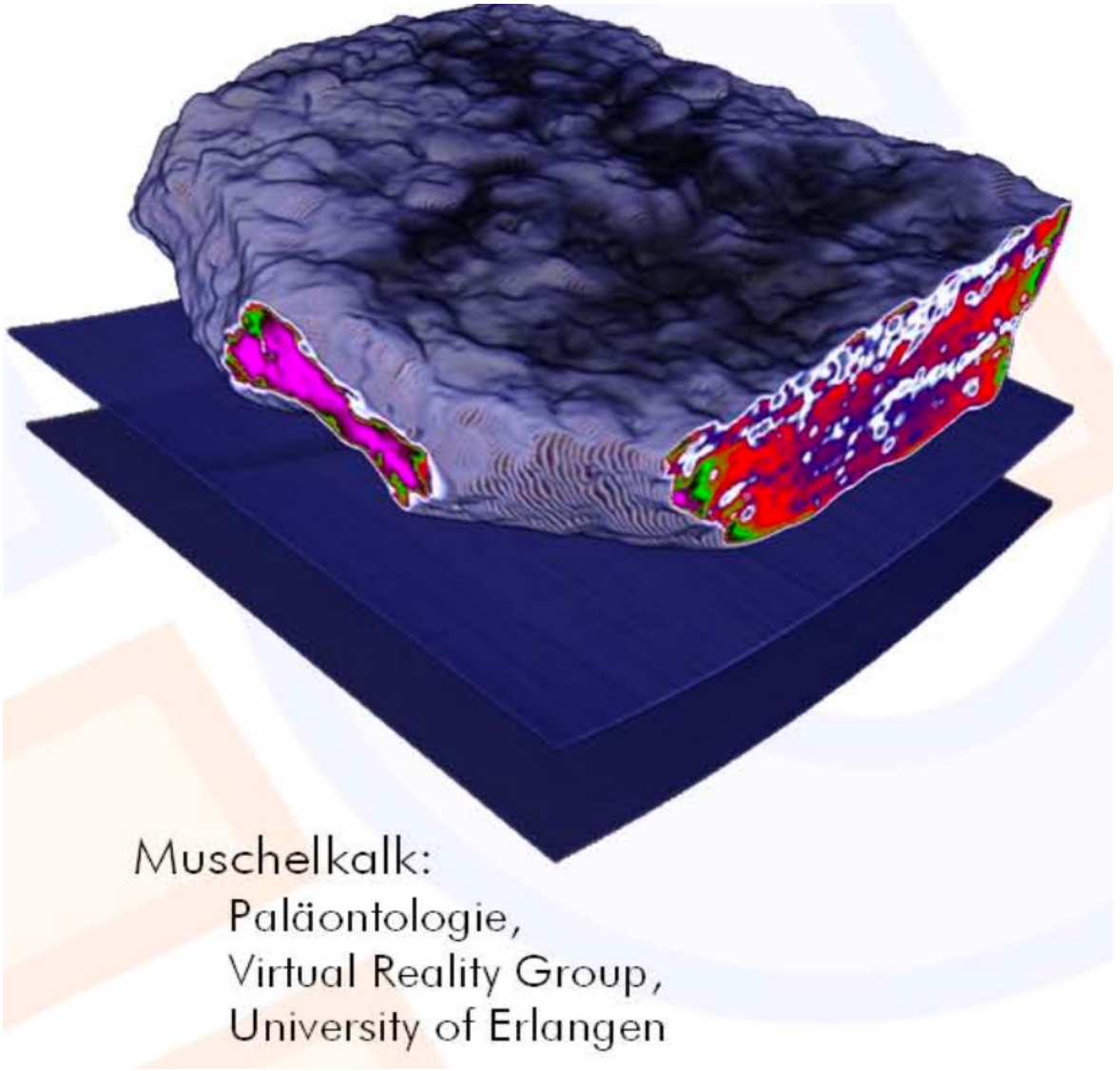


CT Angiography

Applications: Medical Science

Applications: Geology

Deformed plastic model

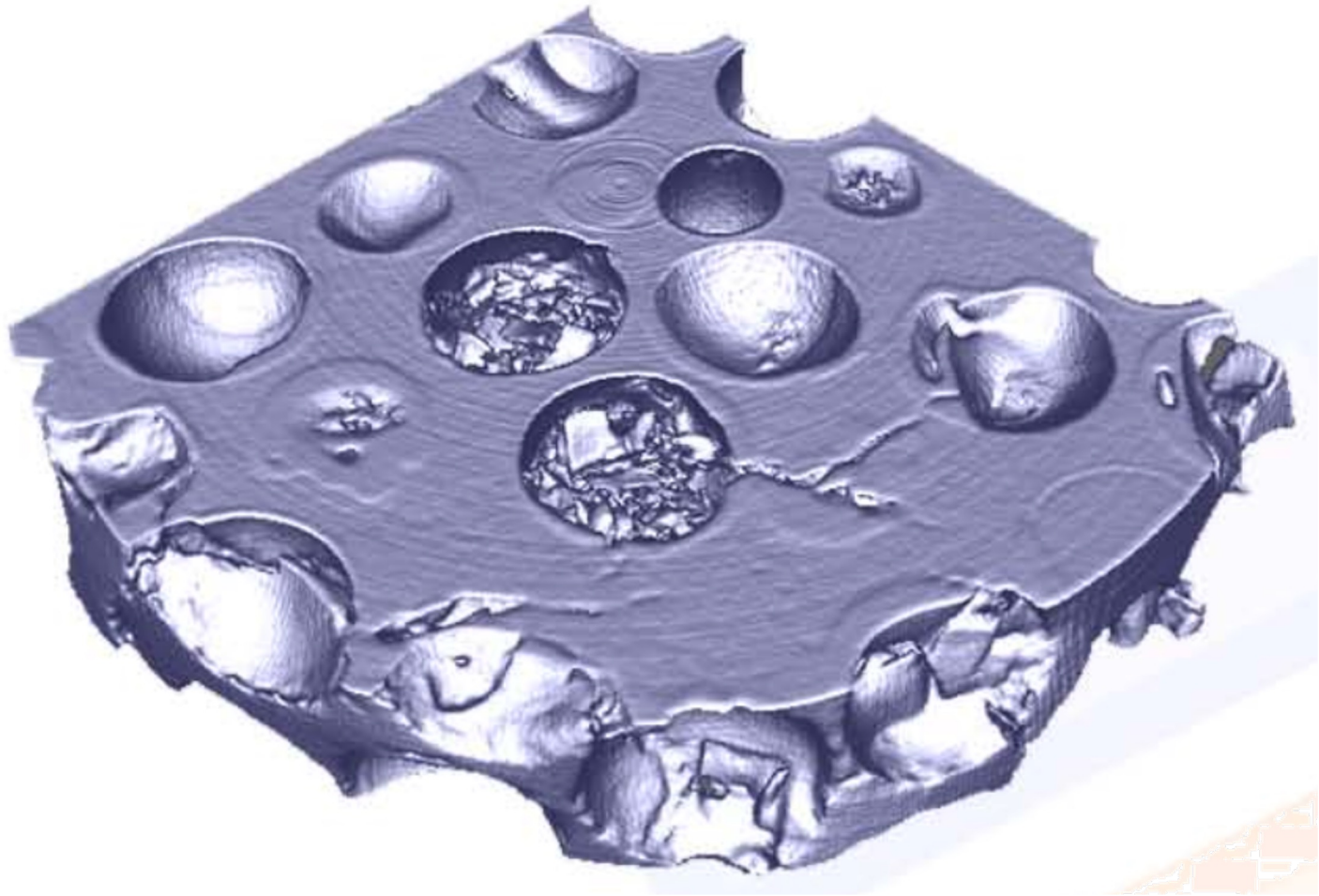


Applications: Archeology

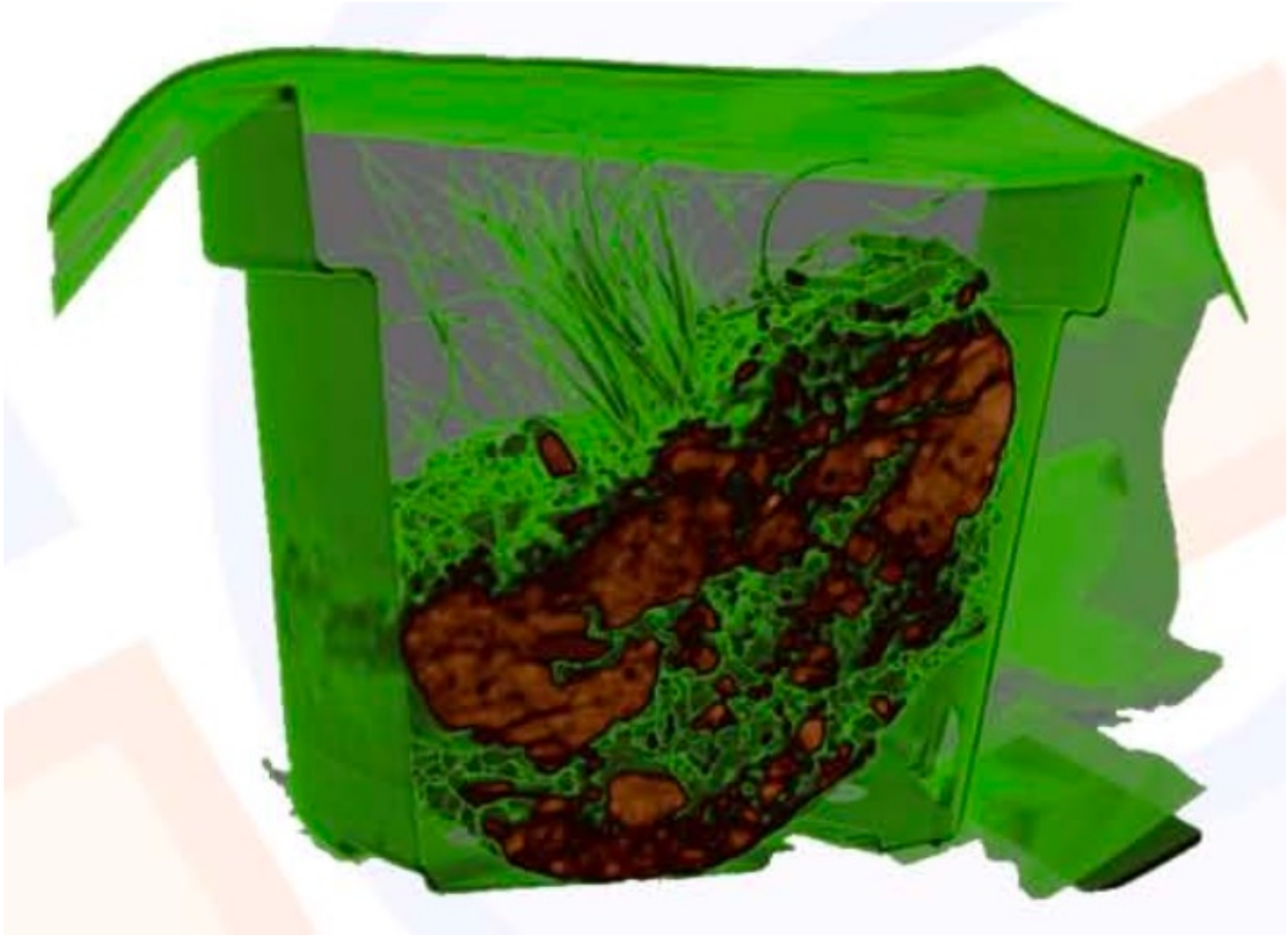
Historical Statute

Applications: Materials Science

Quality control



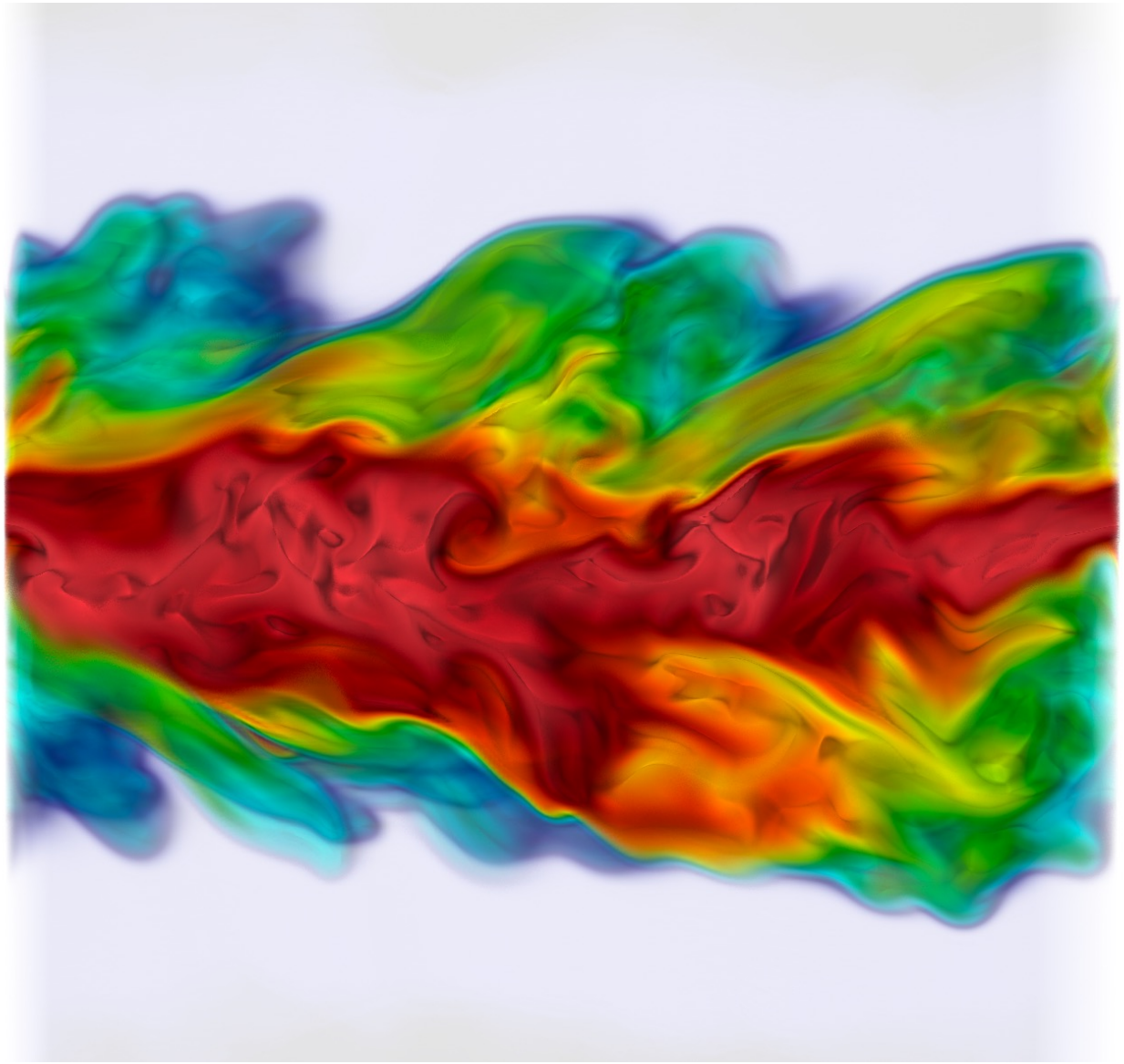
Applications: Biology



Biological soil samples

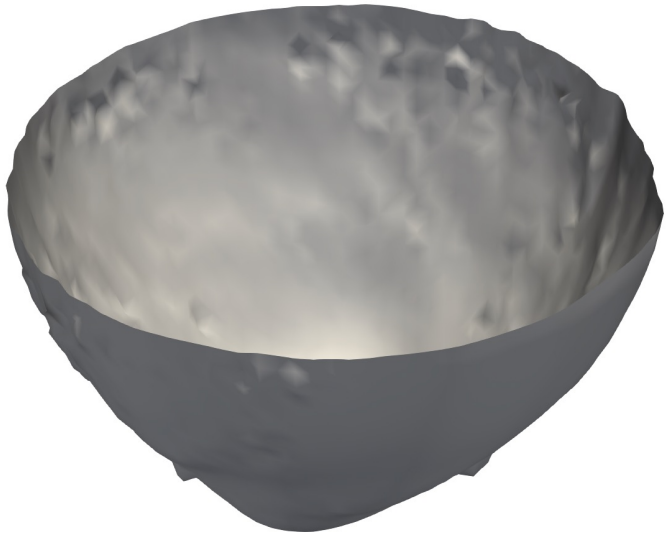
Applications: Computational Sciences

Study Combustion process

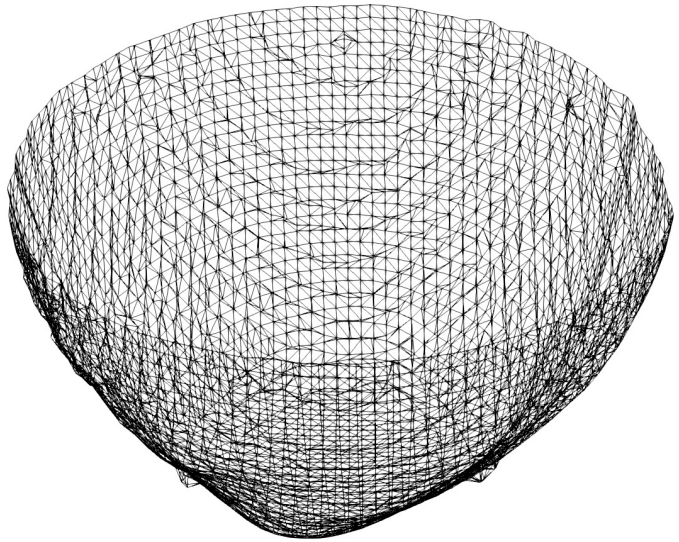


Indirect Visualization of Volume Data

- Isosurface based rendering for 3D data
  - Example of indirect technique for volume data exploration
  - Using geometric representations
    - Points, meshes, surfaces, etc.



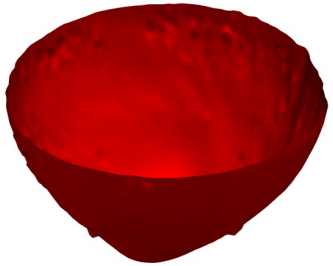
Isosurface



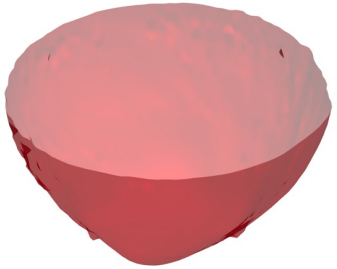
Geometric mesh of the isosurface

Volume Rendering

- Surface visualization is a way of showing data as opaque objects
  - Though you can apply transparency on surfaces
- Many applications demand techniques that allows "see-through" capability
  - Make parts of the data (semi)transparent so the data behind can be seen
- Volume Rendering technique is the answer!
  - Direct mapping of underlying 3D data into an image space
  - Assumes data as a translucent gel that allows light to go through



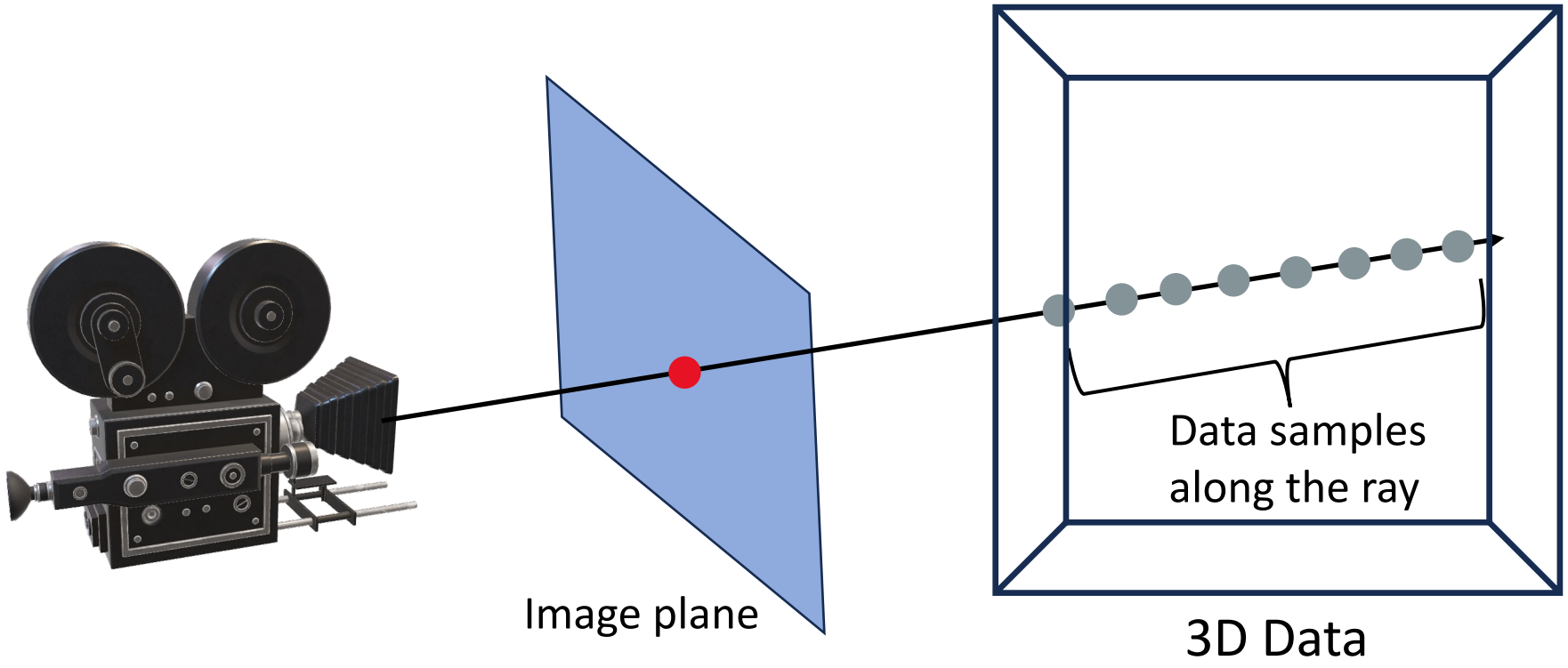
Opaque



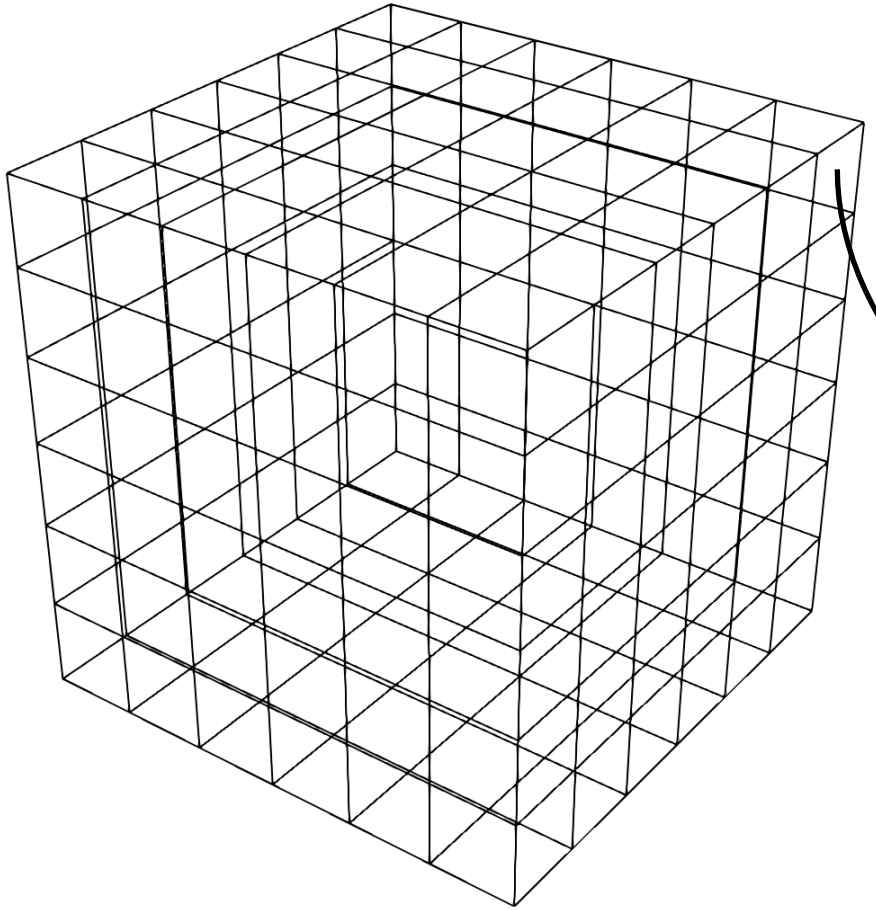
Semi-transparent

Volume Rendering Key Idea

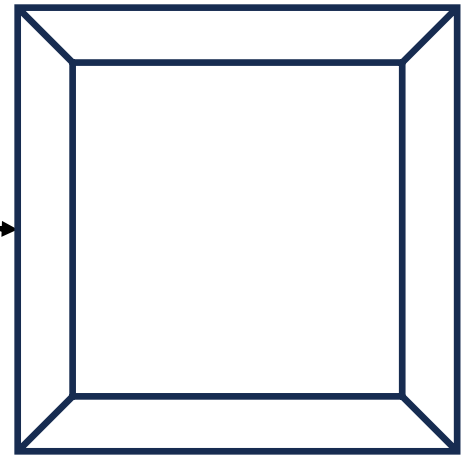
- Data is considered as a translucent gel
- Rays are cast into the volume data through each pixel to observe data values
- Rays accumulate color and opacity values along the ray for final pixel color



Sampling via Trilinear Interpolation

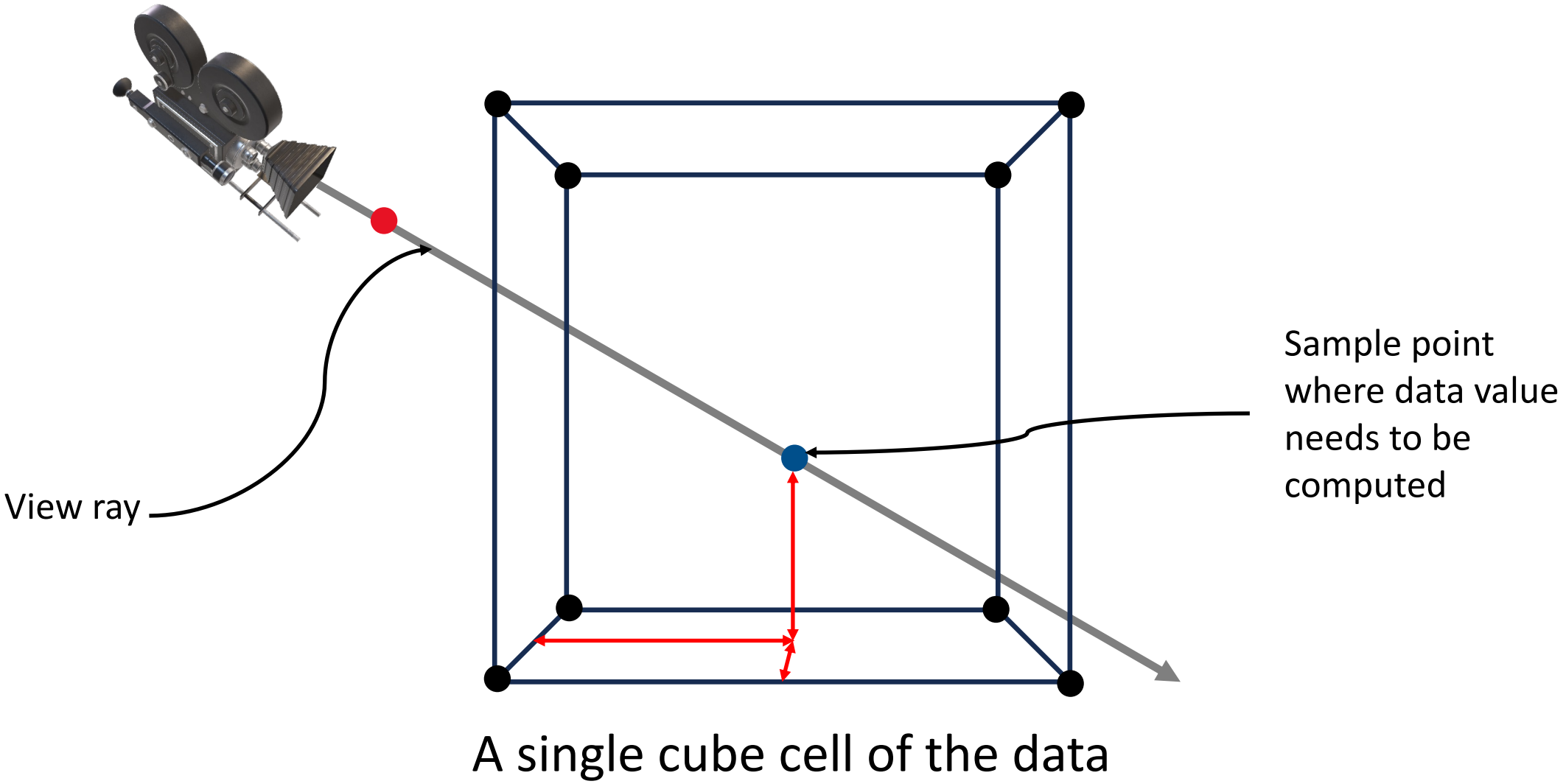


Grid of the data set



A single cube cell of the data

Sampling via Trilinear Interpolation



Different Volume Rendering Algorithms

- Image-order techniques
  - Ray casting approaches
- Object-order techniques
  - Splatting
  - Texture mapping

Different Volume Rendering Algorithms

- **Image-order techniques**
  - **Ray casting approaches**
- Object-order techniques
  - Splatting
  - Texture mapping

Image-Order Volume Rendering Techniques

- Typically known as Ray casting methods
- Given an image plane, for each pixel, we compute the color by casting a ray through each pixel to the data
- We evaluate the data along the ray using some pre-specified functions for computing the final pixel color

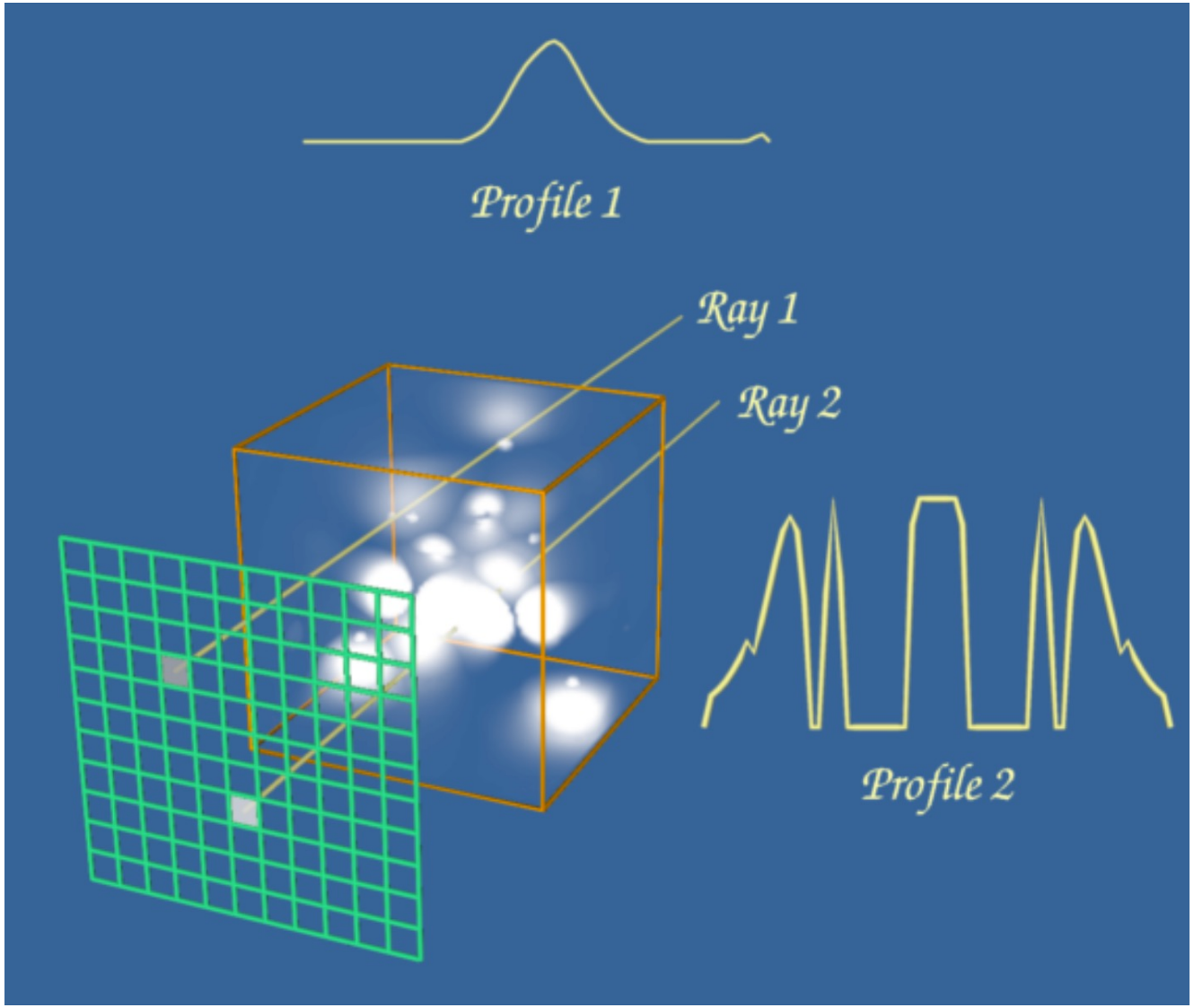
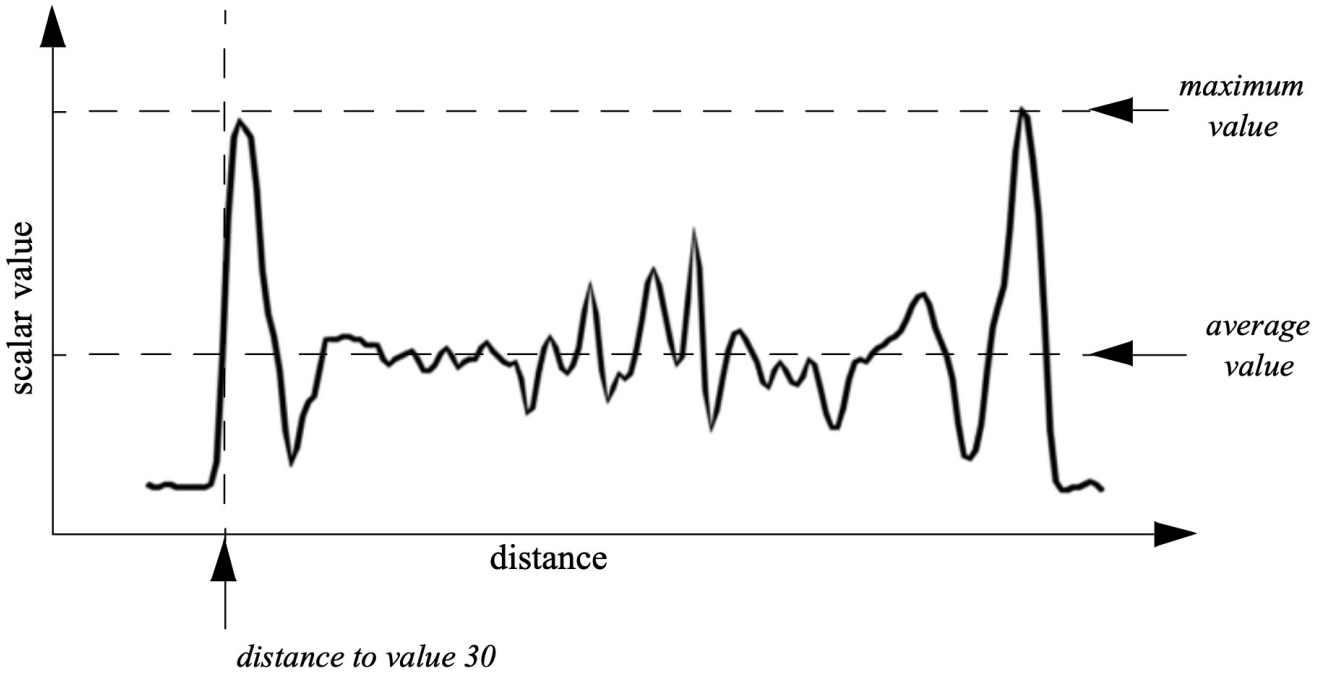


Image-Order Volume Rendering Techniques

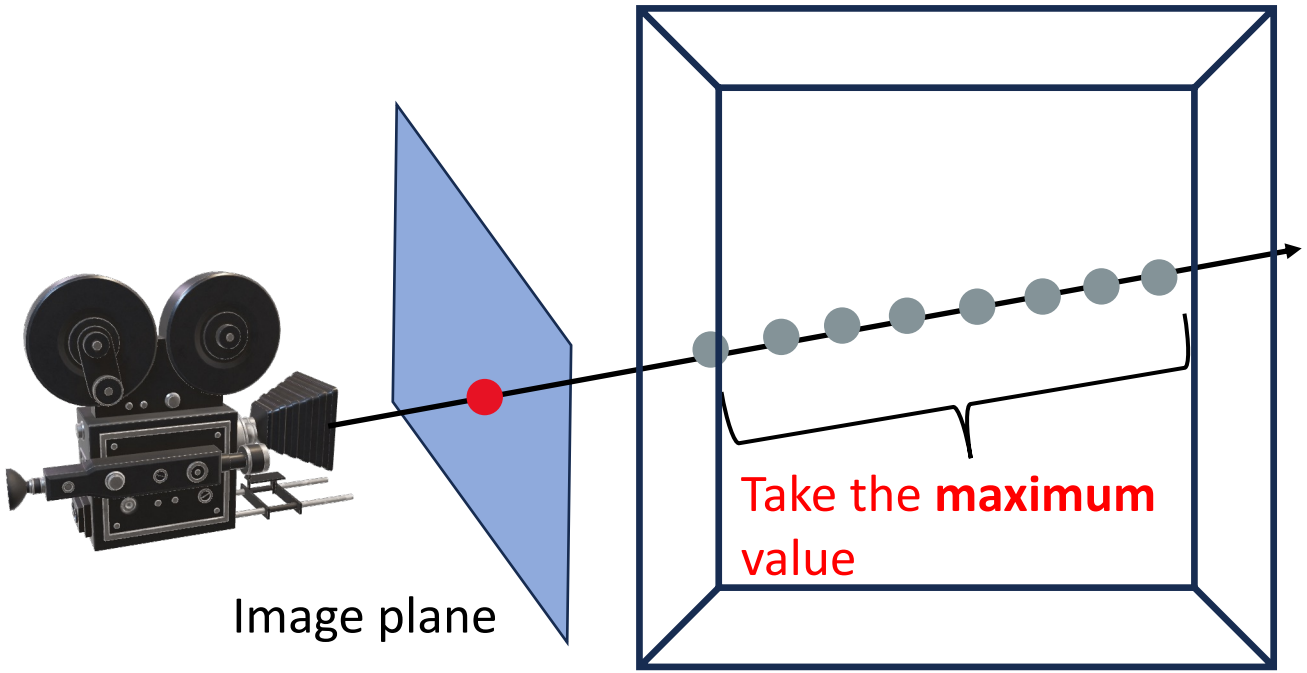
- How do we accumulate the data values along the ray to produce a final pixel color?

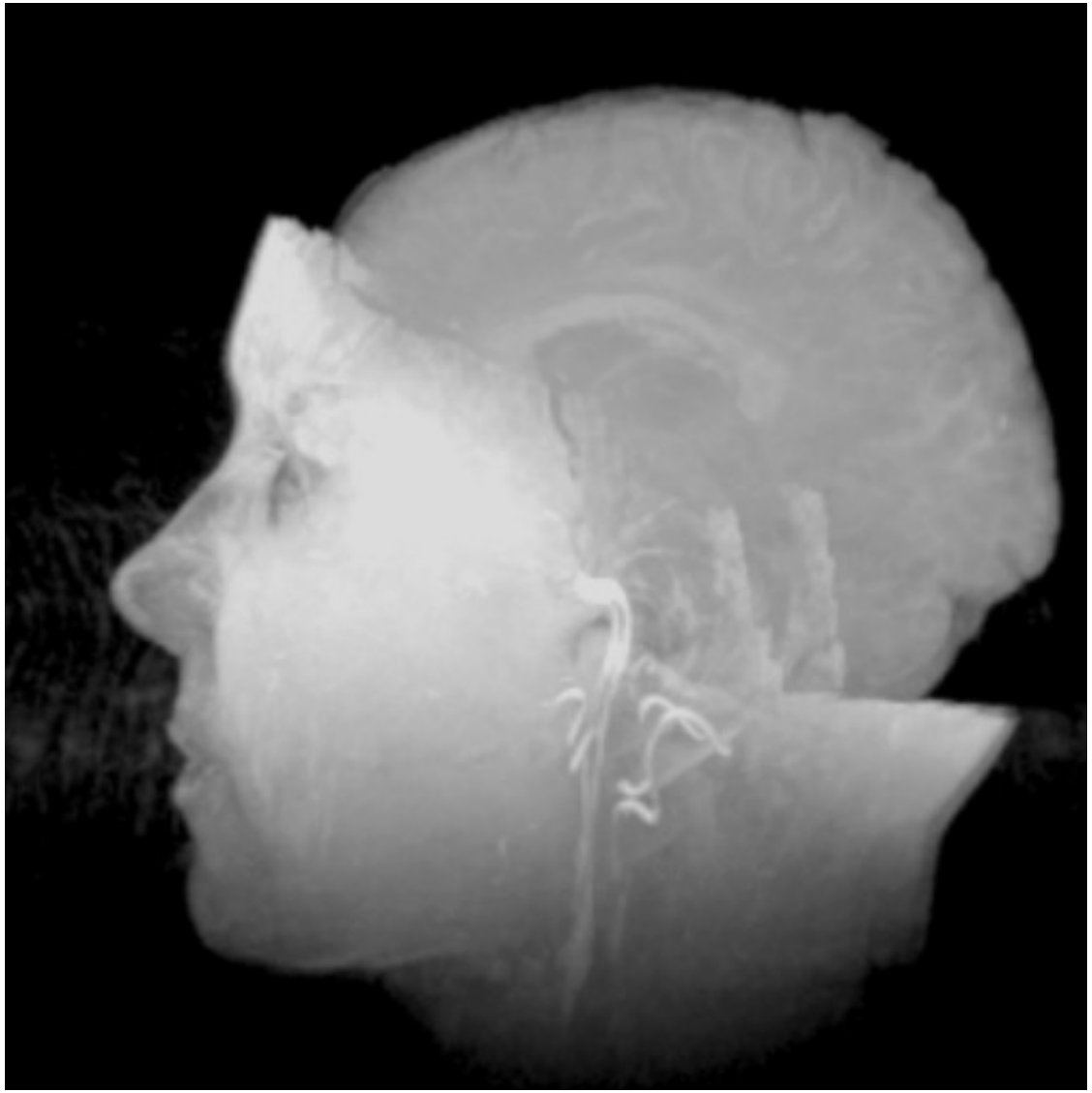
1. Maximum Intensity Projection (MIP)
2. Average Intensity Projection (AIP)
3. Distance to a value
4. Compositing



1. Maximum Intensity Projection (MIP)

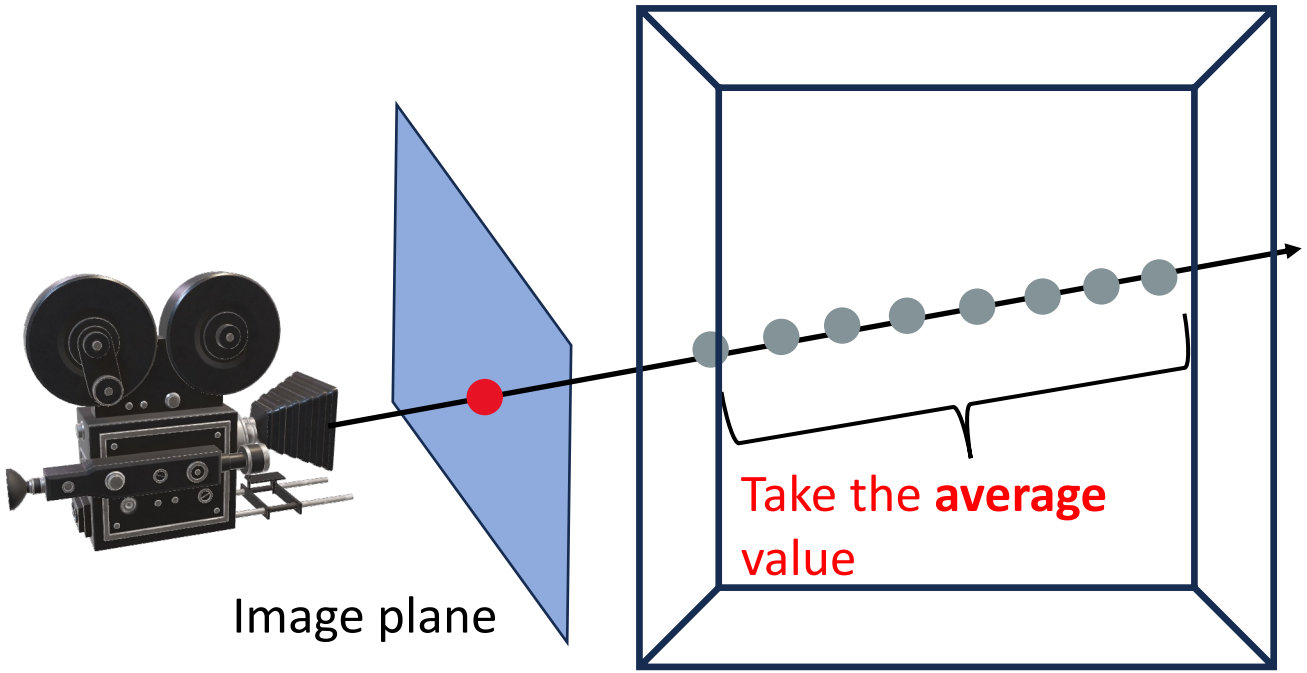
- Compute the **maximum** intensity (data) value along each casted ray and the map the value to a color using a color scale

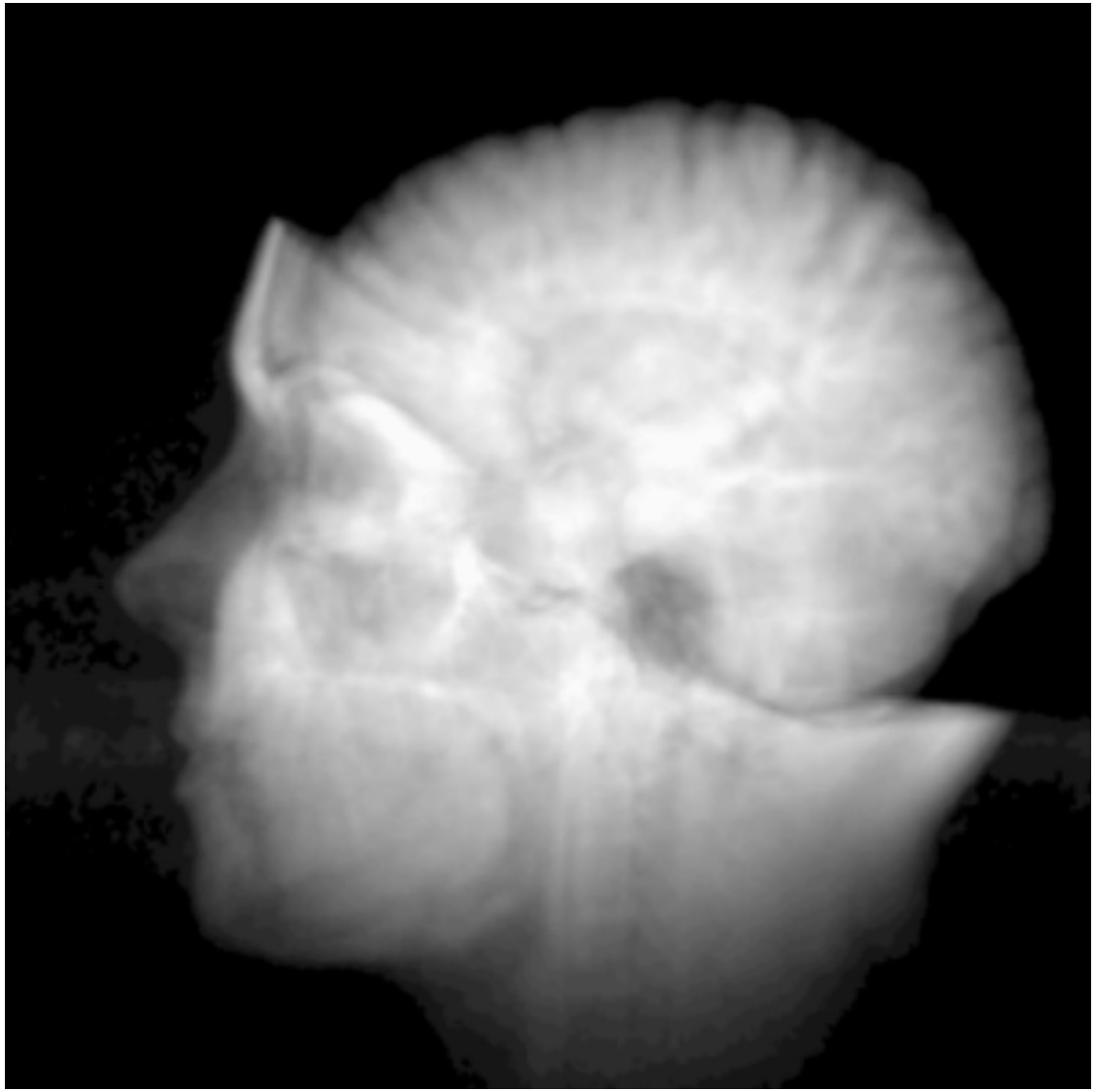




2. Average Intensity Projection (AIP)

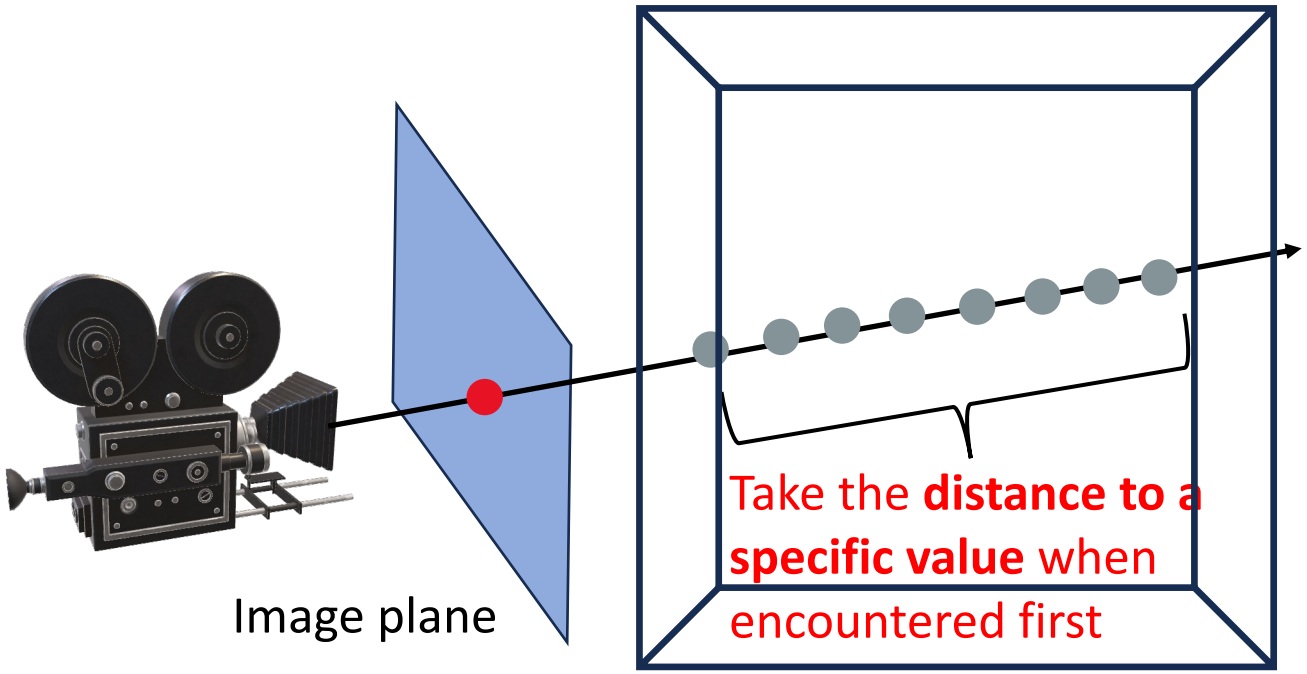
- Compute the **average** intensity (data) value along each casted ray and the map the value to a color using a color scale

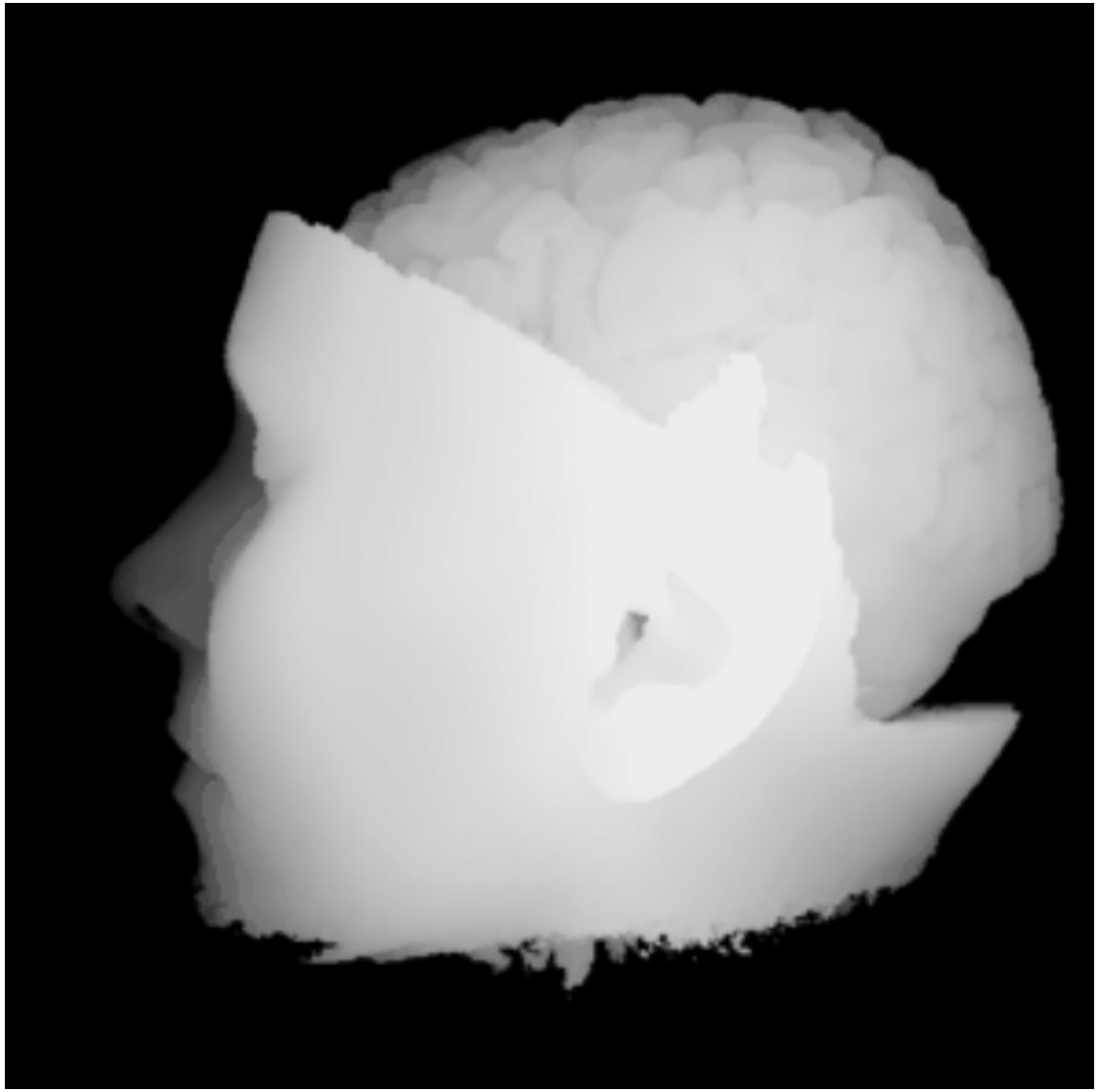




3. Distance to a Value Projection

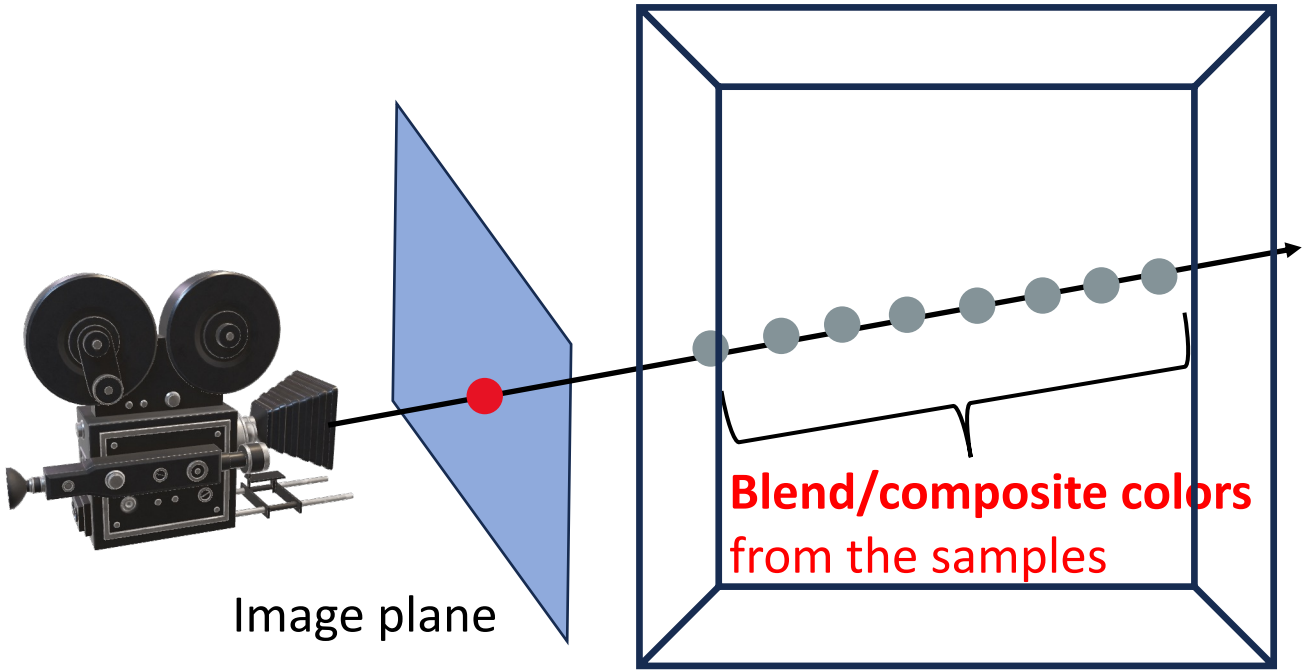
- Distance to value 30 is shown in the rendered image
  - Provides the notion of the depth as to where the data value 30 is encountered

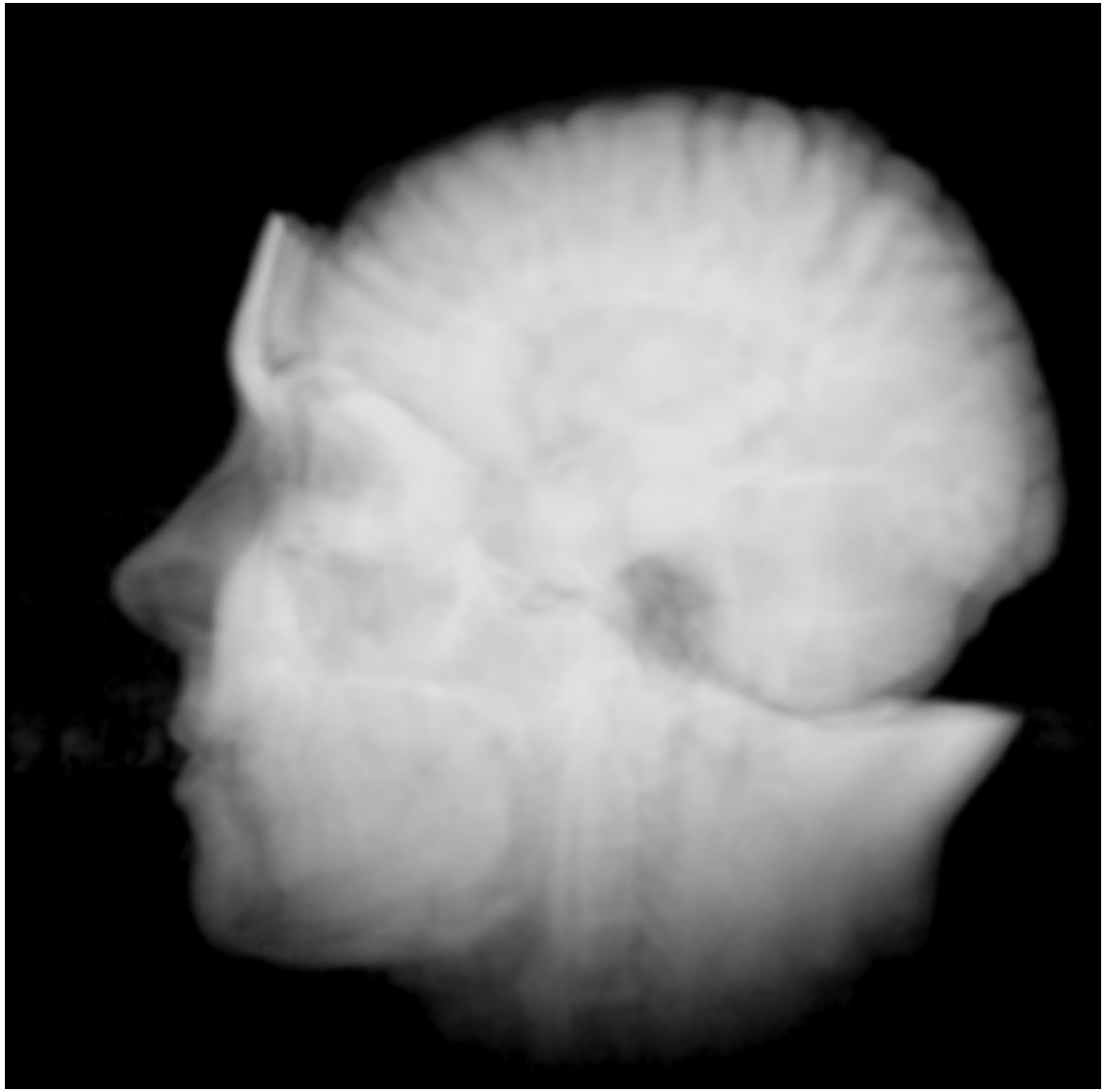




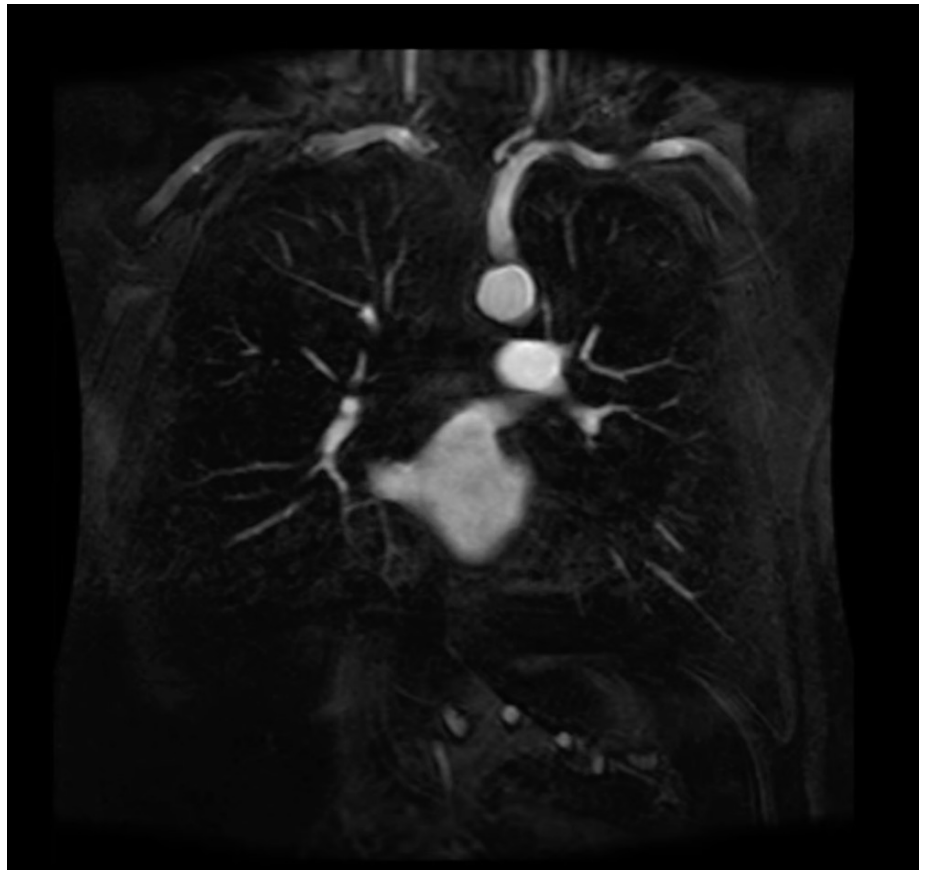
4. Composite Values Along the Ray

- Use an alpha (opacity) composting technique to blend the color values obtained from data samples along the ray





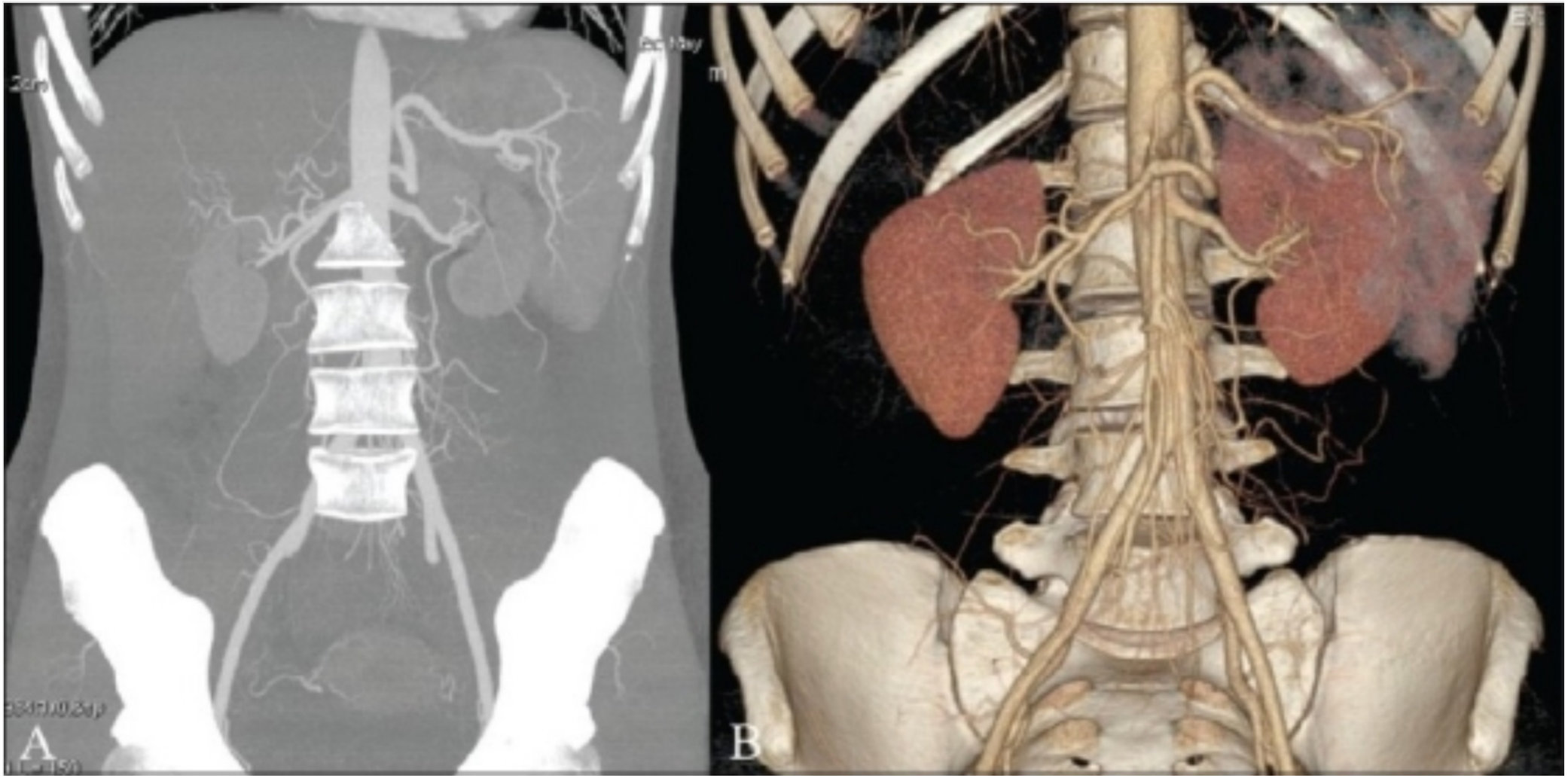
AIP vs MIP



AIP: Lower noise and smoother edges

MIP: Good for bright structures

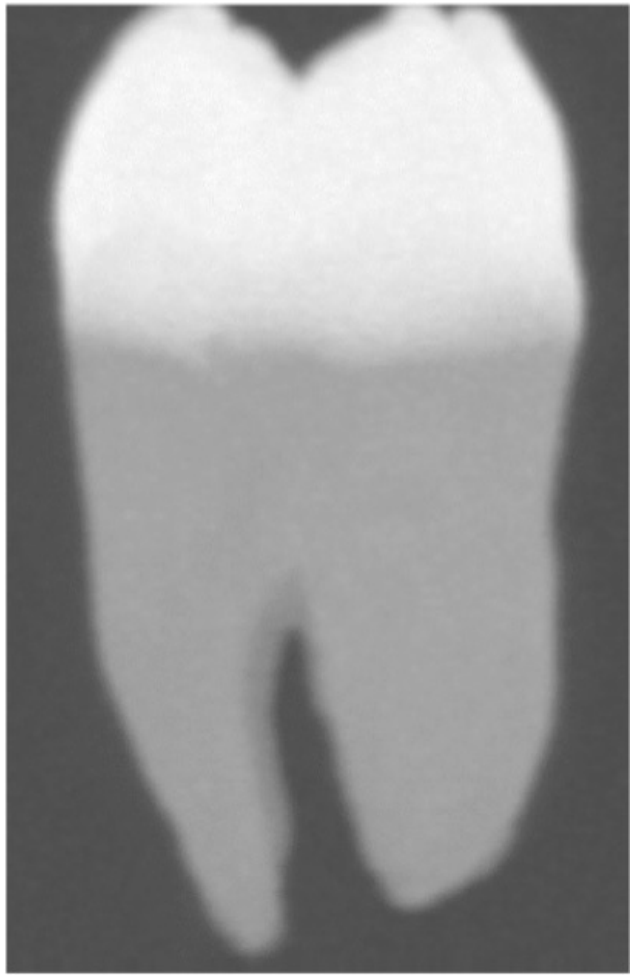
MIP vs Compositing-based Volume Rendering



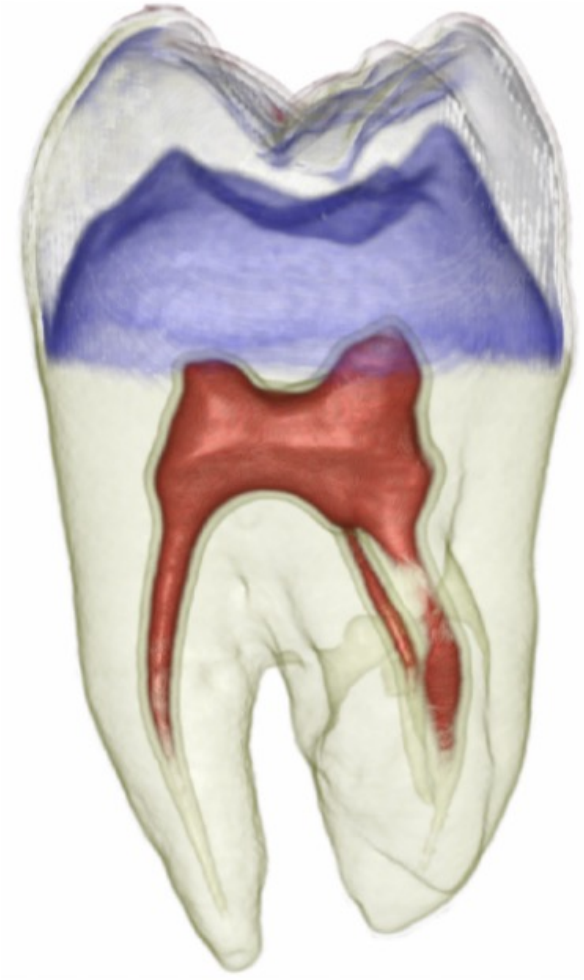
Maximum Intensity Projection

Compositing

MIP vs Compositing-based Volume Rendering



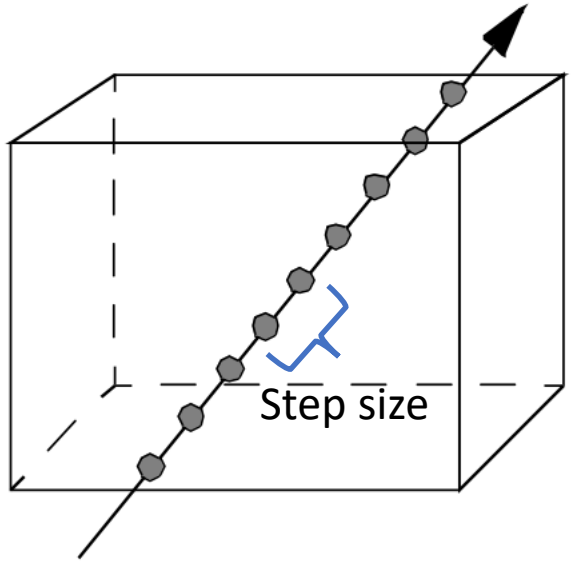
Maximum Intensity Projection

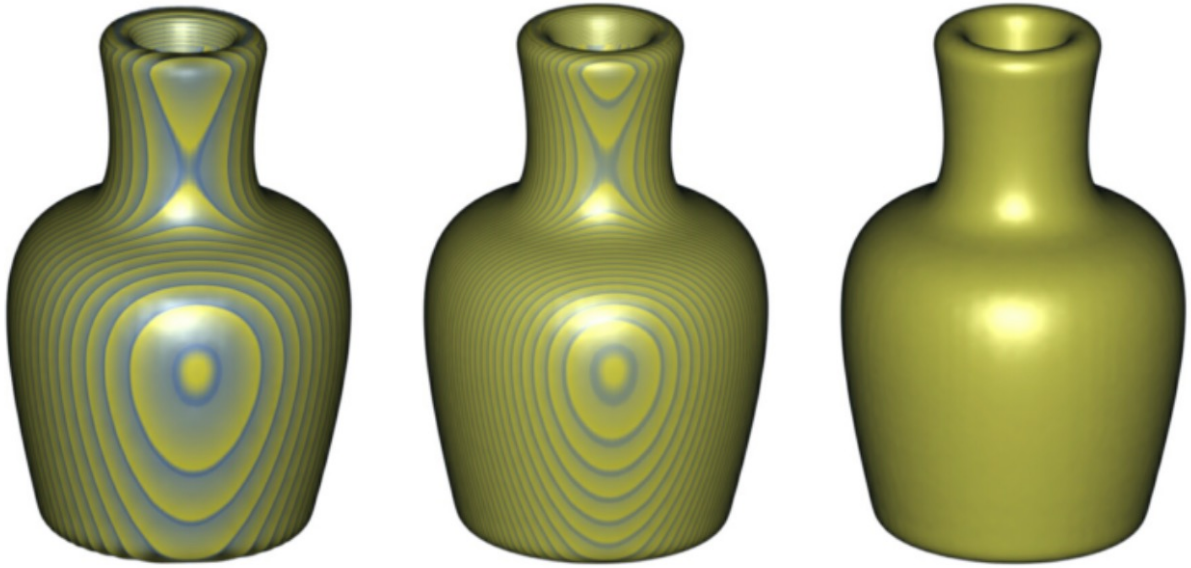


Compositing

Effect of Step Size During Ray Traversal

- The quality of the image produced from the data depends on the step size when each ray is traversed through the data
  - Large hop/step size causes artifacts in the final image
  - Smaller step size makes image more accurate but also computationally more expensive

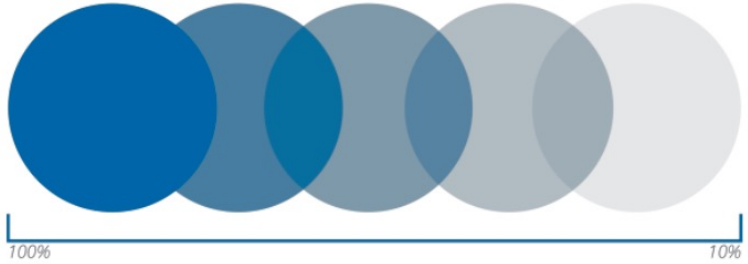




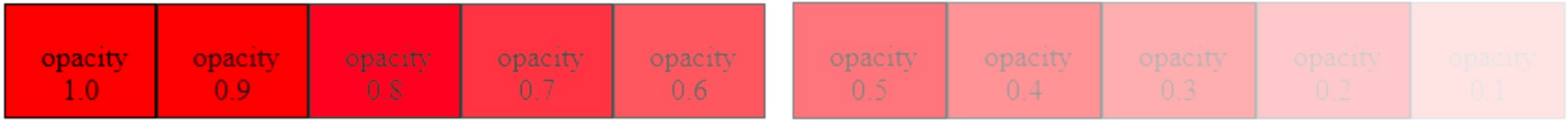
Step size = 2.0 Step size = 1.0 Step size = 0.1

Transparency and Opacity

- Other than RGB color values, there is one more channel – **opacity (A)**
  - Compute RGBA color components
  - Opacity (A) = 1 – transparency (T)
  - Range [0.0 ... 1.0]



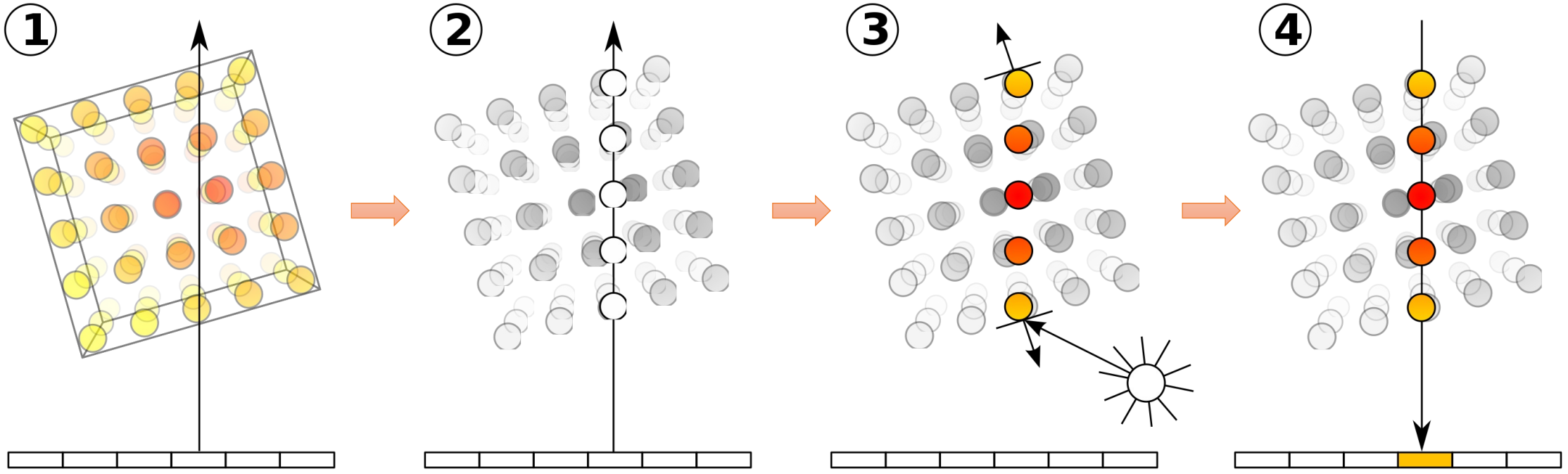
- Opacity (A) multiplied by RGB color creates a weighting effect



Ray Casting and Compositing

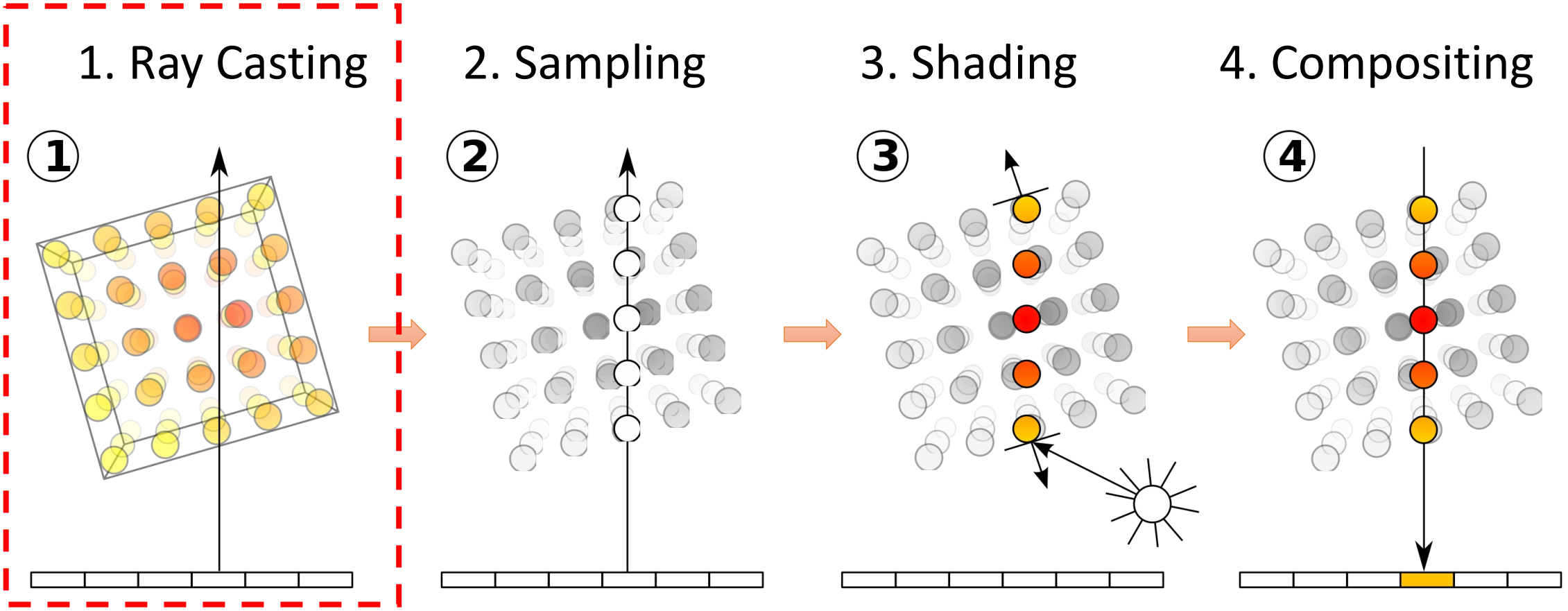
- Direct Volume Rendering

1. Ray Casting
2. Sampling
3. Shading
4. Compositing



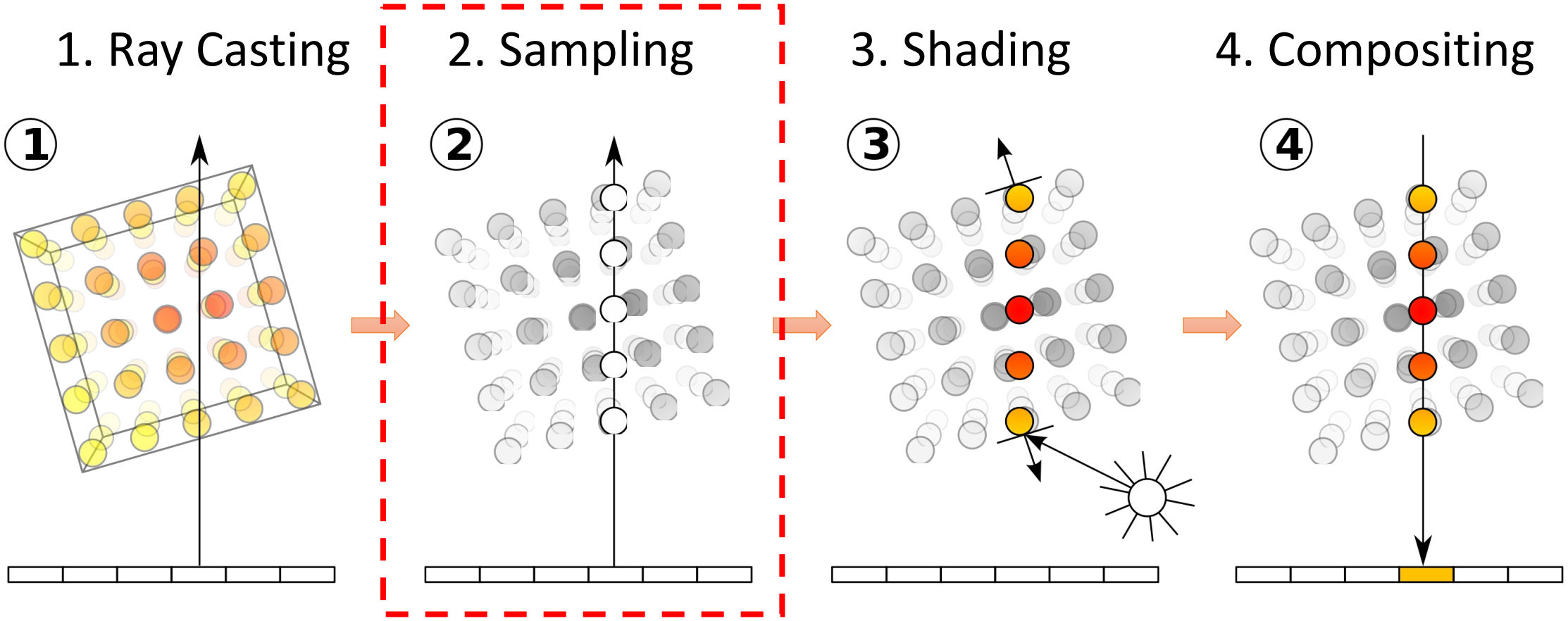
Ray Casting and Compositing

- Direct Volume Rendering



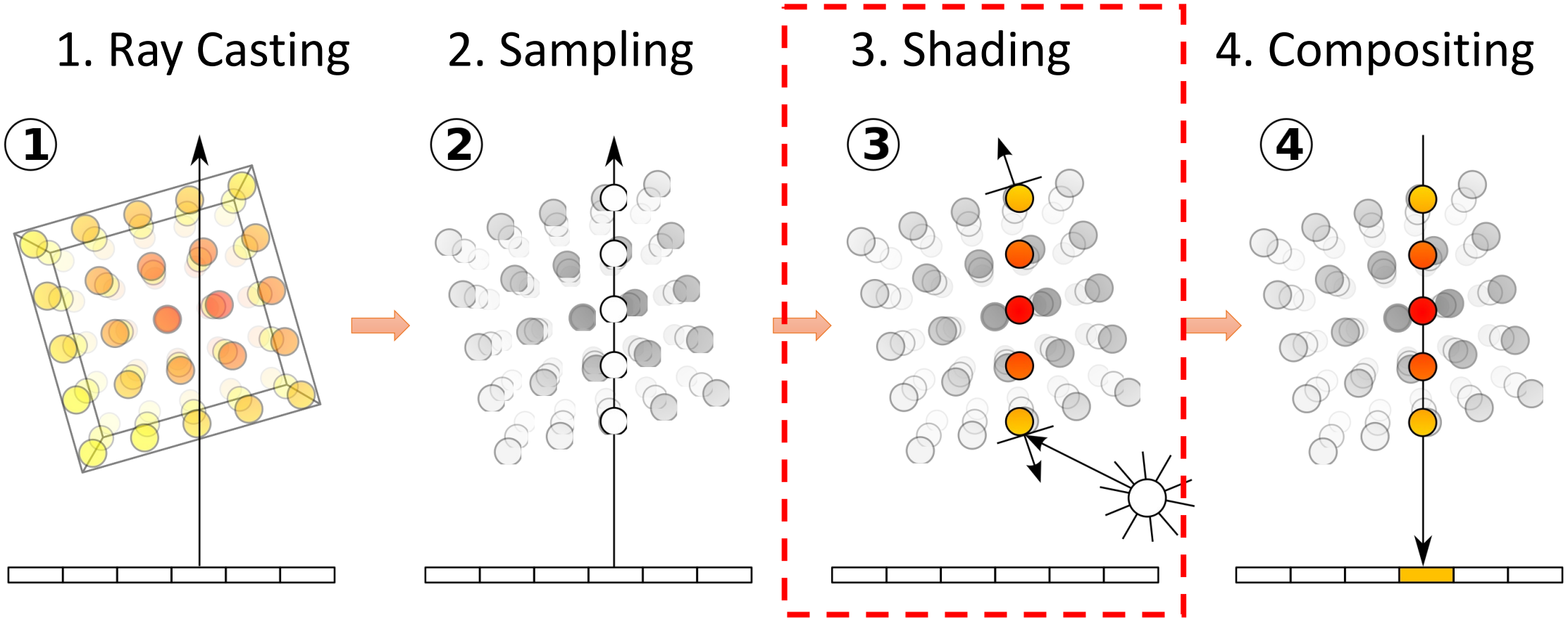
Ray Casting and Compositing

- Direct Volume Rendering



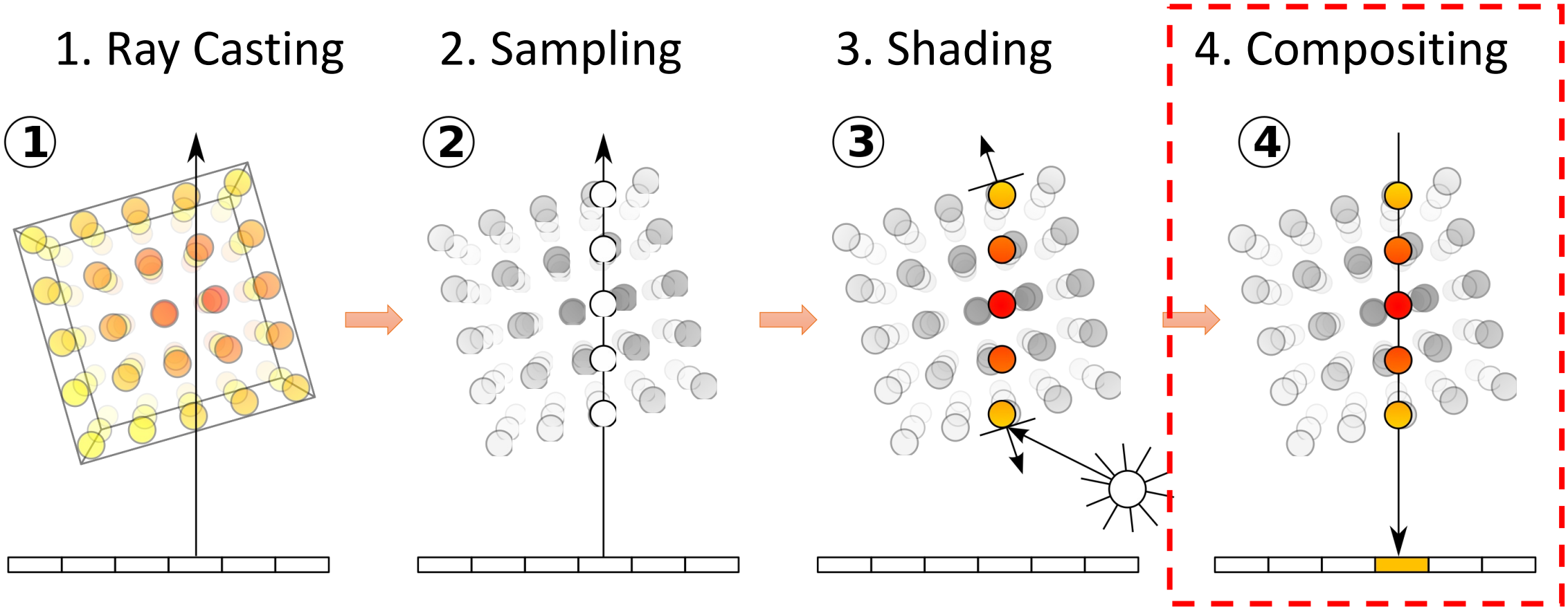
Ray Casting and Compositing

- Direct Volume Rendering



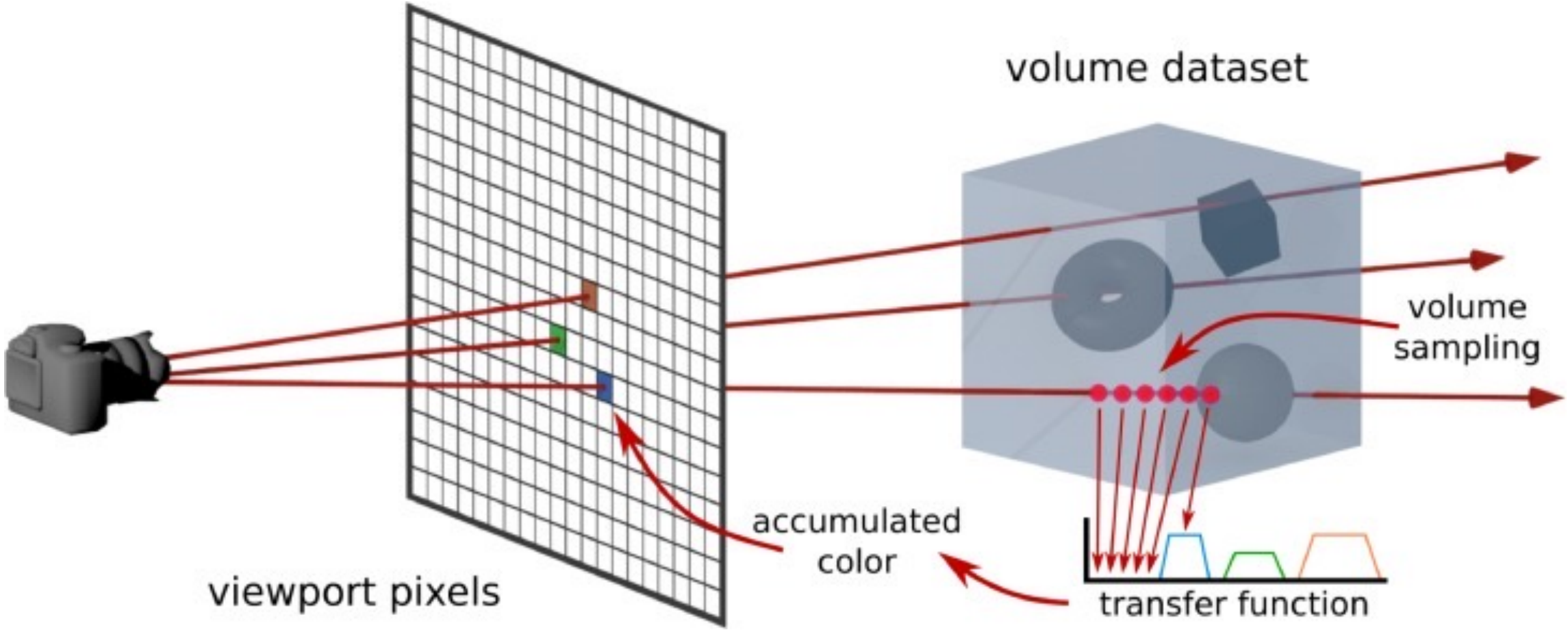
Ray Casting and Compositing

- Direct Volume Rendering



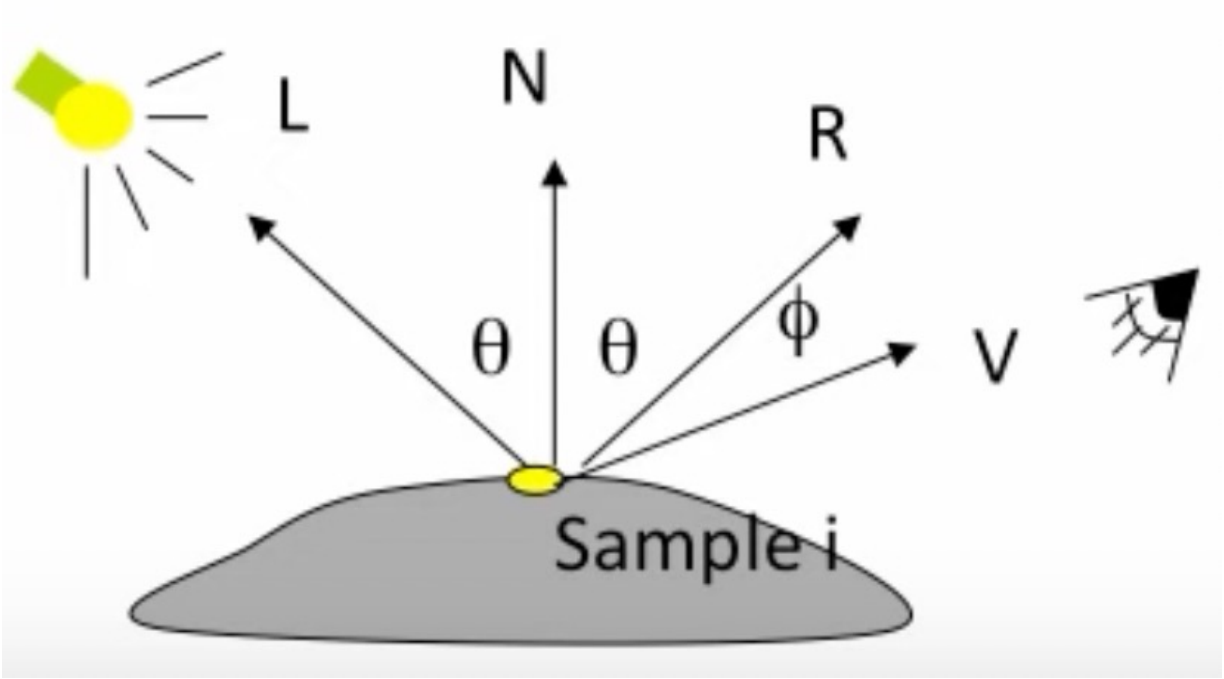
Ray Casting and Compositing

- Direct Volume Rendering



Shading: Phong Illumination Model

- Shading is the process of computing final color for each pixel considering its color, opacity, location of the viewer, distance and direction of the light, etc.
- Phong Illumination = ambient + diffuse + specular



Shading: Normal Vector for a Volume Data?

- Gradient is used as a proxy for the normal vector in scalar data

$$\nabla V(x,y,z) = \left(\frac{\partial V}{\partial x}, \frac{\partial V}{\partial y}, \frac{\partial V}{\partial z}\right)$$

∇V is the gradient vector

- How to compute the gradient vector?
  - Central Differences

$$\frac{\partial V}{\partial x} \approx \frac{V(x+1,y,z) - V(x-1,y,z)}{2\Delta x}$$

$$\frac{\partial V}{\partial y} \approx \frac{V(x,y+1,z) - V(x,y-1,z)}{2\Delta y}$$

$$\frac{\partial V}{\partial z} \approx \frac{V(x,y,z+1) - V(x,y,z-1)}{2\Delta z}$$

Shading: Normal Vector for a Volume Data?

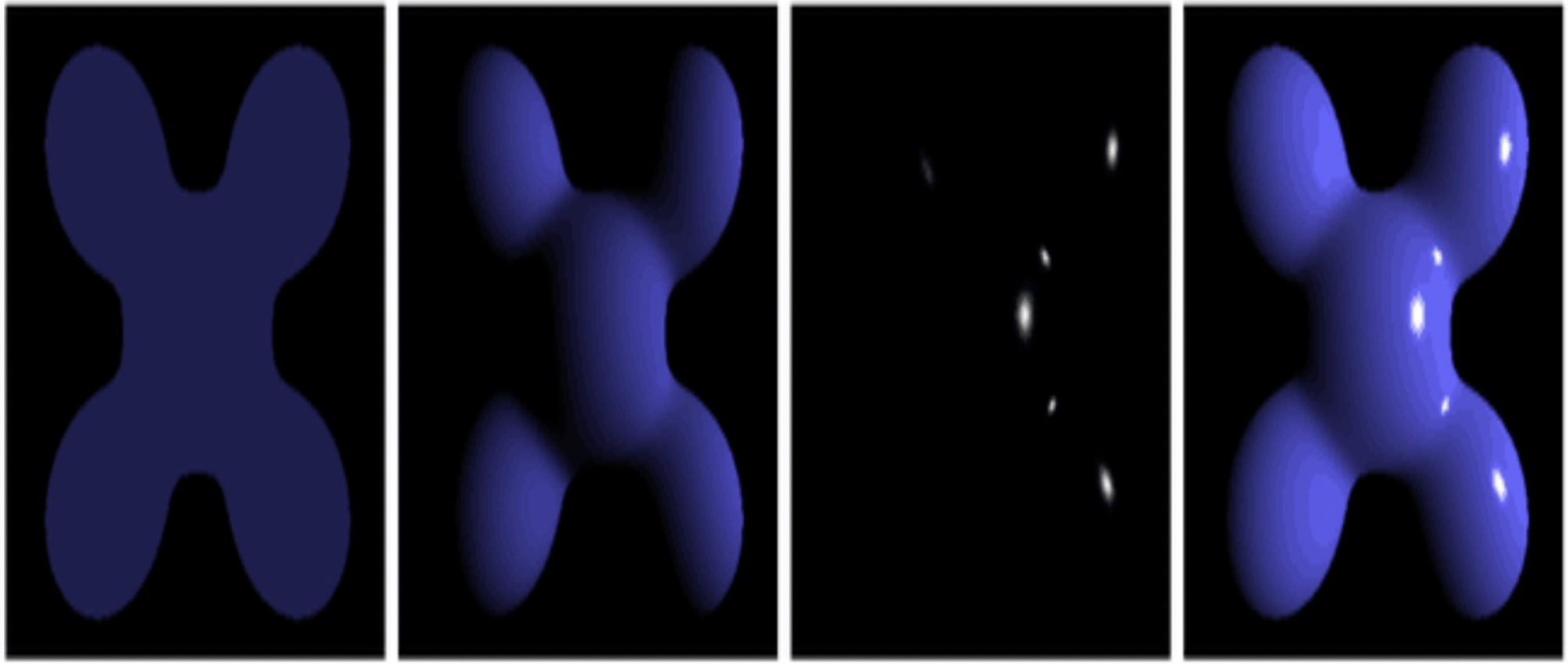
- Why Gradient makes sense to use as the normal vector in scalar data?
  - The gradient vector points in the direction of the maximum change in V
  - The gradient is perpendicular (normal) to the isosurface at a point

$$\nabla V \cdot \mathbf{T} = 0$$ $\mathbf{T}$ is a vector tangent to the surface

  - This perpendicularity ensures that the gradient naturally aligns with the normal vector required for lighting calculations

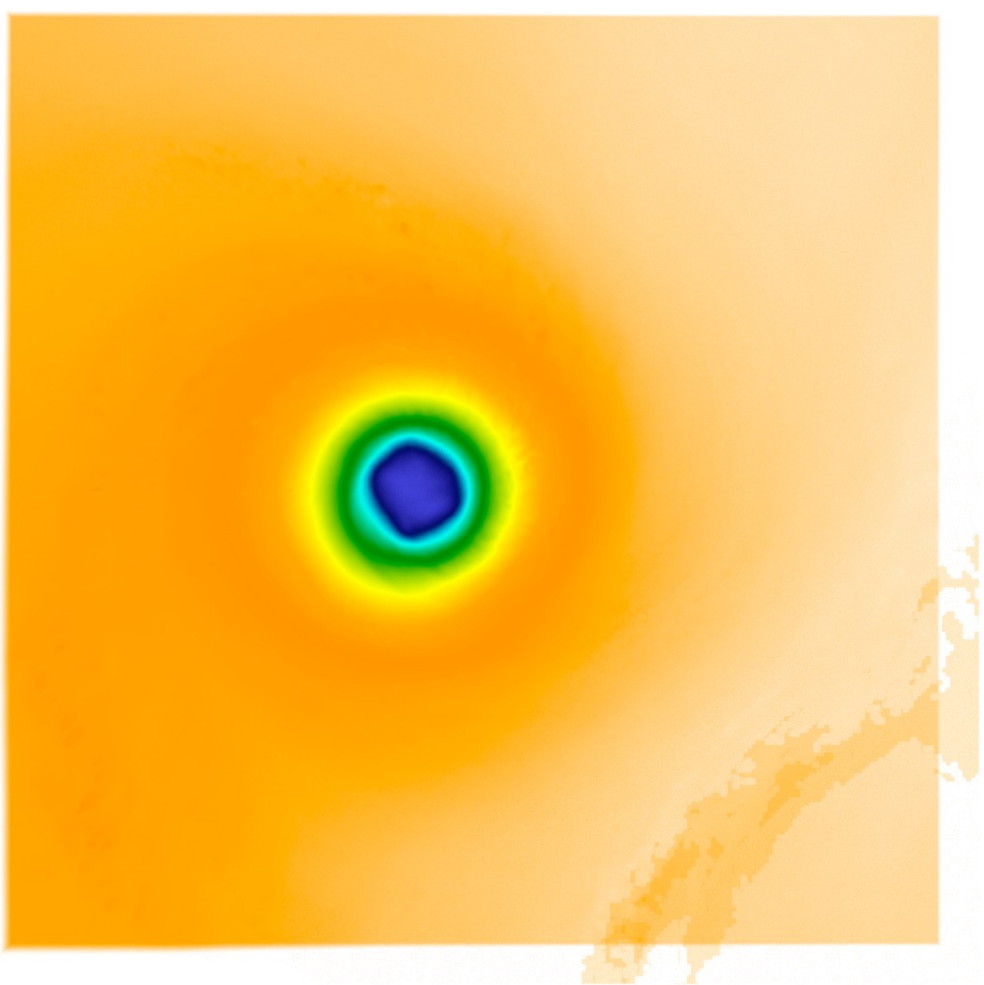
Shading: Phong Illumination Model

- Illumination = ambient + diffuse + specular

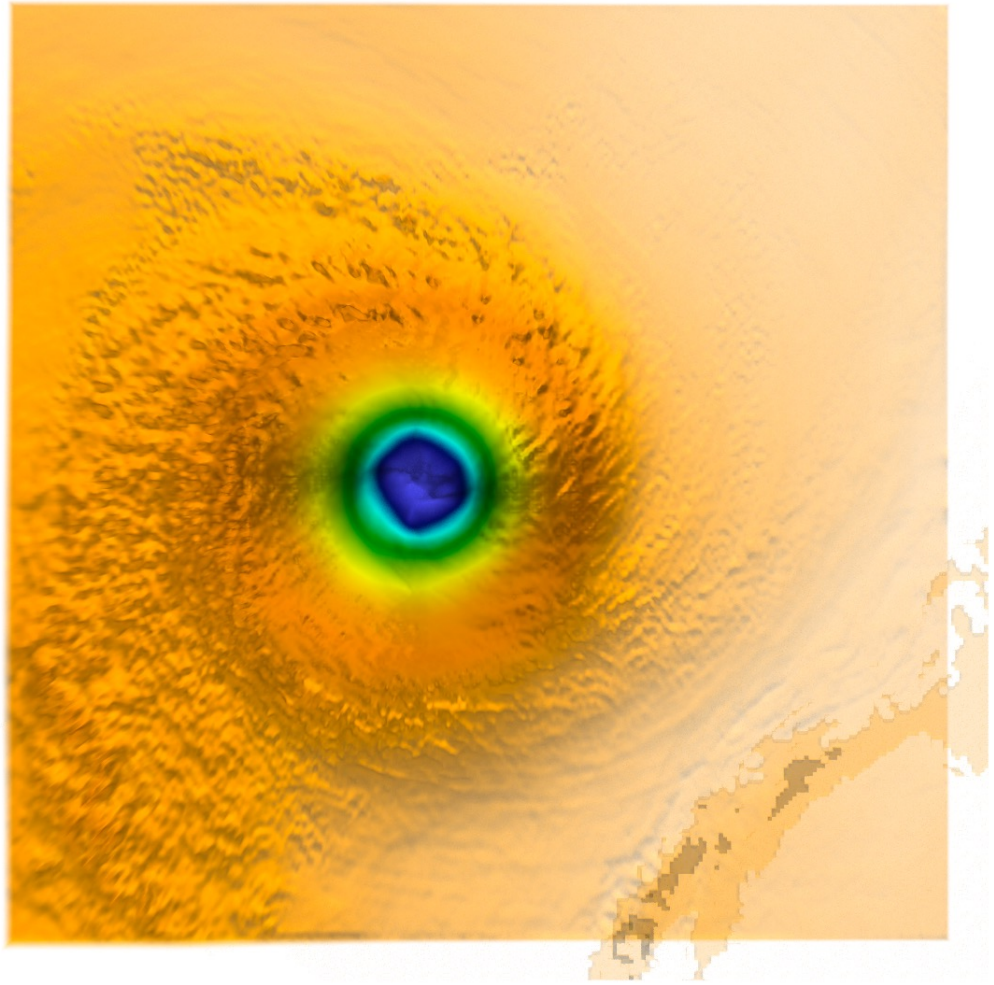


Ambient | Diffuse | Specular | Ambient + Diffuse + Specular

Effect of Shading in Volume Visualization

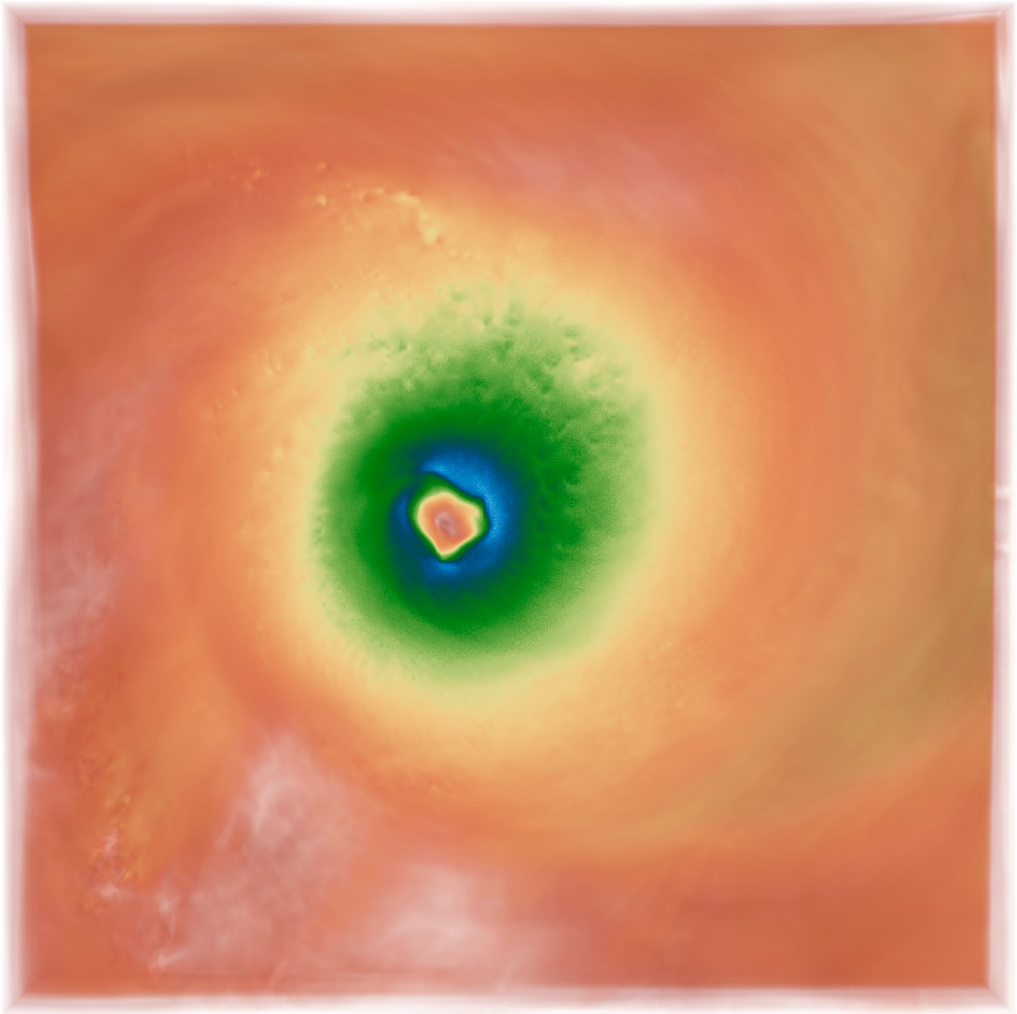


Without shading

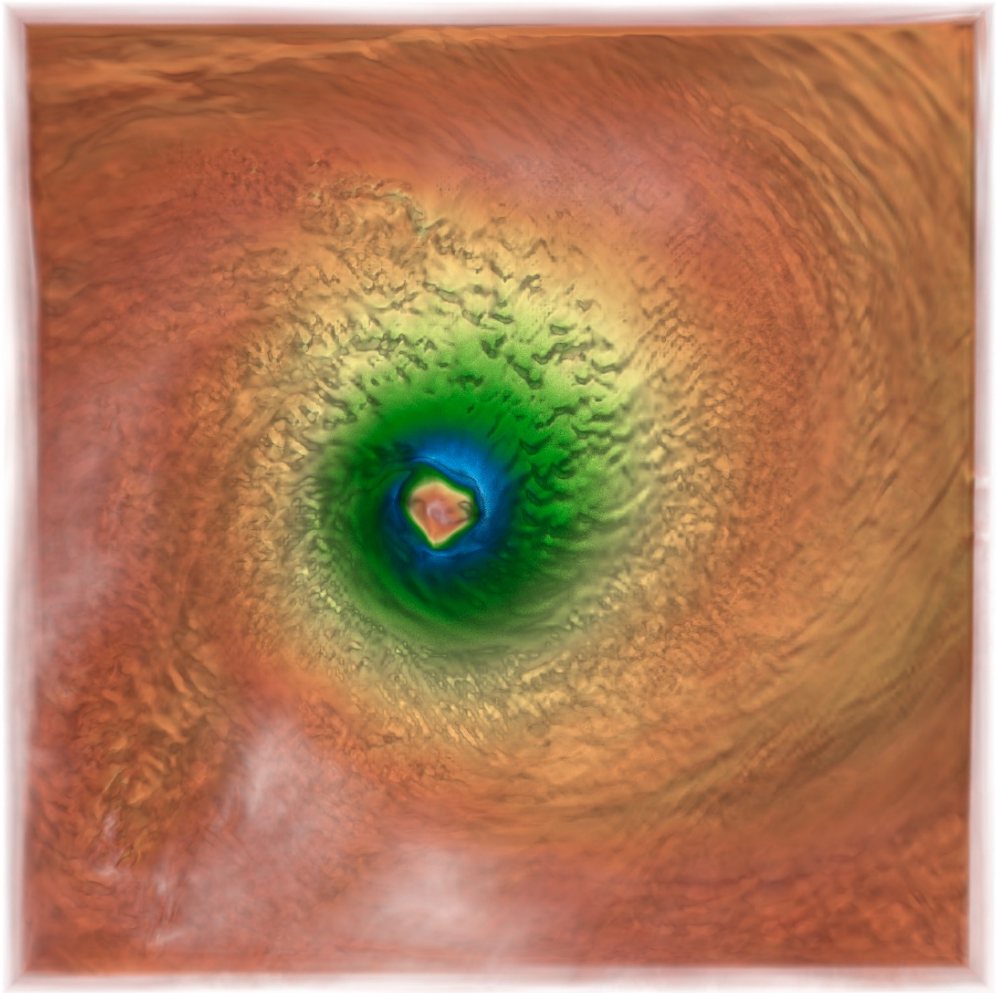


With shading

Effect of Shading in Volume Visualization



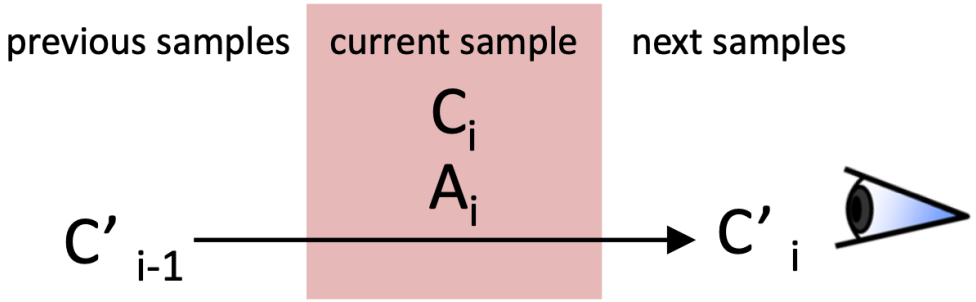
Without shading



With shading

Opacity and Color Blending: Compositing

Back-to-front rendering



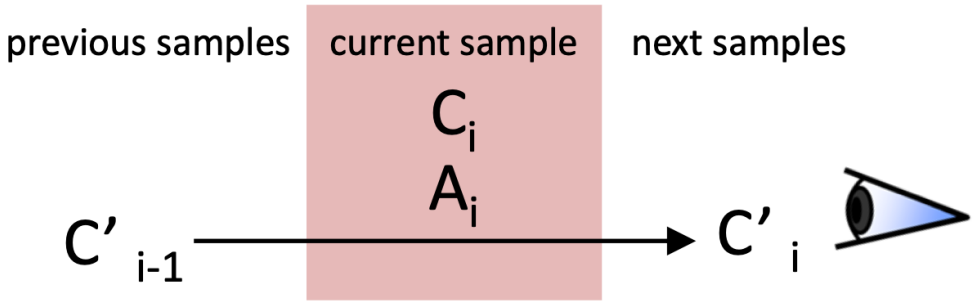
$$C_i^{'} = C_i A_i + (1 - A_i) C_{i-1}^{'}$$

A: Opacity = 1- Transparency = 1 - T

C: Color

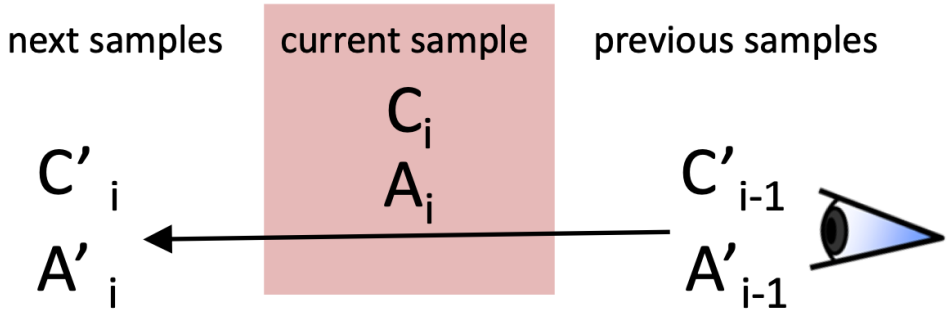
Opacity and Color Blending: Compositing

Back-to-front rendering



$$C_i' = C_i A_i + (1 - A_i) C_{i-1}'$$

Front-to-back rendering



$$C_i' = C_{i-1}' + (1 - A_{i-1}') C_i A_i$$

$$A_i' = A_{i-1}' + (1 - A_{i-1}') A_i$$

A: Opacity = 1- Transparency = 1 - T

C: Color