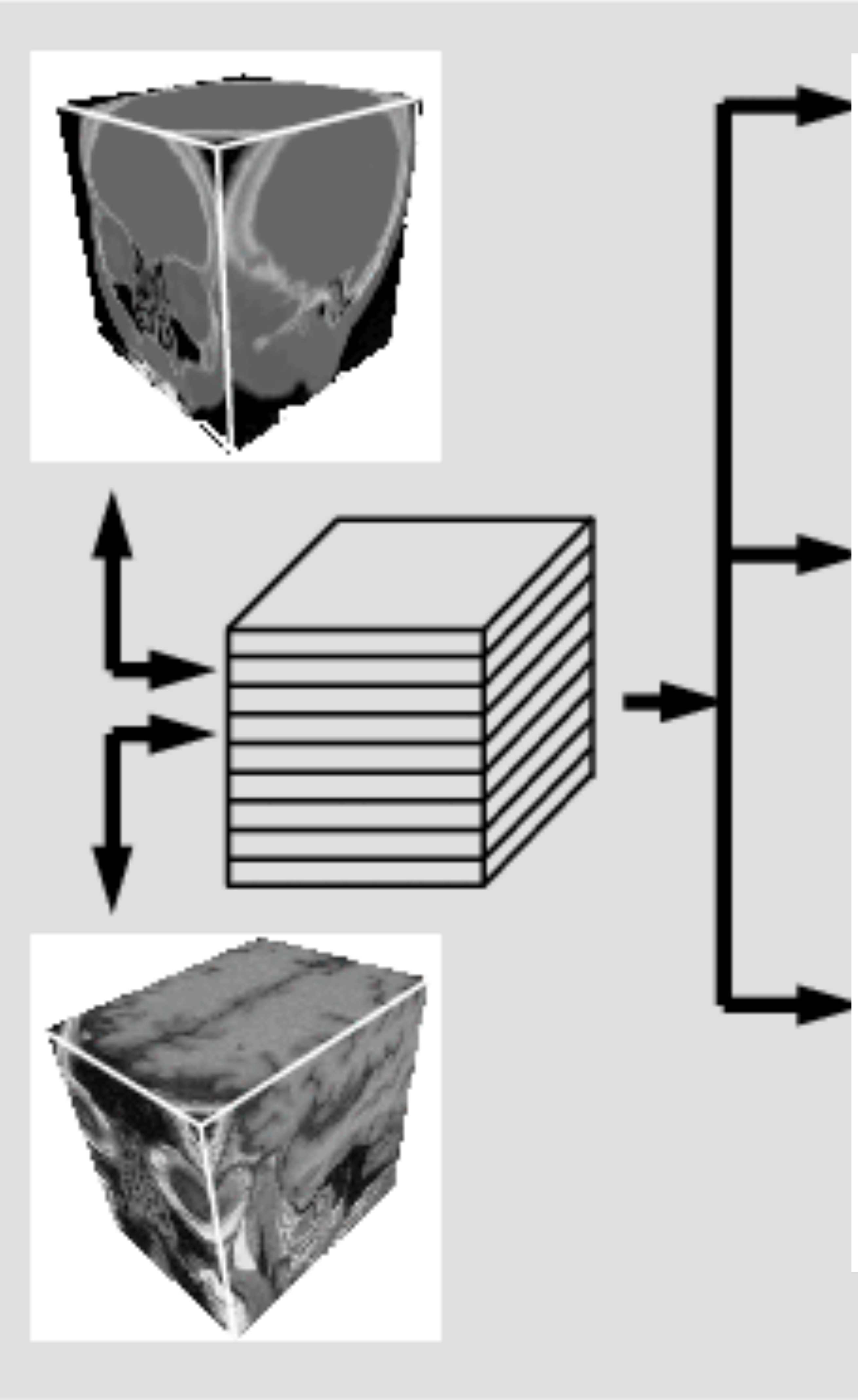
Data Visualization (CSCI 627/490)

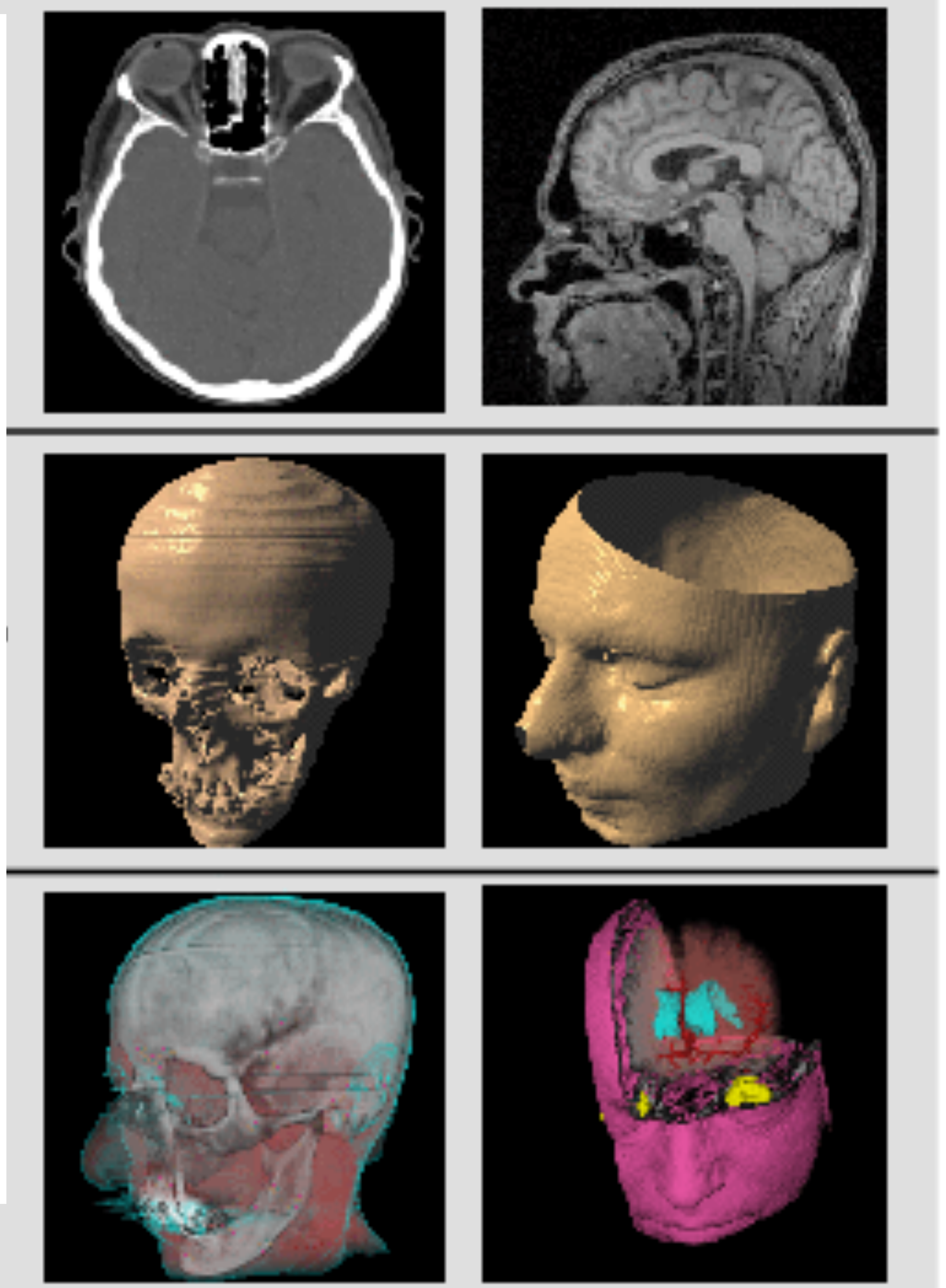
Volume Rendering & Vector Field Visualization

Dr. David Koop

Visualizing Volume (3D) Data

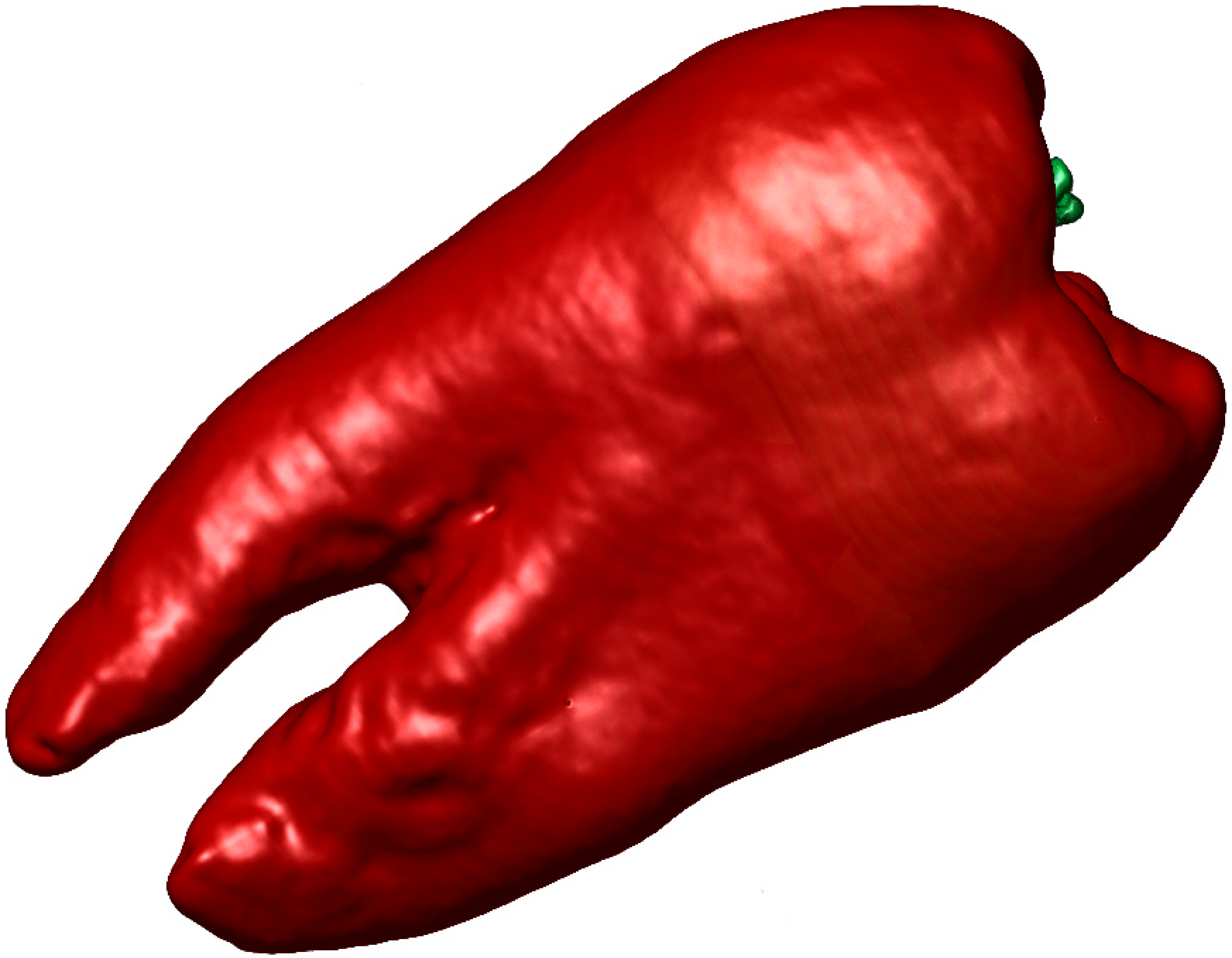


- 2D visualization
  slice images
  (or multi-planar
  reformating MPR)
- *Indirect*
  3D visualization
  isosurfaces
  (or surface-shaded
  display SSD)
- *Direct*
  3D visualization
  (direct volume
  rendering DVR)

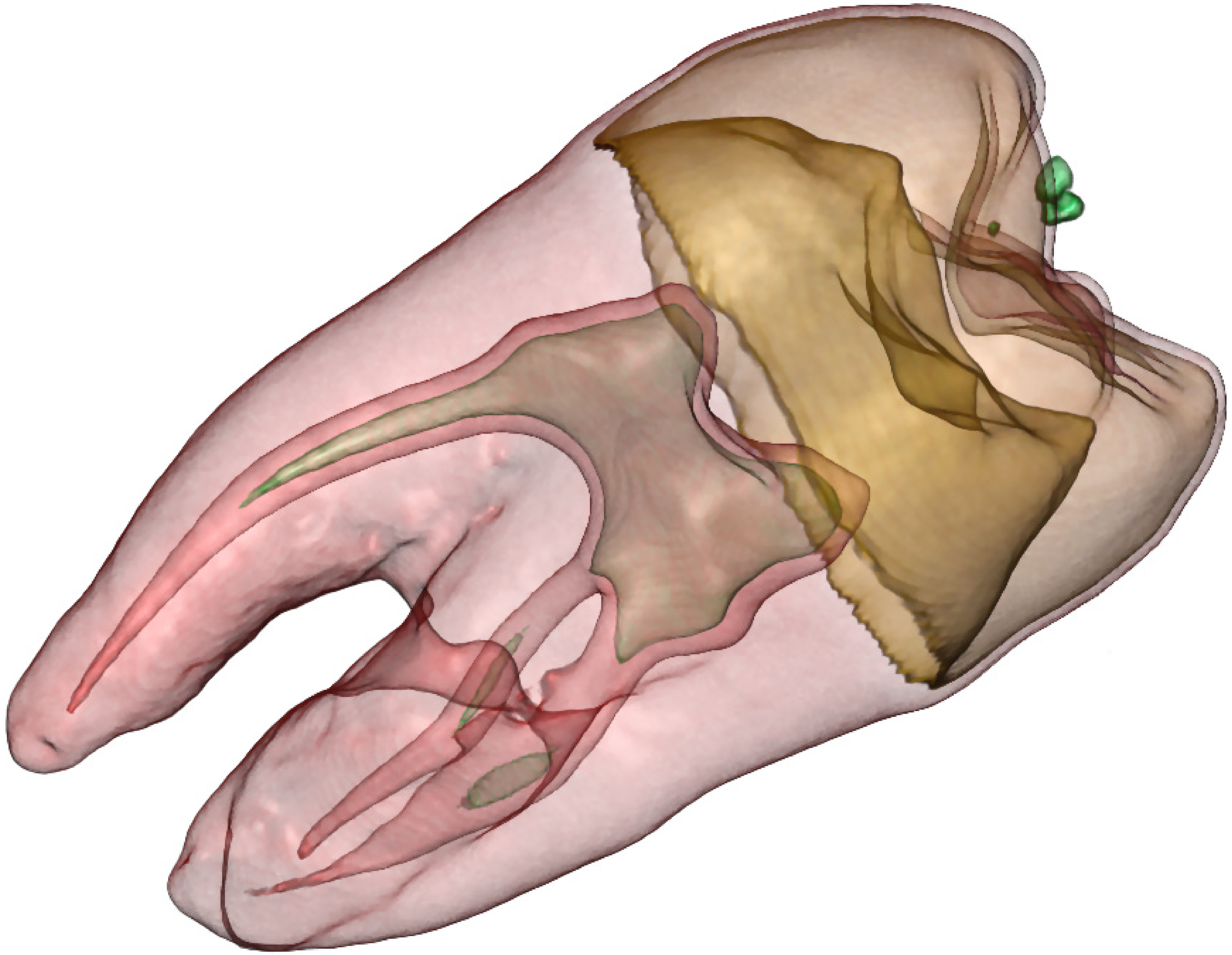


[© Weiskopf/Machiraju/Möller]

Isosurfacing



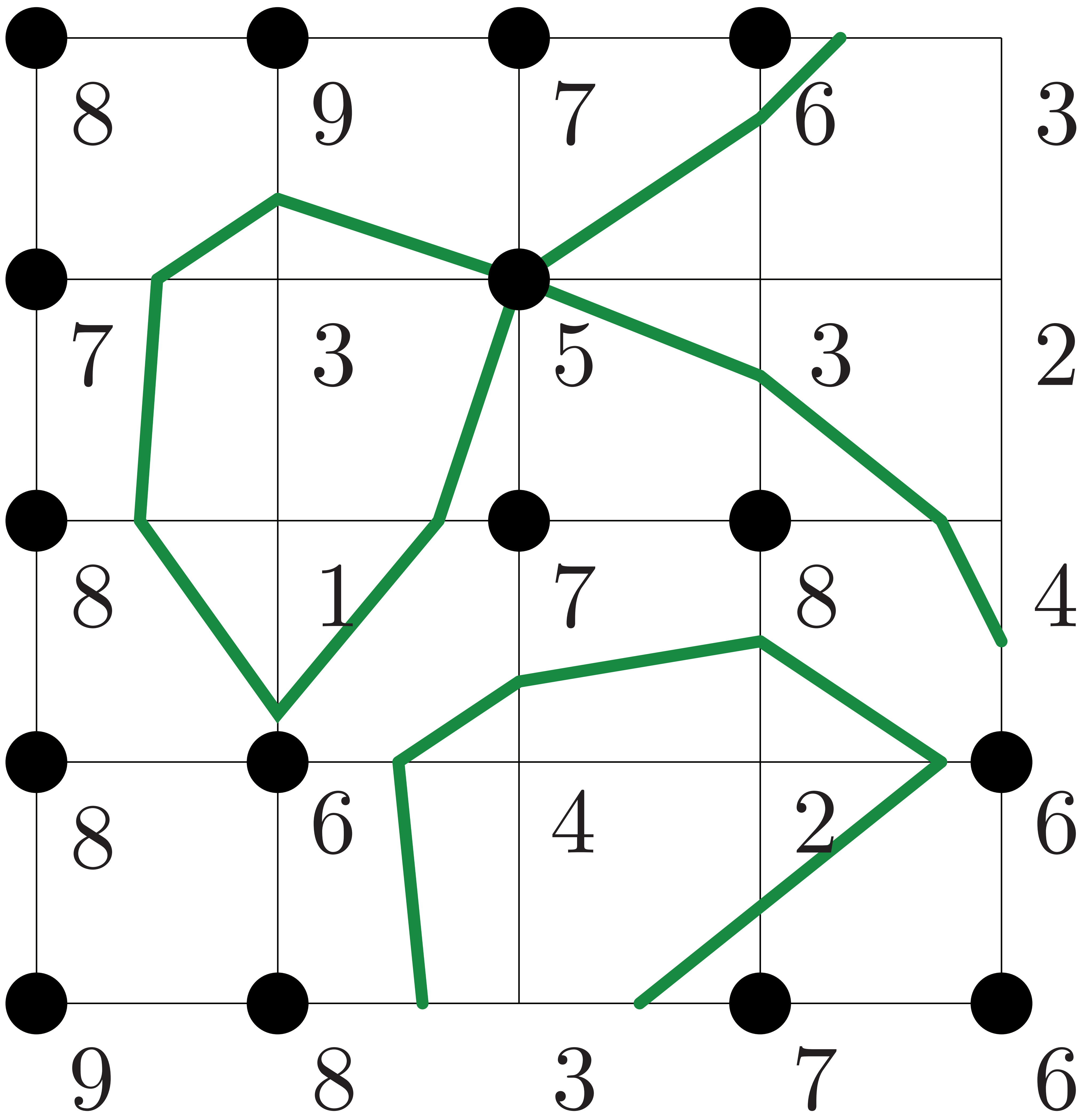
(a) An isosurfaced tooth.



(b) Multiple isosurfaces.

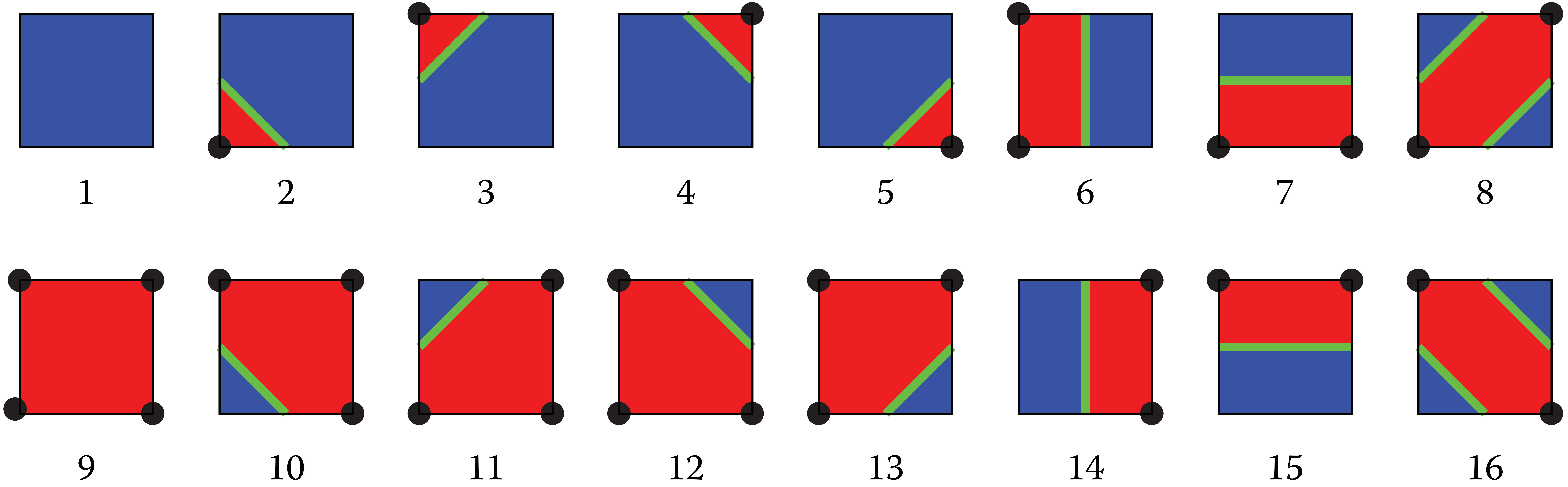
[J. Kniss, 2002]

Generating Isolines



[R. Wenger, 2013]

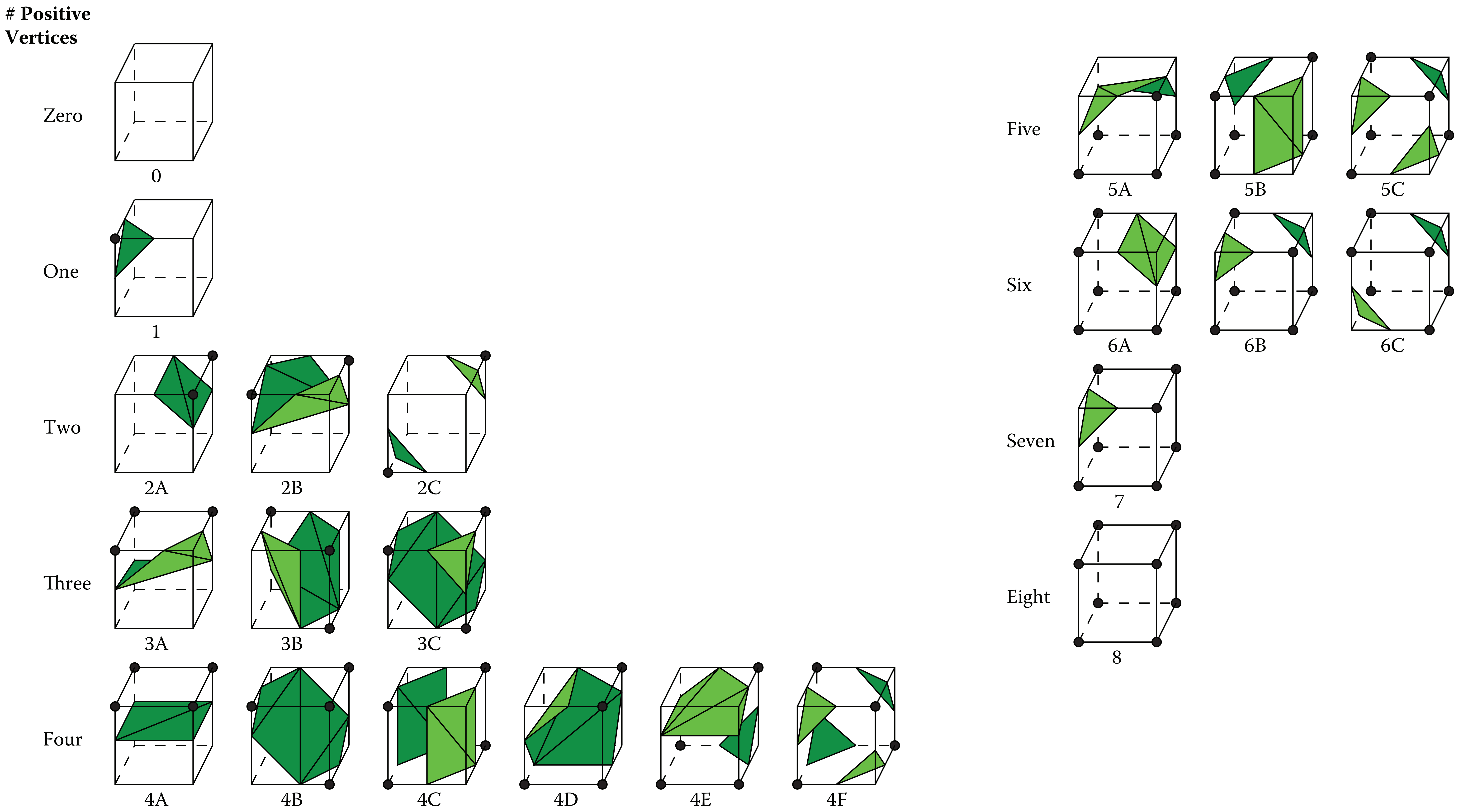
Marching Squares



[R. Wenger, 2013]

3D: Marching Cubes

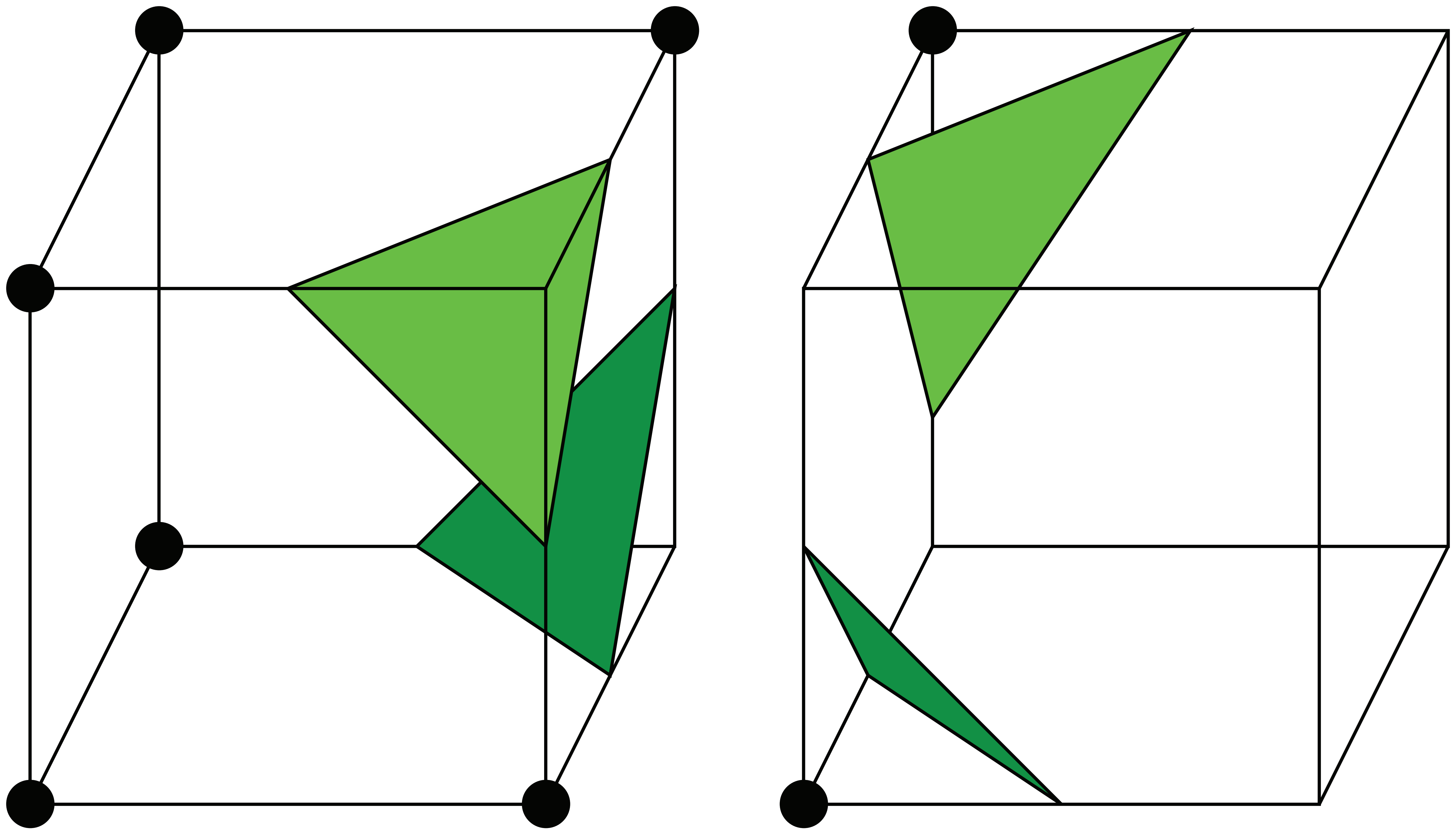
- Same idea, more cases [Lorensen and Cline, 1987]



[R. Wenger, 2013]

Incompatible Choices

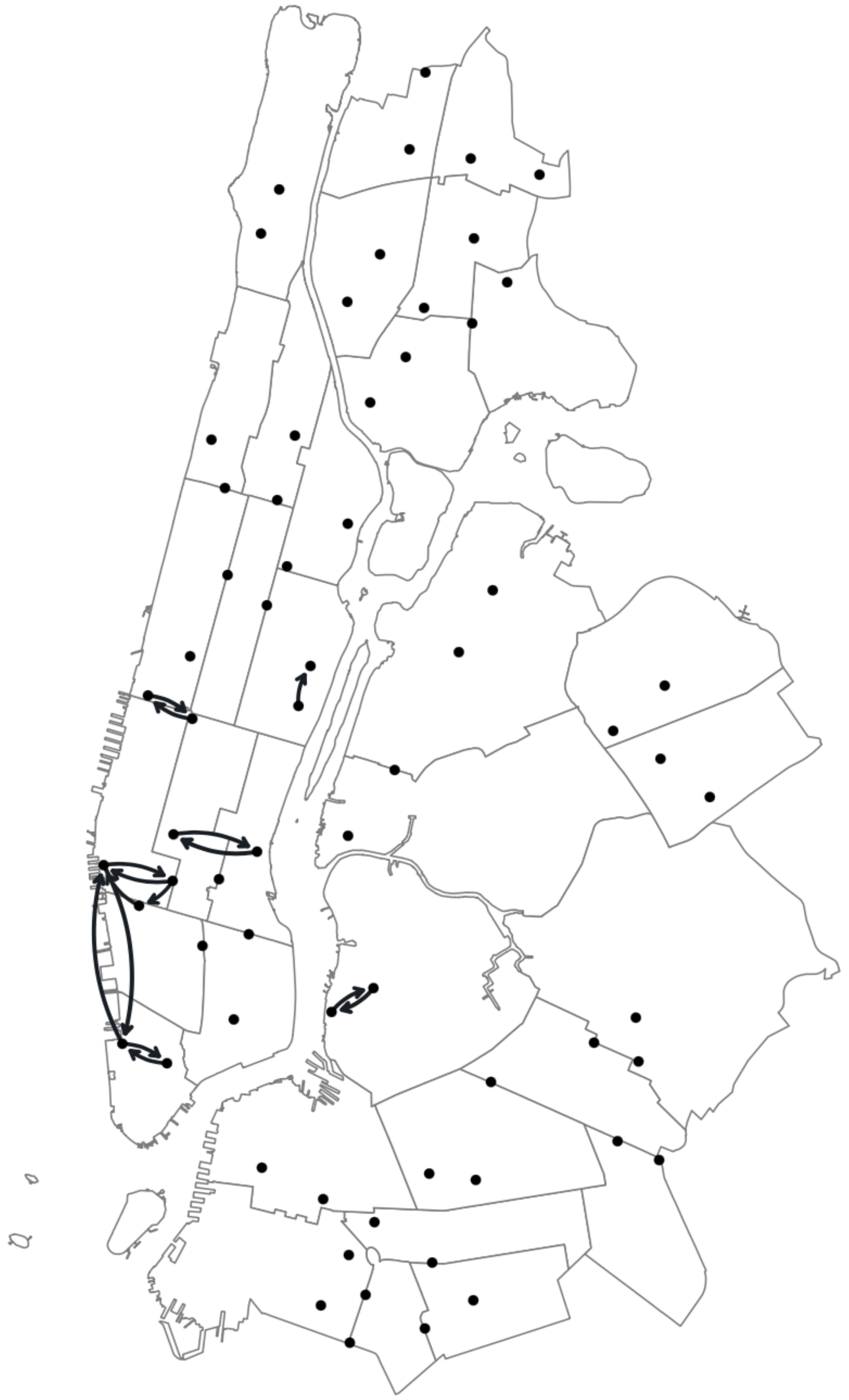
- If we have ambiguous cases where we choose differently for each cell, the surfaces will not match up correctly—there are holes
- Fix with the **asymptotic decider** [Nielson and Hamann,1991]



[R. Wenger, 2013]

Assignment 5

- Create Multiple Views
- Filtering
- Linked Highlighting
- Aggregation



Courselets

- Please provide feedback on the courselets if you have used them
- You can still work through them and complete them
- Extra credit for each completed survey

Final Project

- Designs feedback on Blackboard
- Work on implementations
- Presentations will be next week (April 28 and April 30)
- Submit information to Blackboard later this week
  - Project or link to project
  - Preference for Monday or Wednesday presentation
- Reports due at the end of the class

Final Exam

- Wednesday, May 7, 2025, **8:00-9:50pm**
- Covers all topics but emphasizes second half of the course
- Similar format as Midterm (multiple choice, free response)
- 627 Students will have a extra questions related to the research papers

Volume Rendering

Volume Rendering vs. Isosurfacing

(a) Direct volume rendered

(b) Isosurface rendered

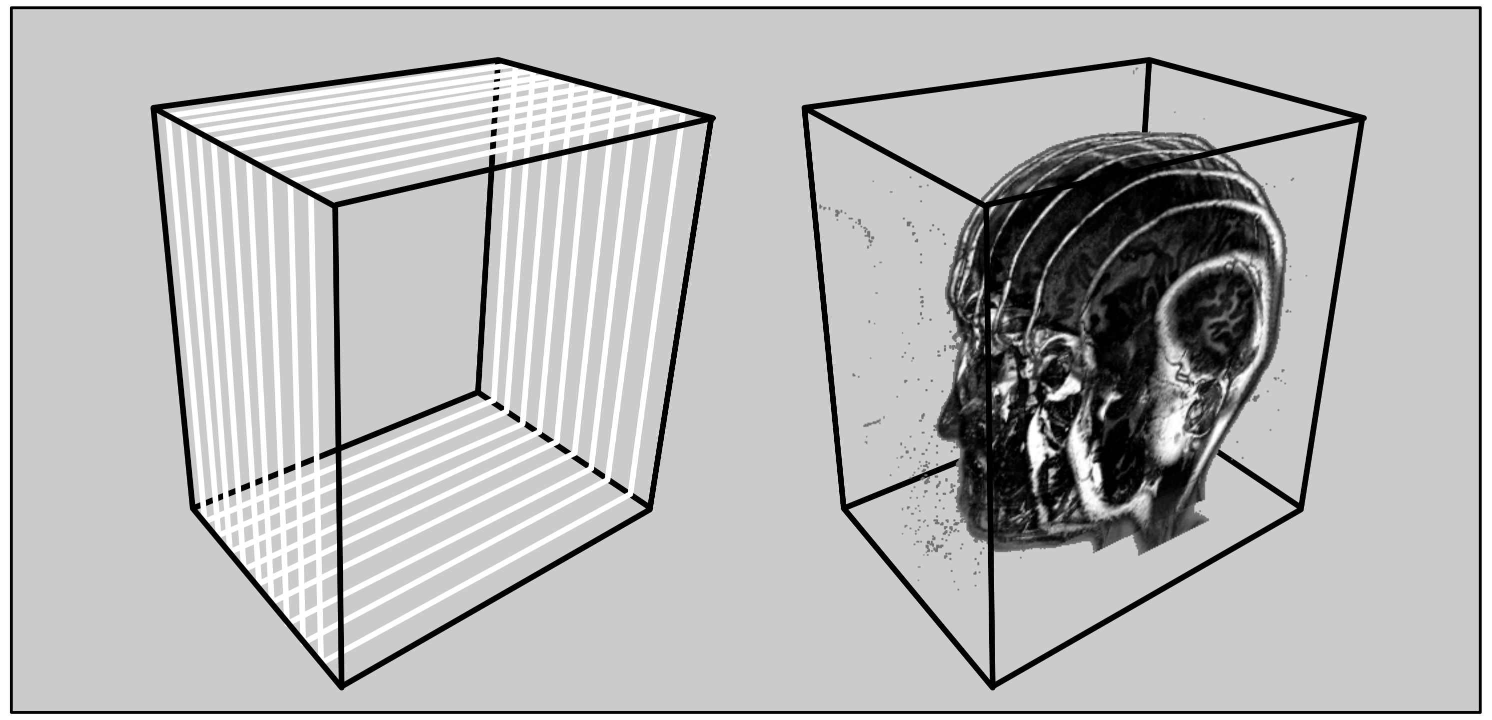
[Kindlmann, 1998]

(Direct) Volume Rendering

- Isosurfacing: compute a surface (triangles) and use standard computer graphics to render the triangles
- Volume rendering: compute the pixels shown directly from the volume information
- Why?
  - No need to figure out precise isosurface boundaries
  - Can work better for data with noise or uncertainty
  - Greater control over appearance based on values

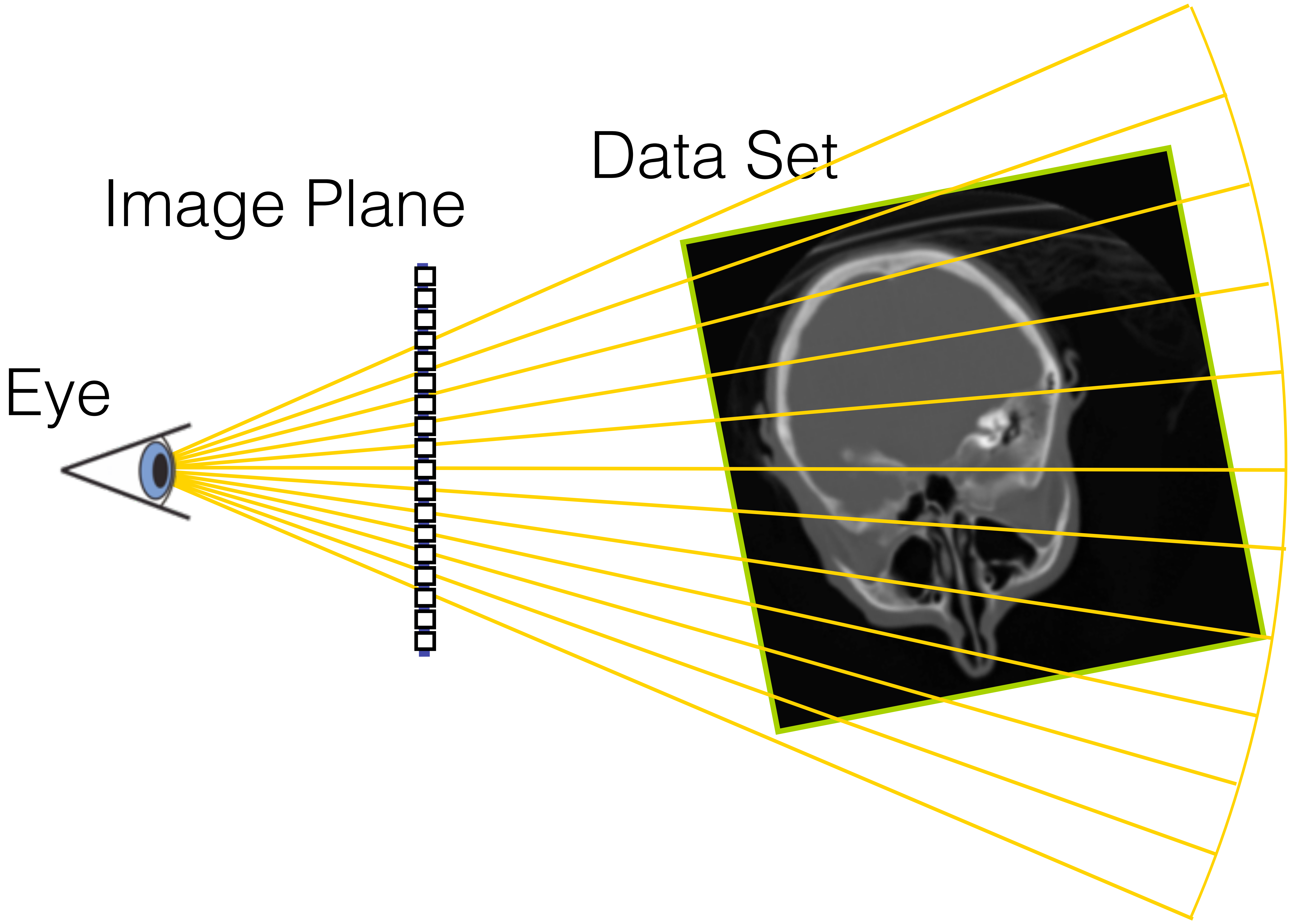
Types of Volume Rendering Algorithms

- Ray casting
  - Similar to ray tracing, but use rays from the viewer
- Splatting:
  - Object-order, voxels splat onto the image plane
- Shear Warp:
  - Object-space, slice-based, parallel viewing rays
- Texture-Based:
  - 2D Slices: stack of texture maps
  - 3D Textures



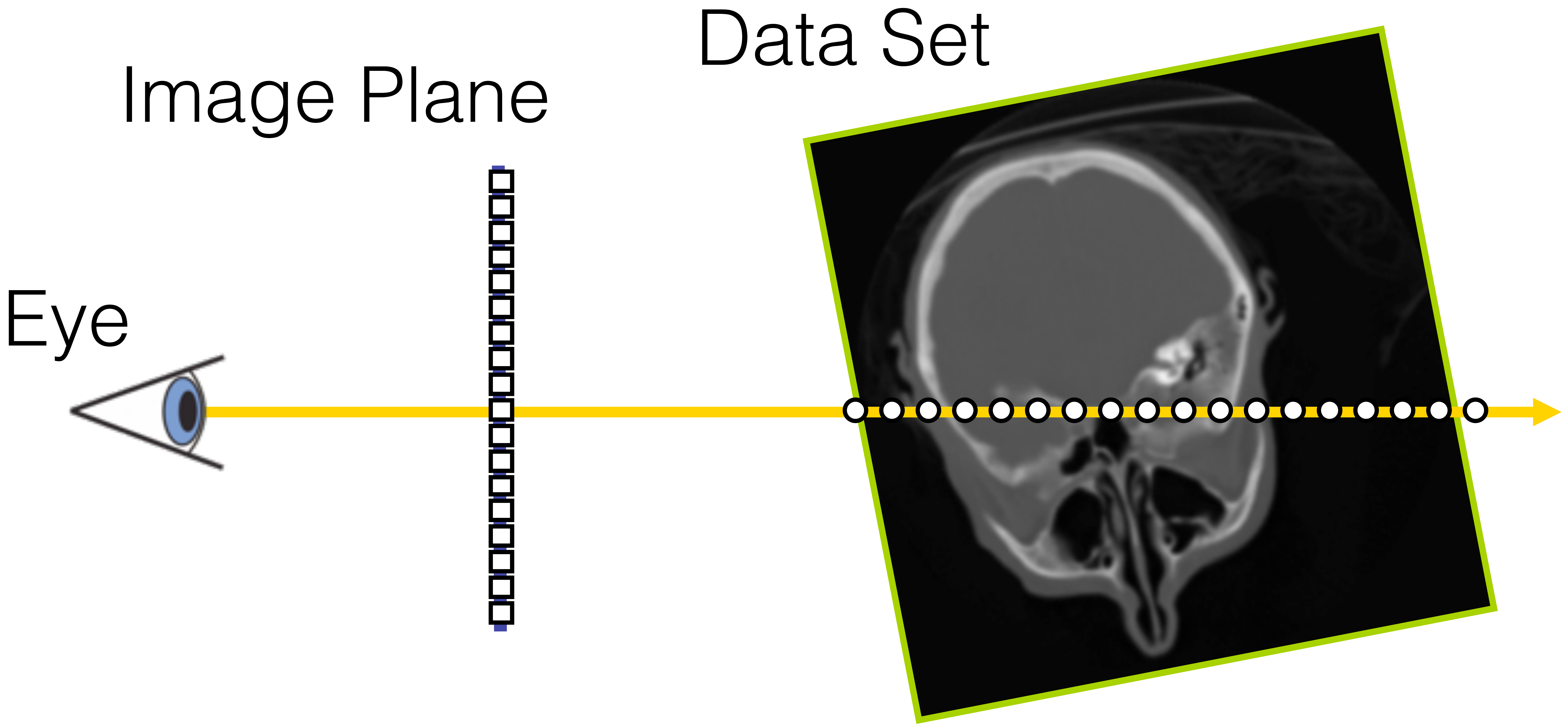
[via Möller]

Volume Ray Casting



[Levine]

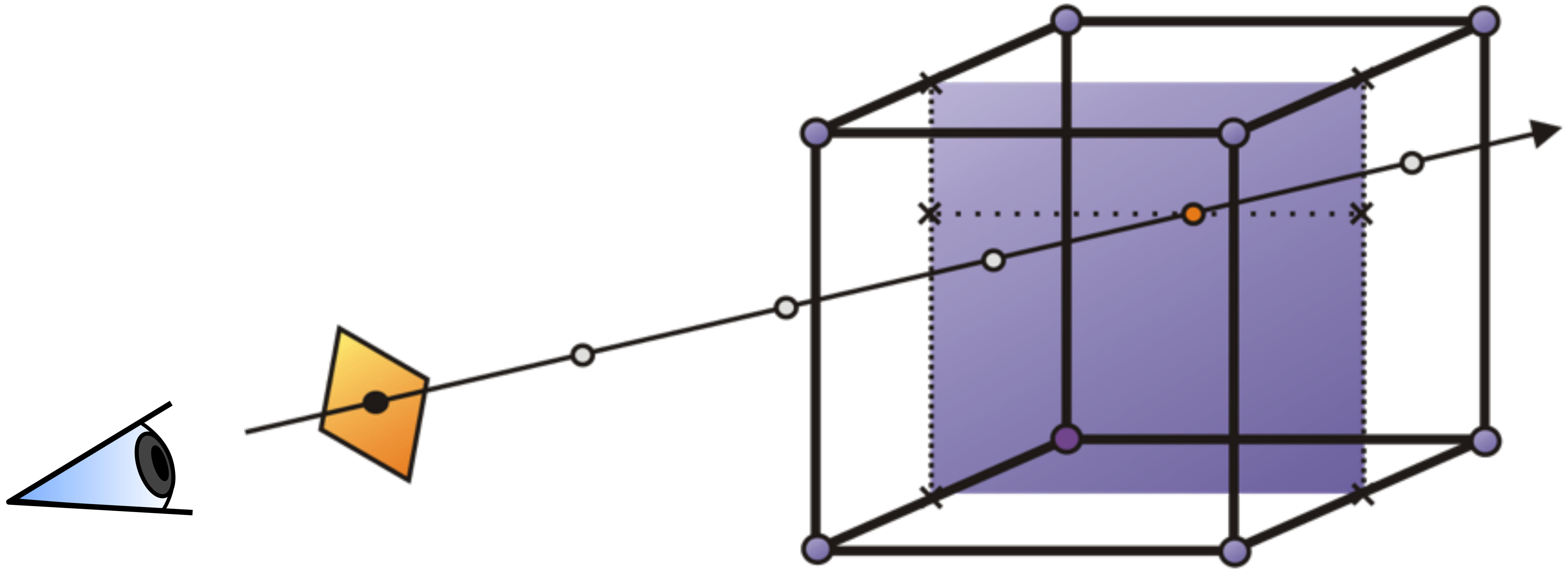
Volume Ray Casting



[Levine]

How?

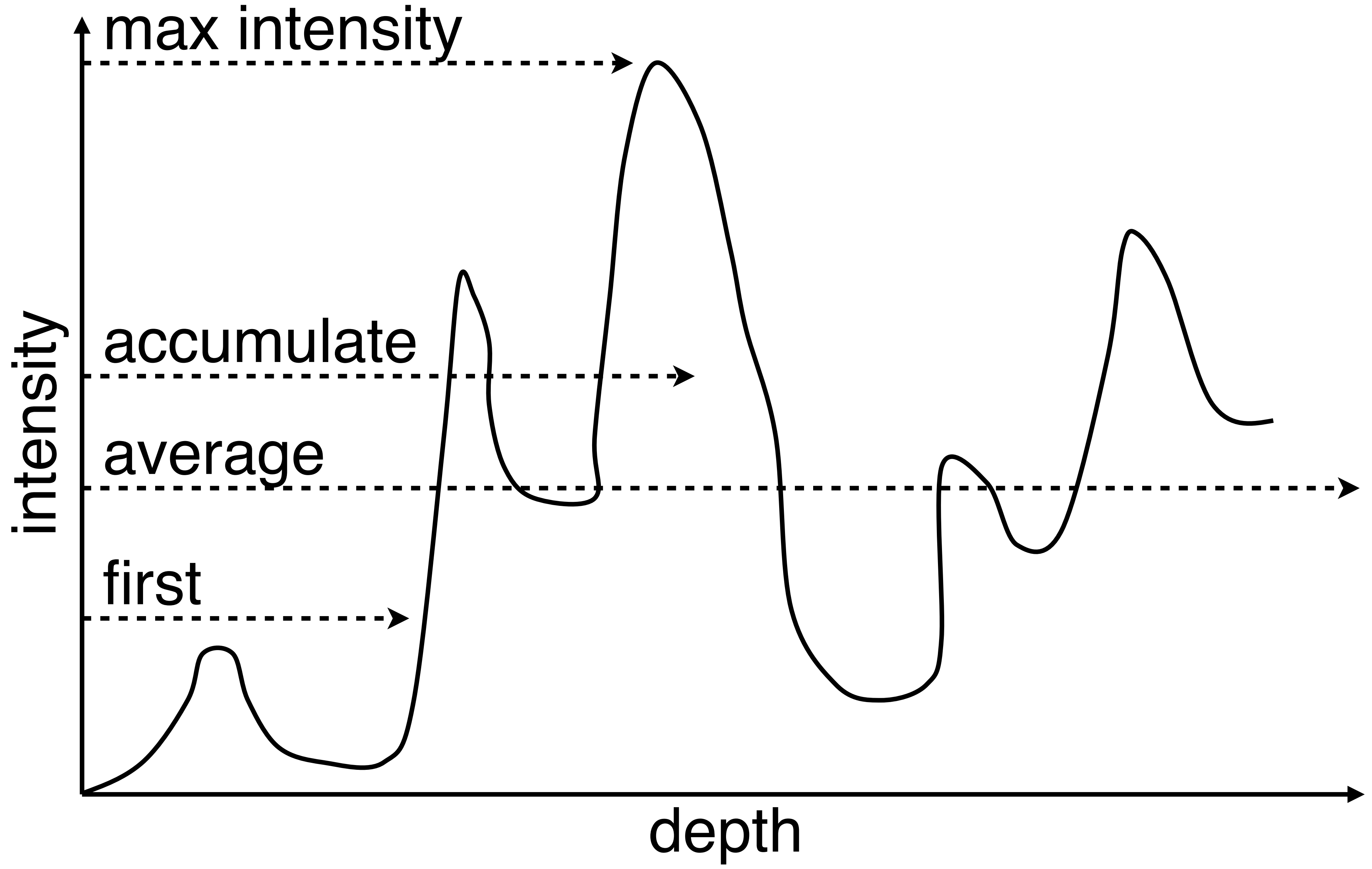
- Approximate volume rendering integral: light absorption & emission
- Sample at regular intervals along each ray
- Trilinear interpolation: linear interpolation along each axes (x,y,z)



- Not the only possibility, also "object order" techniques like splatting or texture-based and combinations like shear-warp

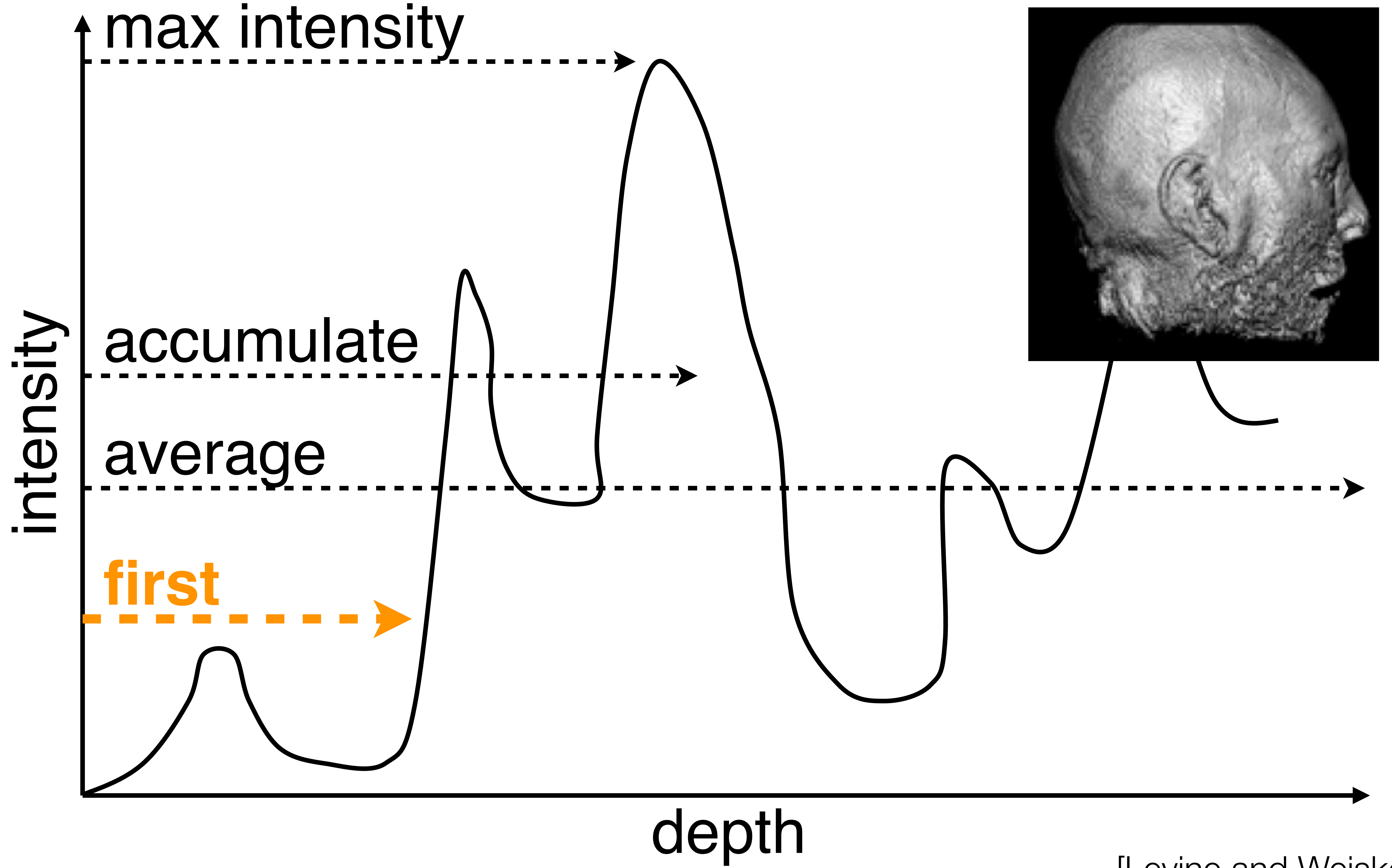
Compositing

- Need **one pixel** from all values along the ray
- Q: How do we "add up" all of those values along the ray?
- A: Compositing!
- Different types of compositing
  - First: like isosurfacing, first intersection at a certain intensity
  - Max intensity: choose highest val
  - Average: mean intensity (density, like x-rays)
  - Accumulate: each voxel has some contribution



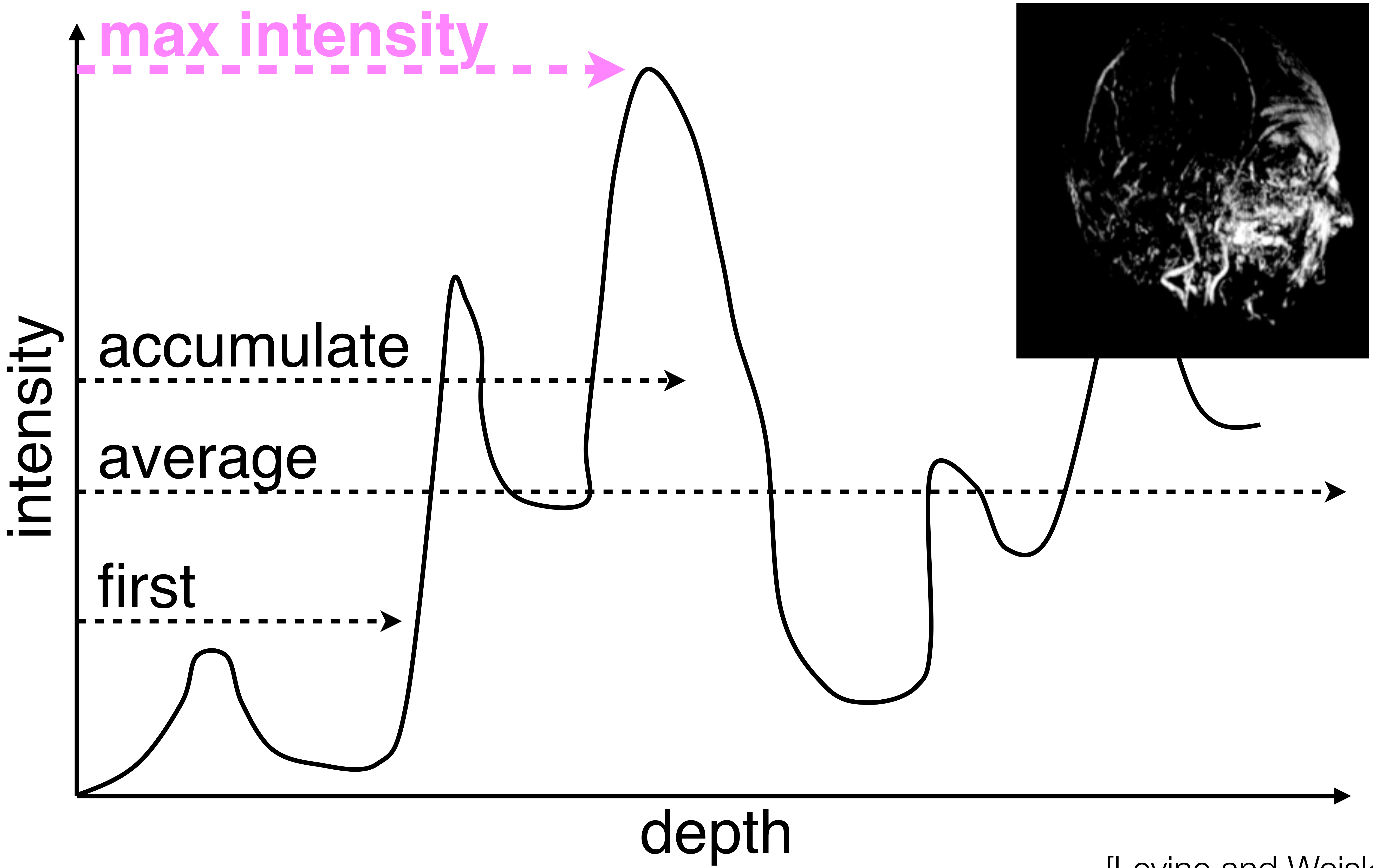
[Levine and Weiskopf/Machiraju/Möller]

Types of Compositing



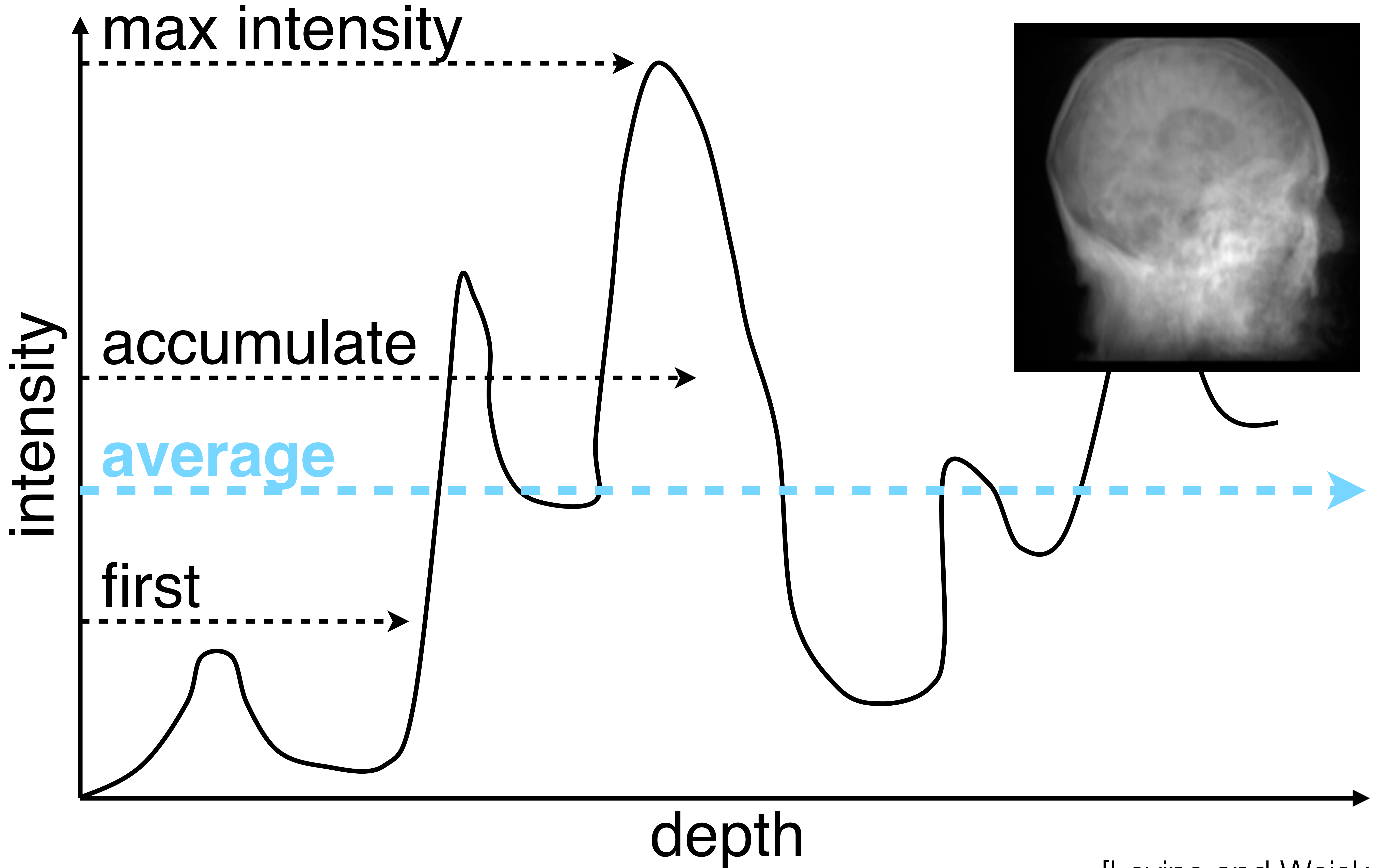
[Levine and Weiskopf/Machiraju/Möller]

Types of Compositing



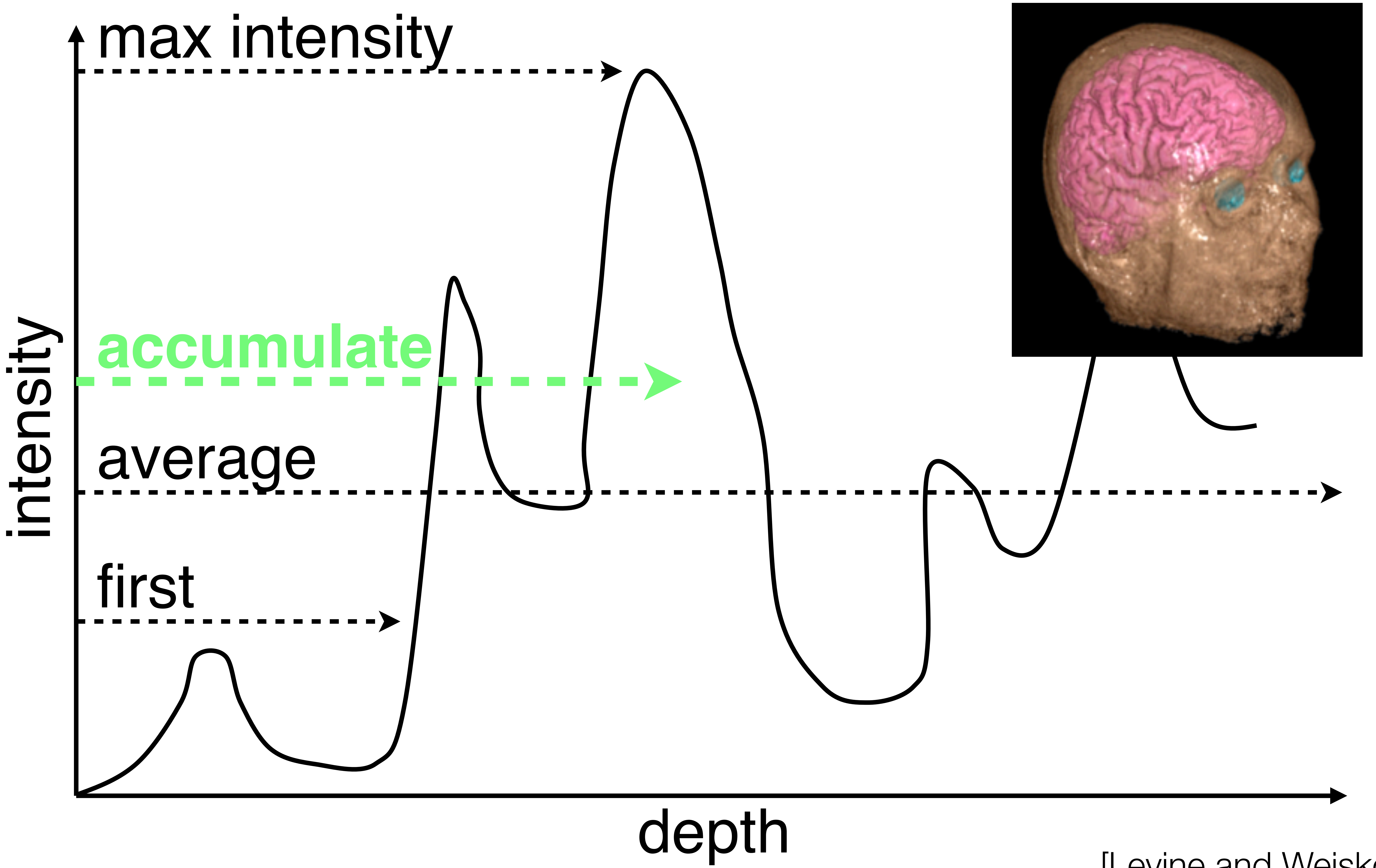
[Levine and Weiskopf/Machiraju/Möller]

Types of Compositing



[Levine and Weiskopf/Machiraju/Möller]

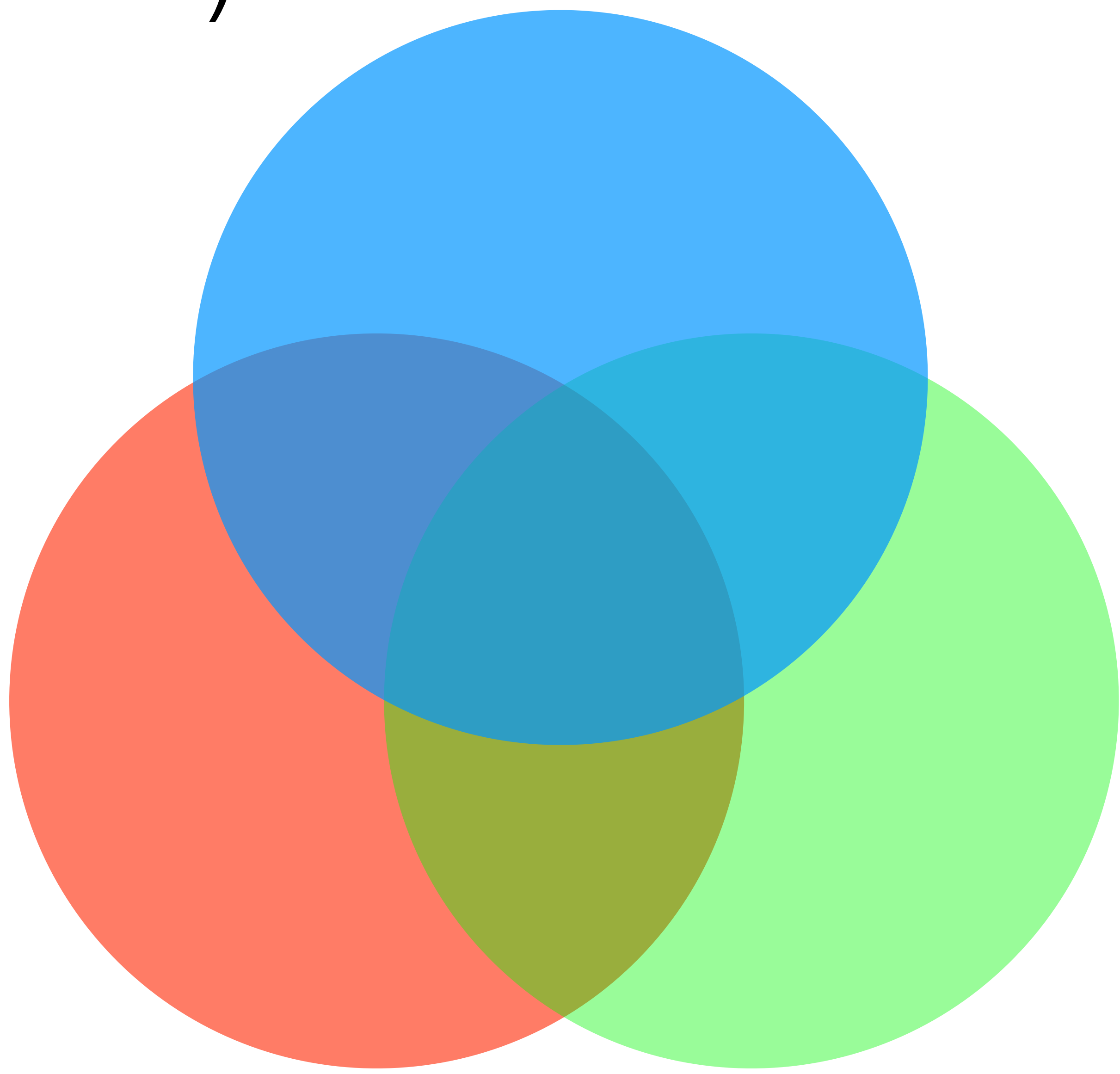
Types of Compositing



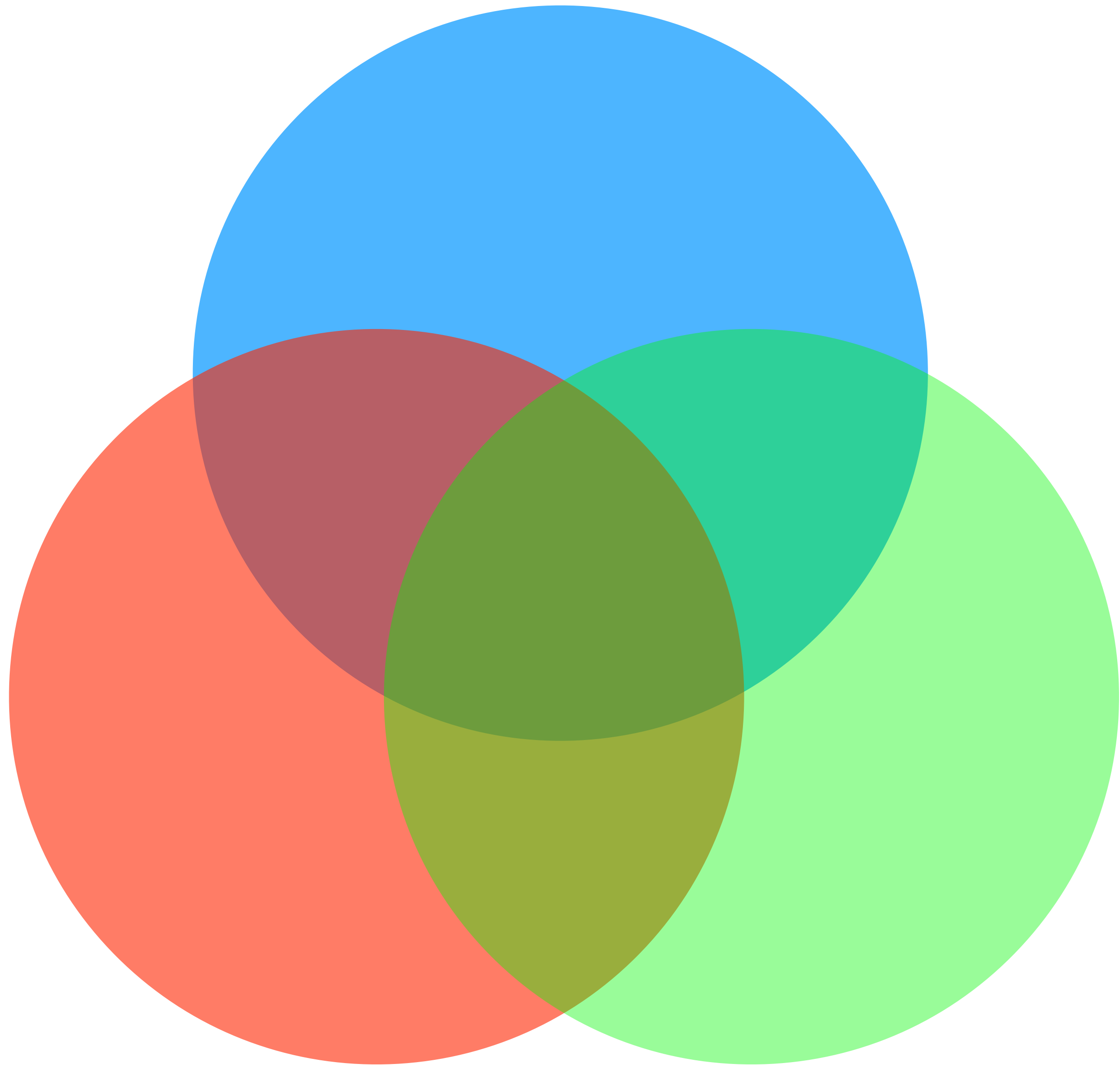
[Levine and Weiskopf/Machiraju/Möller]

Accumulation

- If we're not just calculating a single number (max, average) or a position (first), how do we determine the accumulation?
- Assume each value has an associated color (c) and opacity (α)
- Over operator (back-to-front):
  - $c = \alpha_f \cdot c_f + (1\text{-}\alpha_f) \cdot \alpha_b \cdot c_b$
  - $\alpha = \alpha_f + (1\text{-}\alpha_f) \cdot \alpha_b$
- Order is important!



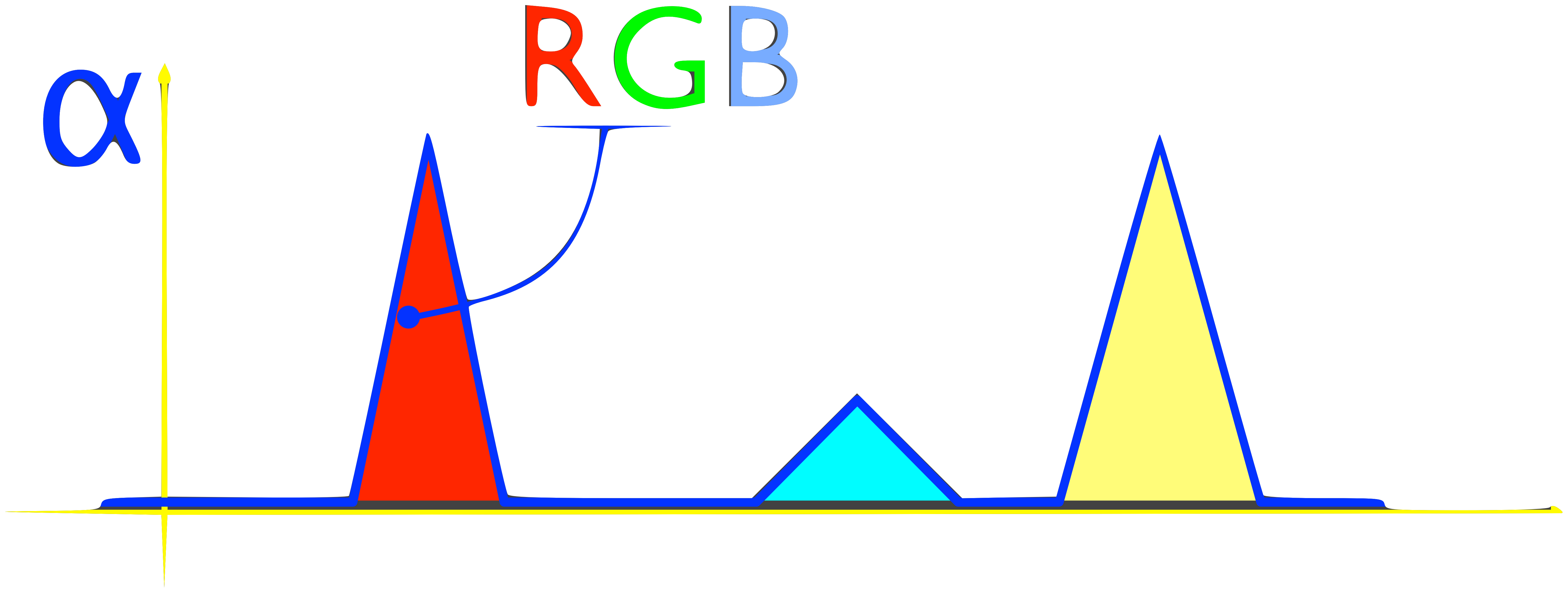
Blue Last



Blue First

Transfer Functions

- Where do the colors and opacities come from?
- Idea is that each voxel emits/absorbs light based on its scalar value
- …but users get to choose how that happens
- x-axis: color region definitions, y-axis: opacity



[Kindlmann]

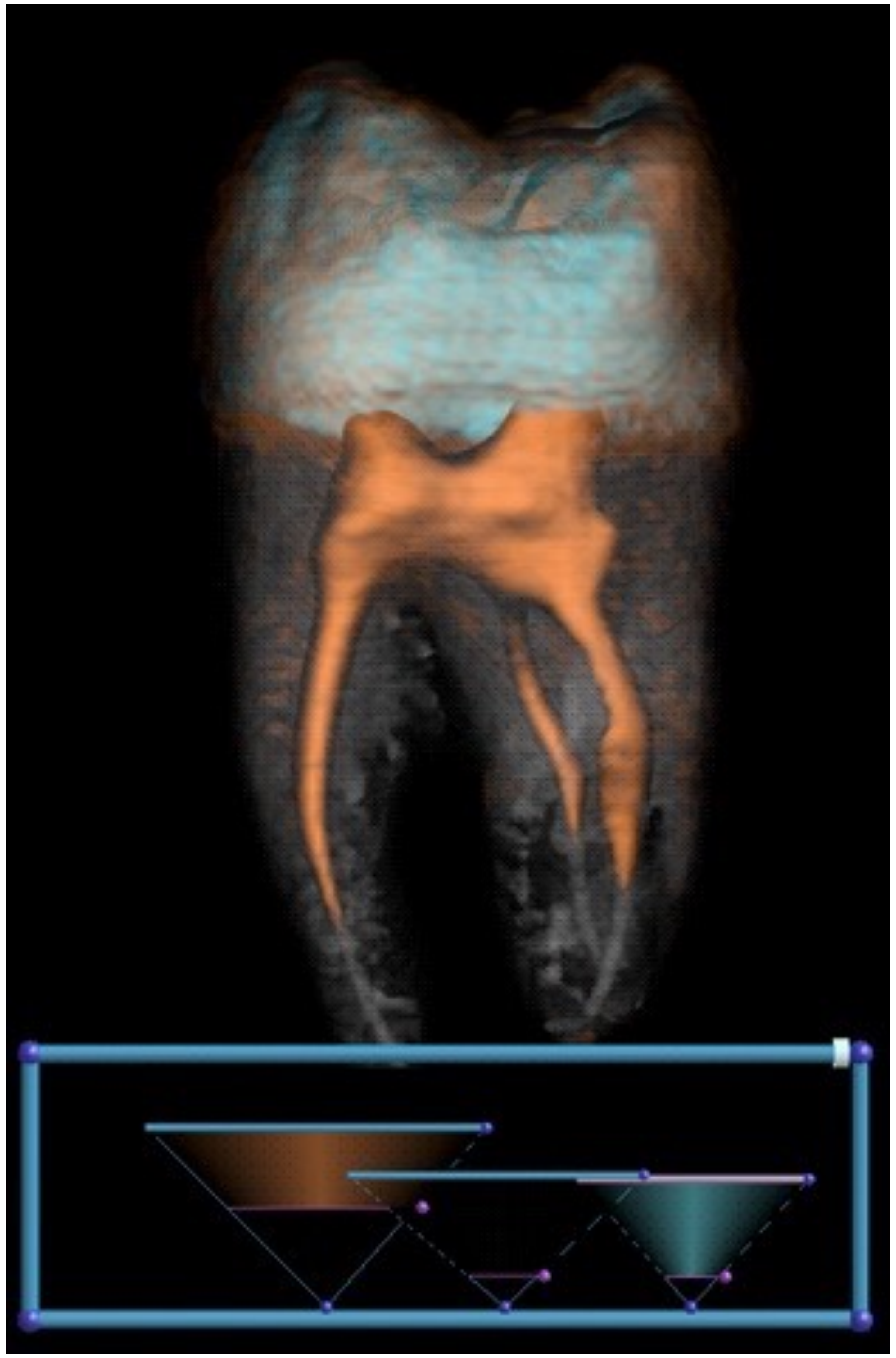
Transfer Function Design

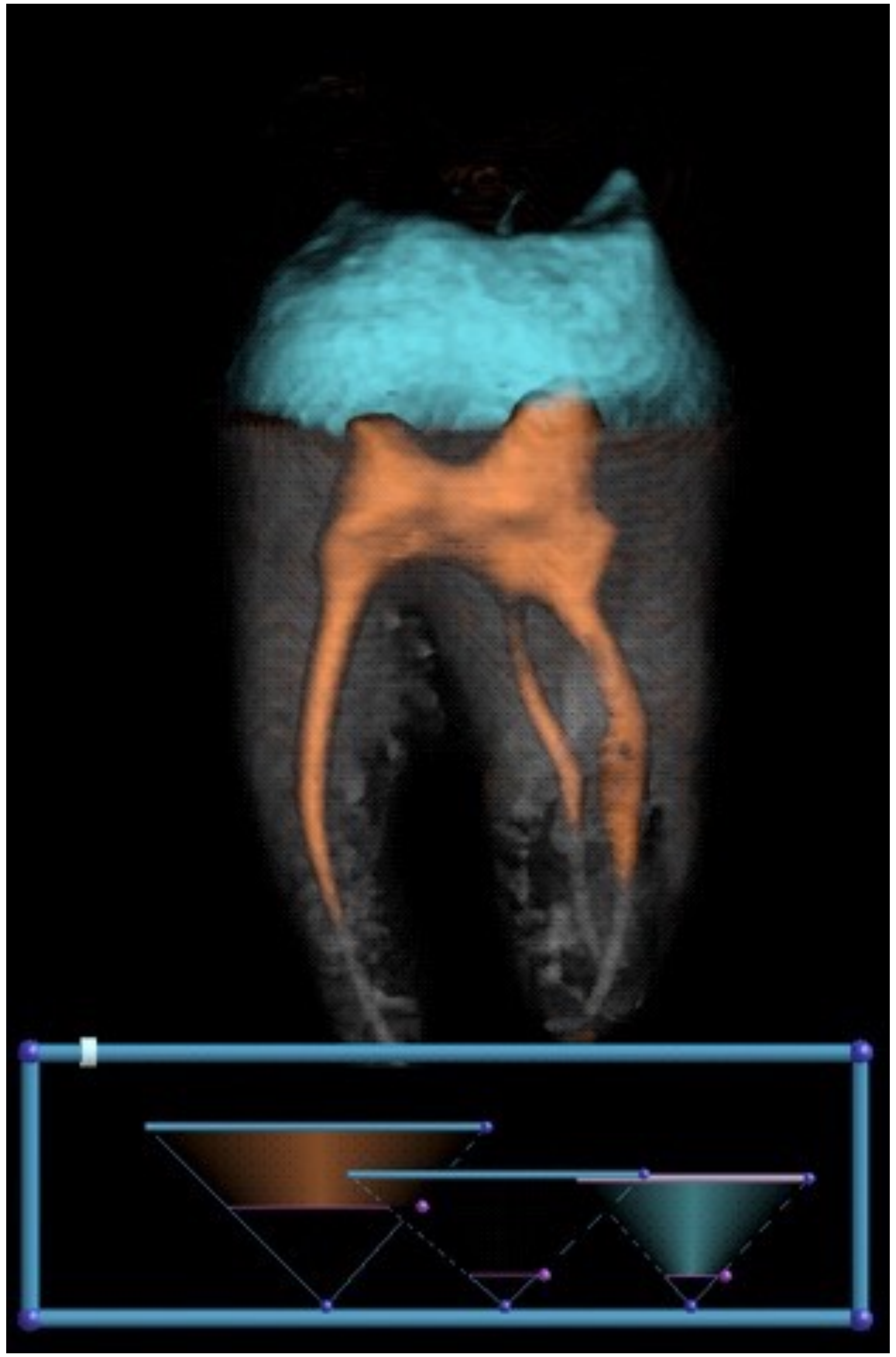
- Transfer function **design** is non-trivial!
- Lots of tools to help visualization designers to create good transfer functions
- Histograms, more attributes than just value like gradient magnitude

Multidimensional Transfer Functions

[J. Kniss]

Multidimensional Transfer Functions





[J. Kniss]

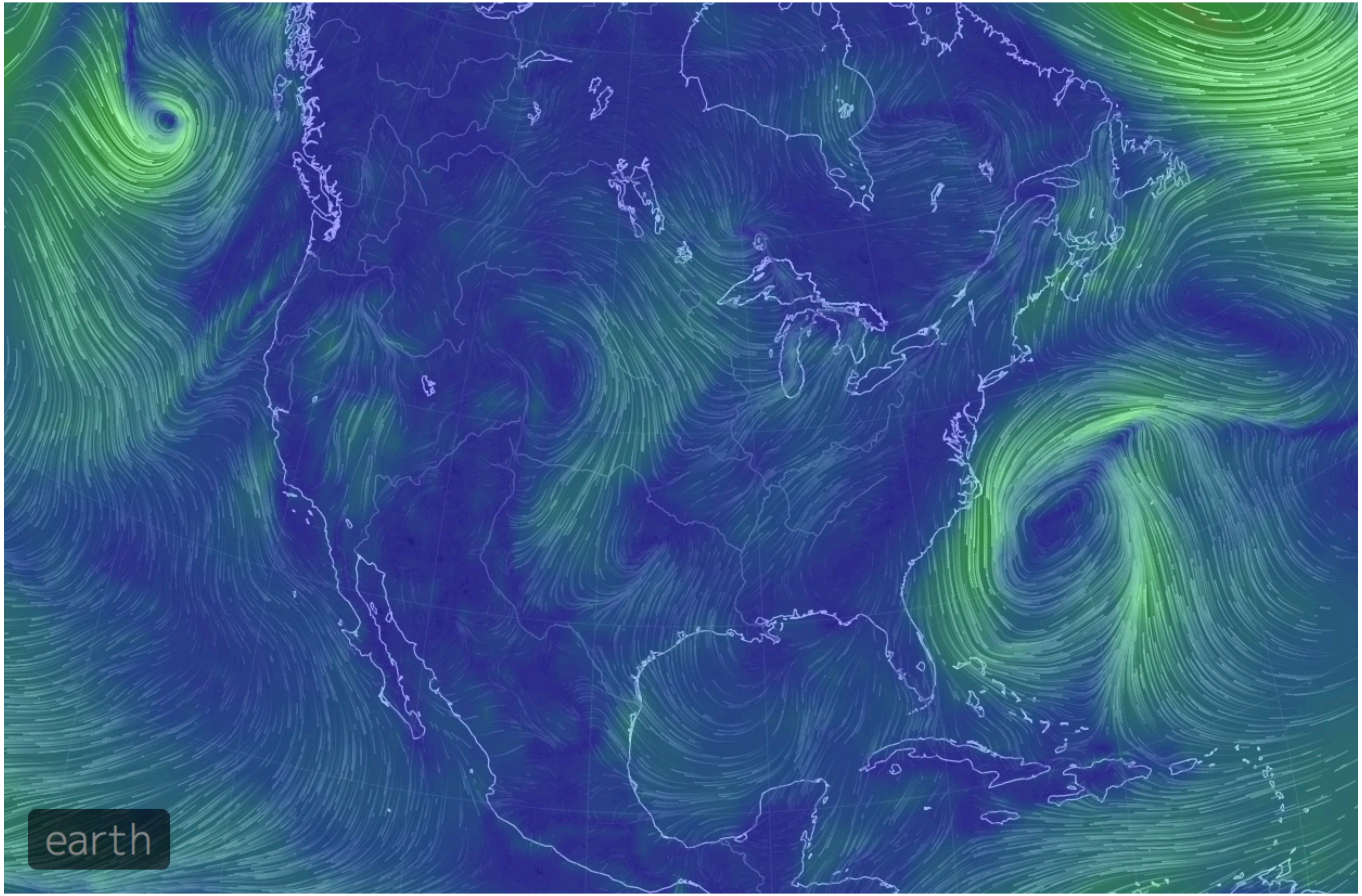
Newer Technology

- Intel OSPRay
- https://www.ospray.org/gallery.html

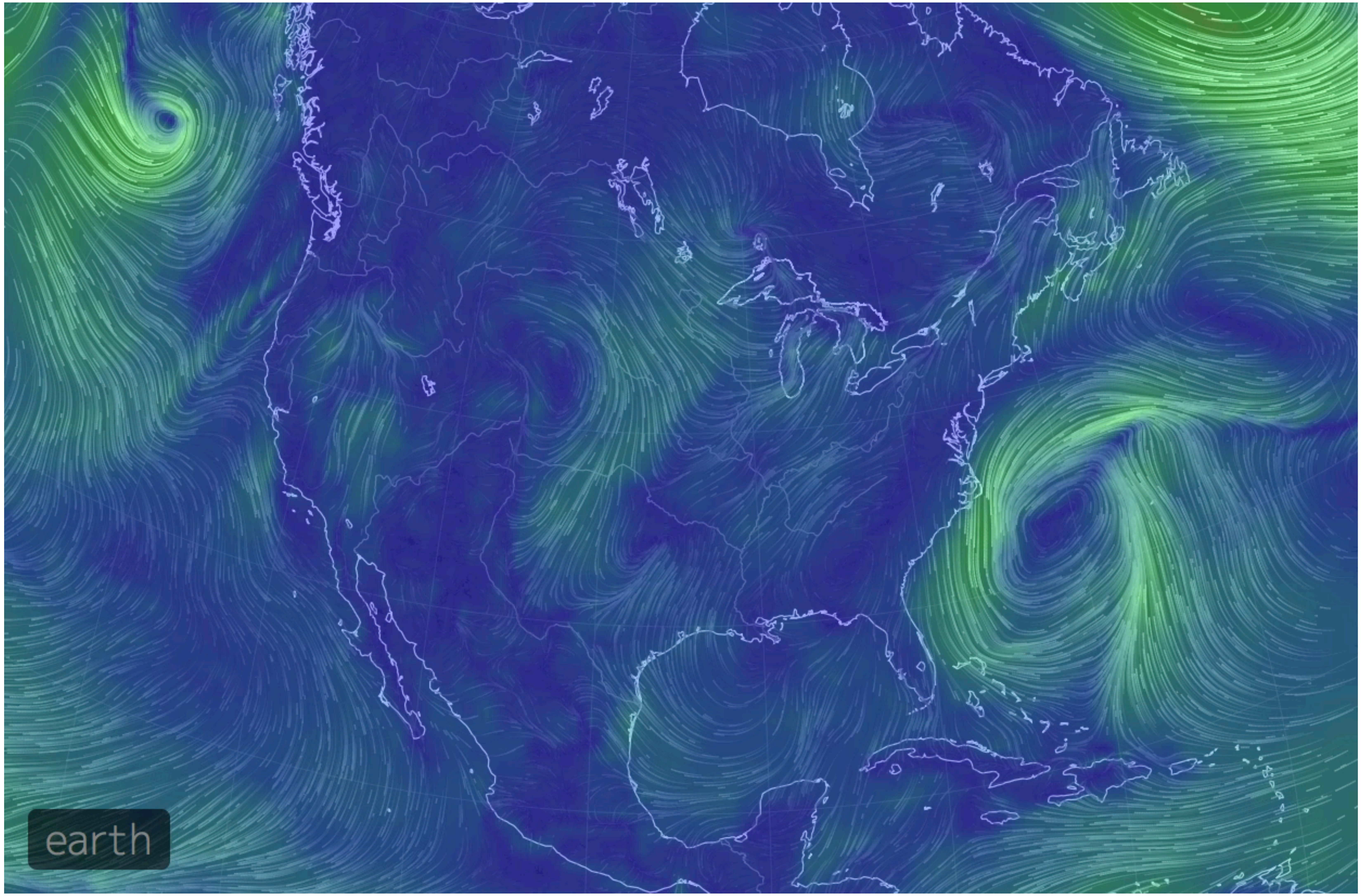
ParaView Examples

Vector Field Visualization

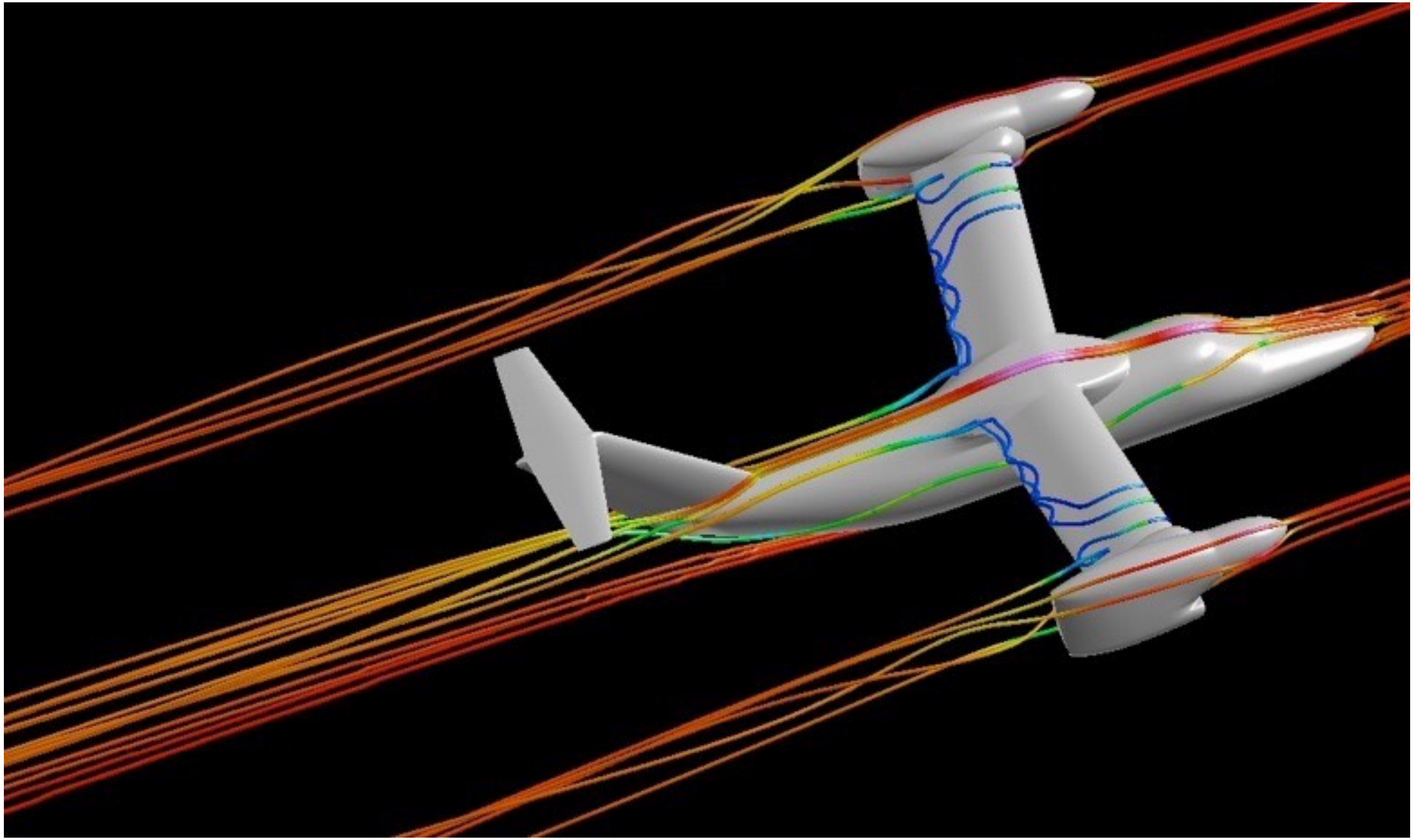
Examples of Vector Fields



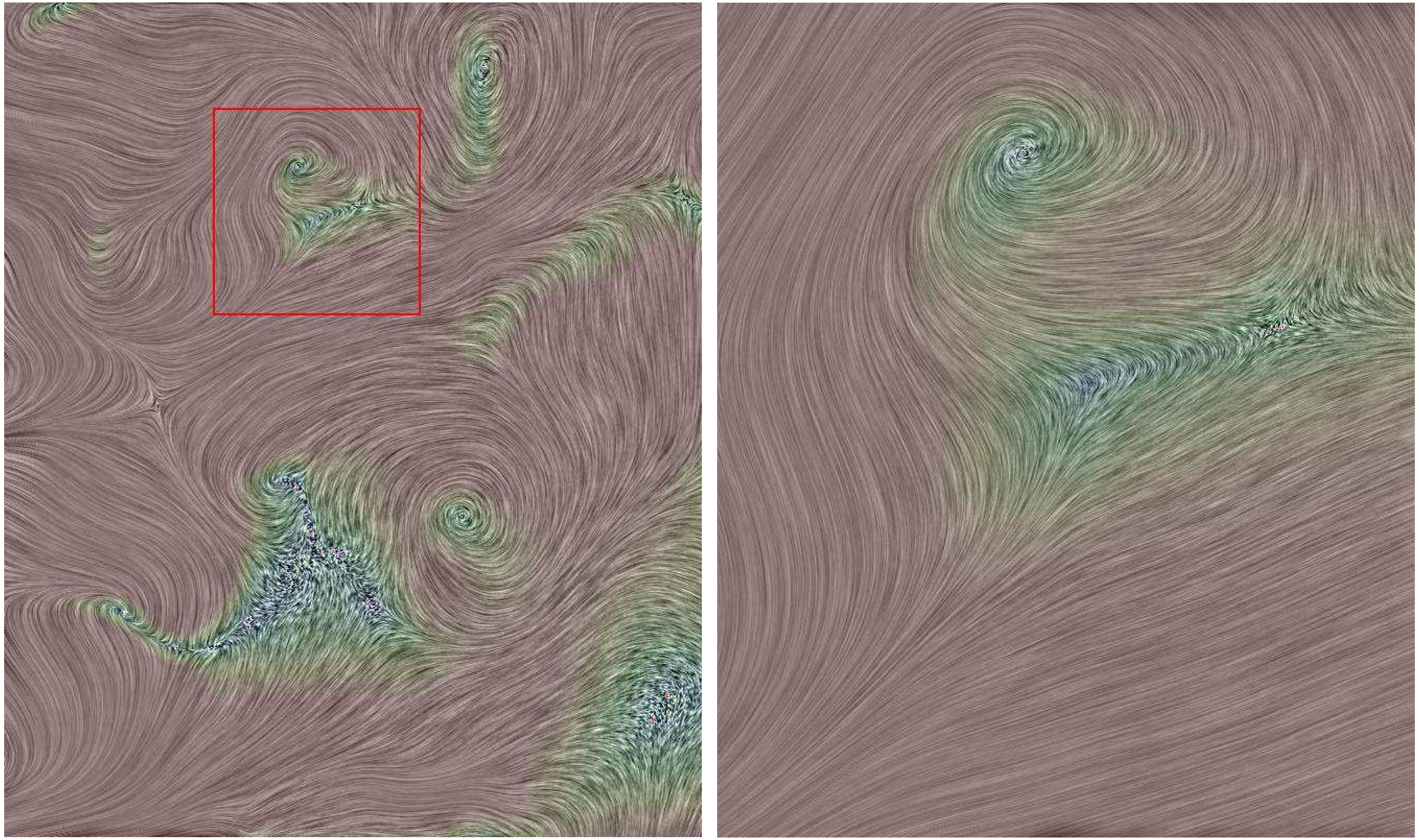
Examples of Vector Fields



Examples of Vector Fields

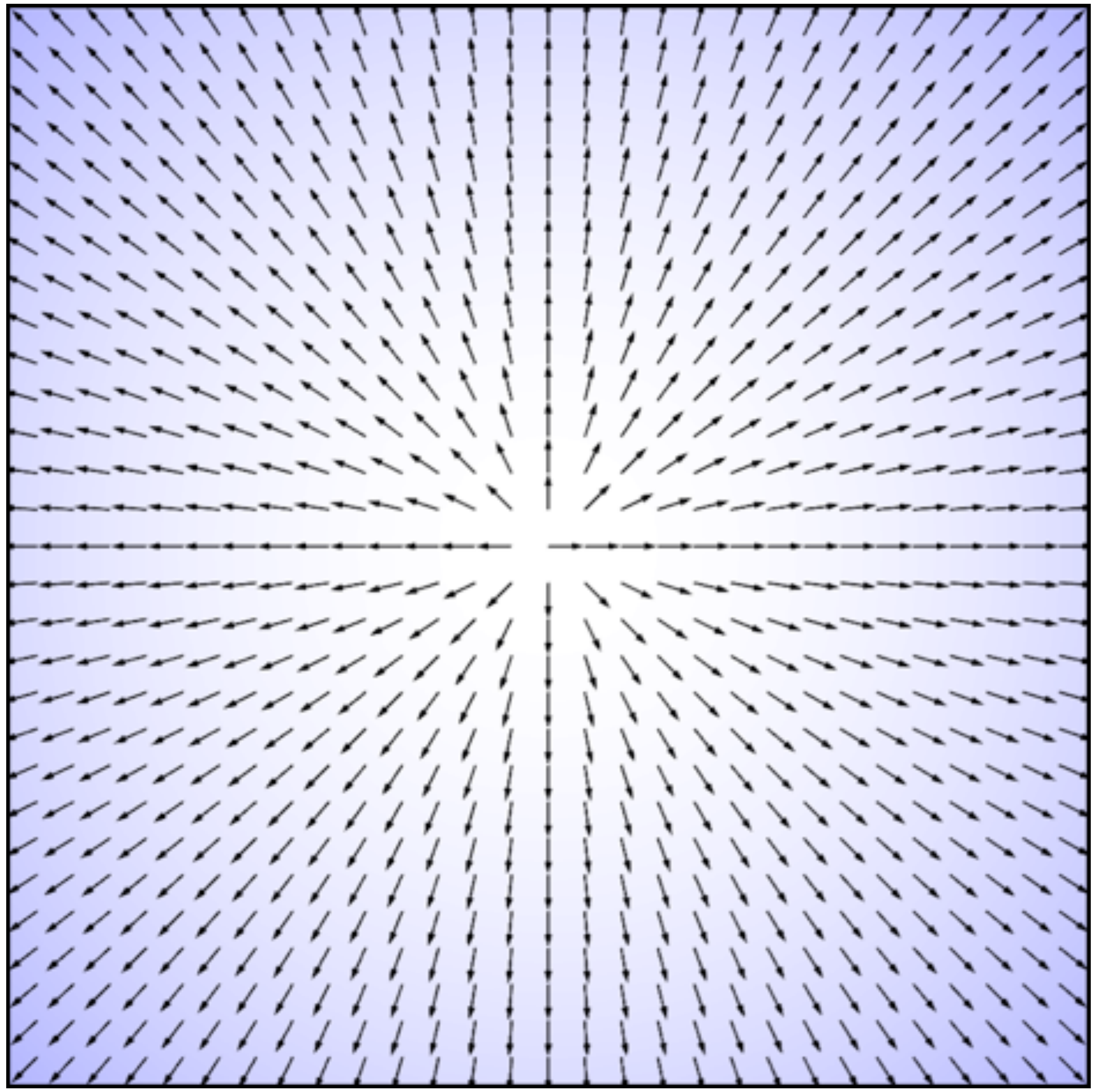


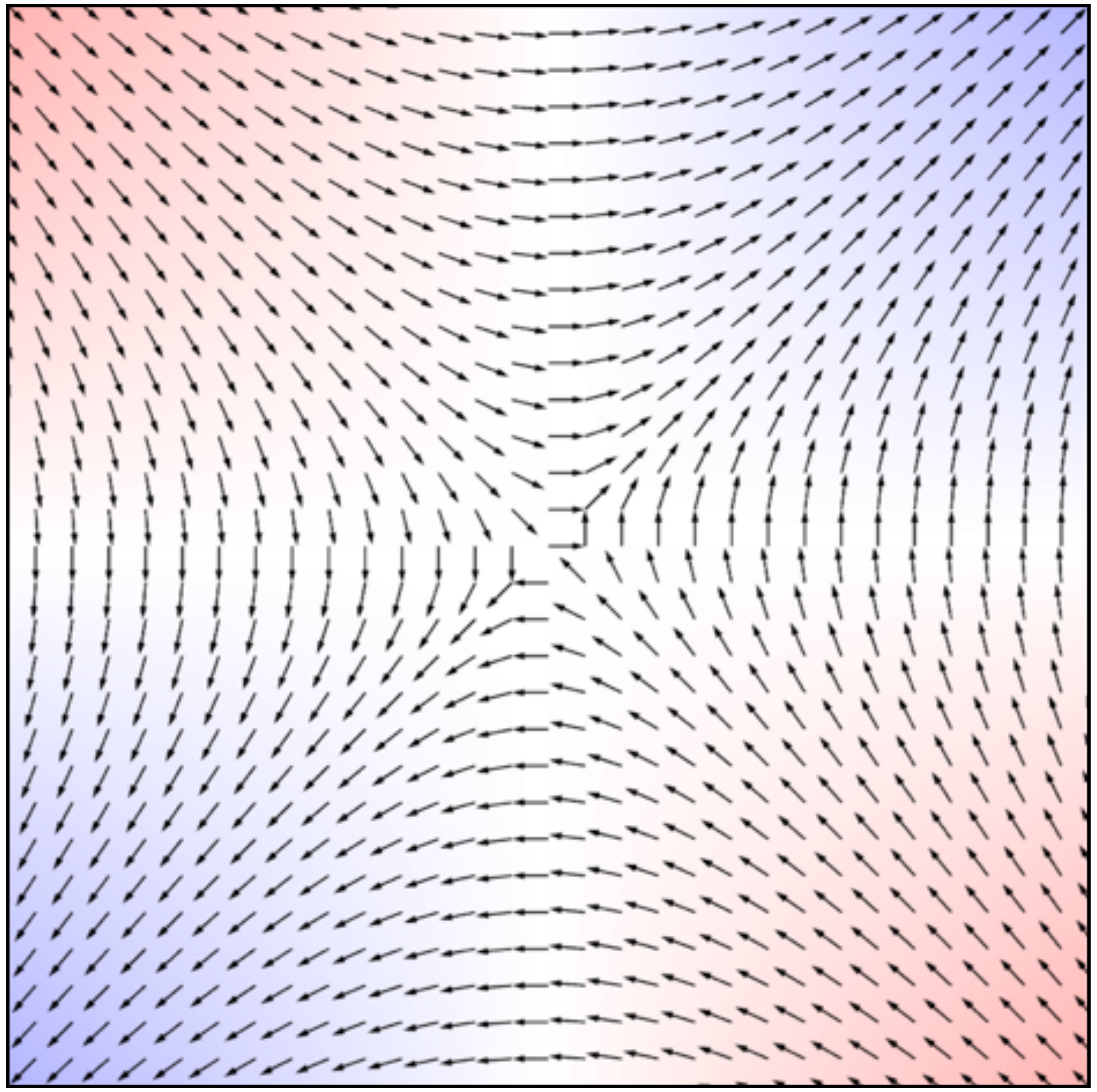
Examples of Vector Fields



Earthquake Ground Surface Movement [H. Yu et. al., SC2004]

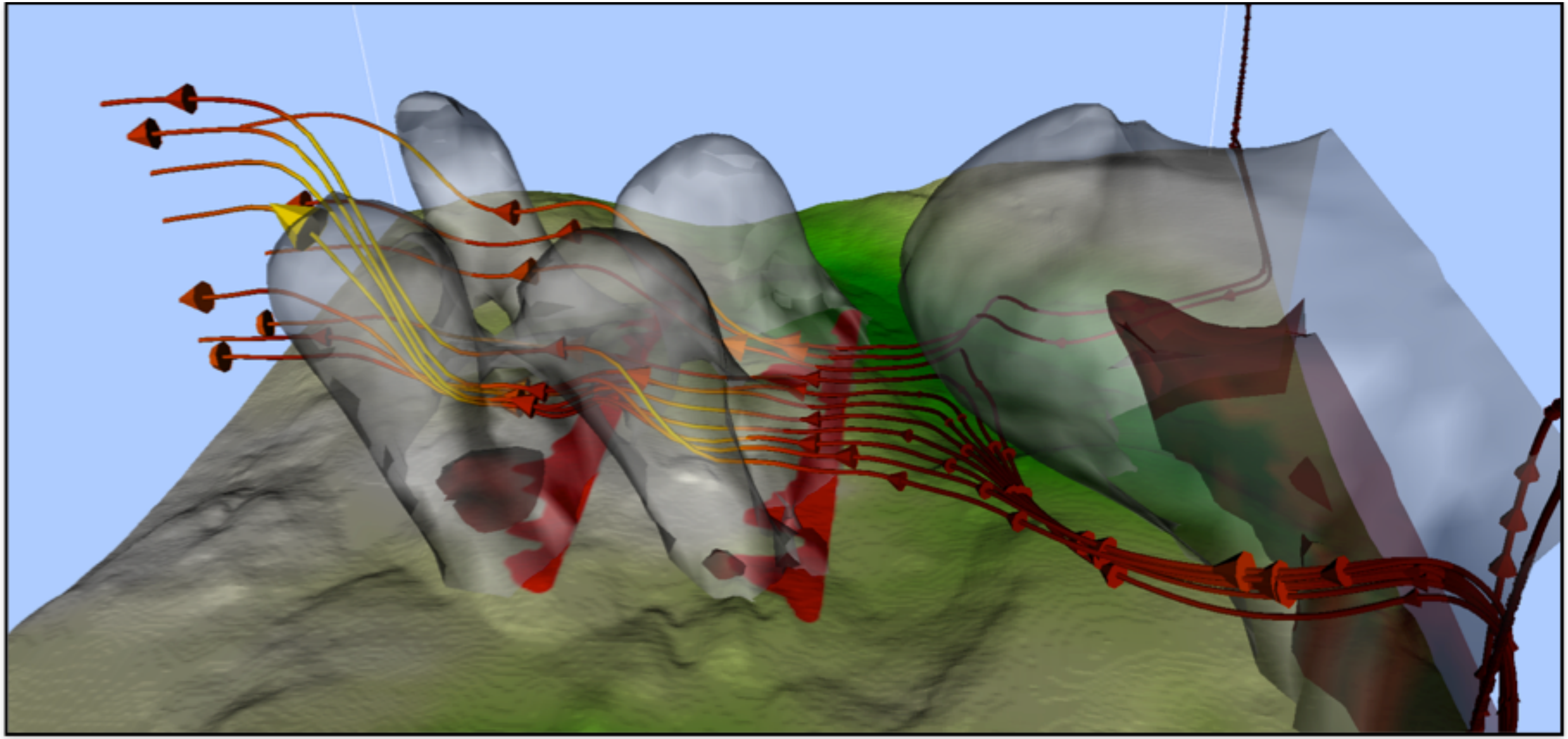
Examples of Vector Fields





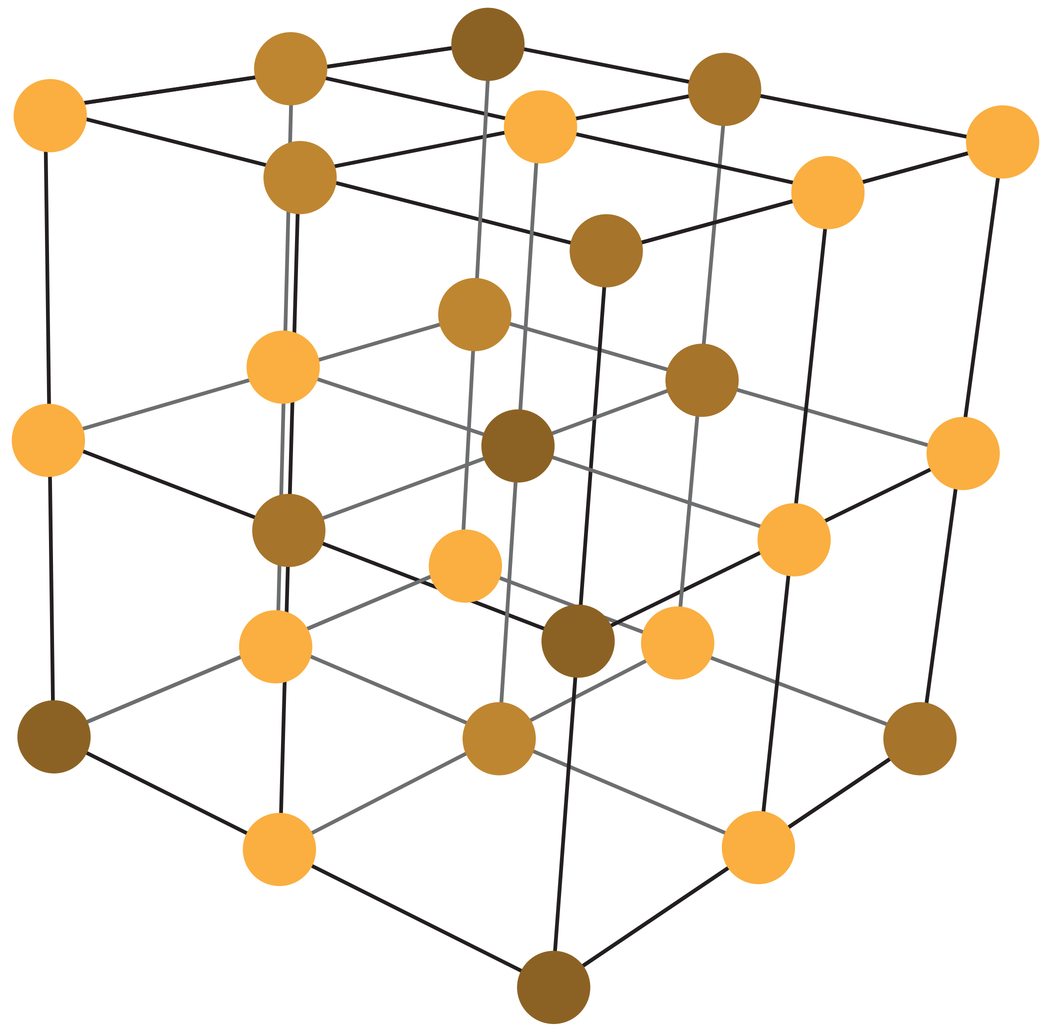
Gradient Vector Fields

Examples of Vector Fields



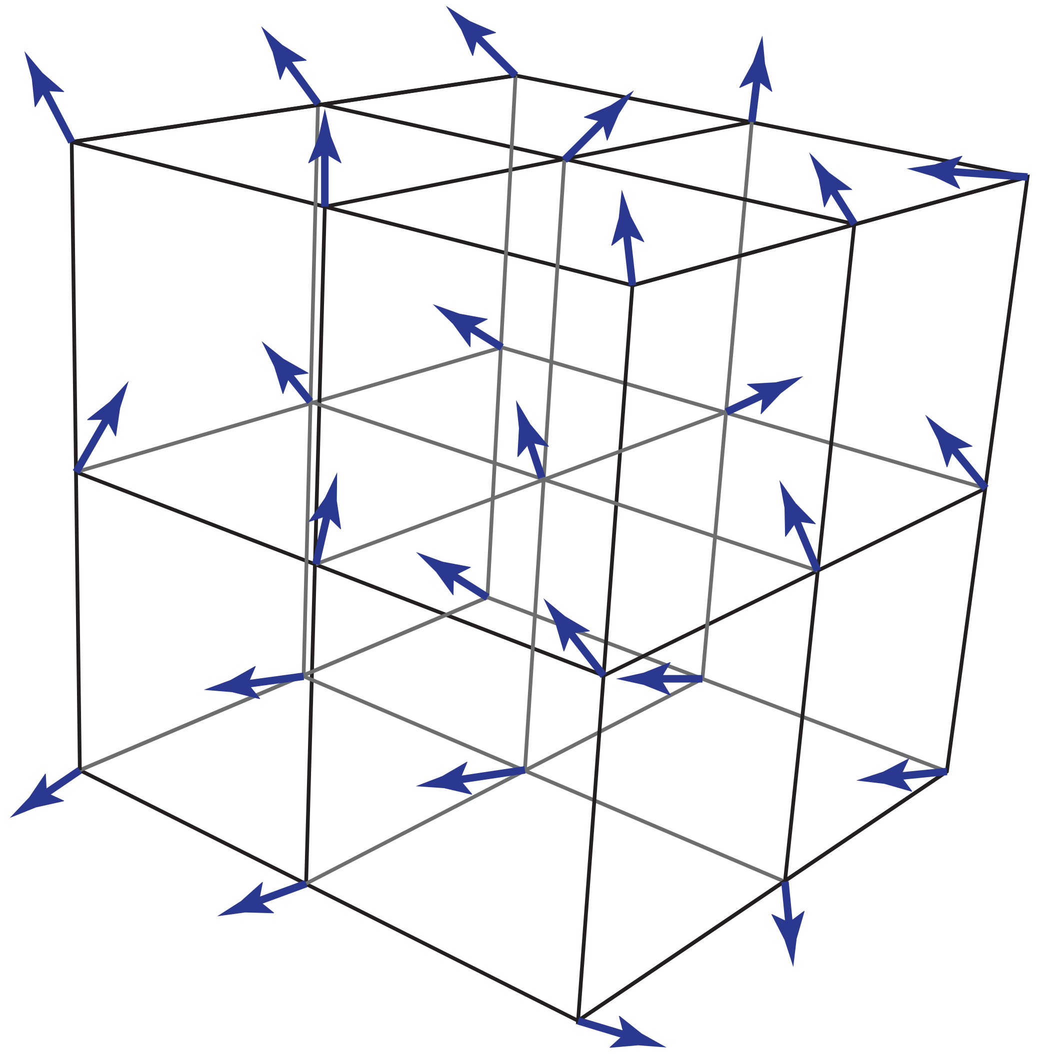
Wildfire Modeling [E. Anderson]

Fields in Visualization



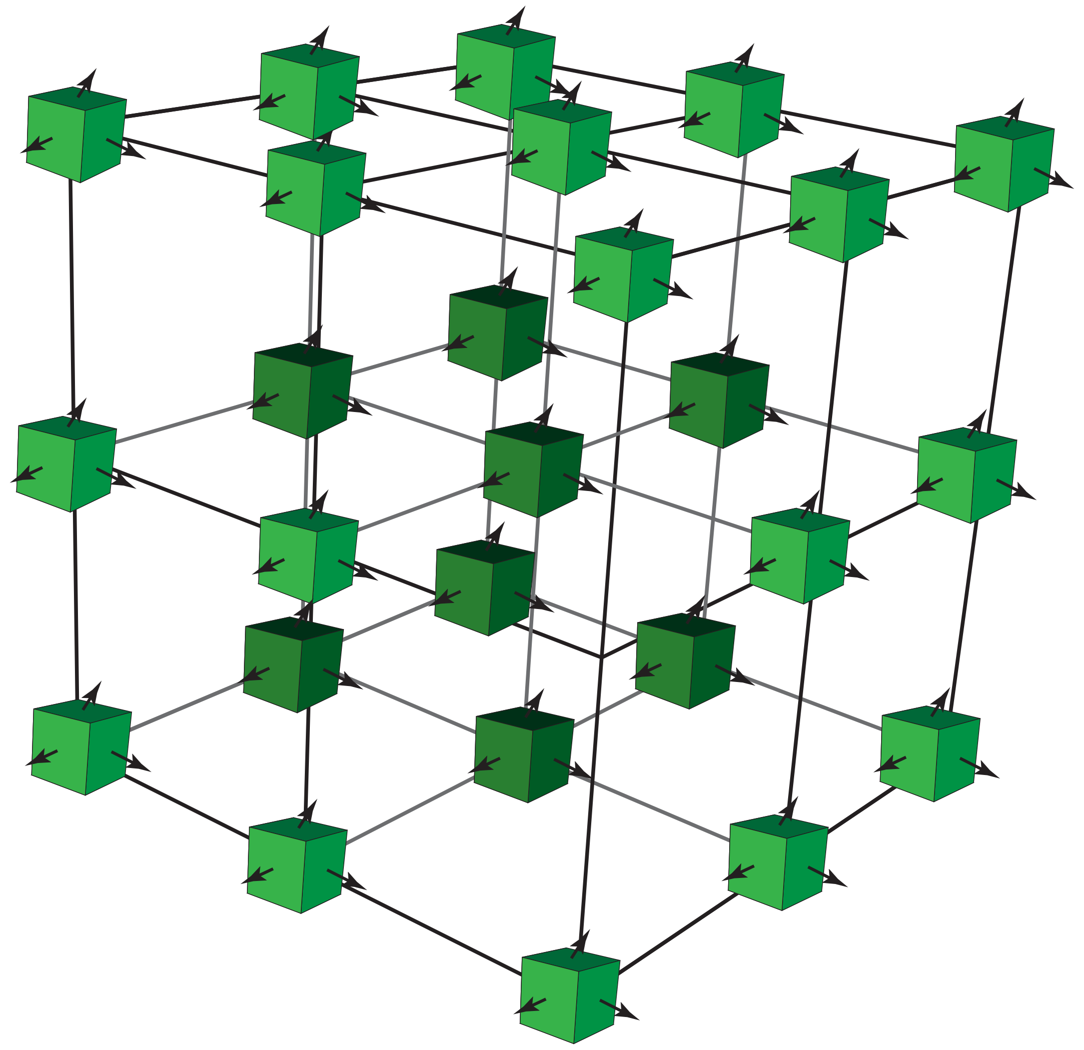
Scalar Fields

(Order-0 Tensor Fields)



Vector Fields

(Order-1 Tensor Fields)



Tensor Fields

(Order-2+)

Each point in space has an associated...

$$s_0$$

Scalar

$$\begin{bmatrix} v_0 \\ v_1 \\ v_2 \end{bmatrix}$$

Vector

$$\begin{bmatrix} \sigma_{00} & \sigma_{01} & \sigma_{02} \\ \sigma_{10} & \sigma_{11} & \sigma_{12} \\ \sigma_{20} & \sigma_{21} & \sigma_{22} \end{bmatrix}$$

Tensor

Visualizing Vector Fields

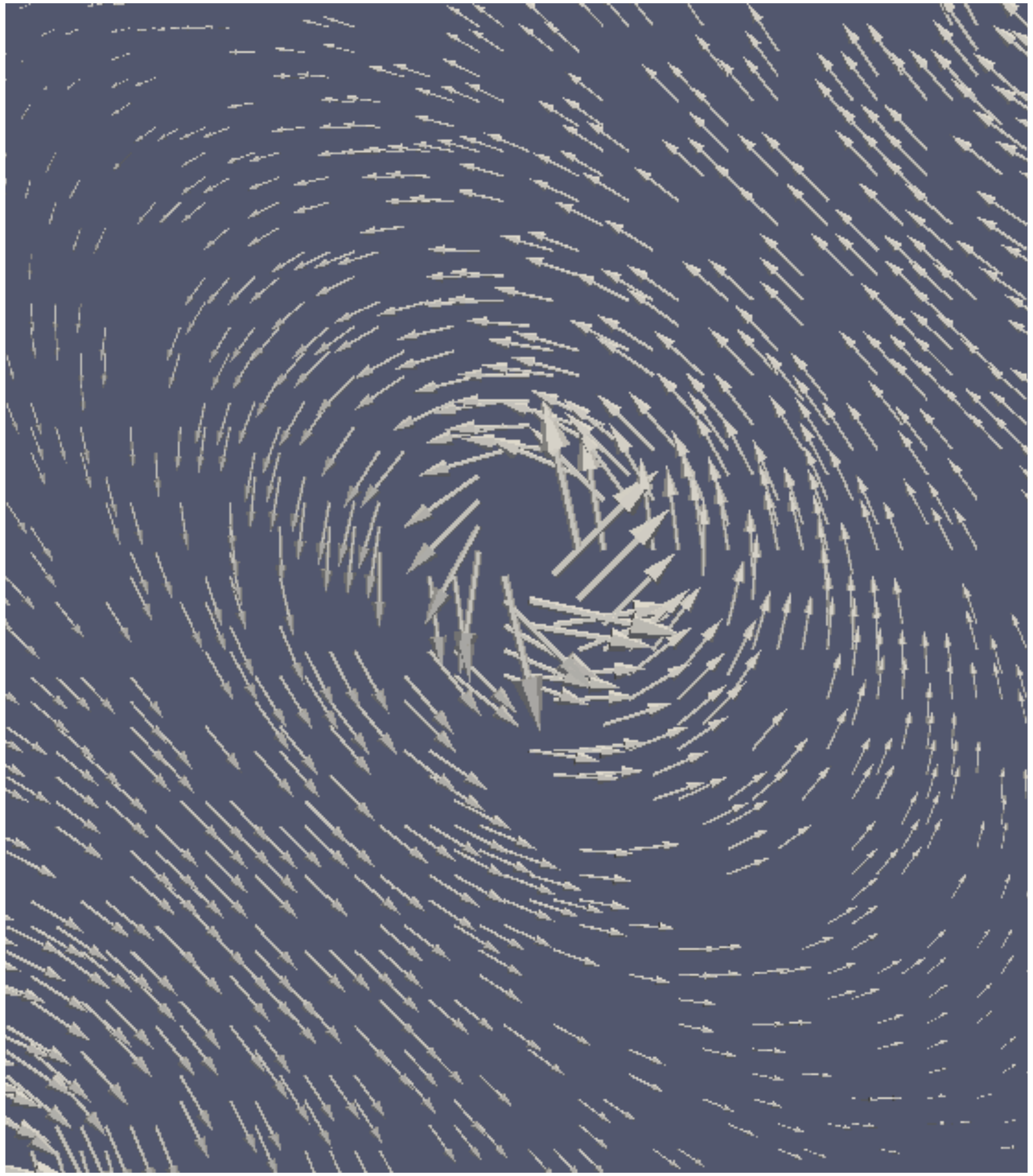
- Direct: Glyphs, Render statistics as scalars
- Geometry: Streamlines and variants
- Textures: Line Integral Convolution (LIC)
- Topology: Extract relevant features and draw them

Glyphs

- Represent each vector with a symbol
- Hedgehogs are primitive glyphs (glyph is a line)
- ParaView Example

Glyphs

- Represent each vector with a symbol
- Hedgehogs are primitive glyphs (glyph is a line)
- Glyphs that show direction and/or magnitude can convey more information
- If we have a separate scalar value, how might we encode that?
- Clutter issues

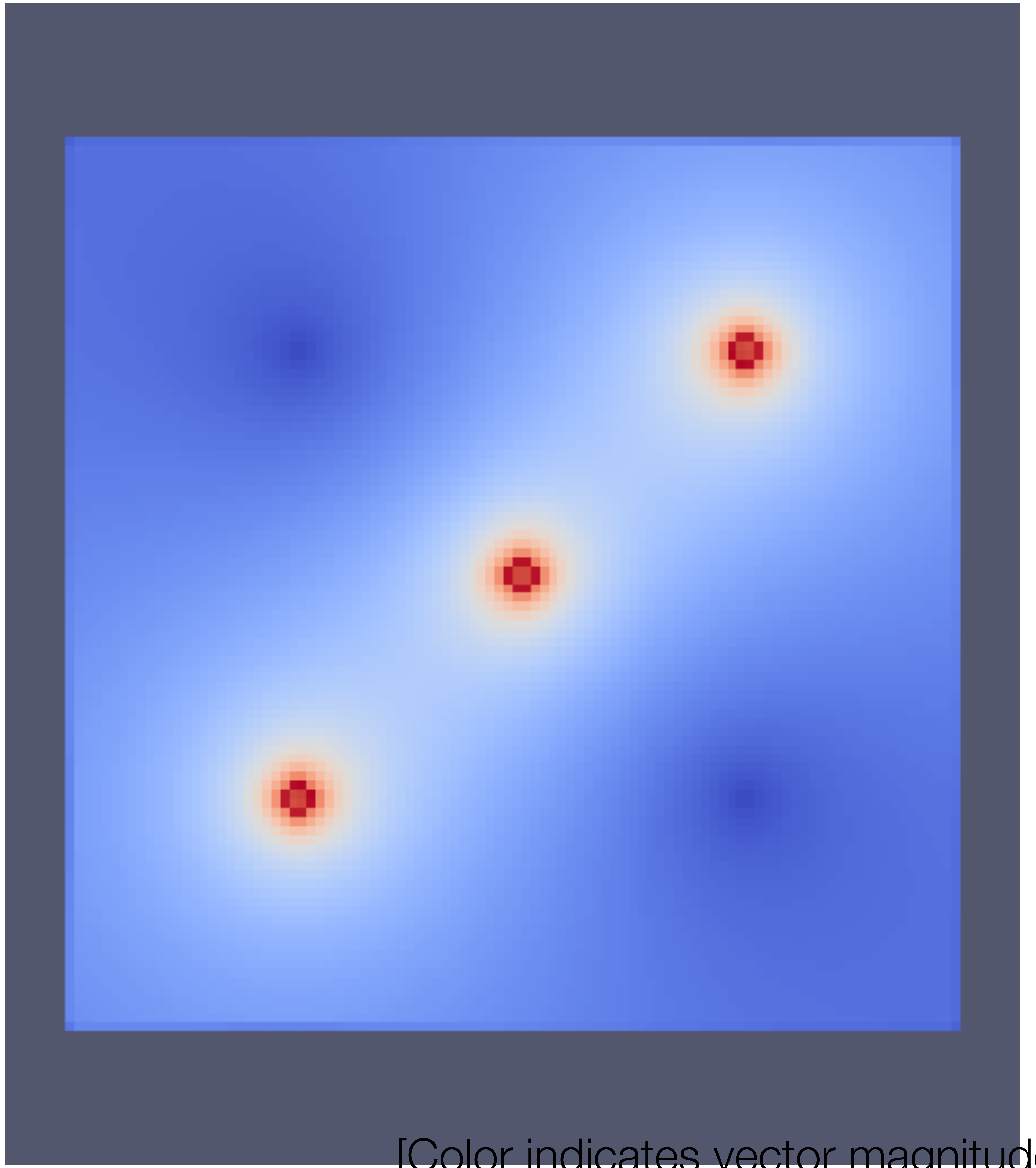


Glyphs

- For vector fields, can encode direction, magnitude, scalar value
- Good:
  - Show precise local measures
  - Can encode scalar information as color
- Bad:
  - Possible sampling issues
  - Clutter (Occlusion): Can remove some points to help
  - Clutter is worse in higher dimensions

Rendering Vector Field Statistics as Scalars

- Many statistics we can compute for vector fields:
  - Magnitude
  - Vorticity
  - Curvature
- These are scalars, can color with our scalar field visualization techniques (e.g. volume rendering)



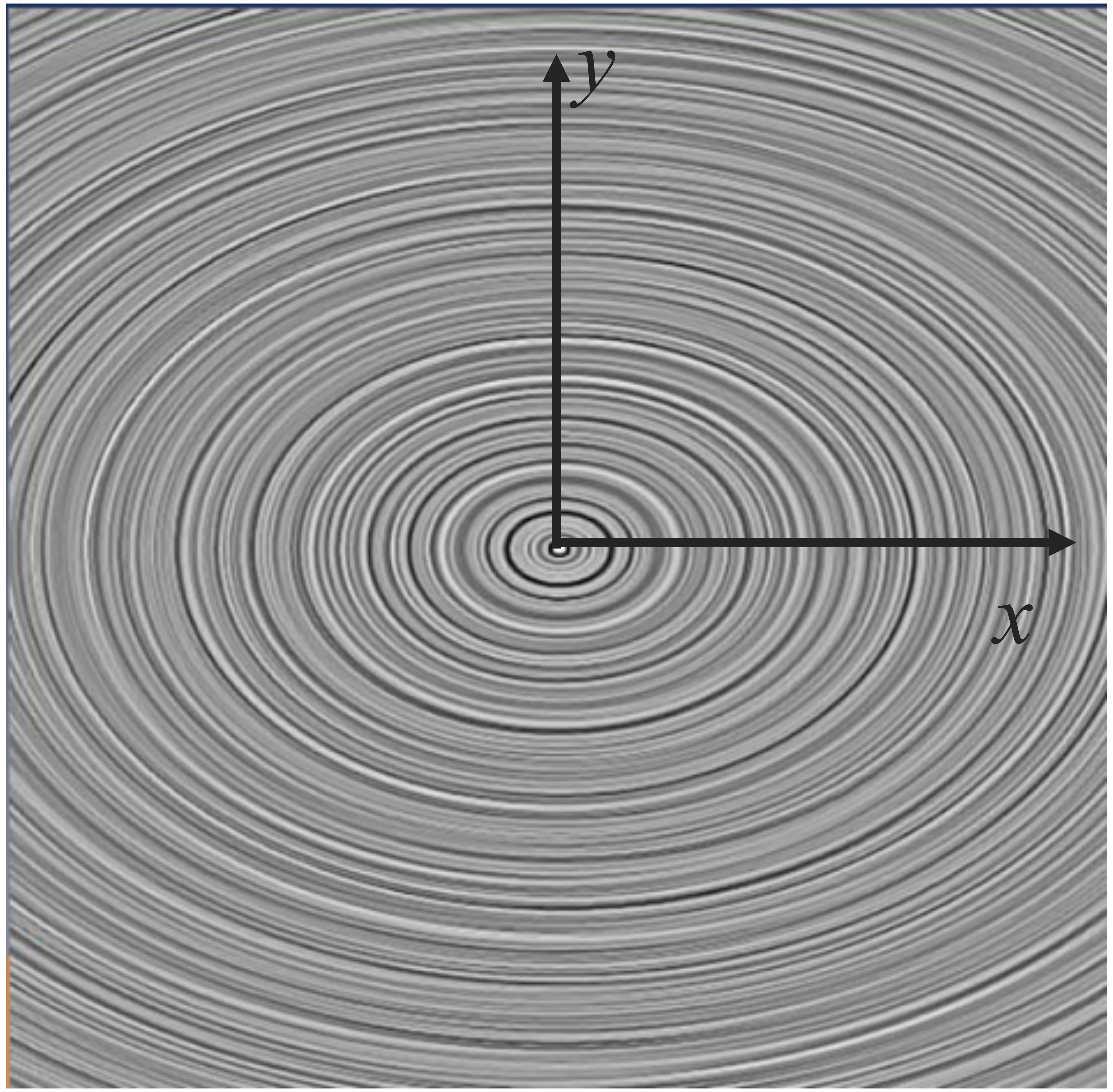
[Color indicates vector magnitude]

Streamlines & Variants

- Trace a line along the direction of the vectors
- Streamlines are always tangent to the vector field
- Basic Particle Tracing:
  1. Set a starting point (seed)
  2. Take a step in the direction of the vector at that point
  3. Adjust direction based on the vector where you are now
  4. Go to Step 2 and Repeat

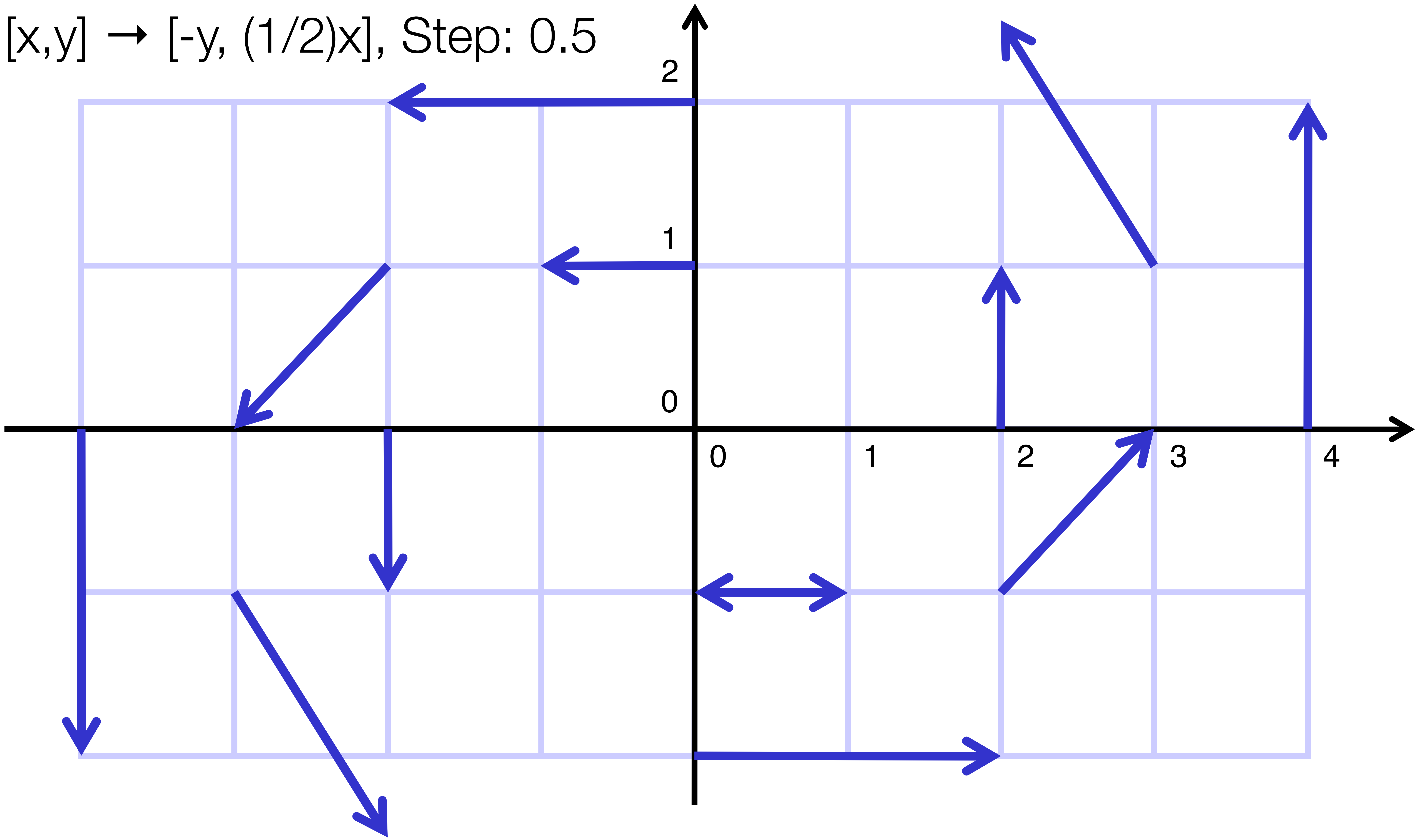
Example

- Elliptical path
- Suppose we have the actual equation
- Given point (x,y), the vector is at that point is $[v_x, v_y]$ where
  - $v_x = -y$
  - $v_y = (1/2)x$
- Want a streamline starting at (0,-1)



[LIC (not streamlines!) via Levine]

Some Glyphs



[via Levine]