2IMV20

Encoding: spatial data

Technische Universiteit
Eindhoven
University of Technology

Recap

Volume rendering equation

- Discretize and simplify

$$I(t) = I_0 e^{-\int_0^t \tau(u)du} + \int_0^t c(s) e^{-\int_s^t \tau(u)du} ds$$

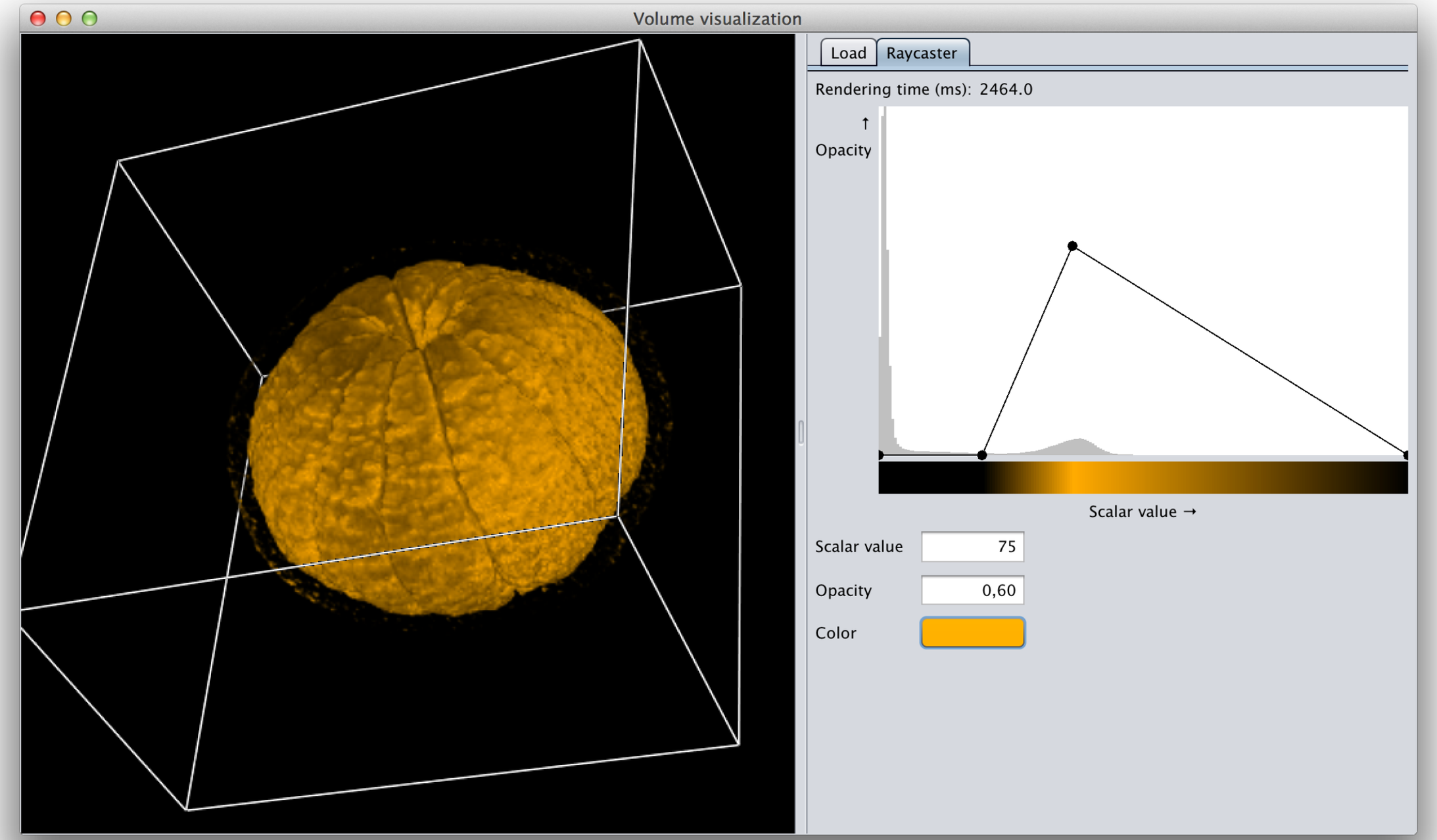
$$I(p) = \sum_{i=0}^{n-1} c_i \prod_{j=i+1}^{n-1} (1 - \tau_j)$$

color **opacity**

- Compositing order (implementation aspect of above equation)
  - back-to-front
  - $$C_i = \tau_i c_i + (1 - \tau_i) C_{i-1}$$

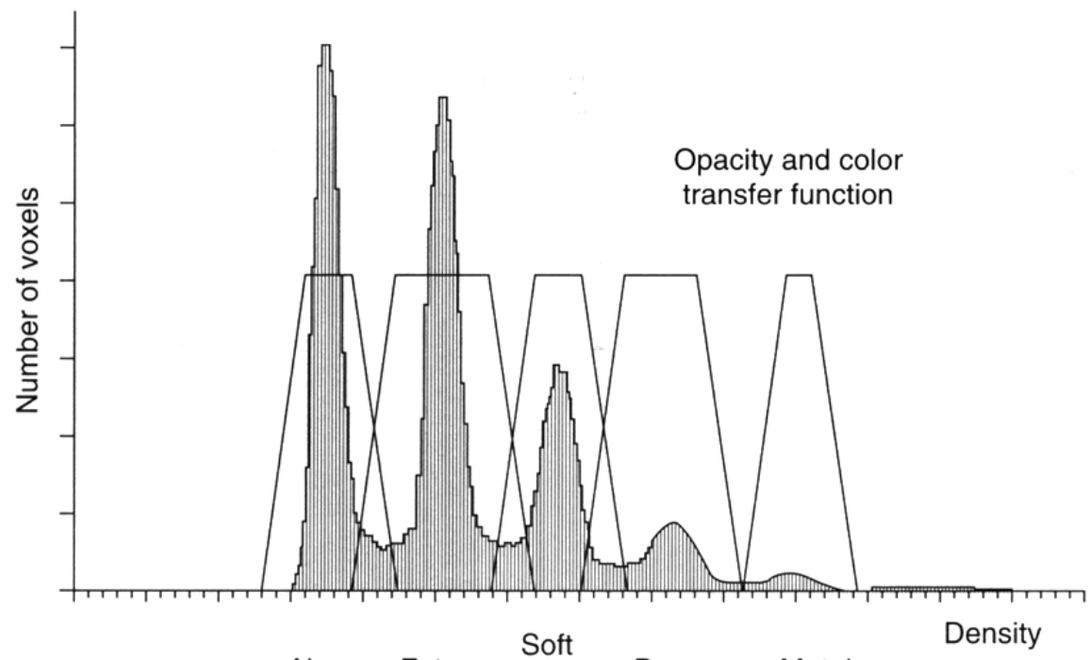
Transfer functions

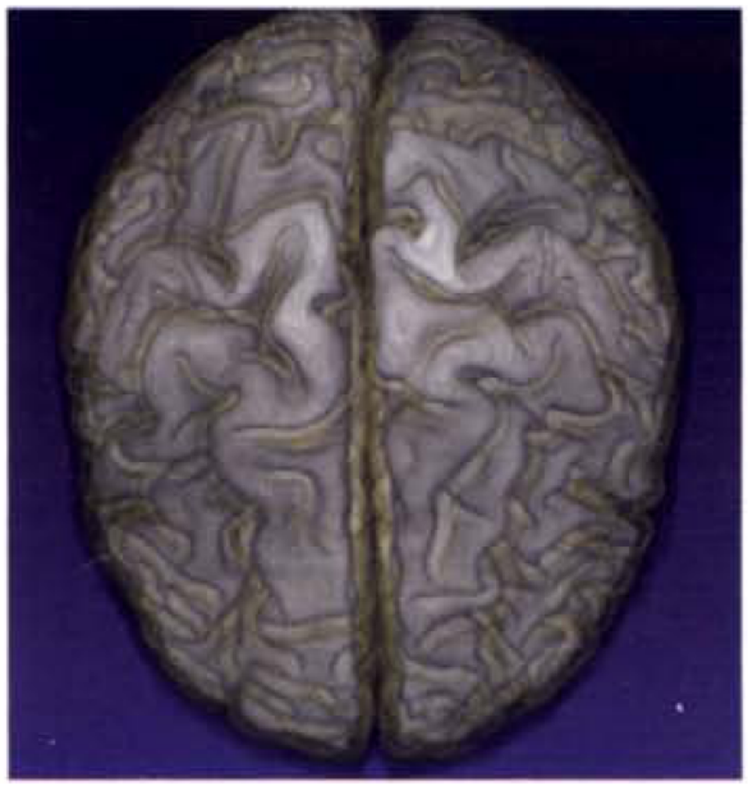
- Map data values to color c_i and opacity τ_i



Transfer function design strategy

- Trial-and-error: repeat until satisfied
- Make use of density histogram
- Try to emphasize boundaries

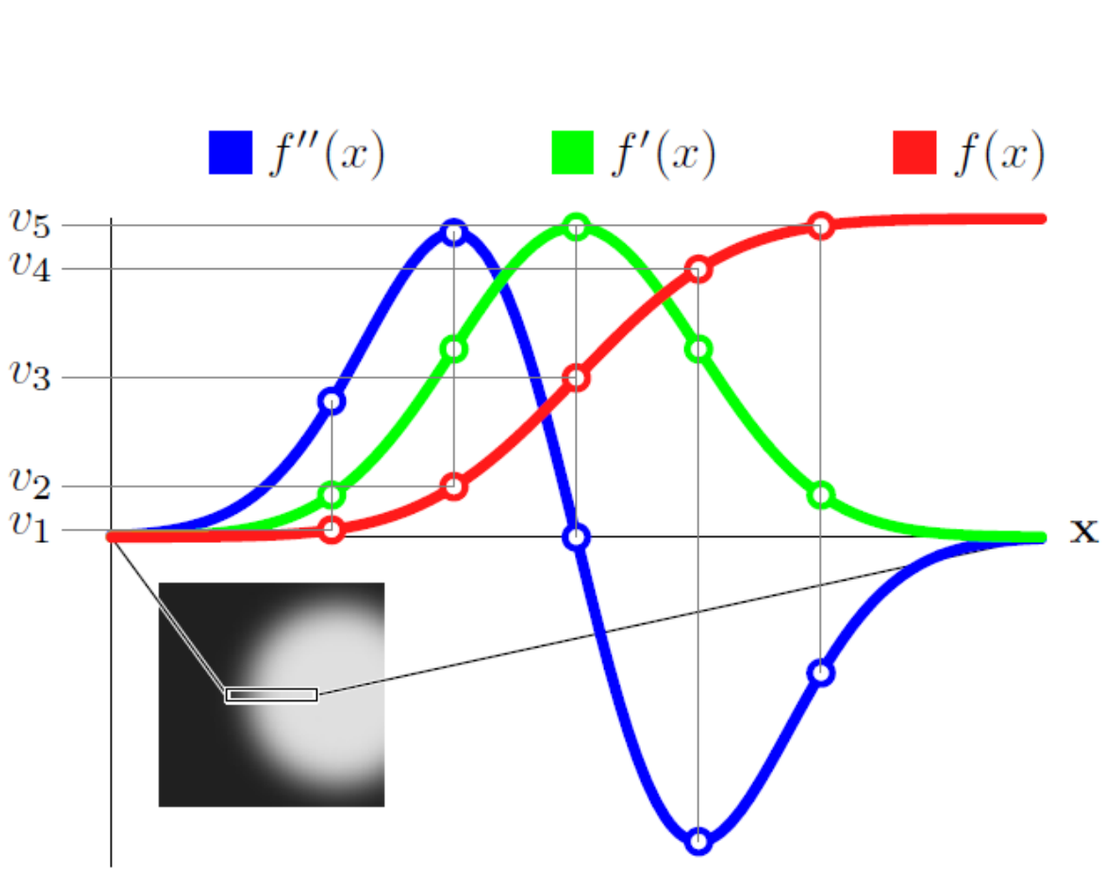




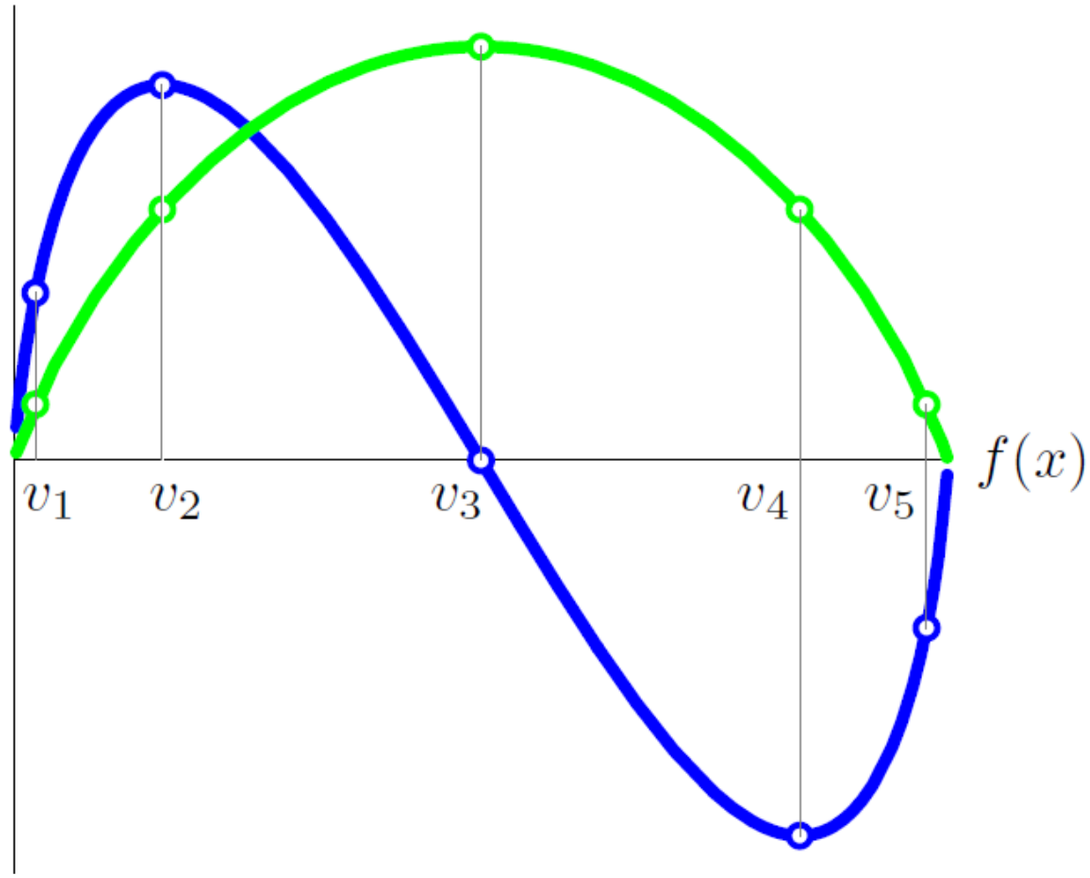
Transfer functions continued

Data-driven transfer function design

- Boundary exists where
  - maximum in first-order derivative
  - zero crossing in second-order derivative

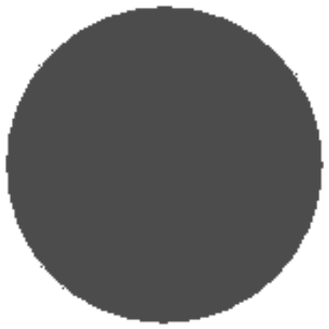


function of position



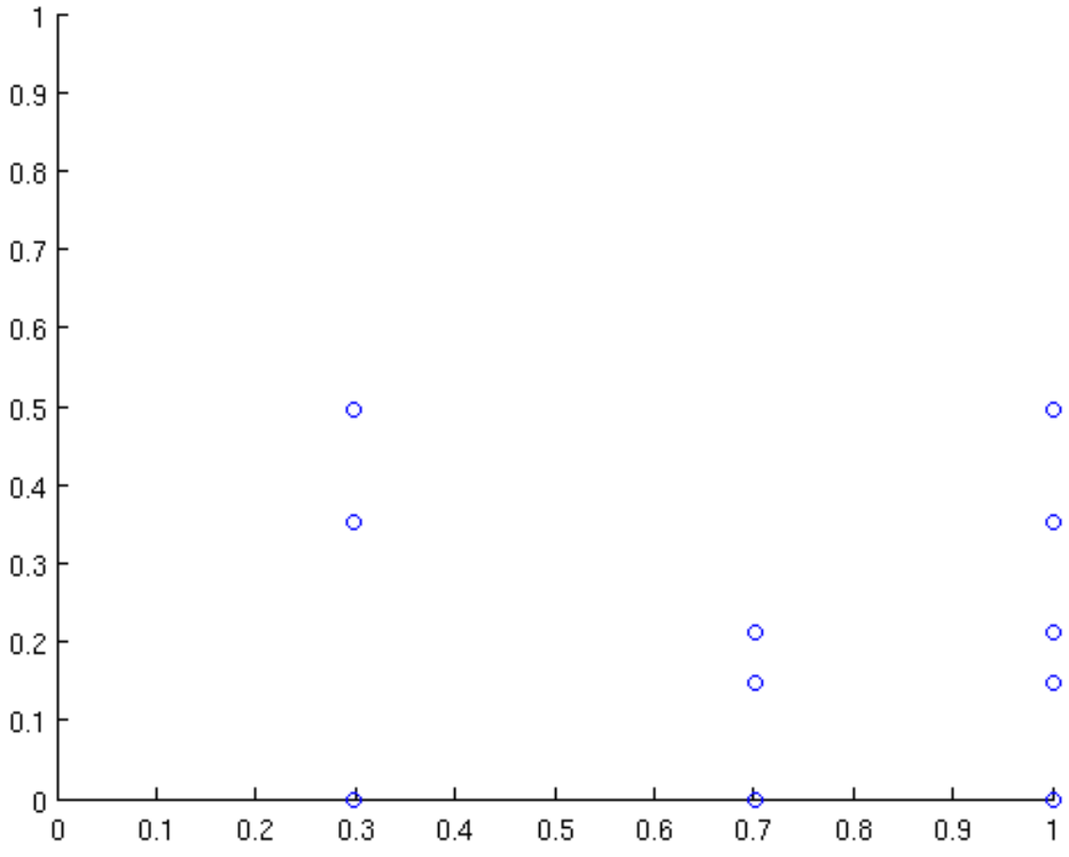
function of value

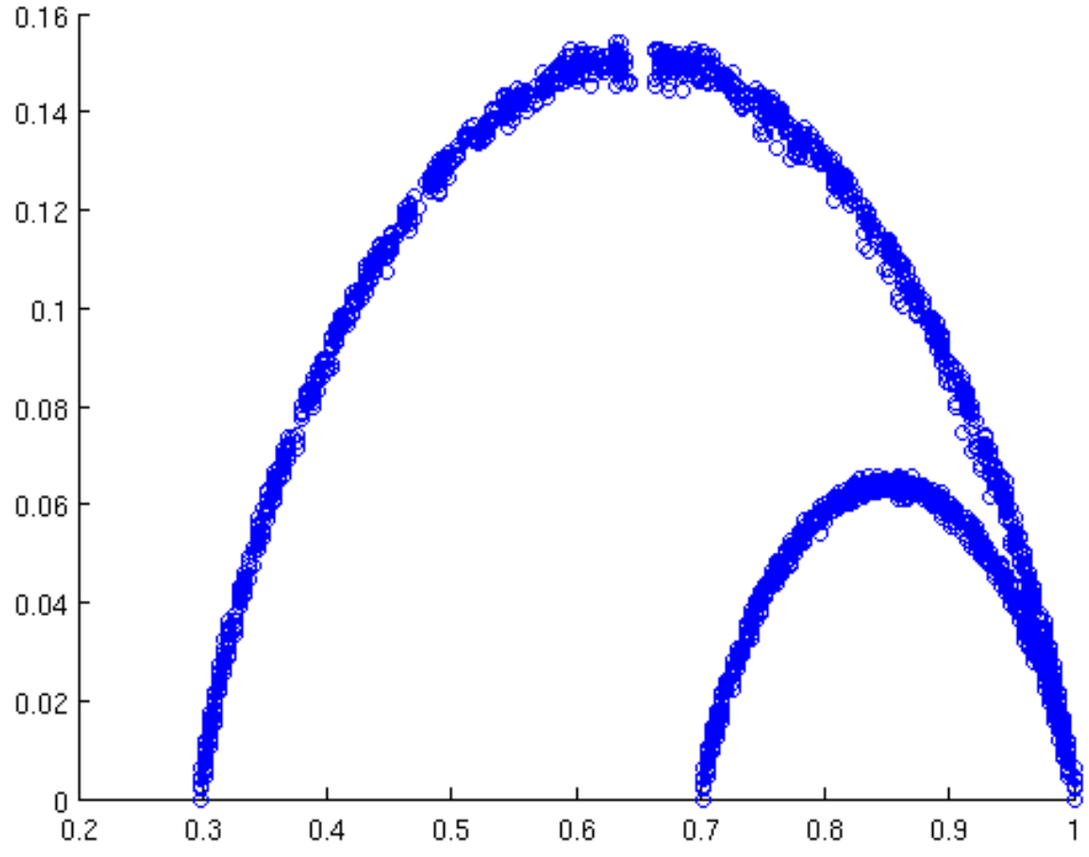
Intensity/gradient magnitude plot



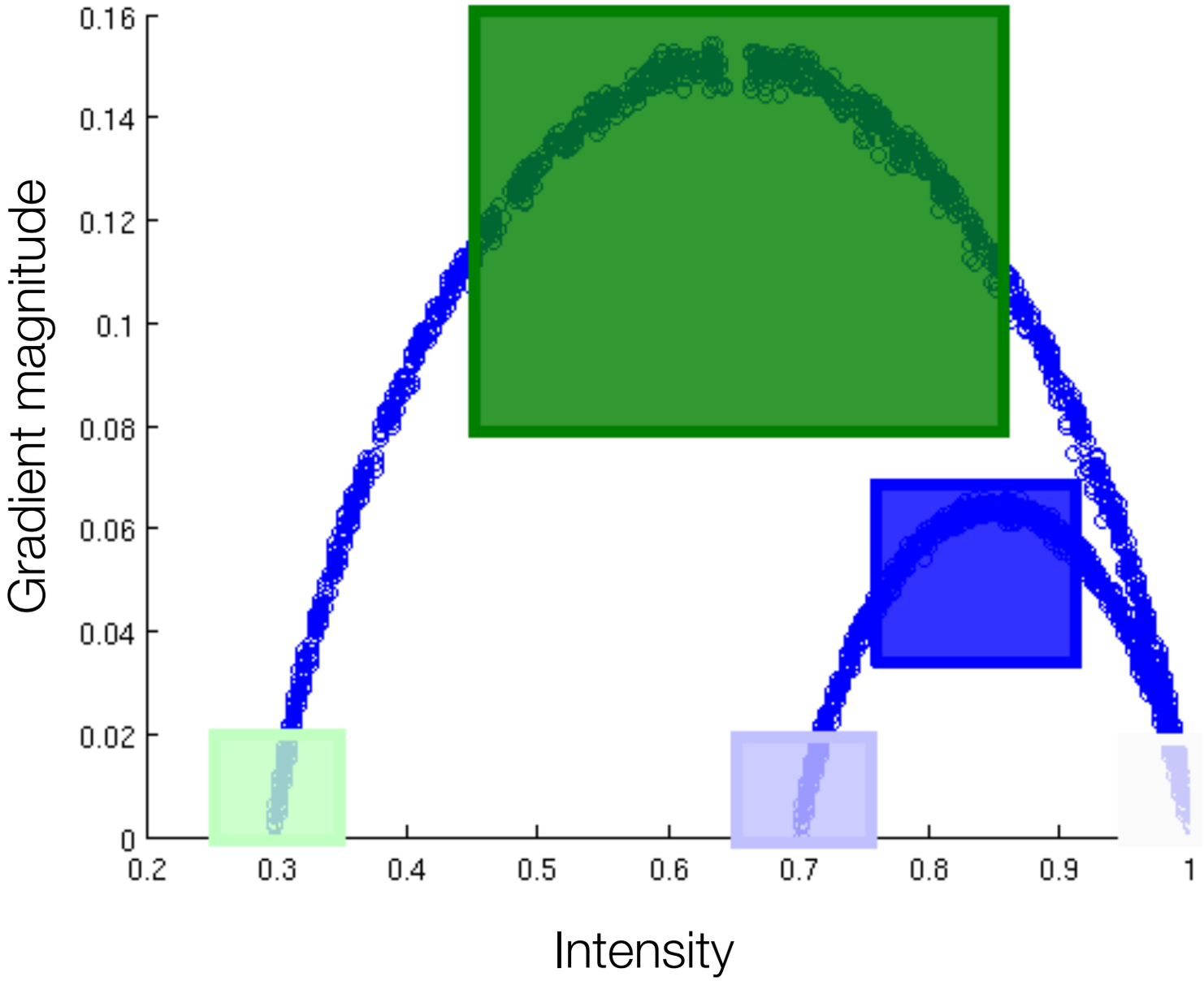
step edges

blurred step edges

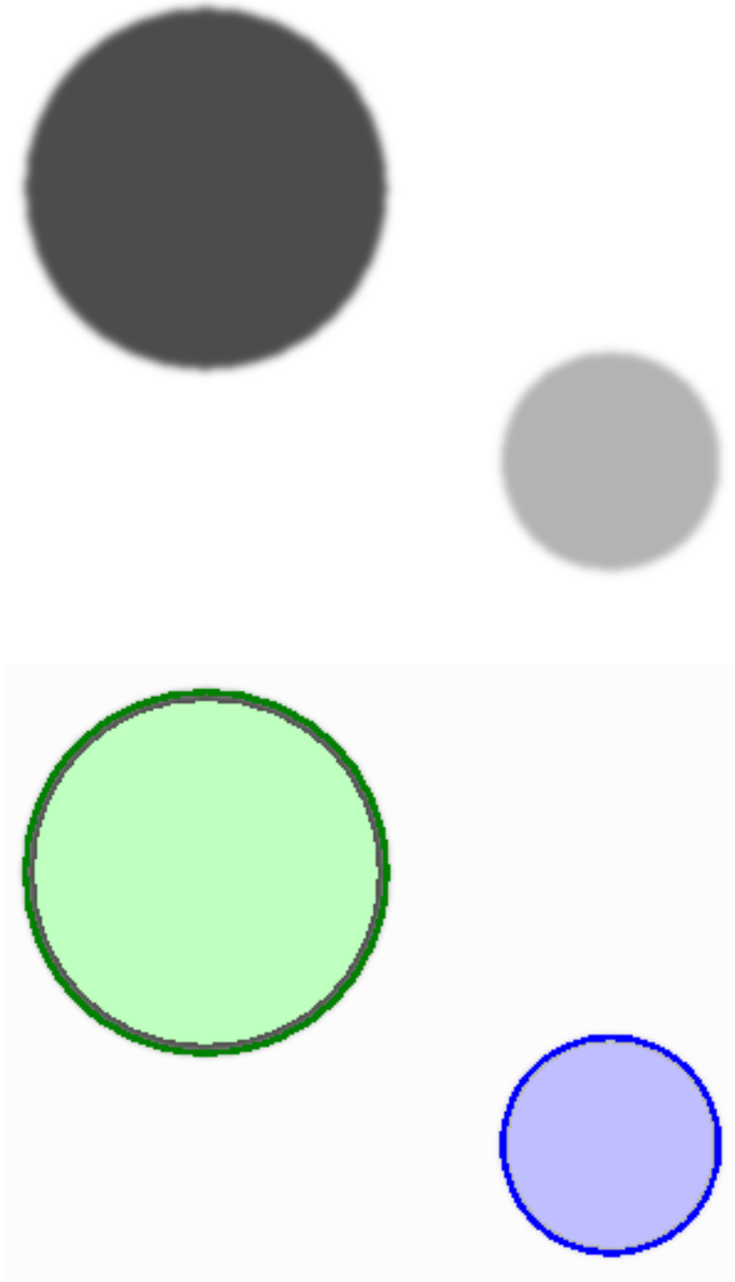




2D transfer function



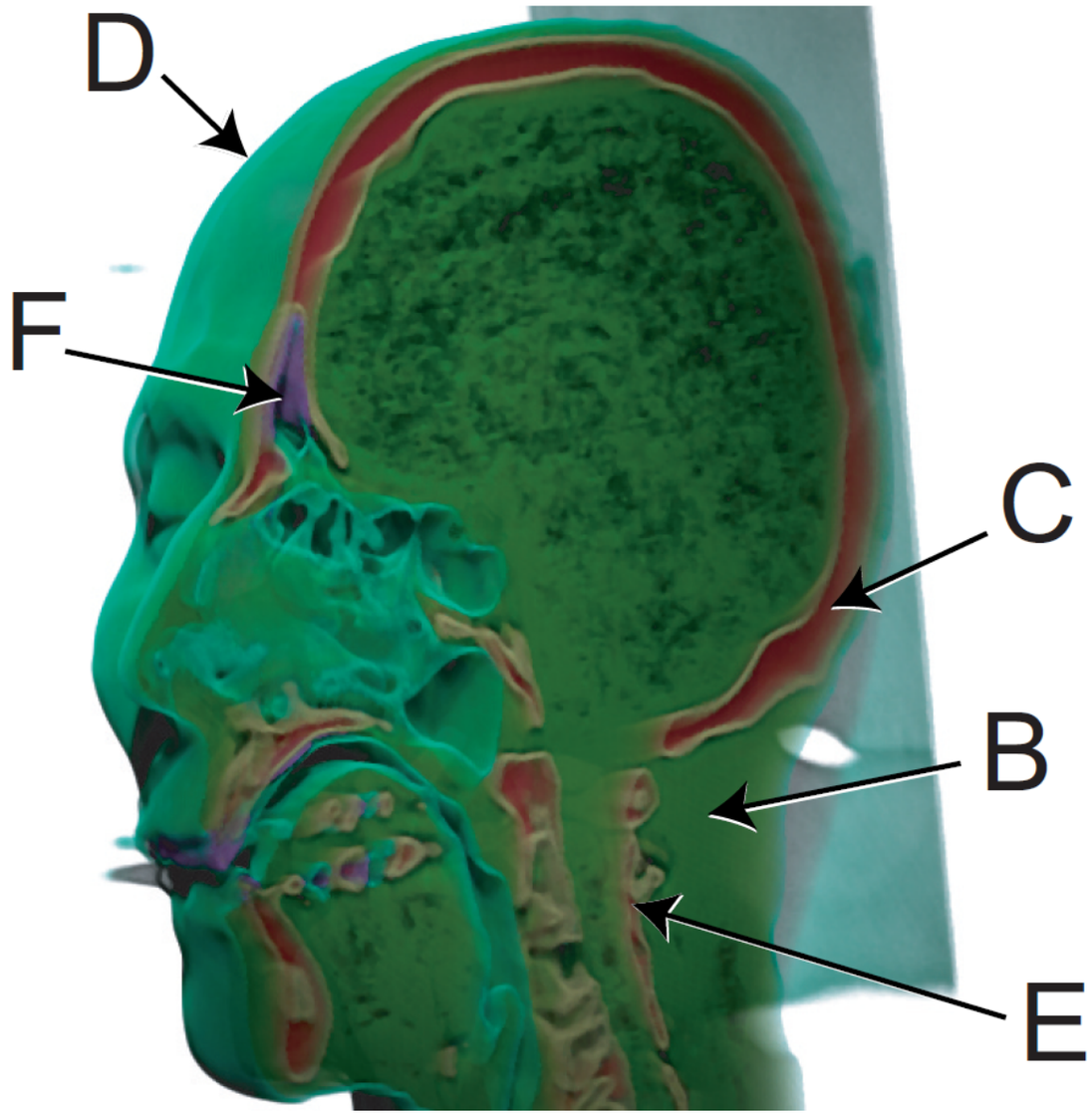
TF design

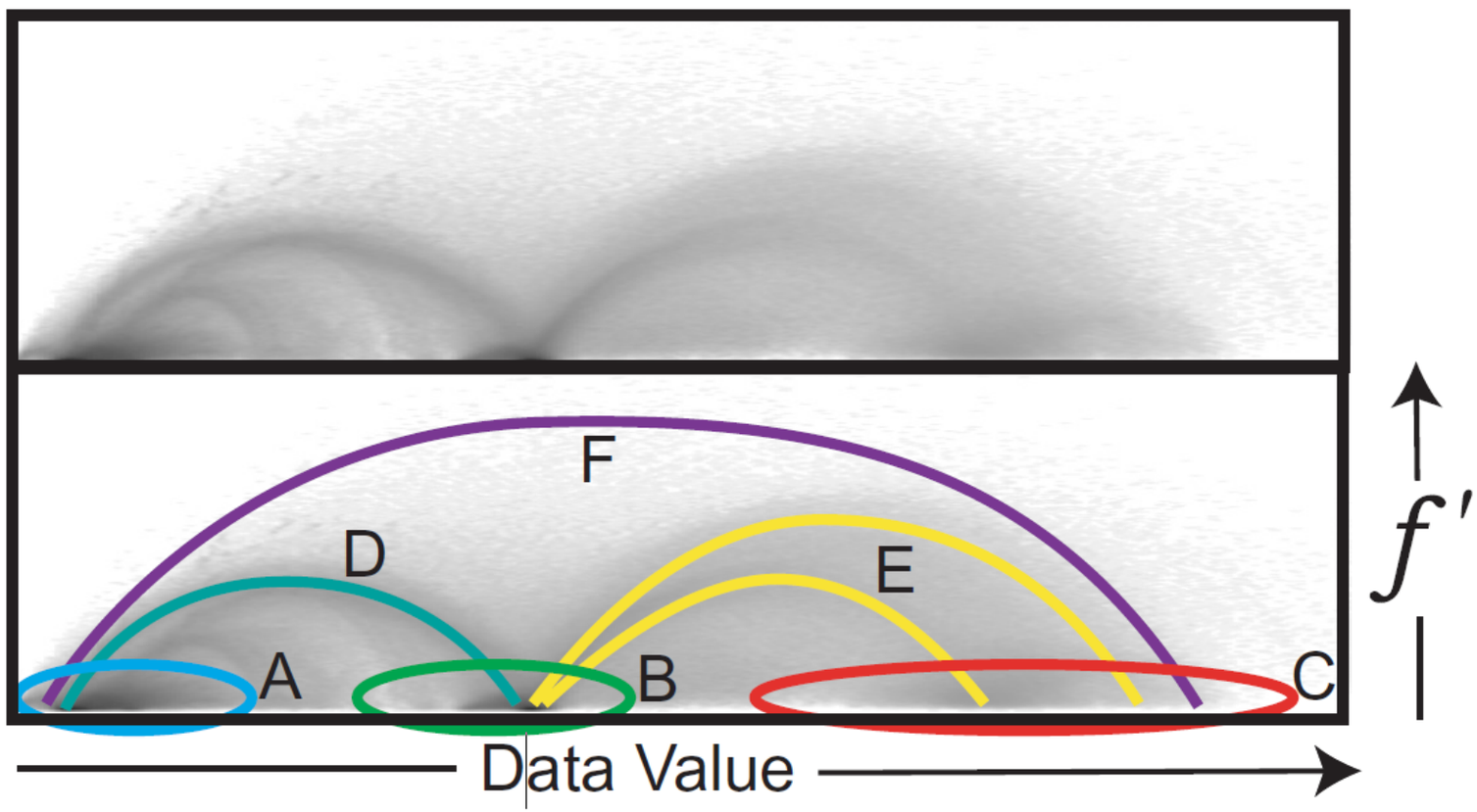


TF applied

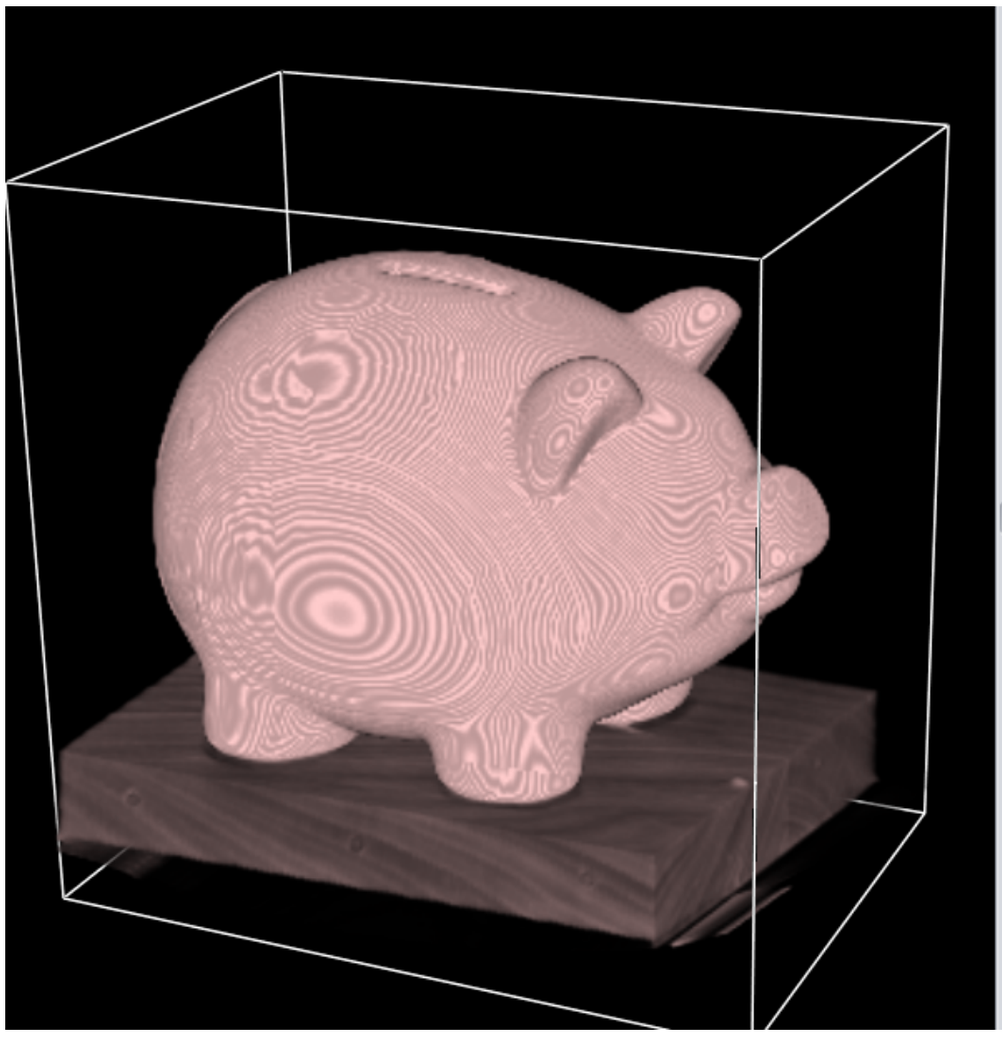
2-D transfer functions

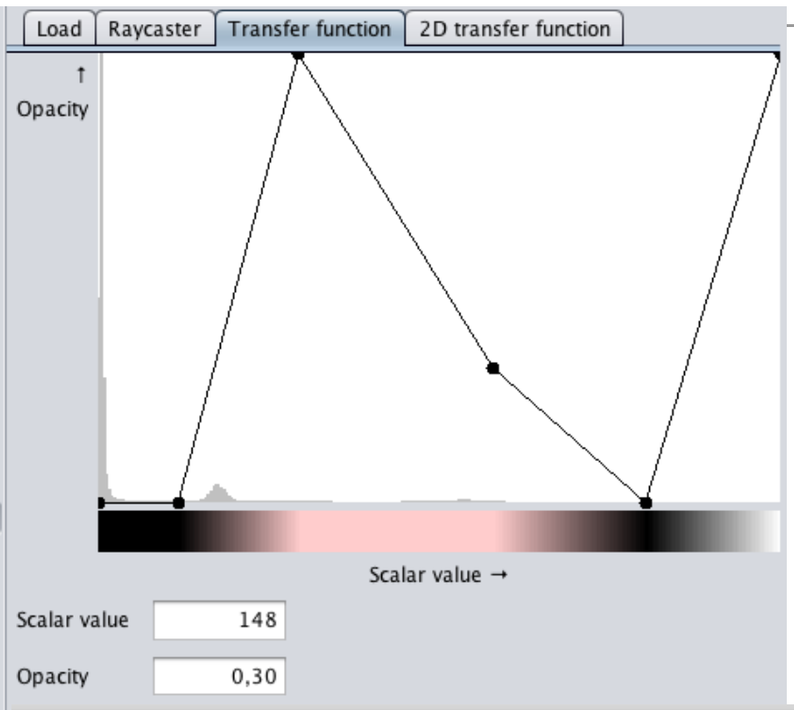
- Histogram of voxel intensity and gradient magnitude
- Identify arcs

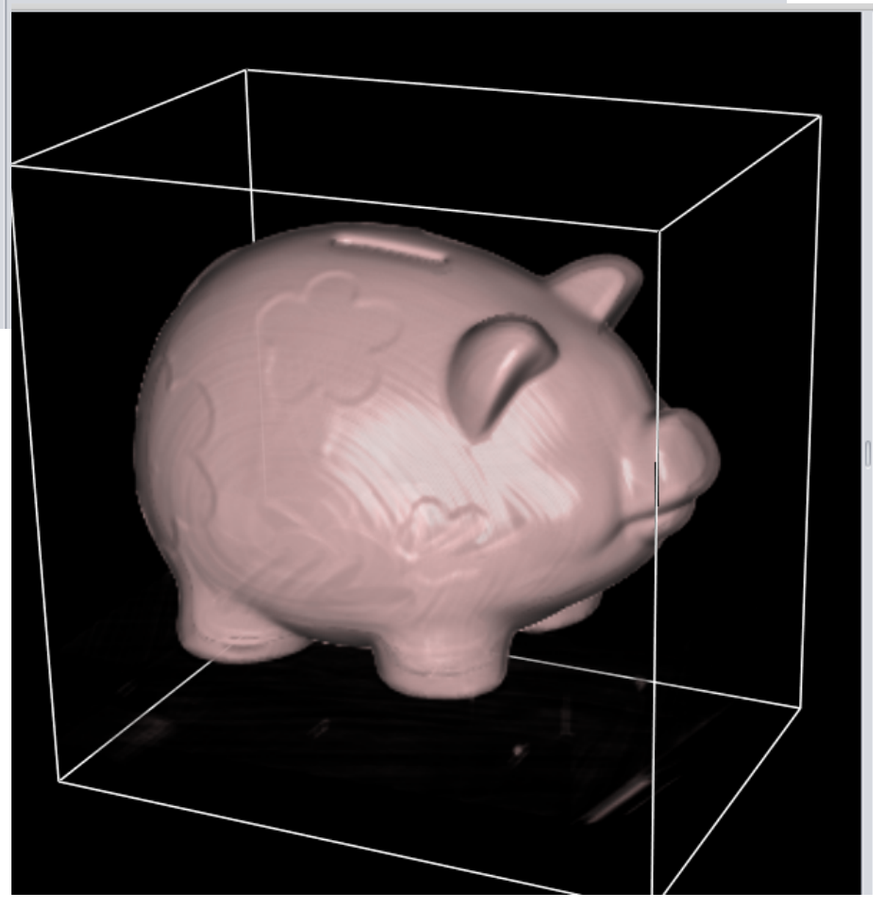


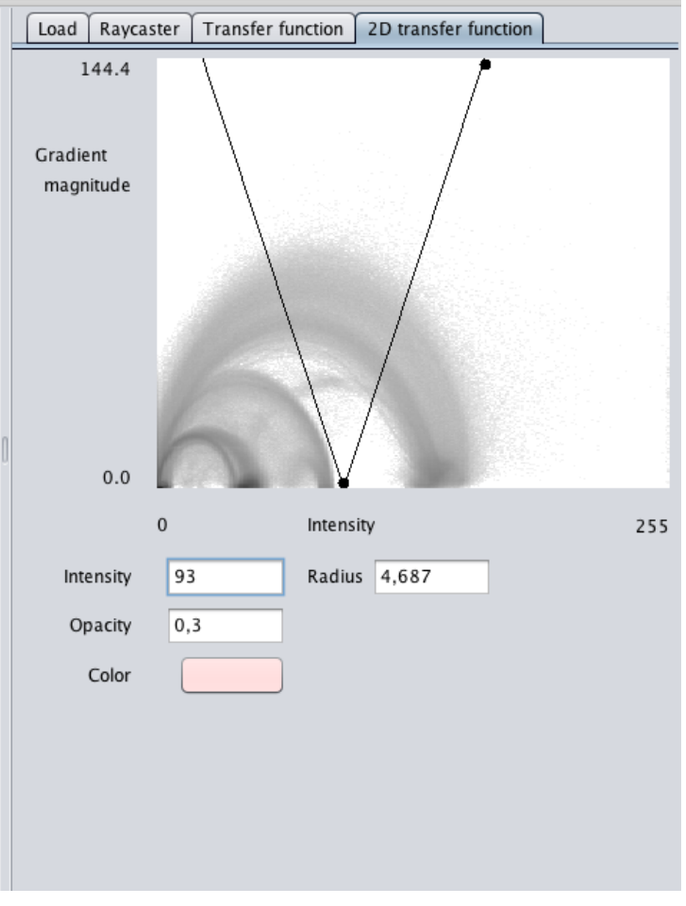


Transfer functions









Levoy's method

$$\alpha(\mathbf{x}_i) = \alpha_v \begin{cases} 1 & \text{if } |\nabla f(\mathbf{x}_i)| = 0 \text{ and } f(\mathbf{x}_i) = f_v \\ 1 - \frac{1}{r}\left|\frac{f_v - f(\mathbf{x}_i)}{|\nabla f(\mathbf{x}_i)|}\right| & \text{if } |\nabla f(\mathbf{x}_i)| > 0 \text{ and } f(\mathbf{x}_i) - r\,|\nabla f(\mathbf{x}_i)| \le f_v \le f(\mathbf{x}_i) + r\,|\nabla f(\mathbf{x}_i)| \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Computation opacity

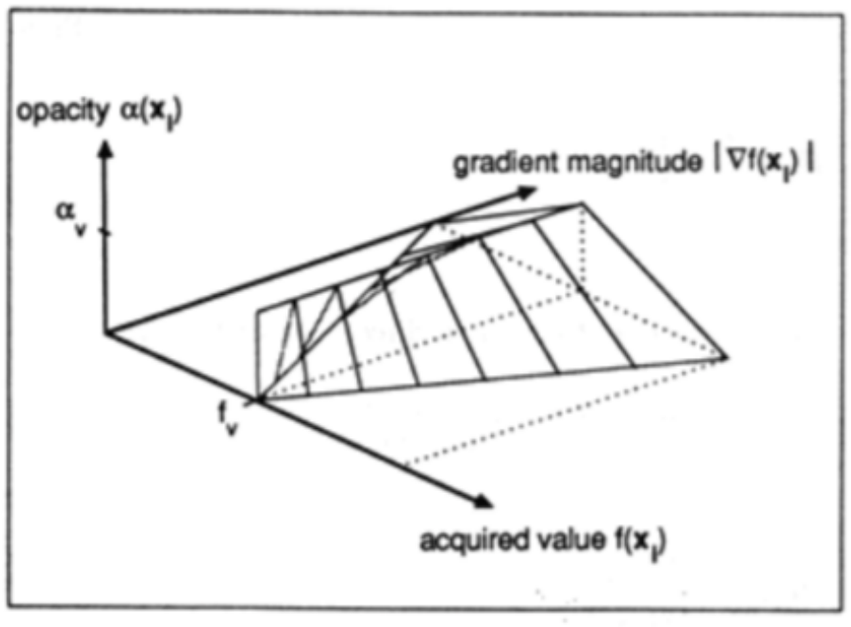
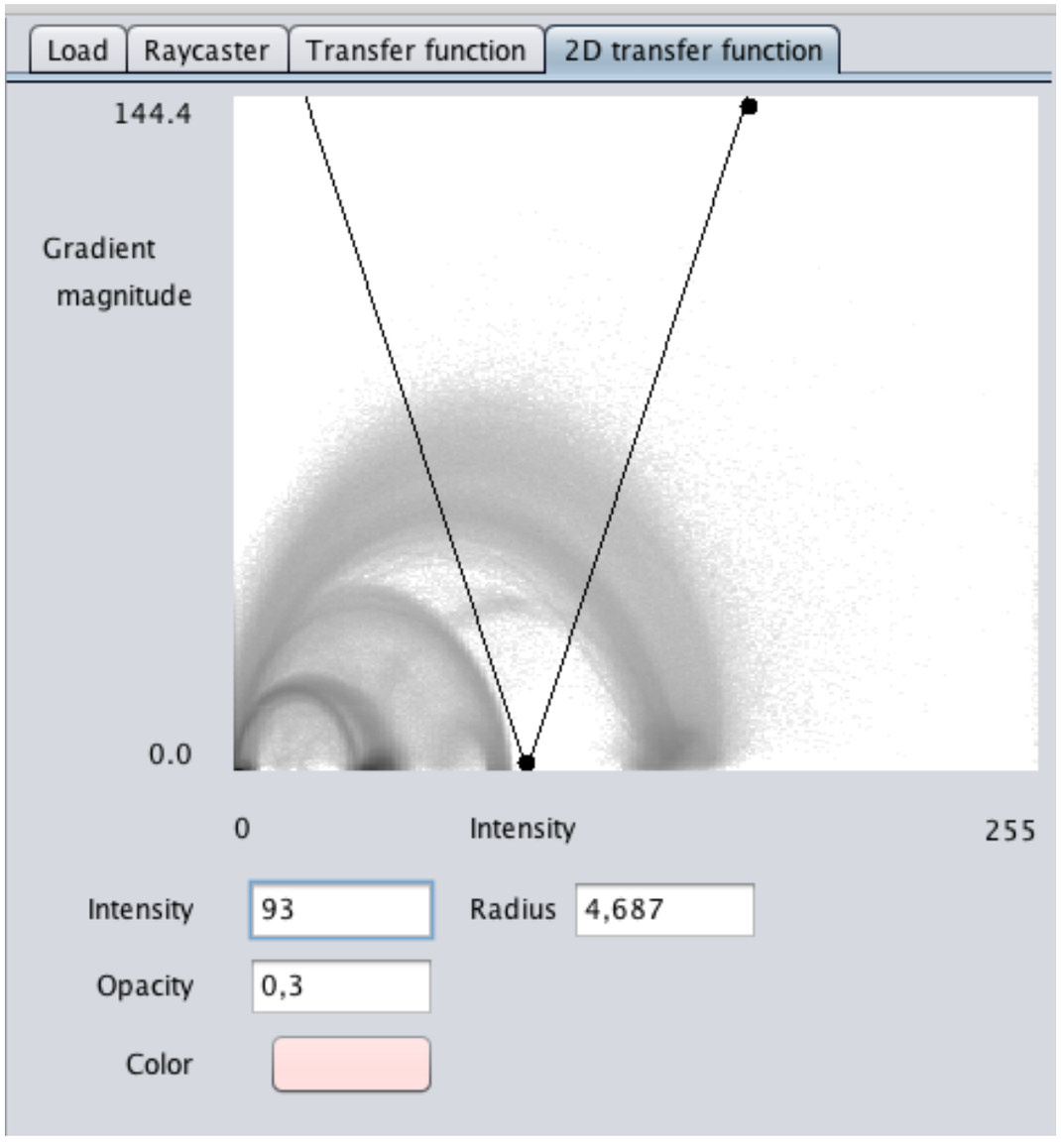


Figure 3. Calculation of opacities for isovalue contour surfaces.

Volume Rendering

Display of Surfaces from Volume Data

Marc Levoy, University of North Carolina

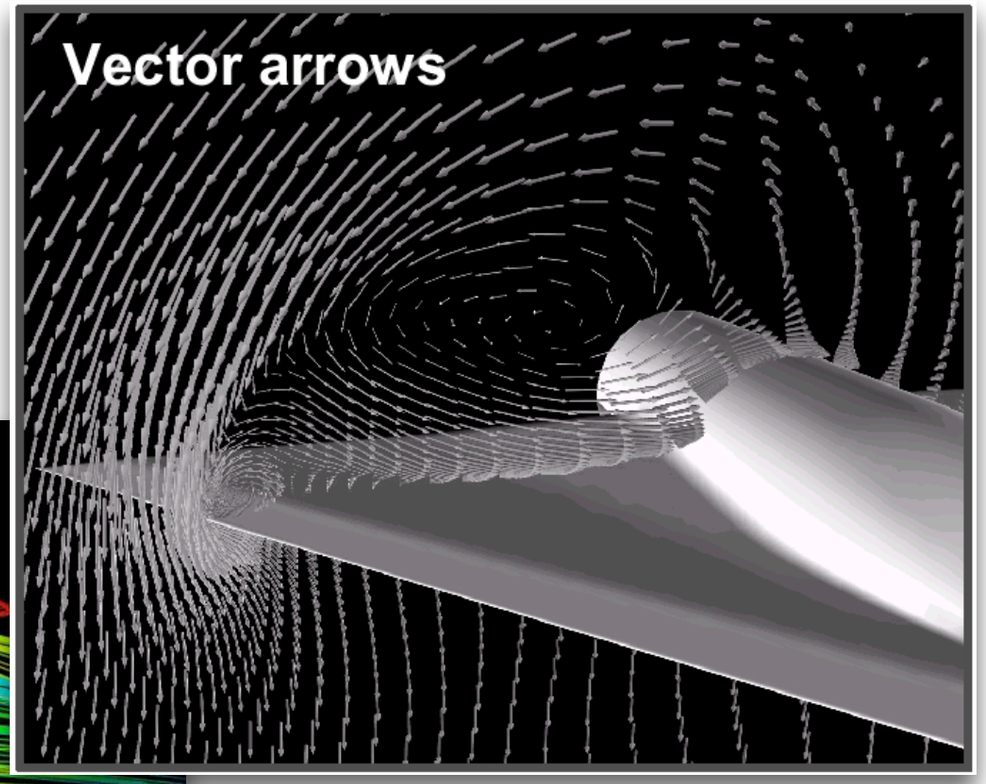


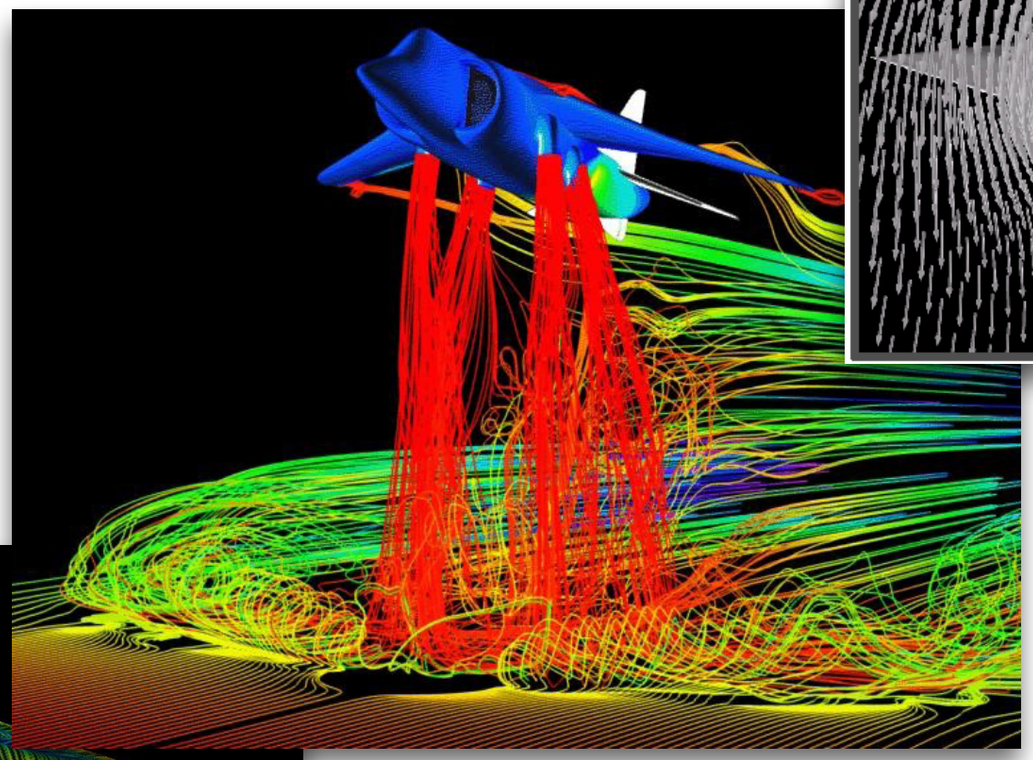
Local illumination in volume rendering

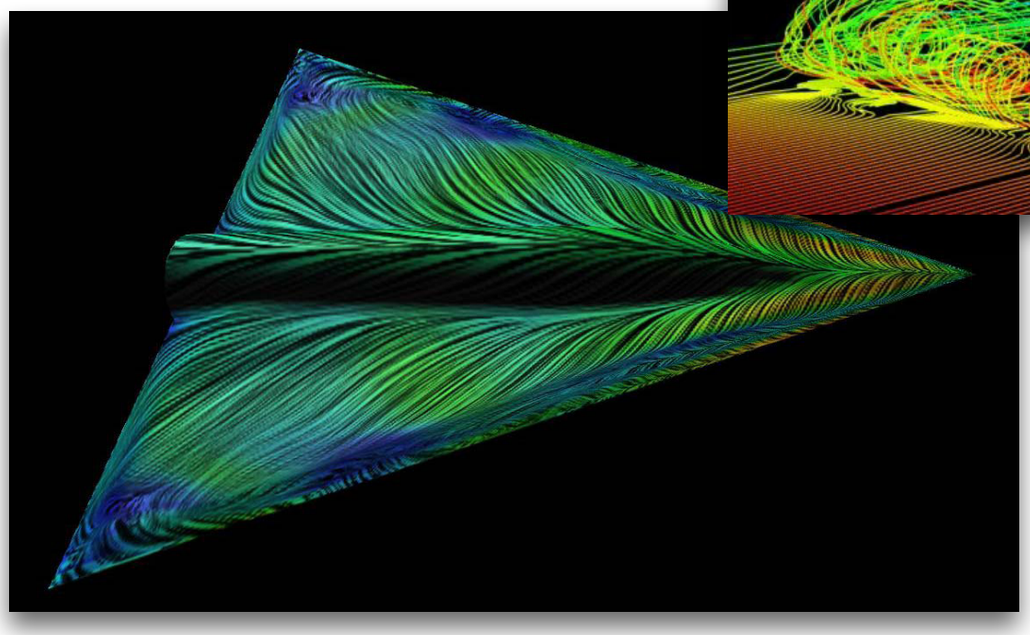
- Use standard Phong model $I = I_a k_{\text{ambient}} + I_l k_{\text{diff}}(\mathbf{L} \cdot \mathbf{N}) + I_l k_{\text{spec}}(\mathbf{V} \cdot \mathbf{R})^{\alpha}$
- Estimate normal vector **N** by computing gradient in each voxel

Vector visualization techniques

- Glyphs
- Streamlines
- Texture based





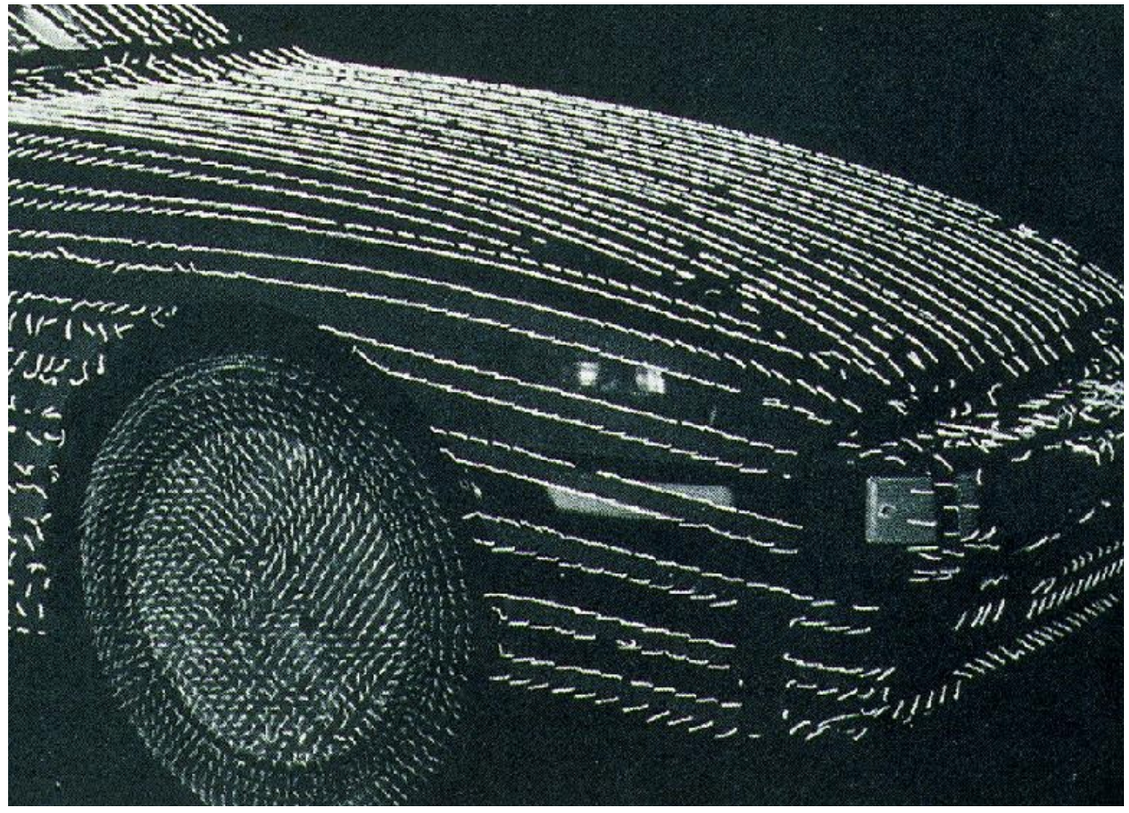


Insight into vector fields

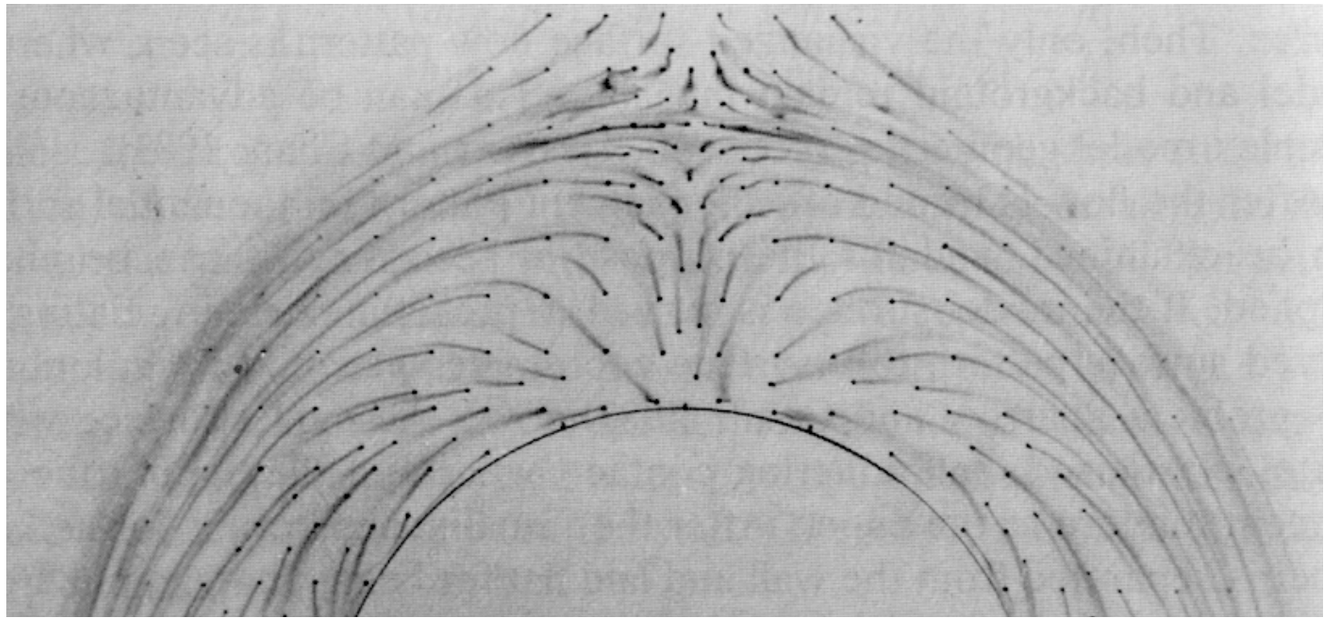
- Properties to visualize
  - global view
  - local view
  - derived properties
- Challenges
  - 2 or 3 components per data point
  - temporal aspects
  - data density

Techniques inspired by experimental visualization

Smoke lines



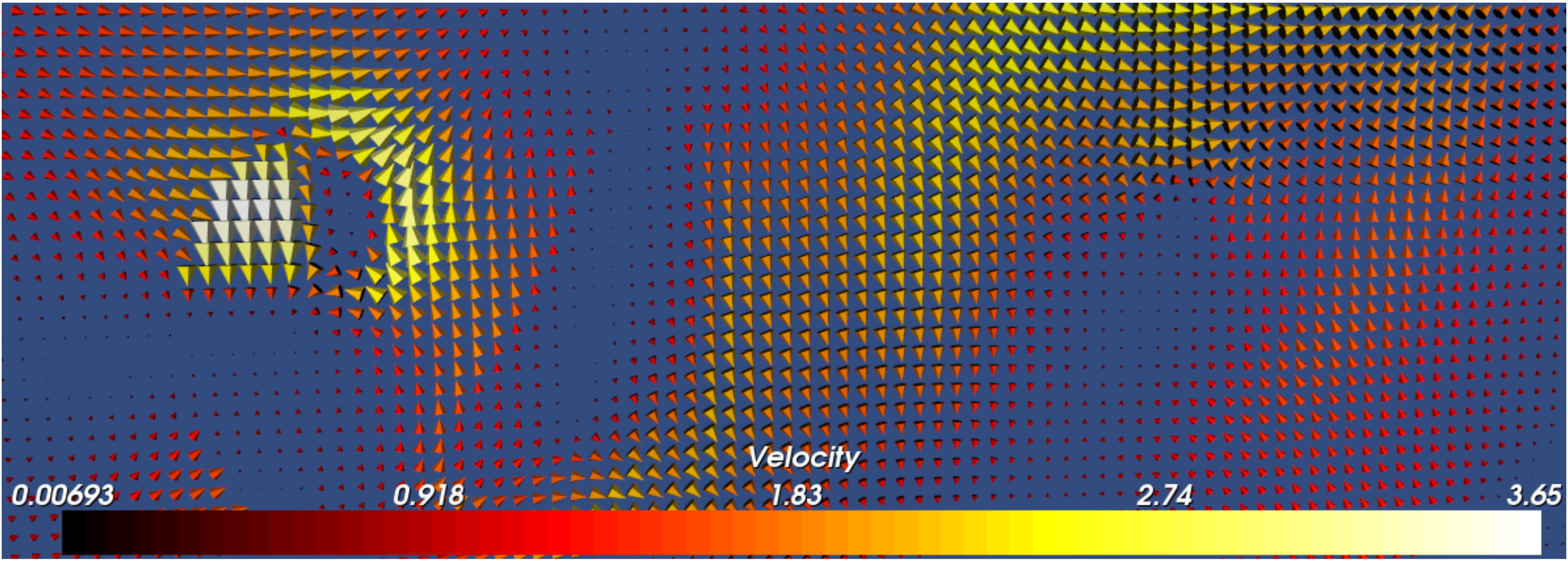
Tufts



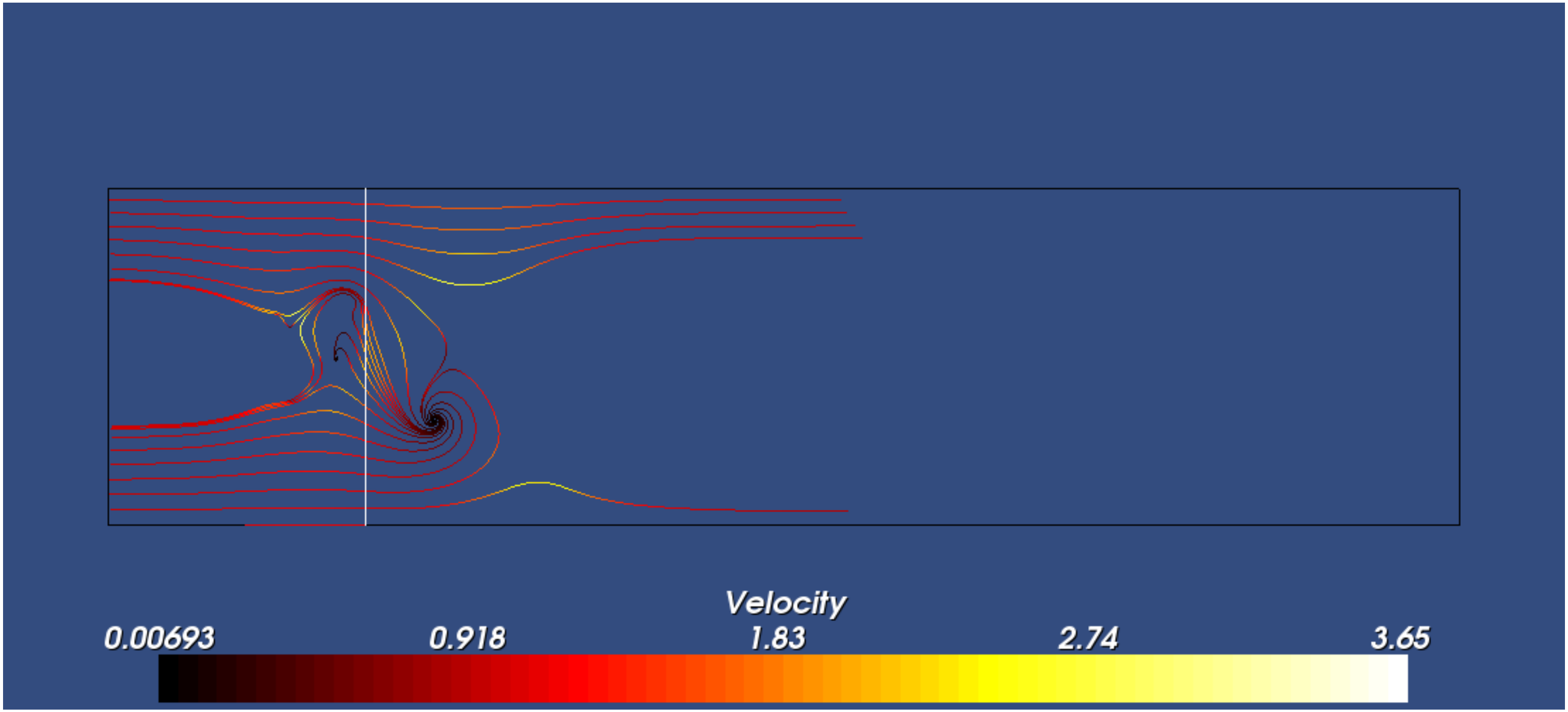
Oil streaks

Vector glyphs

- Assign icon to each sample point
- Icon properties: size, location, direction, orientation, color
- Icons: line segment, arrow, cone, …



Streamlines



Streamlines

- Line followed by one particle during a certain period
- Motion of particle

$$\frac{d\mathbf{x}}{dt} = \vec{v}(\mathbf{x})$$

- Integration required to get next position

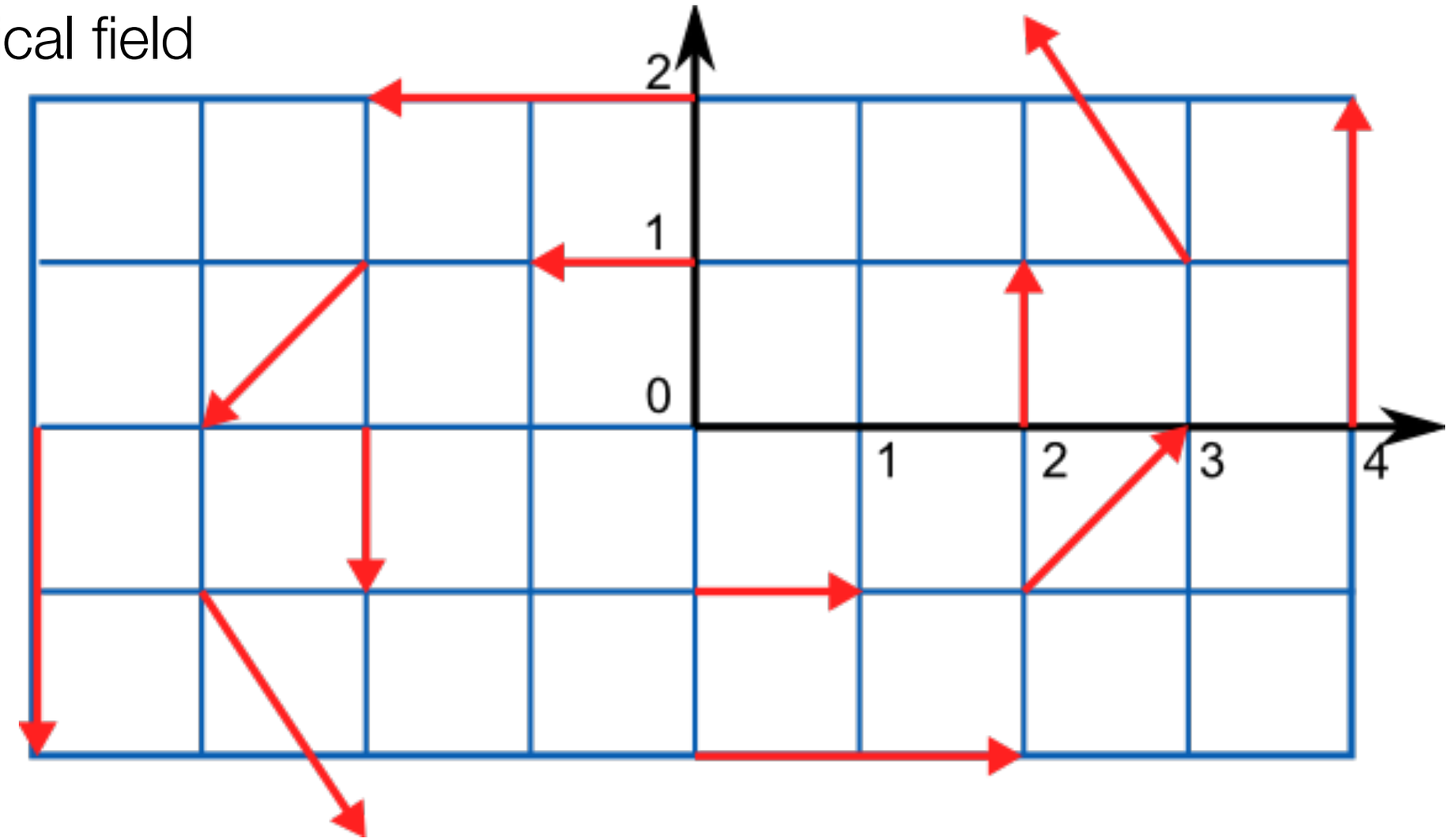
$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \int \vec{v}(\mathbf{x}(t))dt$$

Euler method (simplest numerical scheme)

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \vec{v}(\mathbf{x}(t))\Delta t$$

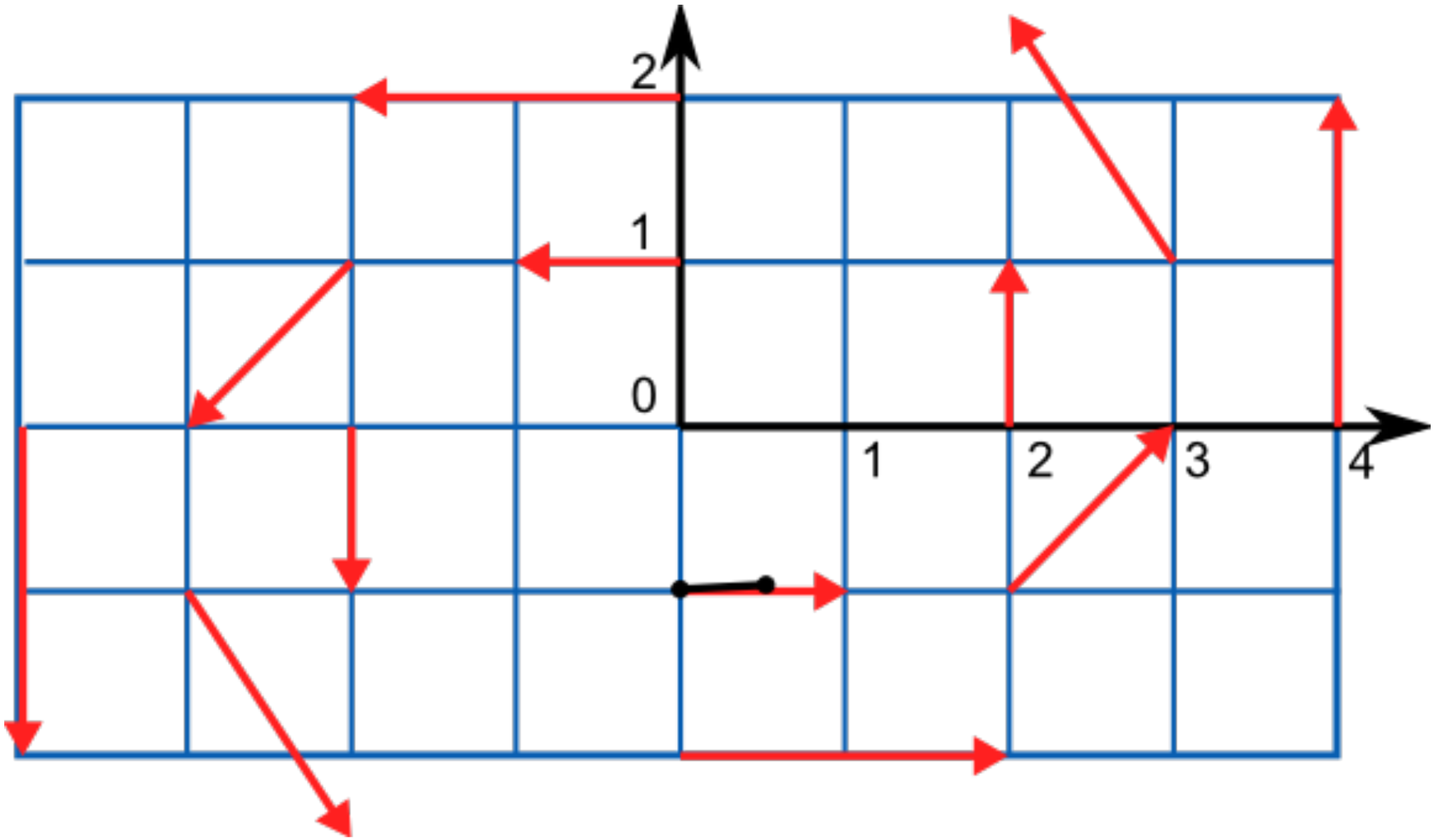
Euler integration example

- Model data $$\vec{v}(x, y) = \begin{pmatrix} -y \\ x/2 \end{pmatrix}$$
- Some sample vectors in red
- Elliptical field



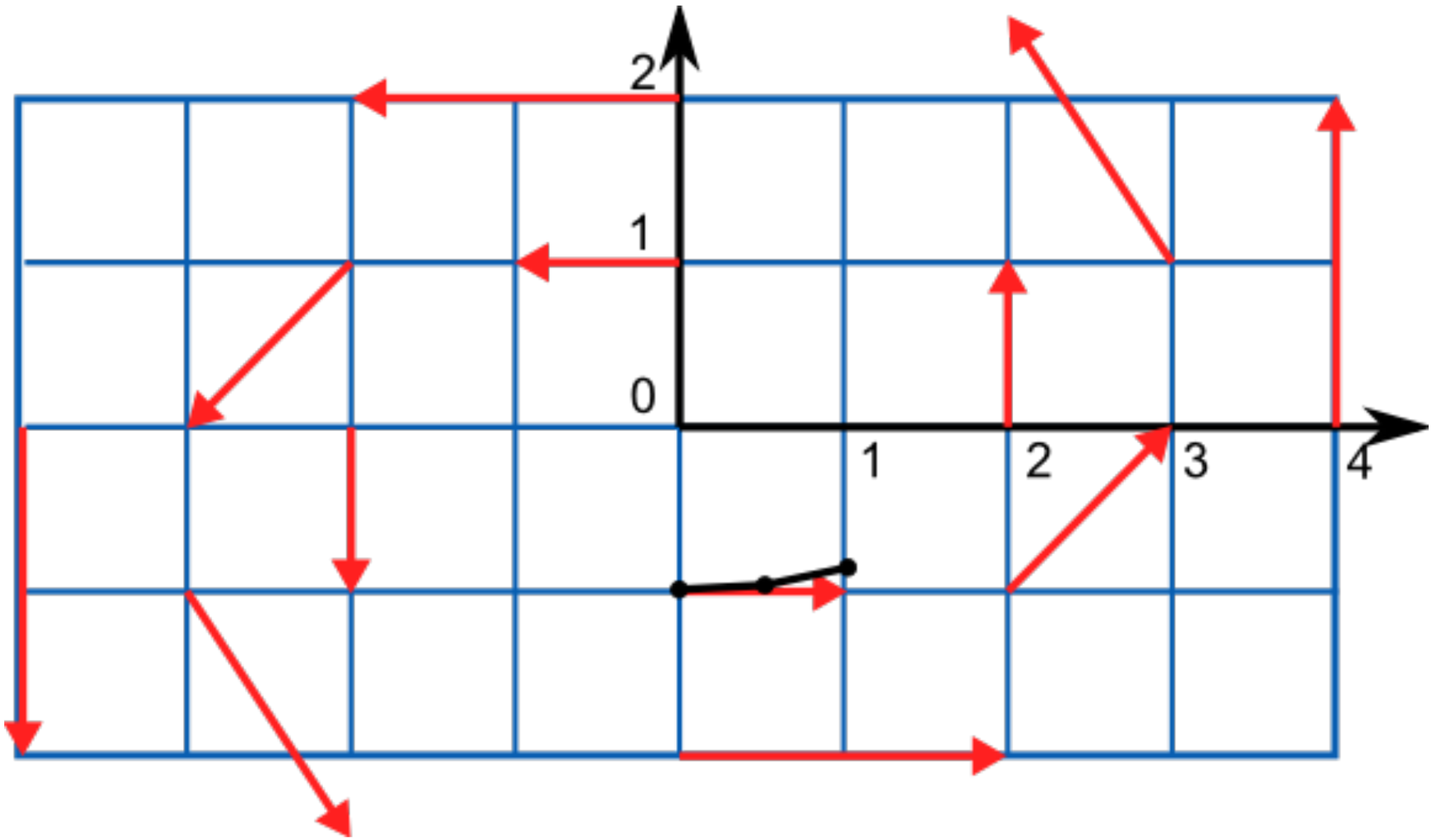
Euler integration example

$$\mathbf{x}(0) = \begin{pmatrix} 0 \\ -1 \end{pmatrix} \qquad \Delta t = \frac{1}{2}$$



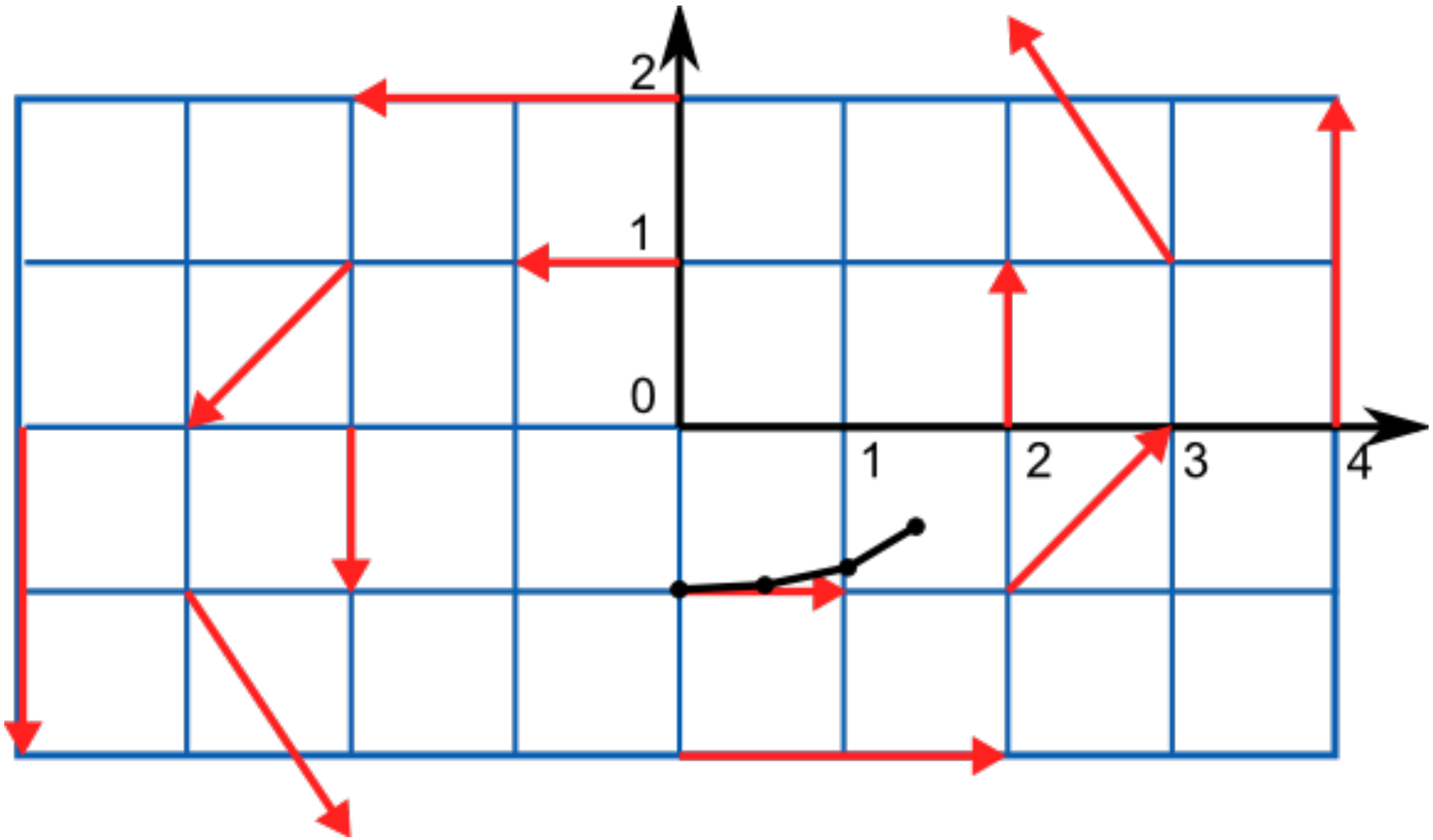
$$\mathbf{x}(1) = \mathbf{x}(0) + \vec{v}(\mathbf{x}(0))\Delta t = \begin{pmatrix} 0 \\ -1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} \frac{1}{2} = \begin{pmatrix} 0.5 \\ -1 \end{pmatrix}$$

Euler integration example

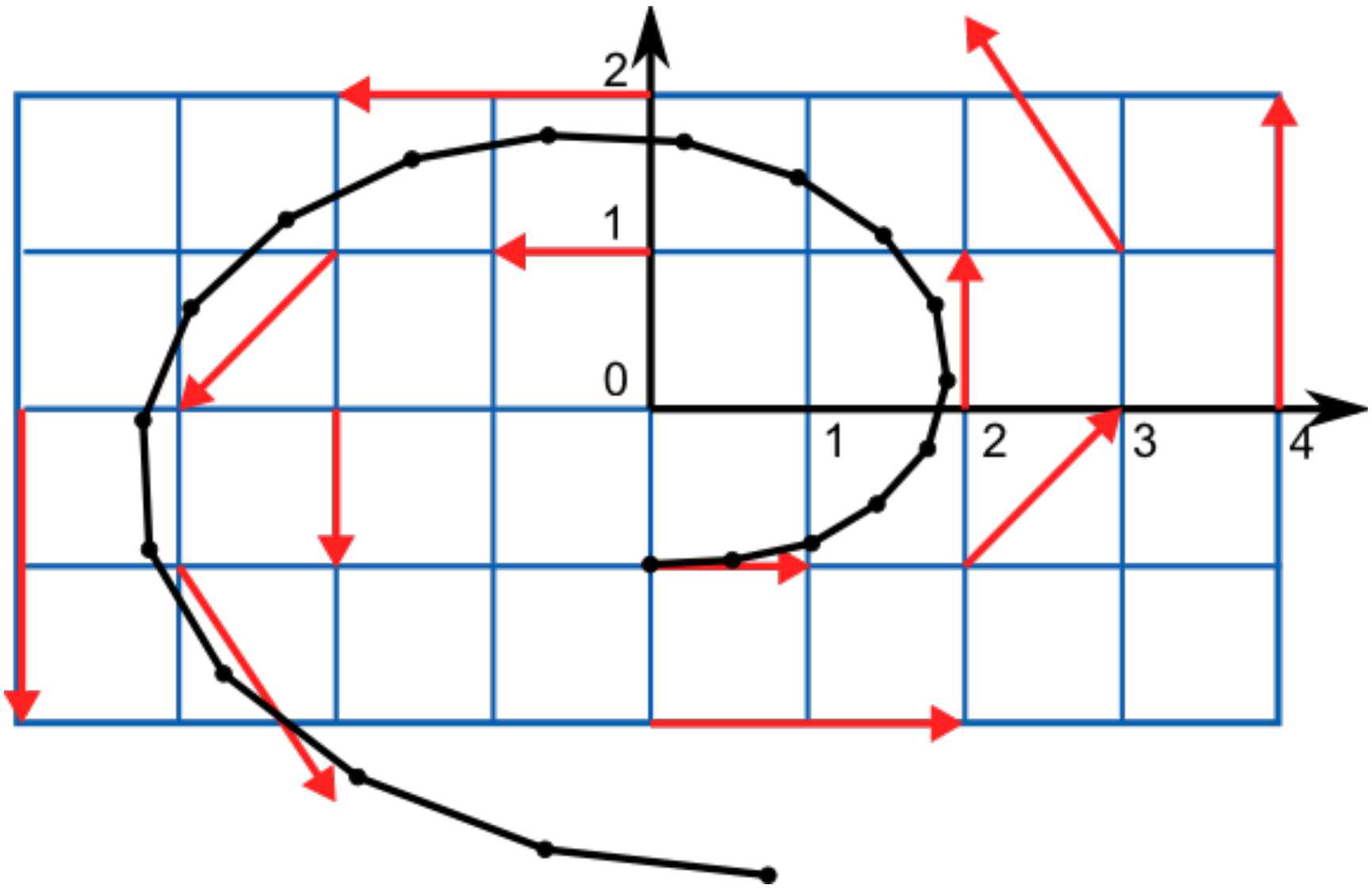


$$\mathbf{x}(2) = \mathbf{x}(1) + \vec{v}(\mathbf{x}(1))\Delta t = \begin{pmatrix} 0.5 \\ -1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0.25 \end{pmatrix} \frac{1}{2} = \begin{pmatrix} 1 \\ -0.875 \end{pmatrix}$$

Euler integration example

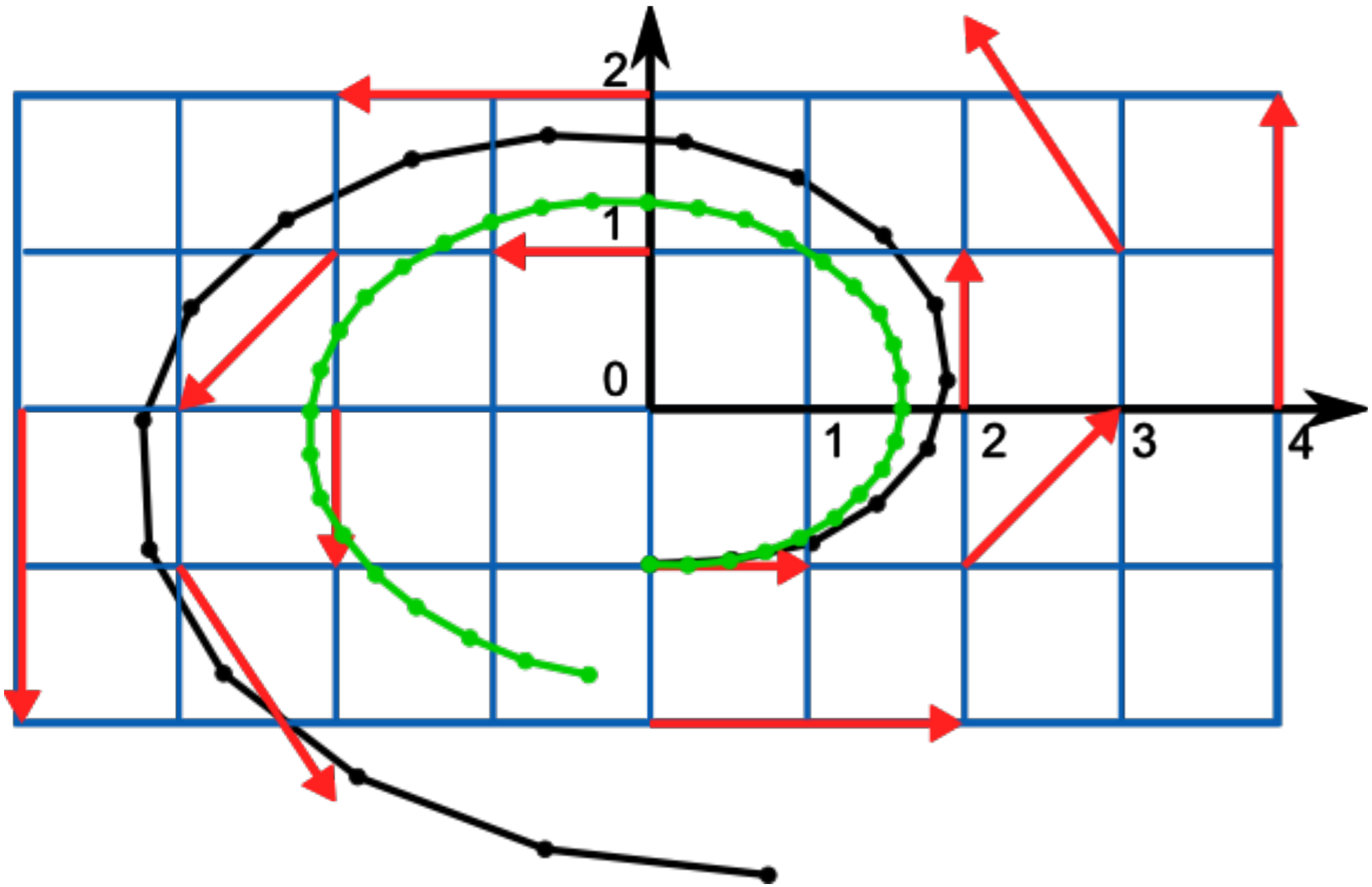


Euler integration example



Euler integration example

Green line: smaller step size $\Delta t = \frac{1}{4}$



Runge-Kutta integration (second order)

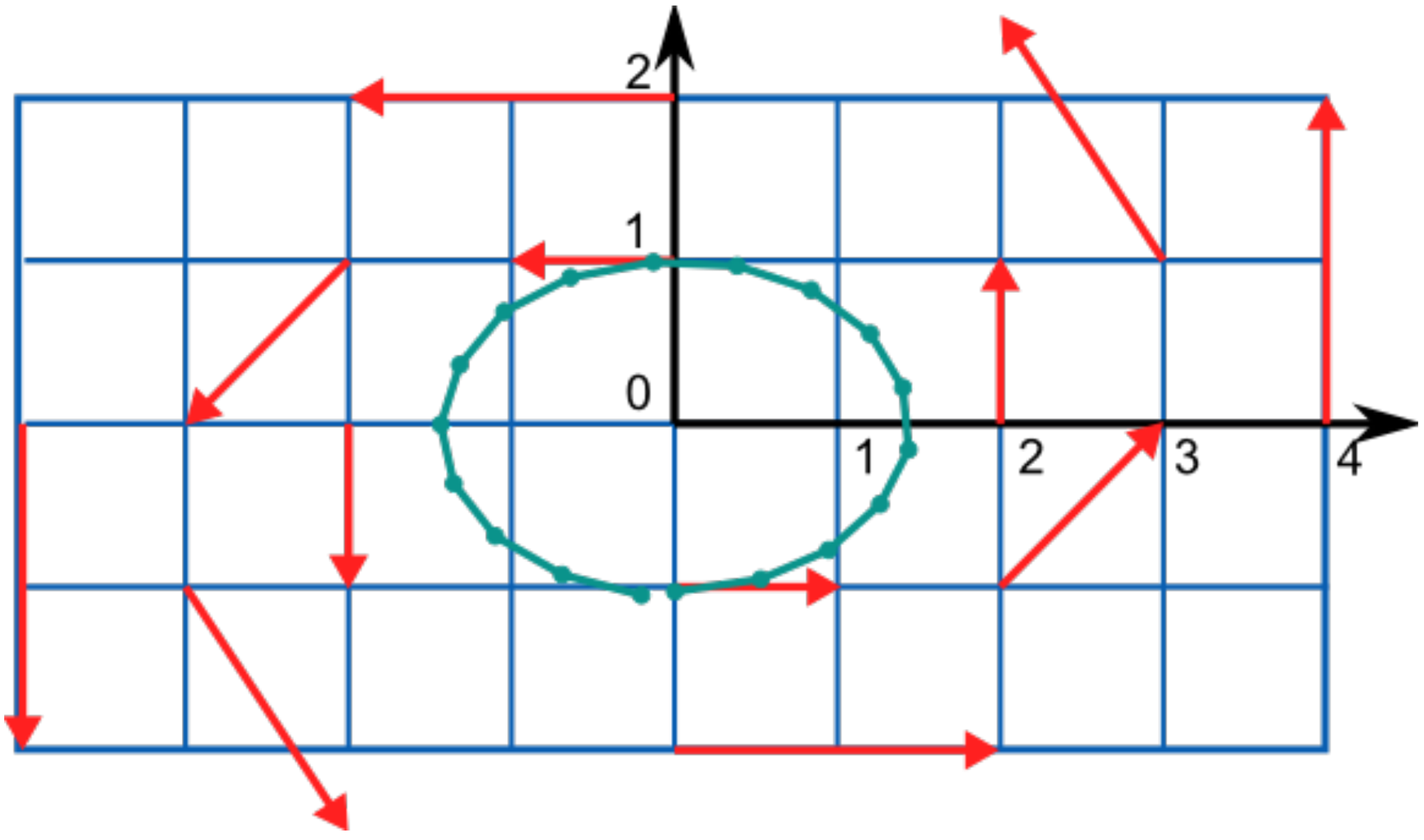
- Predictor-corrector approach

$$\mathbf{x}^*(t+\Delta t) = \mathbf{x}(t) + \vec{v}(\mathbf{x}(t))\Delta t \qquad \text{predictor}$$

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \frac{\vec{v}(\mathbf{x}(t)) + \vec{v}(\mathbf{x}^*(t+\Delta t))}{2}\Delta t \qquad \text{corrector}$$

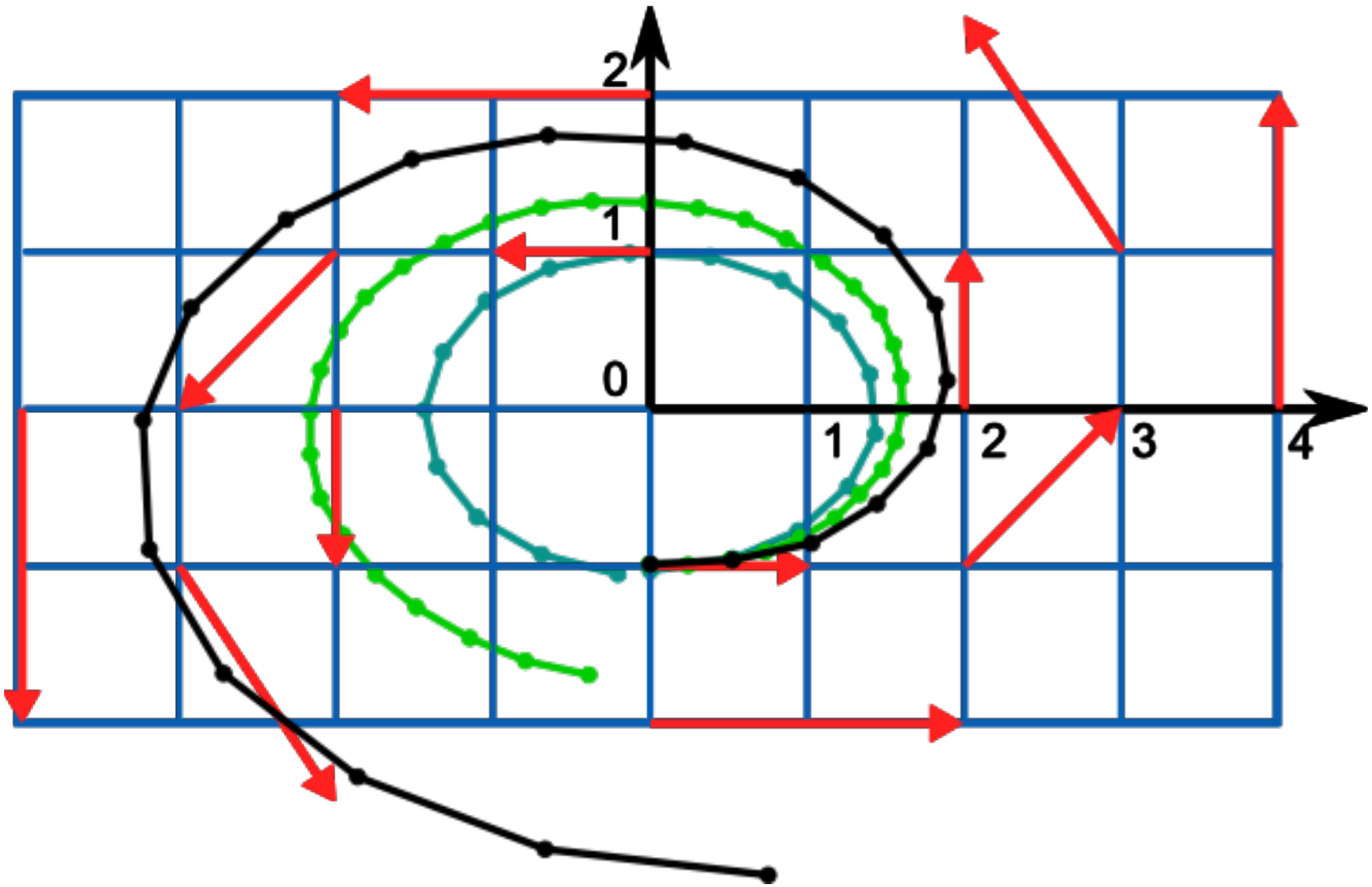
Runge-Kutta example

Runge-Kutta $\Delta t = \frac{1}{2}$ $\Delta t = \frac{1}{4}$ $\Delta t = \frac{1}{2}$



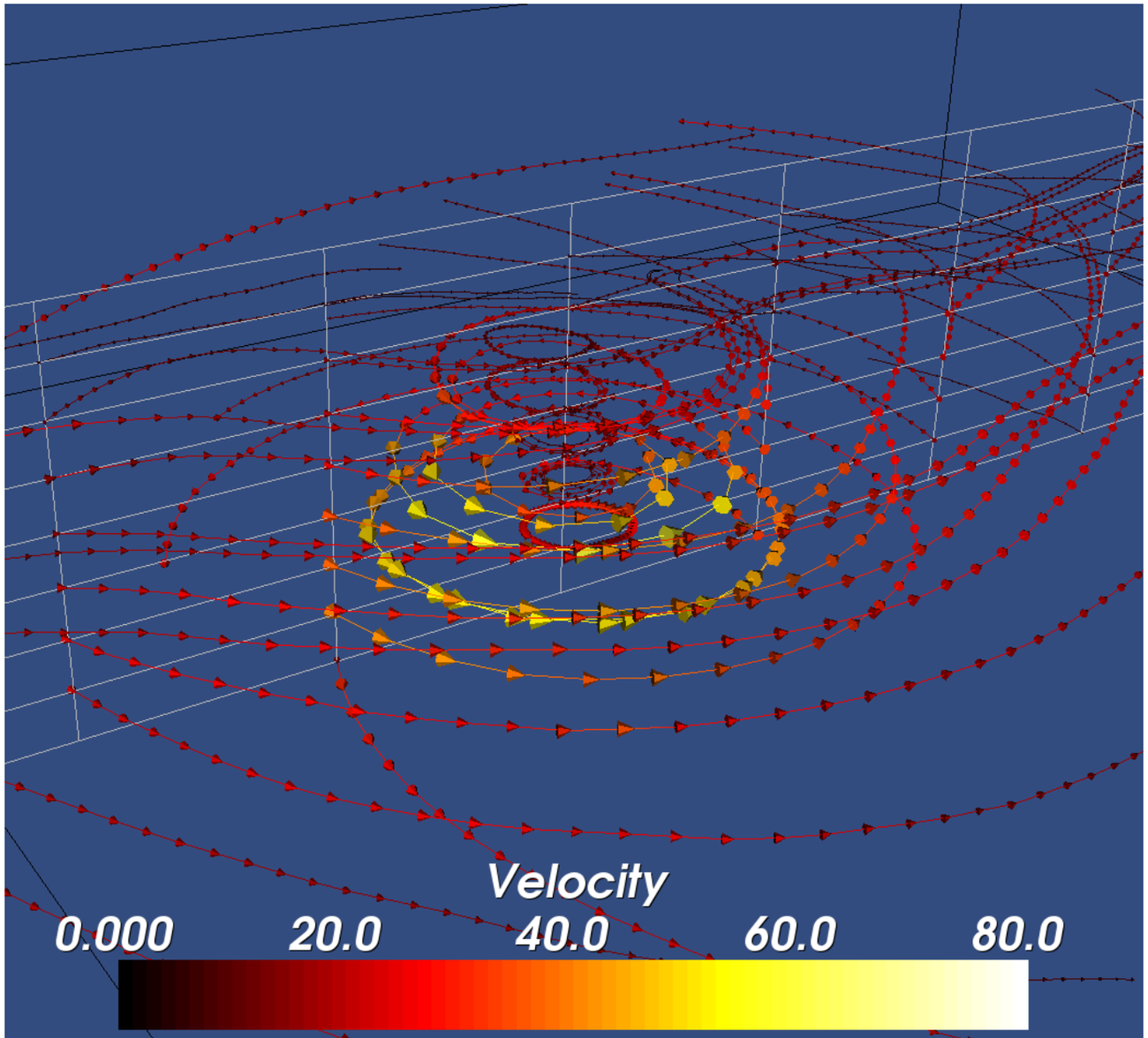
Runge-Kutta example

Runge-Kutta $\Delta t = \frac{1}{2}$ Euler $\Delta t = \frac{1}{4}$ Euler $\Delta t = \frac{1}{2}$



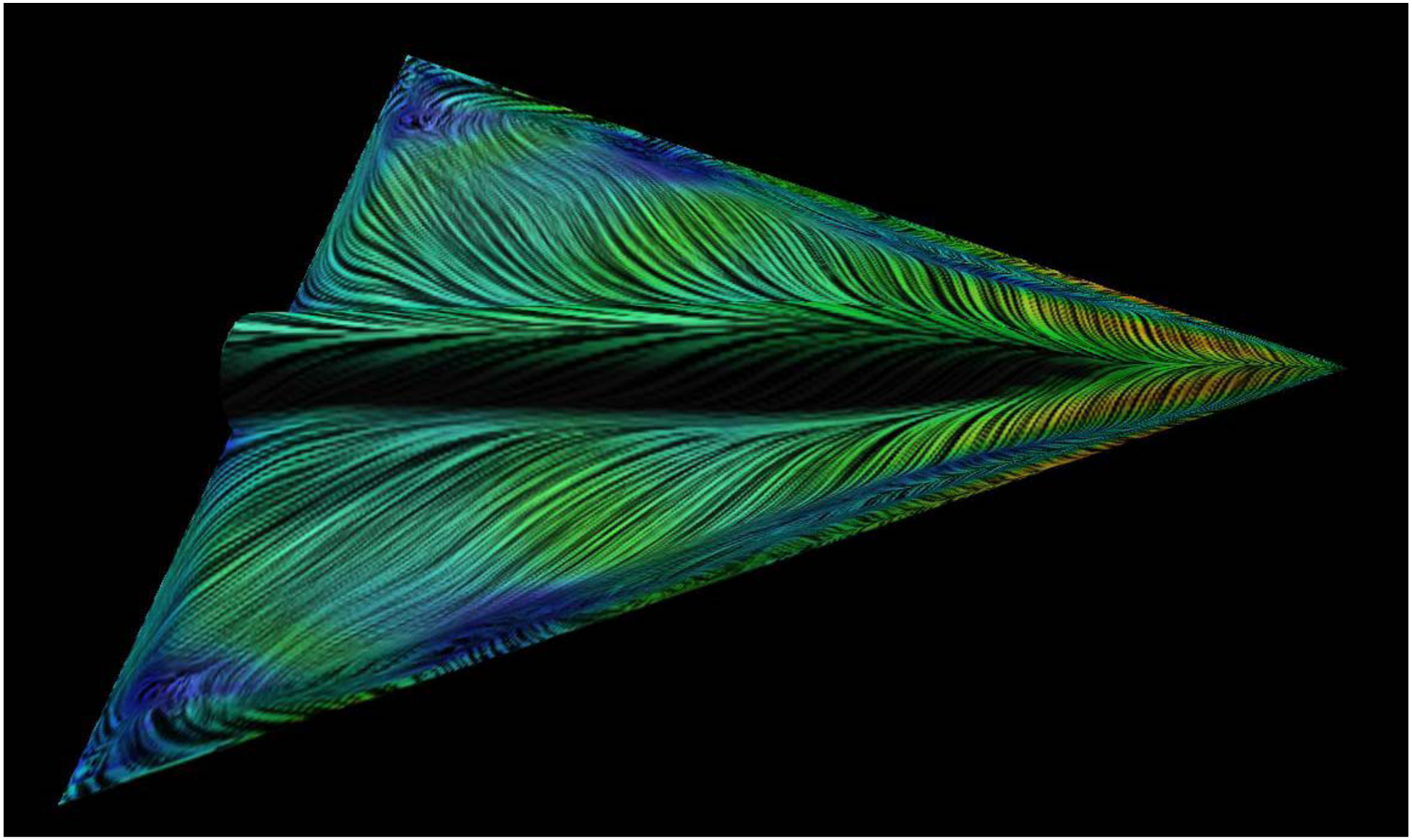
Hurricane Isabel demo

- Simulation of hurricane Isabel
- Grid dimensions 250 x 250 x 50
- Physical scale 2138 km x 2004 km x19.8 km



Texture-based visualization

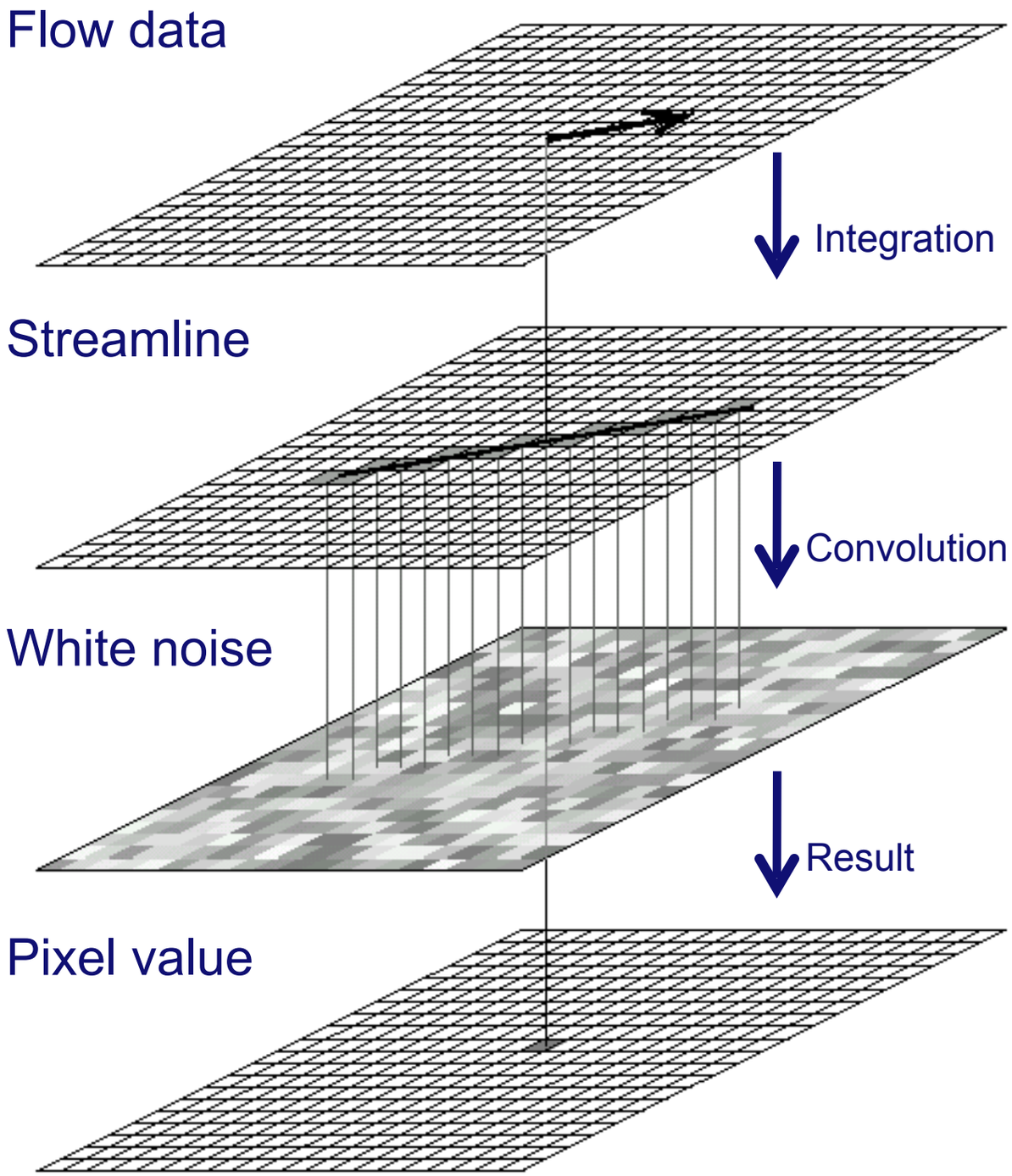
- Goal: generate overview image of the flow
- Approach: blur a noise texture locally according to the vector field



LIC: approach

Line integral convolution (LIC)

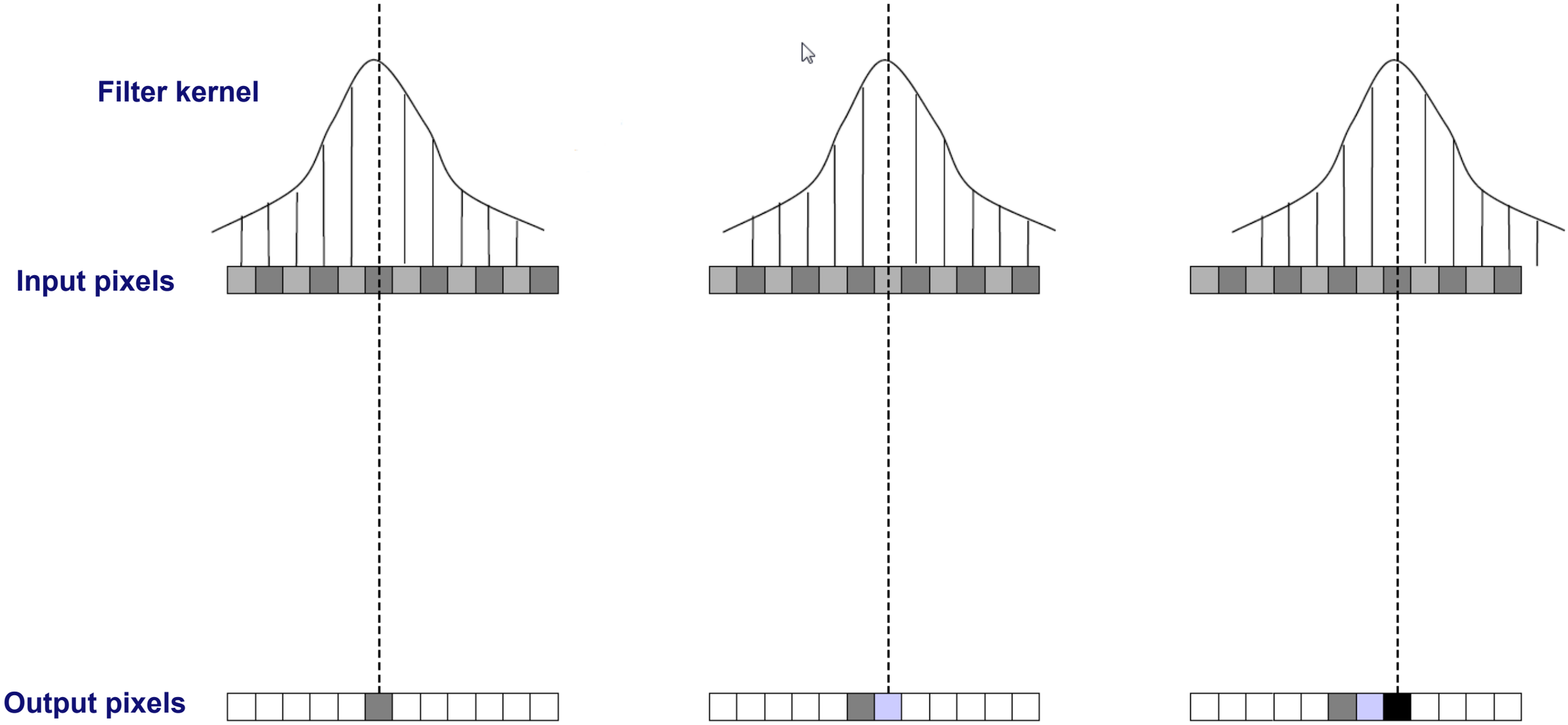
- Integrate streamline from point in forward and backward direction over some length L
- Compute weighted sum of noise texture values at pixel locations of streamline



Cabral et al. (1993), Imaging vector fields using line integral convolution. *Proc. SIGGRAPH'93*, pp. 263-270.

Convolution

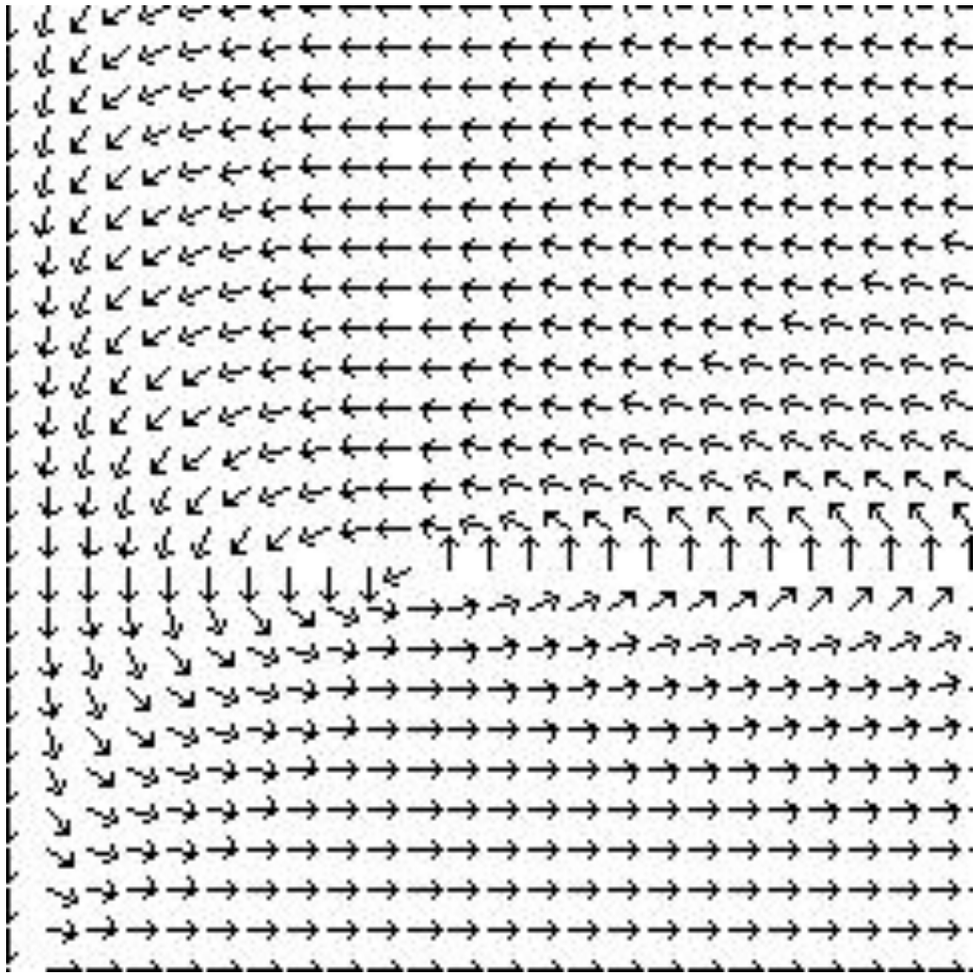
- Output pixel p’ computed as weighted sum of pixel neighborhood of input pixel p

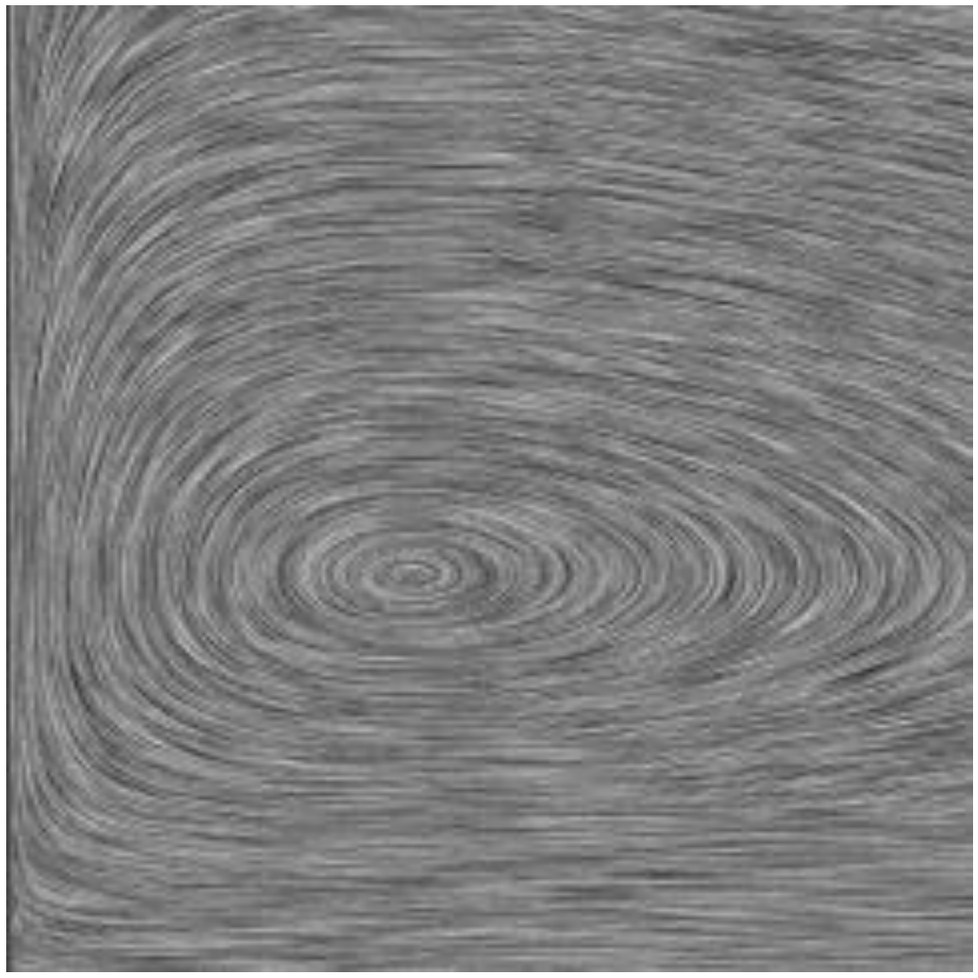


Cabral et al. (1993), Imaging vector fields using line integral convolution. *Proc. SIGGRAPH’93*, pp. 263-270.

LIC: why it works

- Along streamline: little variation in pixel values
- Neighboring streamlines: strong variation

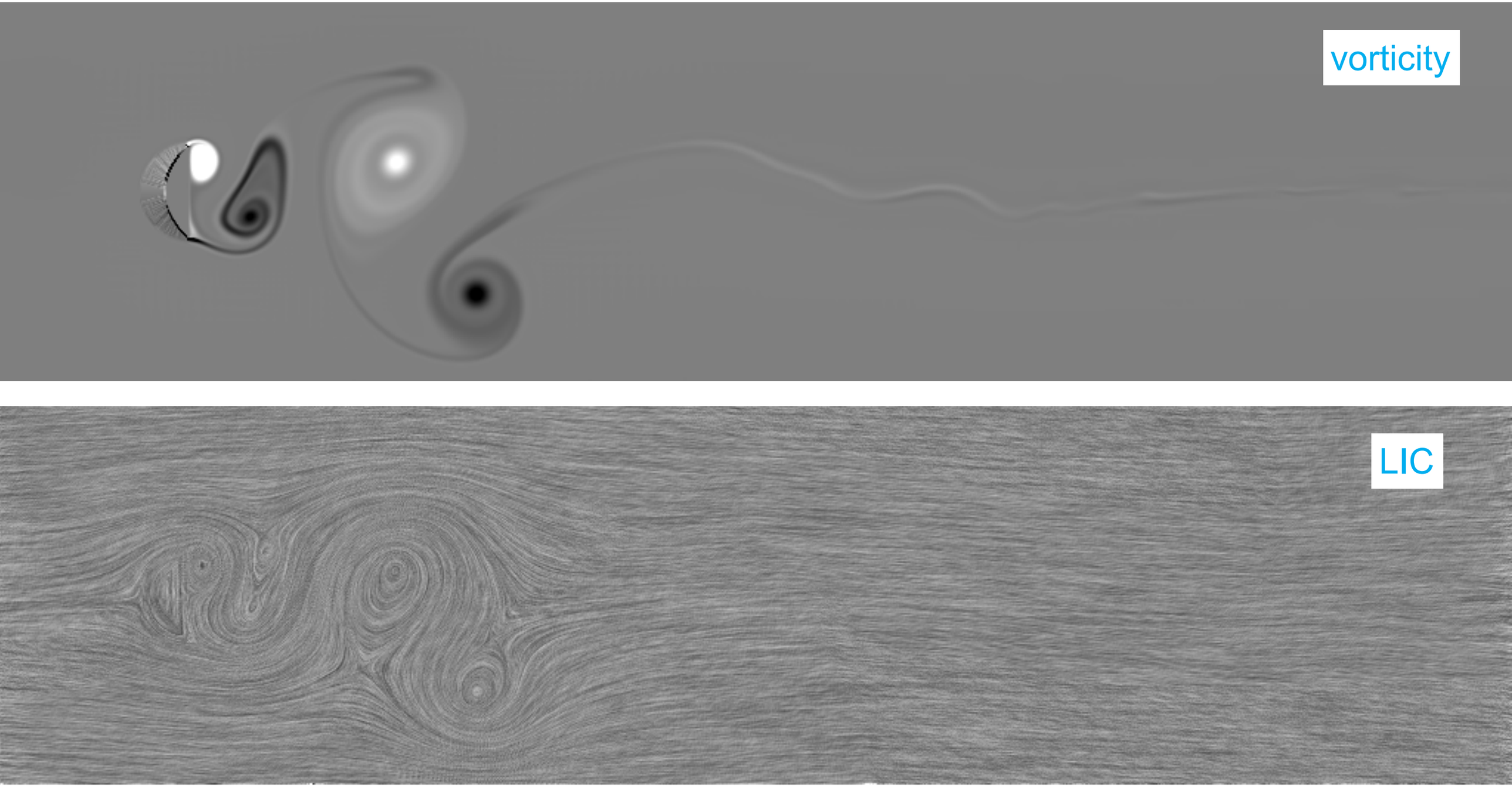




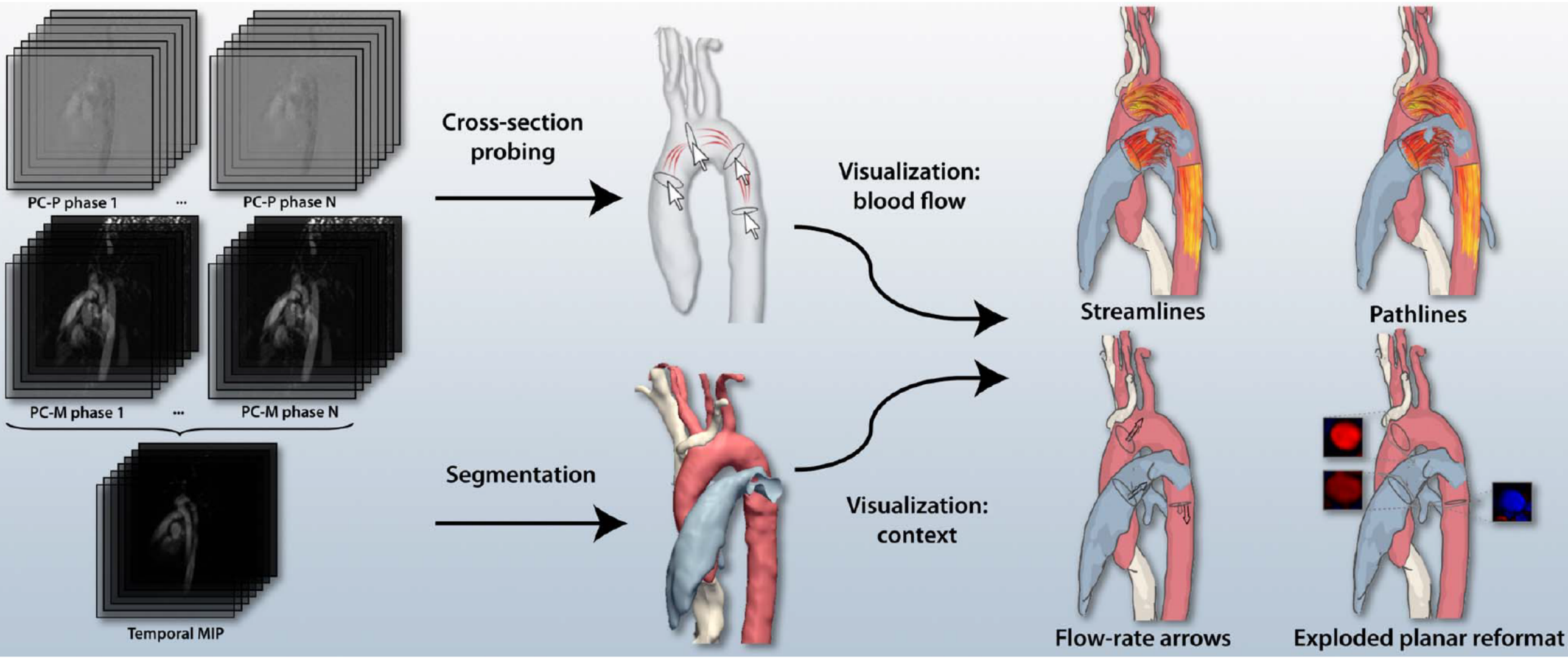
Cabral et al. (1993), Imaging vector fields using line integral convolution. *Proc. SIGGRAPH'93*, pp. 263-270.

LIC: example

- Flow around half-cylinder

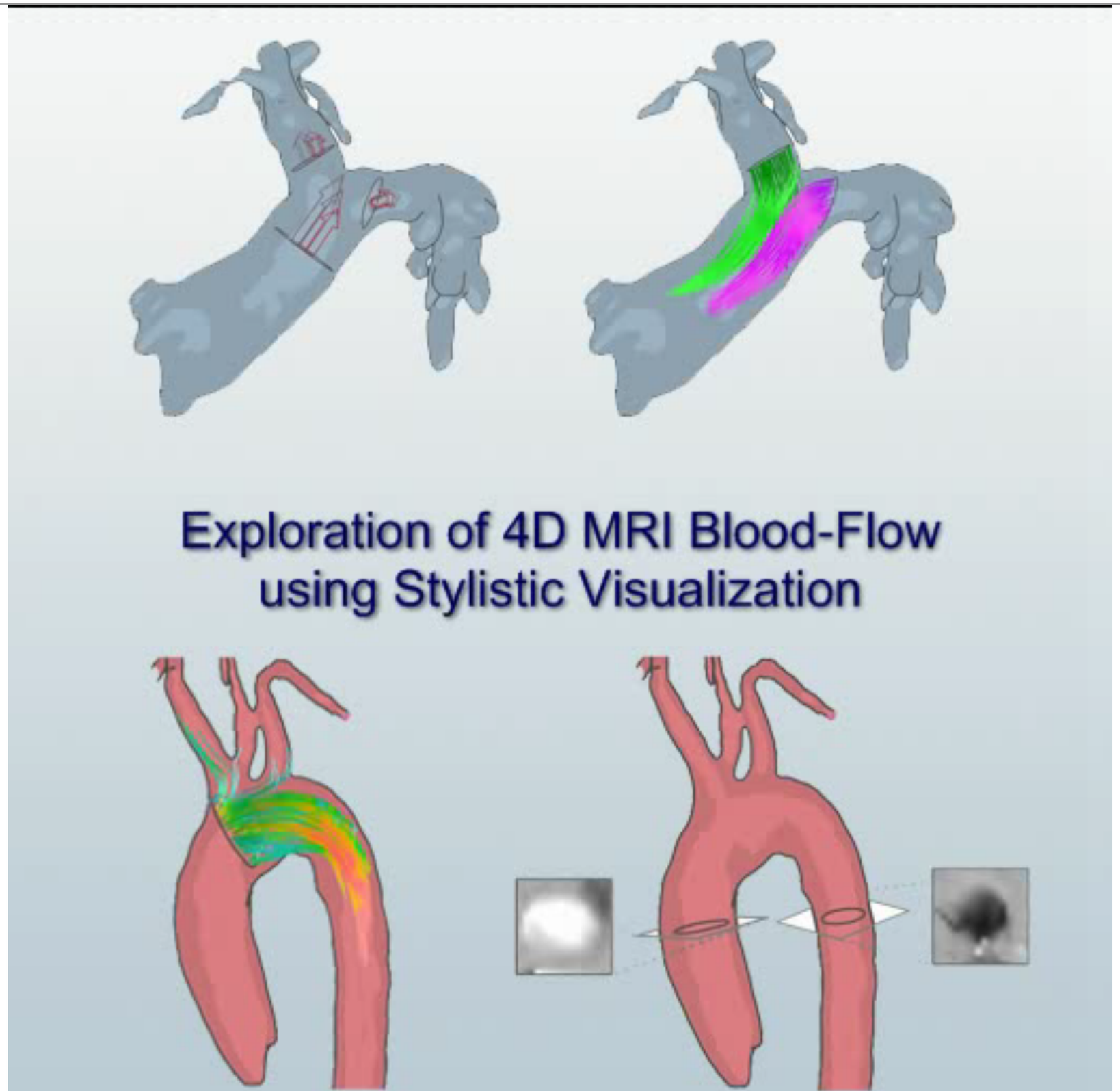


4D blood flow visualization

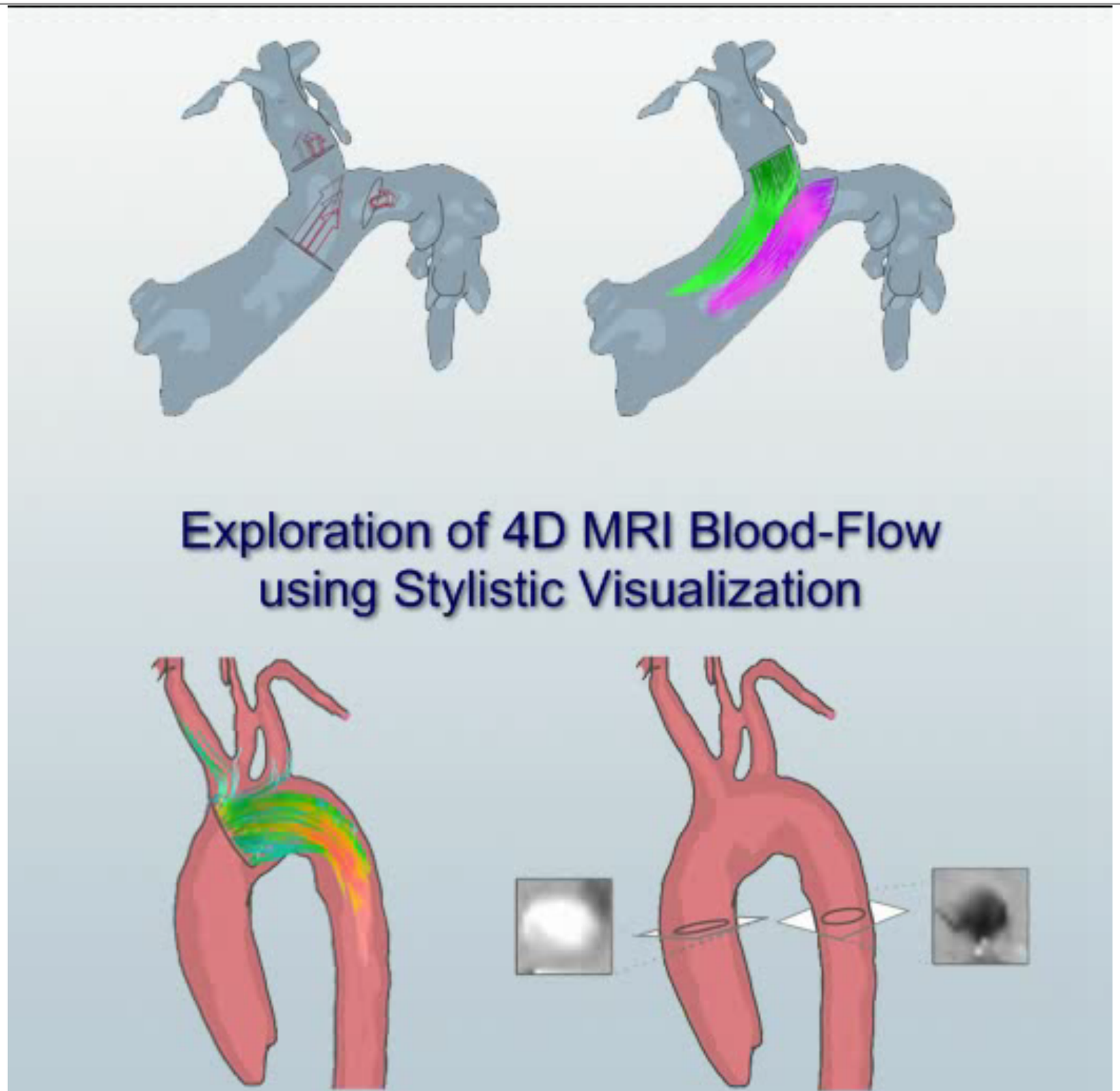


Van Pelt et al., **Exploration of 4D MRI Blood-Flow Using Stylistic Visualization**, IEEE Trans. Visualization and Computer Graphics 16(6):1339-1347, 2010.

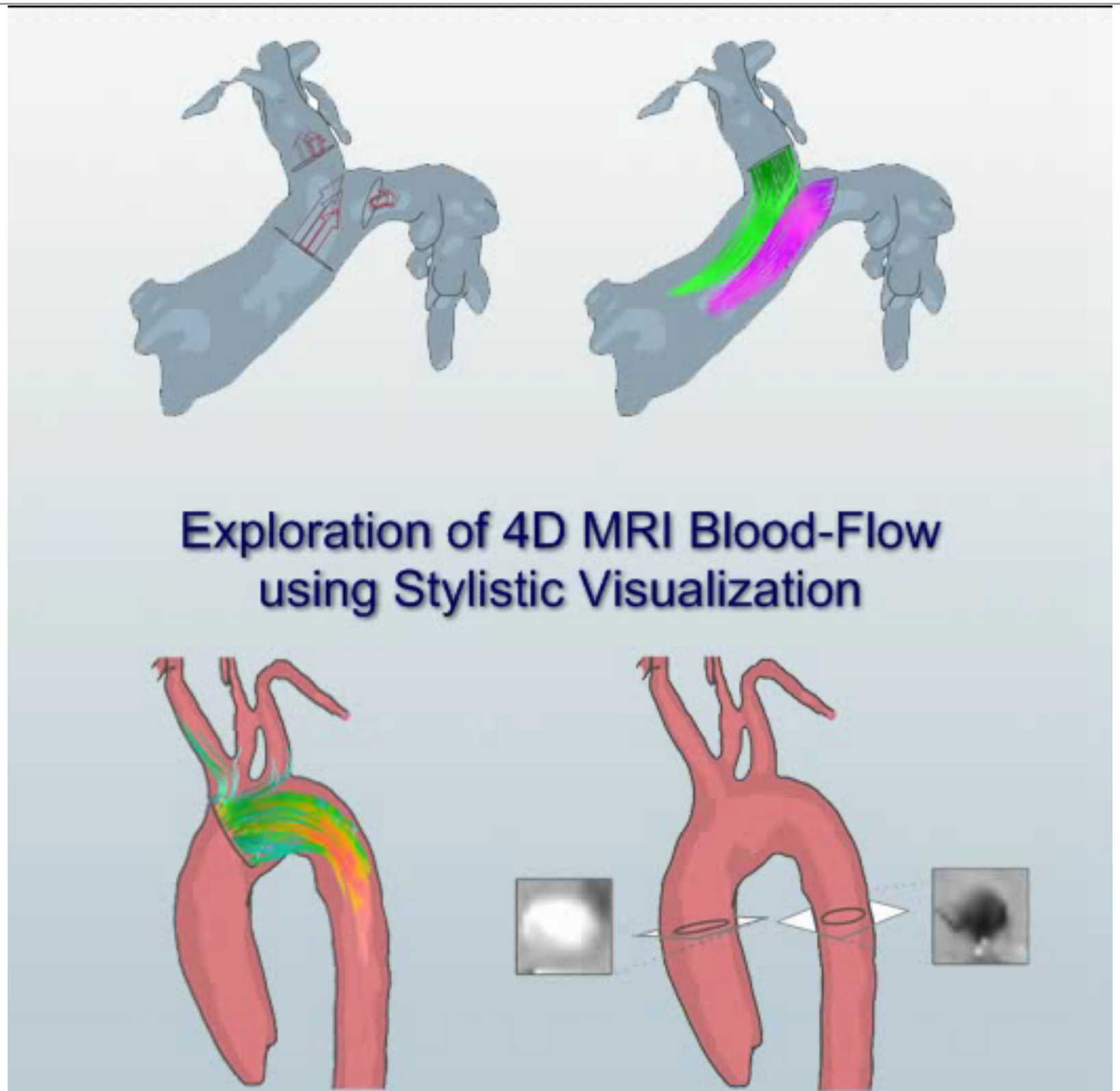
4D blood flow visualization example



4D blood flow visualization example



4D blood flow visualization example



Diffusion tensor imaging

- Measure diffusion of water by MRI
- Diffusion velocity and direction depend on tissue type
- Fluid-filled compartments show isotropic diffusion
- Neural fibers show anisotropic diffusion

DTI data

- Diffusion tensor

$$D = \begin{pmatrix} D_{xx} & D_{xy} & D_{xz} \\ D_{yx} & D_{yy} & D_{yz} \\ D_{zx} & D_{zy} & D_{zz} \end{pmatrix}$$

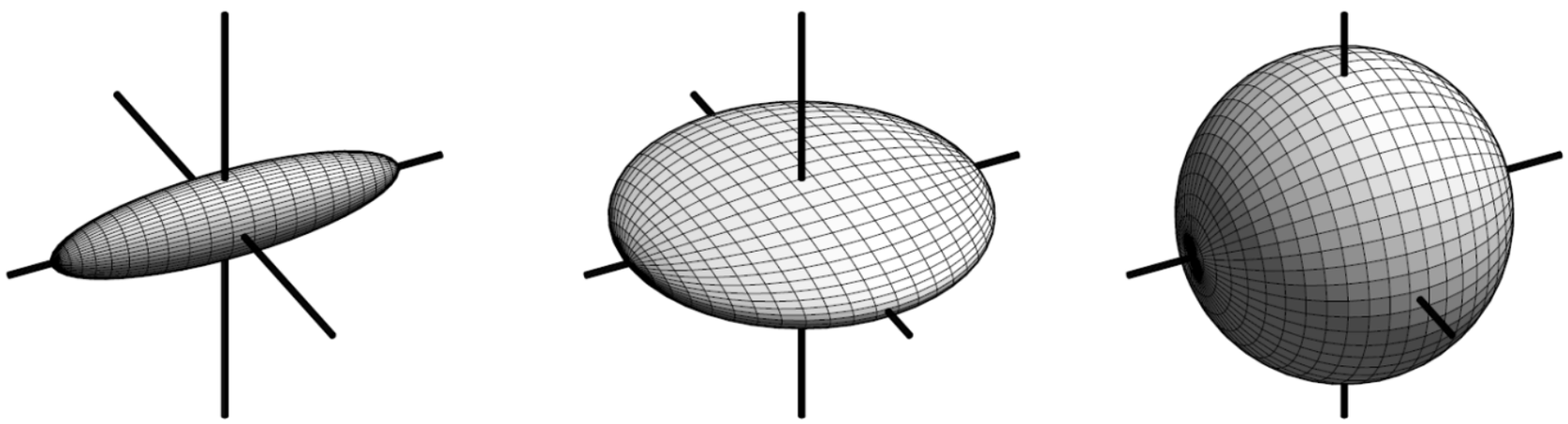
- Low spatial resolution (128x128 and 30-60 slices)
- Voxel spacing 2x2x2 mm
- Matrix elements are with respect to global coordinate system

Eigenanalysis

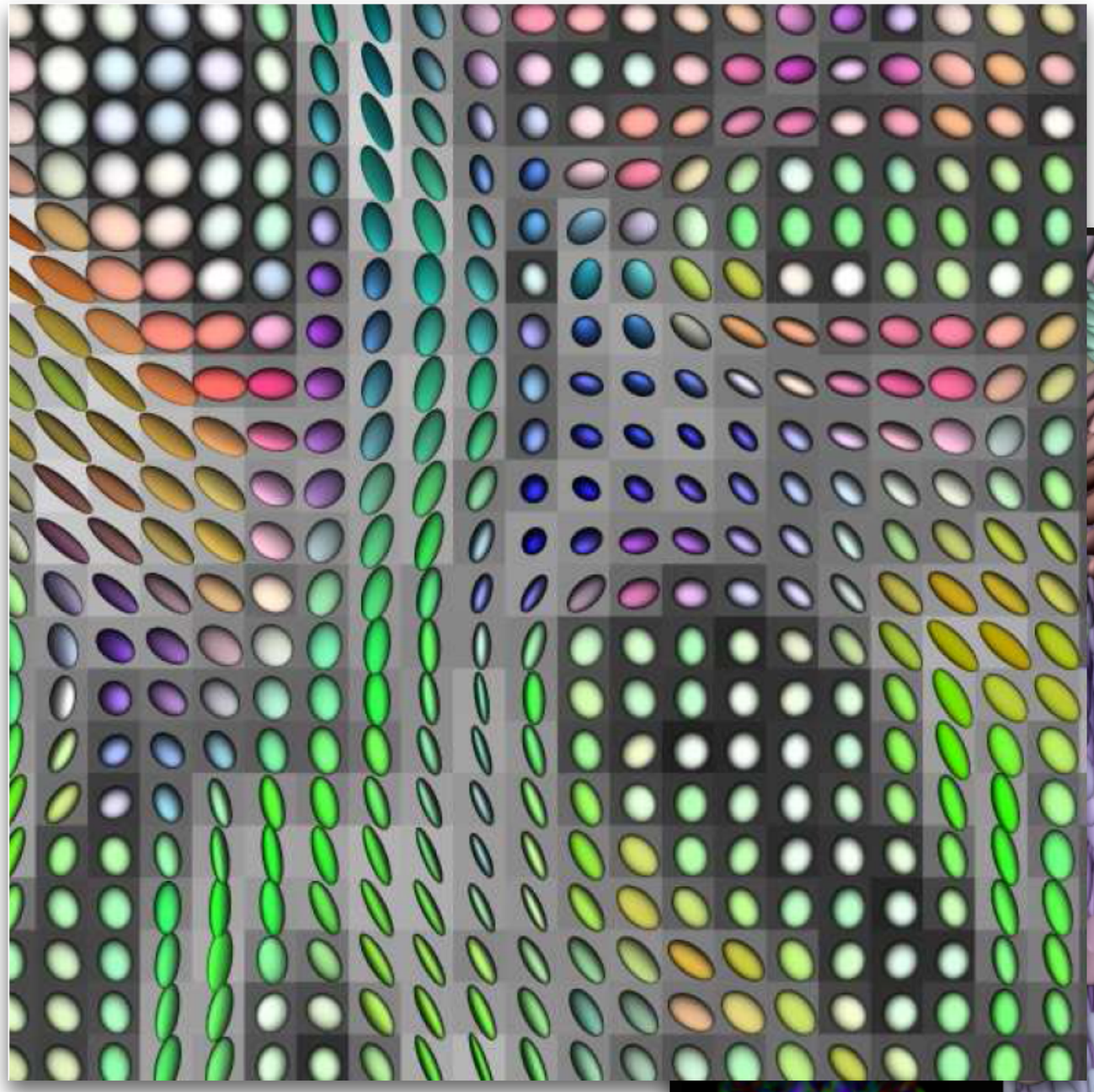
- Diagonal form representation of tensor

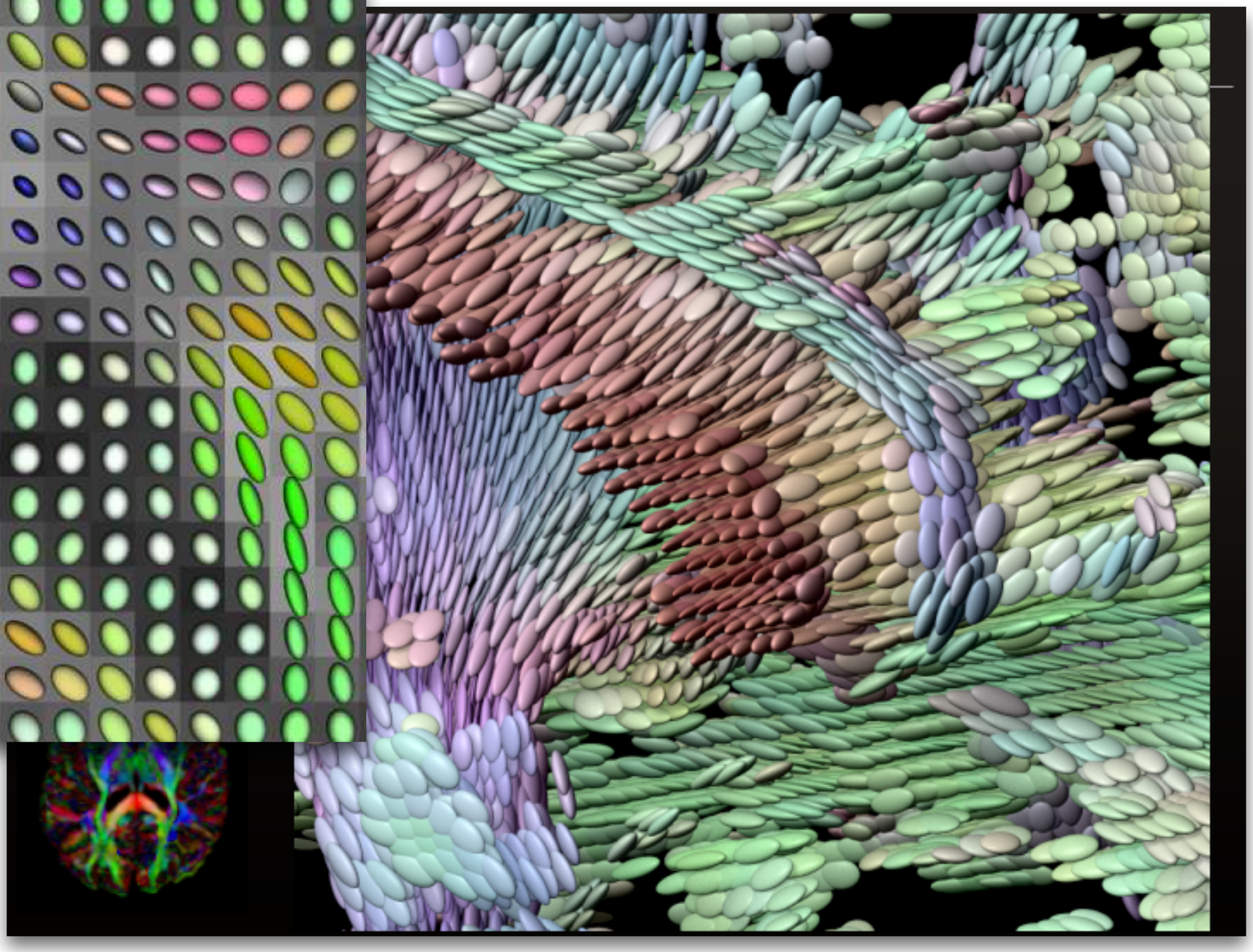
$$D = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\} \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}^T$$

- Geometric interpretation of the diffusion tensor
  - ellipsoid with axes aligned with the three eigenvectors
  - axis length proportional to eigenvalue



Tensor glyphs 2D and 3D





Anisotropy metrics

- Mean diffusivity

$$\lambda_{\text{mean}} = \frac{\lambda_1 + \lambda_2 + \lambda_3}{3}$$

- Fractional anisotropy

$$\text{FA} = \sqrt{\frac{3}{2}} \frac{\sqrt{\sum_{i=1}^{3} (\lambda_i - \lambda_{\text{mean}})^2}}{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}$$

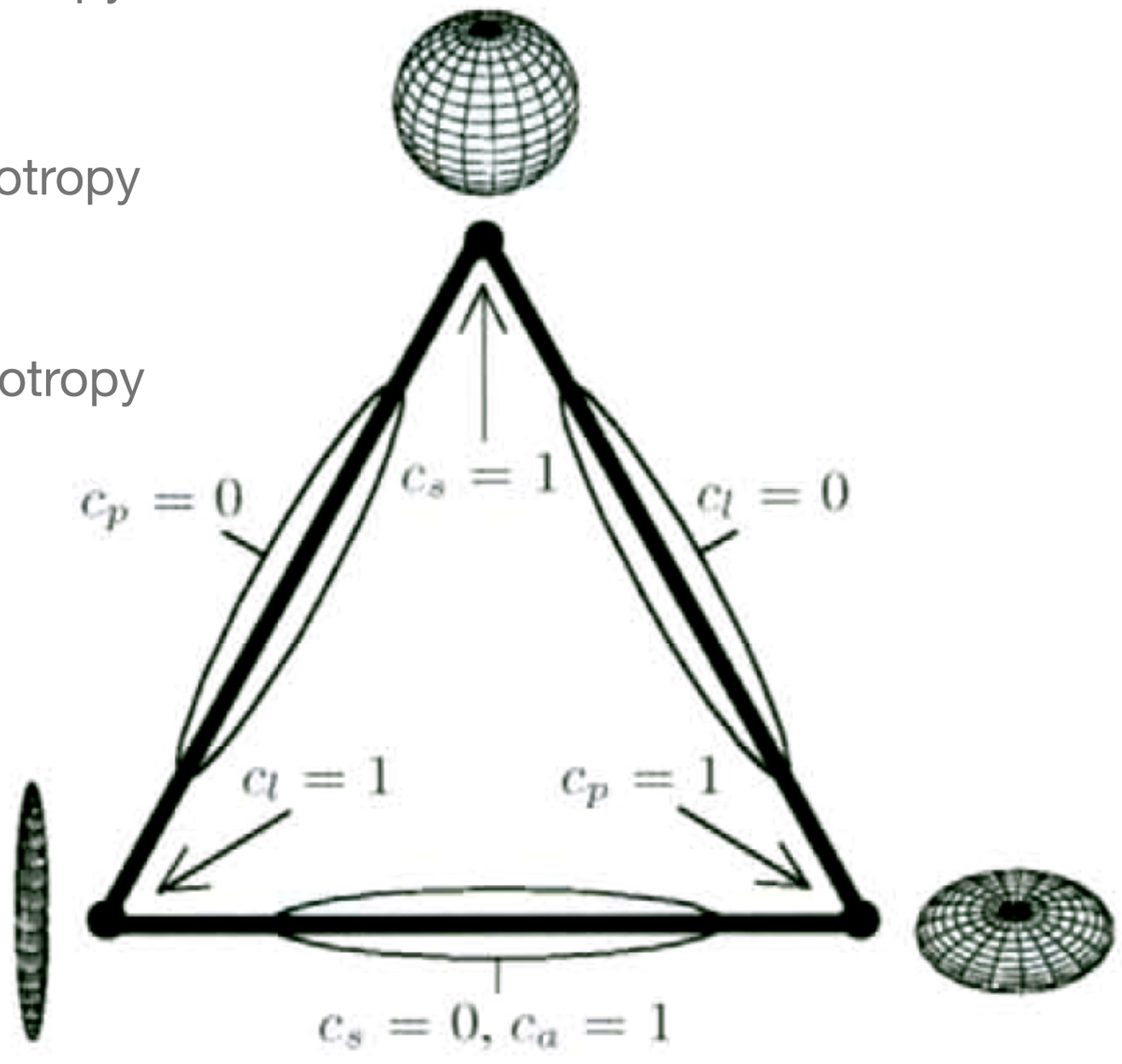
Anisotropy metrics

- Consider sorted eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$

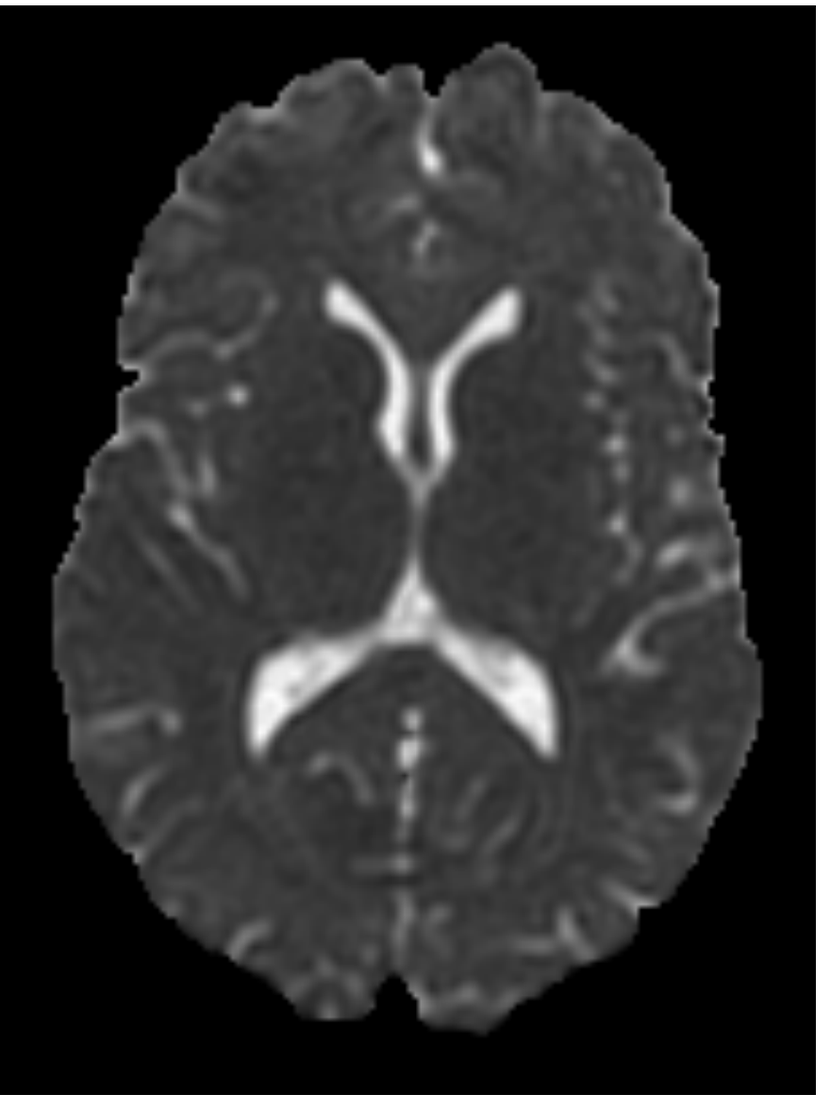
$$c_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3}$$ linear anisotropy

$$c_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3}$$ planar anisotropy

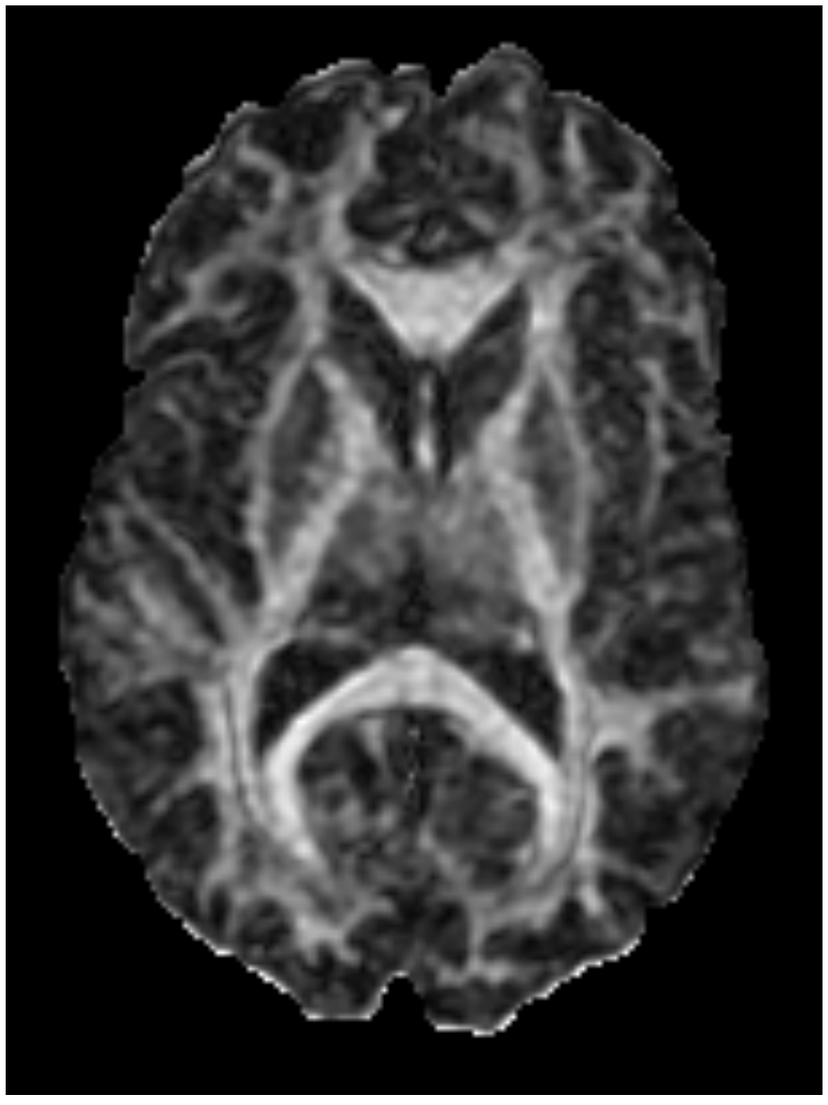
$$c_s = \frac{3\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3}$$ spherical isotropy



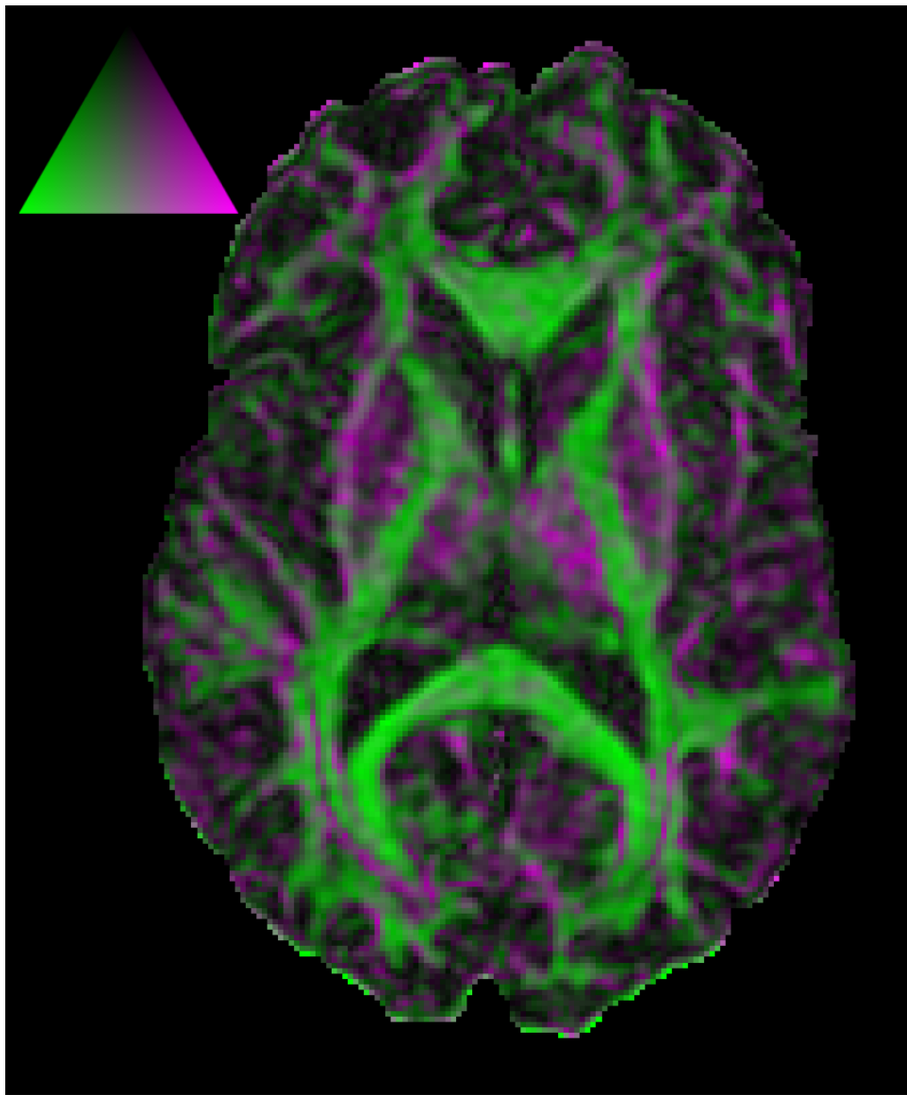
Anisotropy metrics



λ_{mean}



FA



c_l (green)

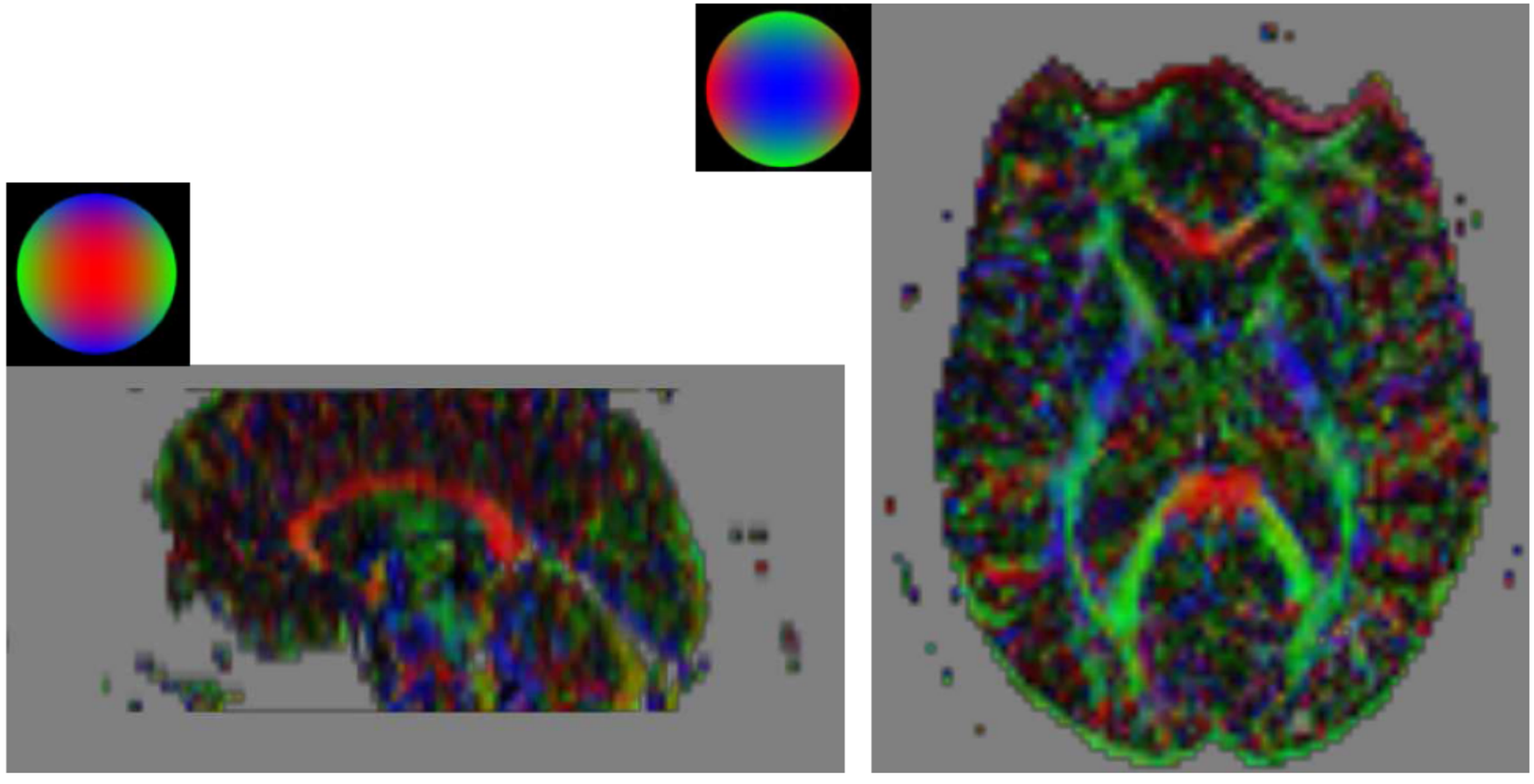
c_p (magenta)

Major eigenvector direction visualization

- Simplify tensor field to vector field
- Consider only main eigenvector $\mathbf{e}_1$
- Map components to RGB colors

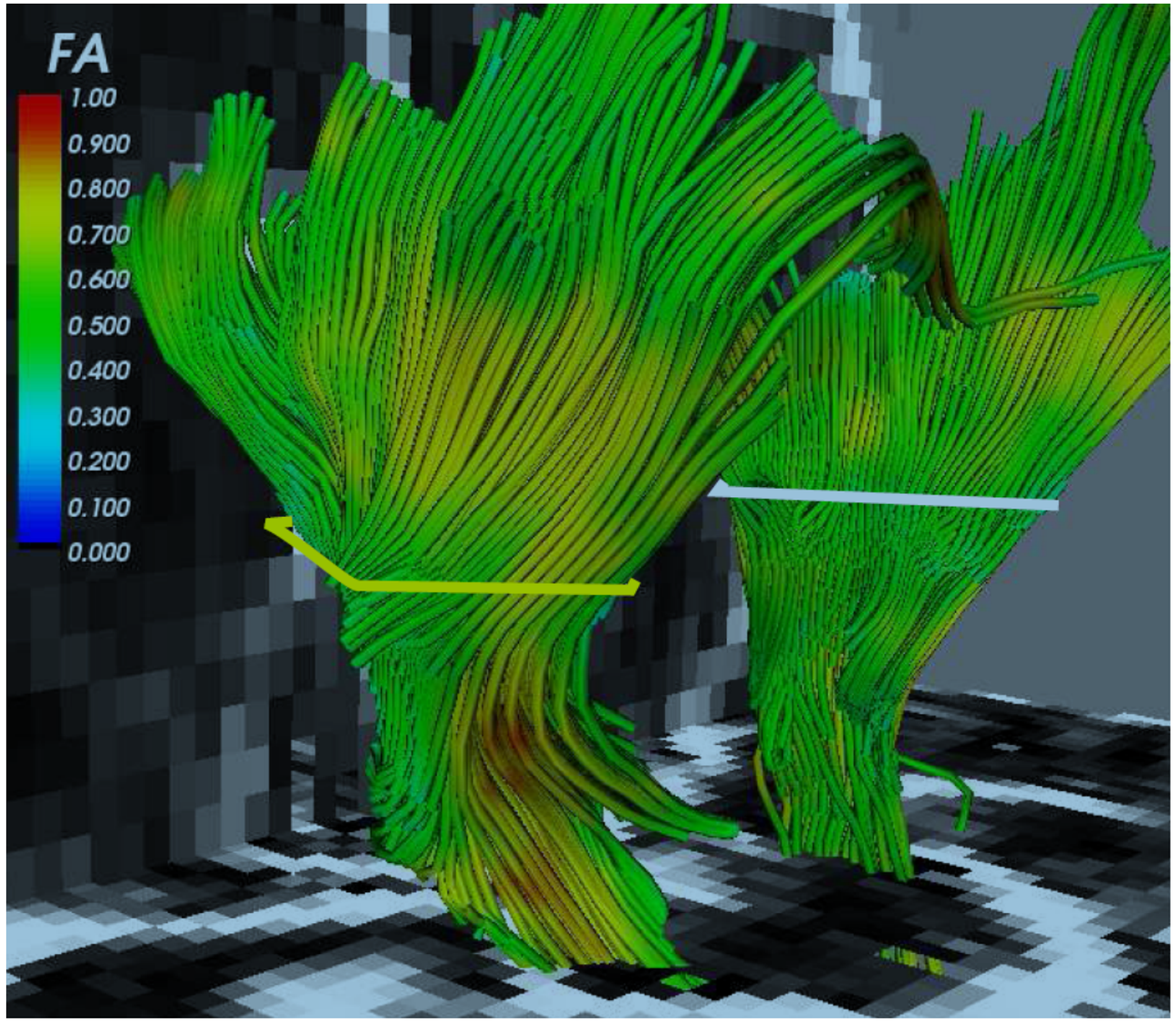
$$R = |\mathbf{e}_1 \cdot \mathbf{x}| \qquad G = |\mathbf{e}_1 \cdot \mathbf{y}| \qquad B = |\mathbf{e}_1 \cdot \mathbf{z}|$$

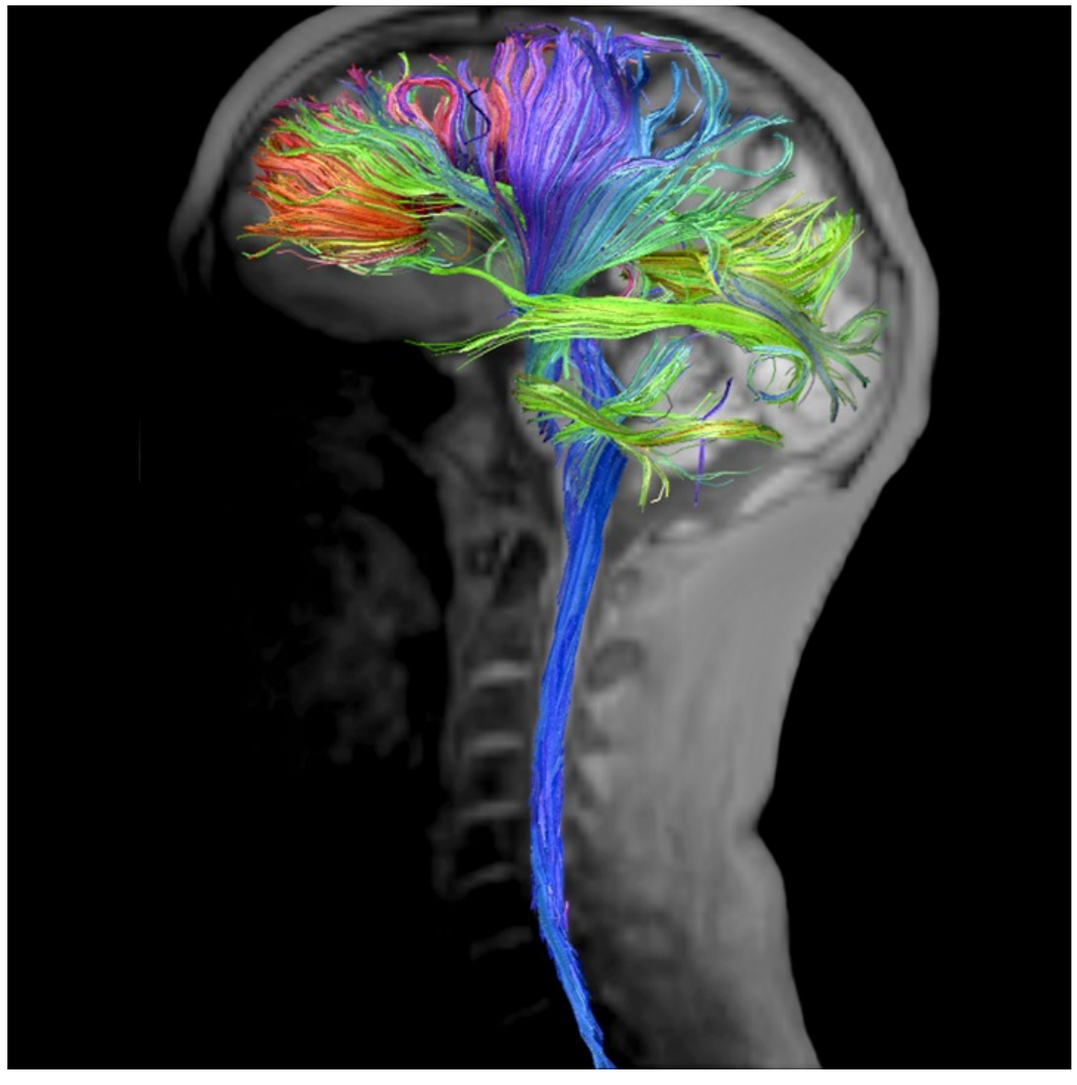
- Saturation weighted by anisotropy



Fiber tracking

- Trace streamlines in the $\boldsymbol{e}_1$ vector field
- Terminology: fiber tracking, tractography





Readings

Volume Rendering

Display of Surfaces from Volume Data

Marc Levoy, University of North Carolina

In this article we will explore the application of *volume rendering* techniques to the display of surfaces from sampled scalar functions of three spatial dimensions. It is not necessary to fit geometric primitives to the sampled data. Images are formed by directly shading each sample and projecting it onto the picture plane.

Surface-shading calculations are performed at every voxel with local gradient vectors serving as surface normals. In a separate step, surface classification operators are applied to compute a partial opacity for every voxel. We will look at operators that detect isovalue contour surfaces and region boundary surfaces. Independence of shading and classification calculations ensure an undistorted visualization of 3D shape. Nonbinary classification operators ensure that small or poorly defined features are not lost. The resulting colors and opacities are composited from back to front along viewing rays to form an image.

The technique is simple and fast, yet displays surfaces exhibiting smooth silhouettes and few other aliasing artifacts. We will also describe the use of selective blurring and supersampling to further improve image quality. Examples from two applications are given: molecular graphics and medical imaging.

Visualization of scientific computations is a rapidly growing application of computer graphics. A large subset of these applications involves sampled functions of three spatial dimensions, also known as volume data. Surfaces are commonly used to visualize volume data because they succinctly present the 3D configuration of complicated objects. In this article we explore the use of isovalue contour surfaces to visualize electron density maps for molecular graphics, and the use of region boundary surfaces to visualize computed tomography (CT) data for medical imaging.

The currently dominant techniques for displaying surfaces from volu
detector to the san
tives to the detect
primitives using
algorithms. The te
mainly in the cho
which they are de

In the medical in
to apply thresholdi
binary representa
1-voxels as opaque
If this binary repr
local gray scale g
improvements in s
Alternatively, edg
slice to yield a se
interest. Then a m
connecting the co
scale of voxels ap

May 1988

0272-1716/88/0500-0029$01.00 1988 IEEE

Multidimensional Transfer Functions for Interactive Volume Rendering

Joe Kniss, *Student Member*, *IEEE Computer Society*,
Gordon Kindlmann, *Student Member*, *IEEE Computer Society*, and
Charles Hansen, *Member*, *IEEE Computer Society*

Abstract—Most direct volume renderings produced today employ one-dimensional transfer functions which assign color and opacity to the volume based solely on the single scalar quantity which comprises the data set. Though they have not received widespread attention, multidimensional transfer functions are a very effective way to extract materials and their boundaries for both scalar and multivariate data. However, identifying good transfer functions is difficult enough in one dimension, let alone two or three dimensions. This paper demonstrates an important class of three-dimensional transfer functions for scalar data, and describes the application of multidimensional transfer functions to multivariate data. We present a set of direct manipulation widgets that make specifying such transfer functions intuitive and convenient. We also describe how to use modern graphics hardware to both interactively render with multidimensional transfer functions and to provide interactive shadows for volumes. The transfer functions, widgets, and hardware combine to form a powerful system for interactive volume exploration.

Index Terms—Volume visualization, direct volume rendering, multidimensional transfer functions, direct manipulation widgets, graphics hardware.

1 Introduction

Direct volume rendering has proven to be an effective and flexible visualization method for three-dimensional (3D) scalar fields. Transfer functions are fundamental to direct volume rendering because their role is essentially to make the data visible: By assigning optical properties like color and opacity to the voxel data, the volume can be rendered with traditional computer graphics methods. Good transfer functions reveal the important structures in the data without obscuring them with unimportant regions. To date, transfer functions have generally been limited to one-dimensional (1D) domains, meaning that the 1D space of scalar data value has been the only variable to which opacity and color are assigned. One aspect of direct volume rendering which has received little attention is the use of multidimensional transfer functions.

Often, there are features of interest in volume data that are difficult to extract and visualize with 1D transfer functions. Many medical data sets created from CT or MRI scans contain a complex combination of boundaries between multiple materials. This situation is problematic for 1D transfer functions because of the potential for overlap between the data value intervals spanned by the different boundaries. When one data value is associated with multiple boundaries, a 1D transfer function is unable to render them in isolation. Another benefit of higher dimensional transfer functions is their ability to portray subtle variations in properties of a single boundary, such as its thickness. When working with multivariate data, a similar difficulty arises with features that can be identified only by their unique combination of multiple data values. A 1D transfer function is simply not capable of capturing this relationship.

Unfortunately, using multidimensional transfer functions in volume rendering is complicated. Even when the transfer function is only 1D, finding an appropriate transfer function is generally accomplished by trial and error. This is one of the main challenges in making direct volume rendering an effective visualization tool. Adding dimensions to the transfer function domain only compounds the problem. While this is an ongoing research area, many of the proposed methods for transfer function generation and manipulation are not easily extended to higher dimensional transfer functions. In addition, fast volume rendering algorithms that assume the transfer function can be implemented as a linear lookup table (LUT) can be difficult to adapt to multidimensional transfer functions due to the linear interpolation imposed on such LUTs.

A previous paper [19] demonstrated the importance and power of multidimensional transfer functions. This paper extends that work with a more detailed exposition of the multidimensional transfer function concept, a generalization of multidimensional transfer functions for both scalar and multivariate data, as well as a novel technique for the interactive generation of volumetric shadows. To resolve the potential complexities in a user interface for multidimensional transfer functions, we introduce a set of direct manipulation widgets which make finding and experimenting with transfer functions an intuitive, efficient, and informative process. In order to make this process genuinely interactive, we exploit the fast rendering capabilities

• *The authors are with the Scientific Computing and Image Institute, School of Computing, University of Utah, 50 S. Central Campus Dr., Salt Lake City, UT 84112. E-mail: {jmk, gk, hansen}@cs.utah.edu.*

Manuscript received 15 Feb. 2002; revised 15 Mar. 2002; accepted 2 Apr. 2002.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number 116208.

DOI: 10.1111/cgf.12934
EuroVis 2016
R. Maciejewski, T. Ropinski, and A. Vilanova
(Guest Editors)

Volume 35 (2016), Number 3
STAR – State of The Art Report

State of the Art in Transfer Functions for Direct Volume Rendering

Patric Ljung[1] Jens Krüger[2,3] Eduard Gröller[4,5] Markus Hadwiger[6] Charles D. Hansen[3] Anders Ynnerman[1]

[1]Linköping University, Sweden [2]CoViDAG, University of Duisburg-Essen
[3]Scientific Computing and Imaging Institute, University of Utah [4]TU Wien, Austria [5]University of Bergen, Norway
[6]King Abdullah University of Science and Technology

Abstract

A central topic in scientific visualization is the transfer function (TF) for volume rendering. The TF serves a fundamental role in translating scalar and multivariate data into color and opacity to express and reveal the relevant features present in the data studied. Beyond this core functionality, TFs also serve as a tool for encoding and utilizing domain knowledge and as an expression for visual design of material appearances. TFs also enable interactive volumetric exploration of complex data. The purpose of this state-of-the-art report (STAR) is to provide an overview of research into the various aspects of TFs, which lead to interpretation of the underlying data through the use of meaningful visual representations. The STAR classifies TF research into the following aspects: dimensionality, derived attributes, aggregated attributes, rendering aspects, automation, and user interfaces. The STAR concludes with some interesting research challenges that form the basis of an agenda for the development of next generation TF tools and methodologies.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications—Volume Rendering I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture I.4.10 [Computer Graphics]: Image Representation—Volumetric

roduction

sfer function (TF) maps volumetric data to optical properties part of the traditional visualization pipeline: data acquisi- rocessing, visual mapping, and rendering. Volumetric data is ered to be a scalar function from a three-dimensional spatial n with a one-dimensional range (e.g., density, flow magnitude, mage generation involves mapping data samples through the here they are given optical properties such as color and opacity, mpositing them into the image.

F simultaneously defines (1) which parts of the data are es- l to depict and (2) how to depict these, often small, portions volumetric data. Considering the first step, a TF is a special, portant, case of a segmentation or classification. With classifi- , certain regions in a three-dimensional domain are identified ng to the same material, such as bone, vessel, or soft tissue, dical imaging. A plethora of classification and segmentation thms have been developed over the last decades, with semi- atic approaches often tailored to specific application scenarios. entation algorithms can be quite intricate since information he three-dimensional spatial domain and the one-dimensional nge are taken into account. TFs in their basic form are, on the hand, restricted to using only the data ranges. In comparison eneral classification algorithms, this characteristic makes a TF less powerful with respect to identifying relevant parts of the data. The advantage of a TF is, however, a substantial performance gain as classification based on the one-dimensional data range reduces the complexity tremendously. This gain is the result of the three-dimensional domain being typically two orders of magnitude larger than the small data range. Histograms are an example of discarding potentially complex spatial information and aggregating only the data values to binned-frequency information. In the same spirit, TFs classify interesting parts of the data by considering data values alone. The second functionality of a TF deals with specifying optical properties for portions of the data range previously identified as being relevant.

A survey of the works published in the field of TFs reveals that in the three decades since the earliest published techniques for direct volume rendering [KVH84, DCH88, Lev88], over a hundred studies have been published in the most influential journals and conferences. In 2001, the Transfer Function Bake-Off examined four of the then most promising approaches to TF design [PLB*01]: trial and error with a TF editor; data-centric using computed metrics over the scalar field; data-centric using a material boundary model; and image-centric using an organized sampling of exemplar images. In 2010, Arens and Domik [AD10] authored a survey of TFs for volume rendering in which they subdivided TFs into the follow-