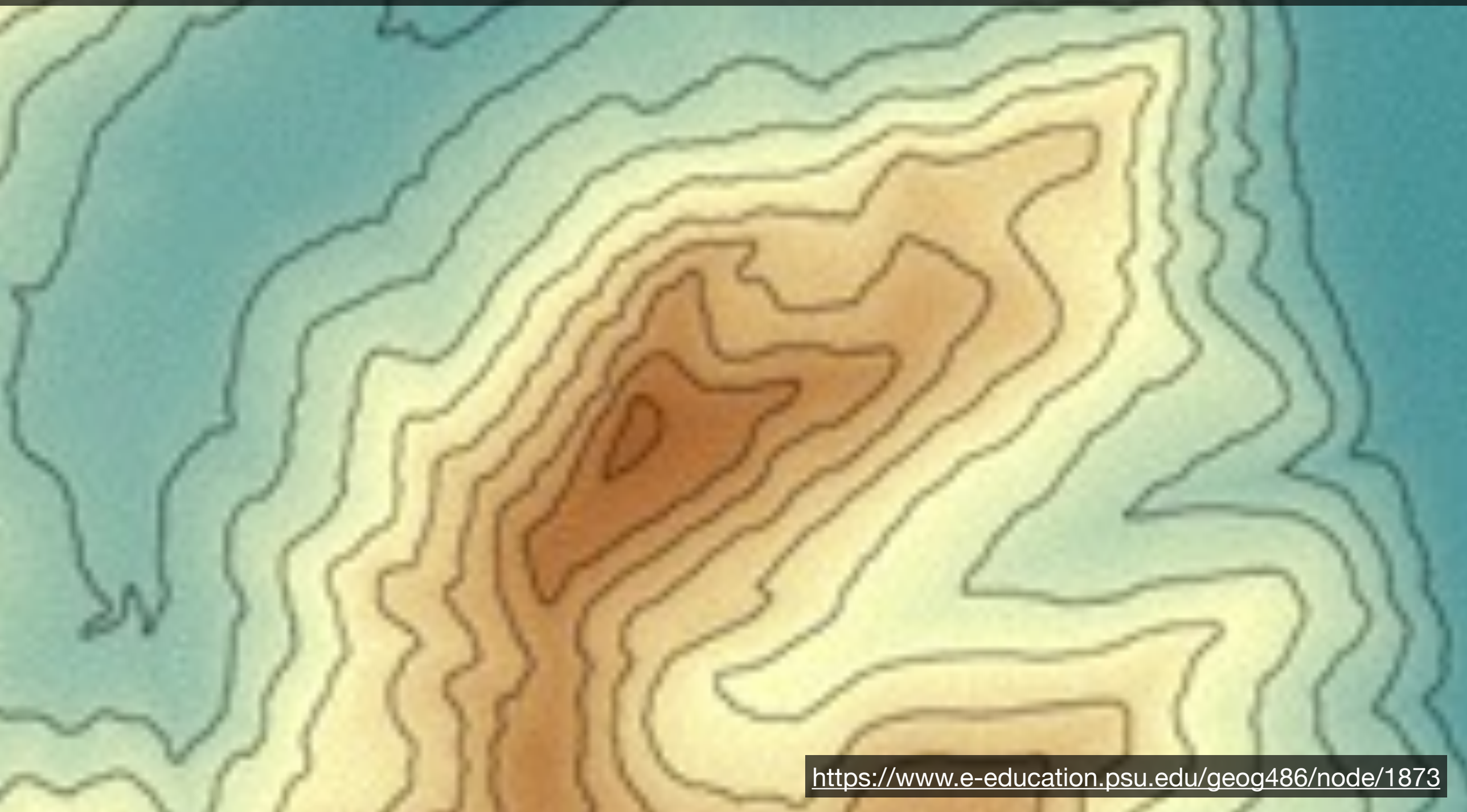
Spatial Data: Scalar Fields wrap-up

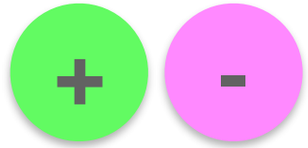
CSC544

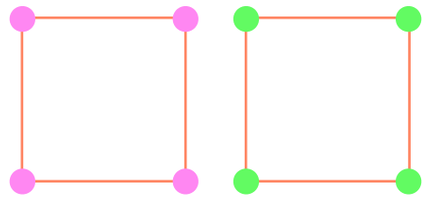
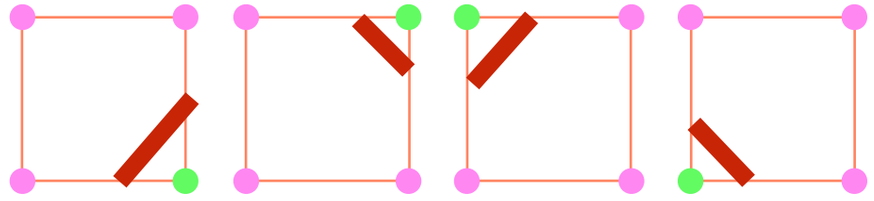
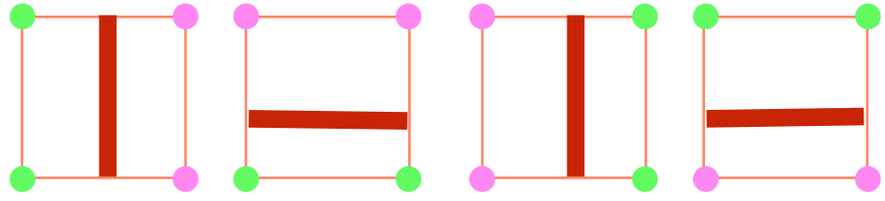
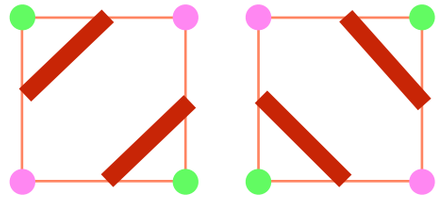
Recap: 2D contouring



https://www.e-education.psu.edu/geog486/node/1873

Recap: 2D contouring



Case	Polarity	Rotation	Total	
No Crossings	x2		2	
Singlet	x2	x4	8	 (x2 for polarity)
Double adjacent	x2	x2 (4)	4	
Double Opposite	x2	x1 (2)	2	
			16 = 2^4	

Recap: 2D contouring

The Asymptotic Decider:
Resolving the Ambiguity in Marching Cubes

Gregory M. Nielson Bernd Hamann

Computer Science
Arizona State University
Tempe, AZ 85287-5406

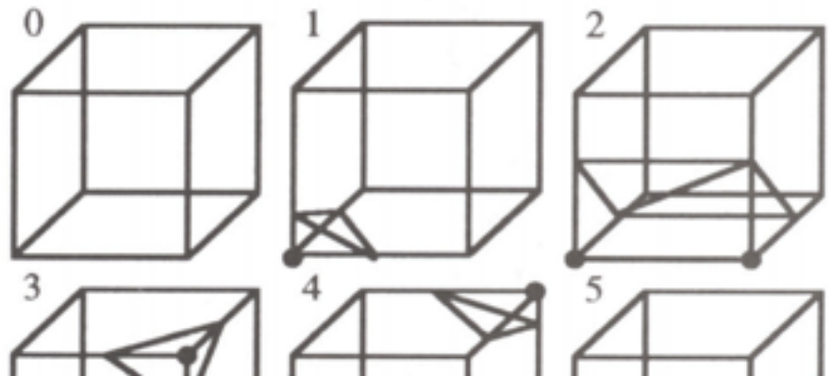
Abstract

A method for computing isovalue or contour surfaces of a trivariate function is discussed. The input data are values of the trivariate function, F_{ijk}, at the cuberille grid points (x_i, y_j, z_k) and the output is a collection of triangles representing the surface consisting of all points where $F(x, y, z)$ is a constant value. The method described here is a modification that is intended to correct a problem with a previous method.

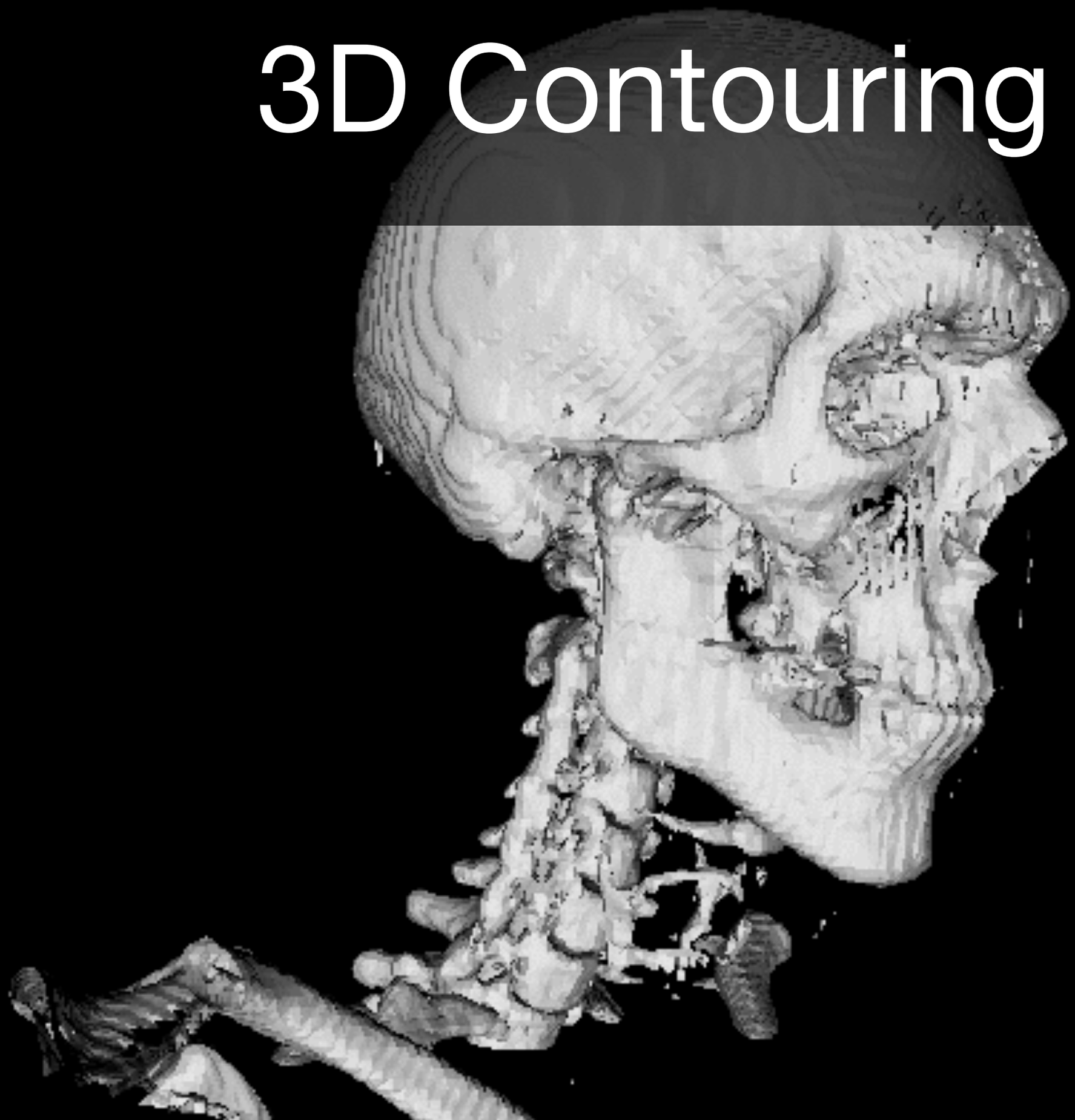
1.0 Introduction

The purpose of this paper is to describe a method for computing contour or isovalue surfaces of a trivariate function F(x, y, z). It is assumed that the function is continuous and that samples over a cuberille grid (see Figure 1) are available. These values are denoted by $F_{ijk} = F(x_i, y_j, z_k)$; i = 1, ... , Nx, j = 1, ... , Ny, k = 1, ... , Nz. The problem is to compute the isovalue or contour surface

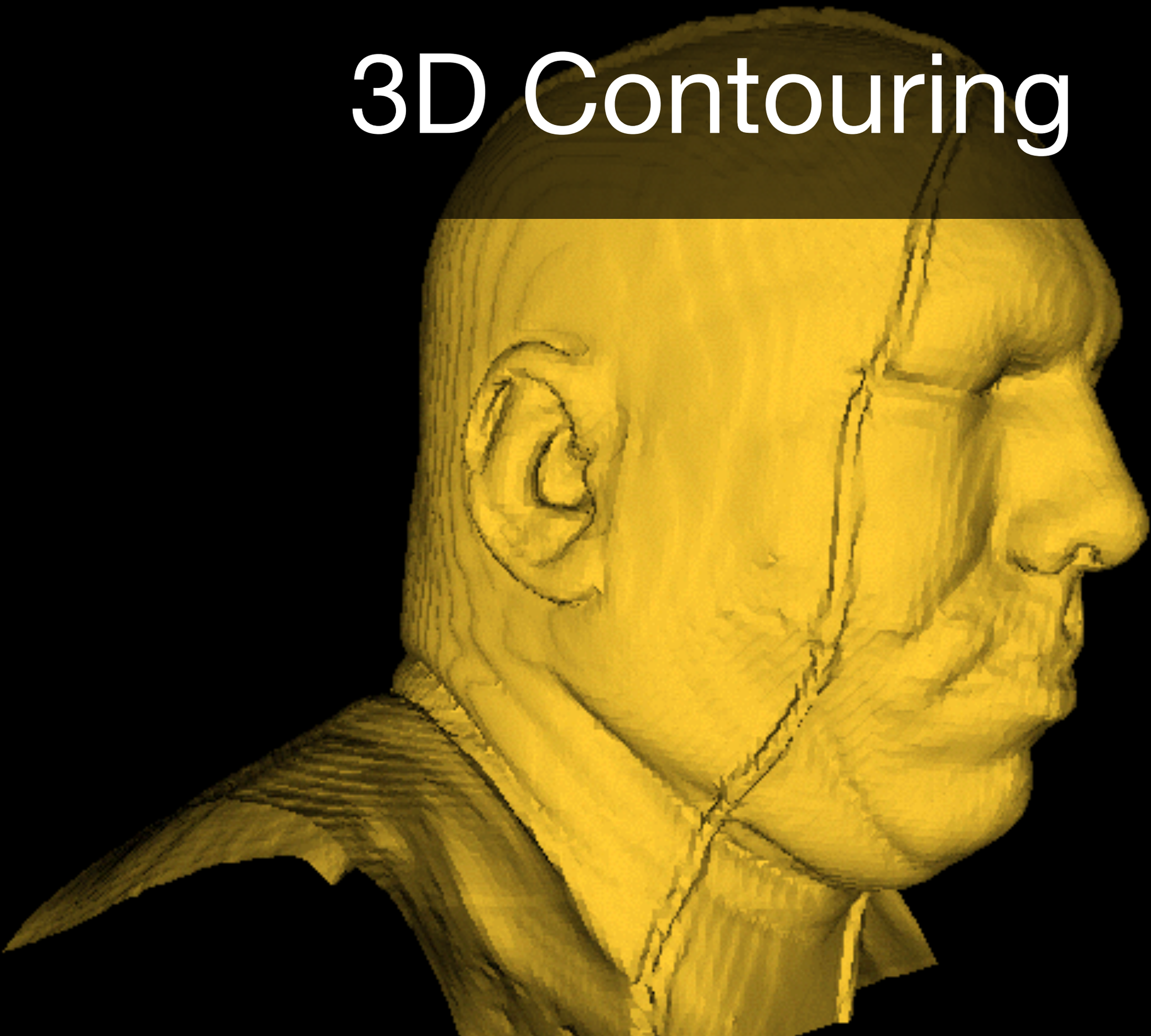
marked indicates $F_{ijk} > \alpha$. While there are $2^8 = 256$ possible configurations, there are only 15 shown in Figure 2. This is because some configurations are equivalent with respect to certain operations. First off, the number can be reduced to 128 by assuming two configurations are equivalent if marked grid points and unmarked grid points are switched. This means that we only have to consider cases where there are four or fewer marked grid points. Further reduction to the 15 cases shown is possible by equivalence due to rotations.



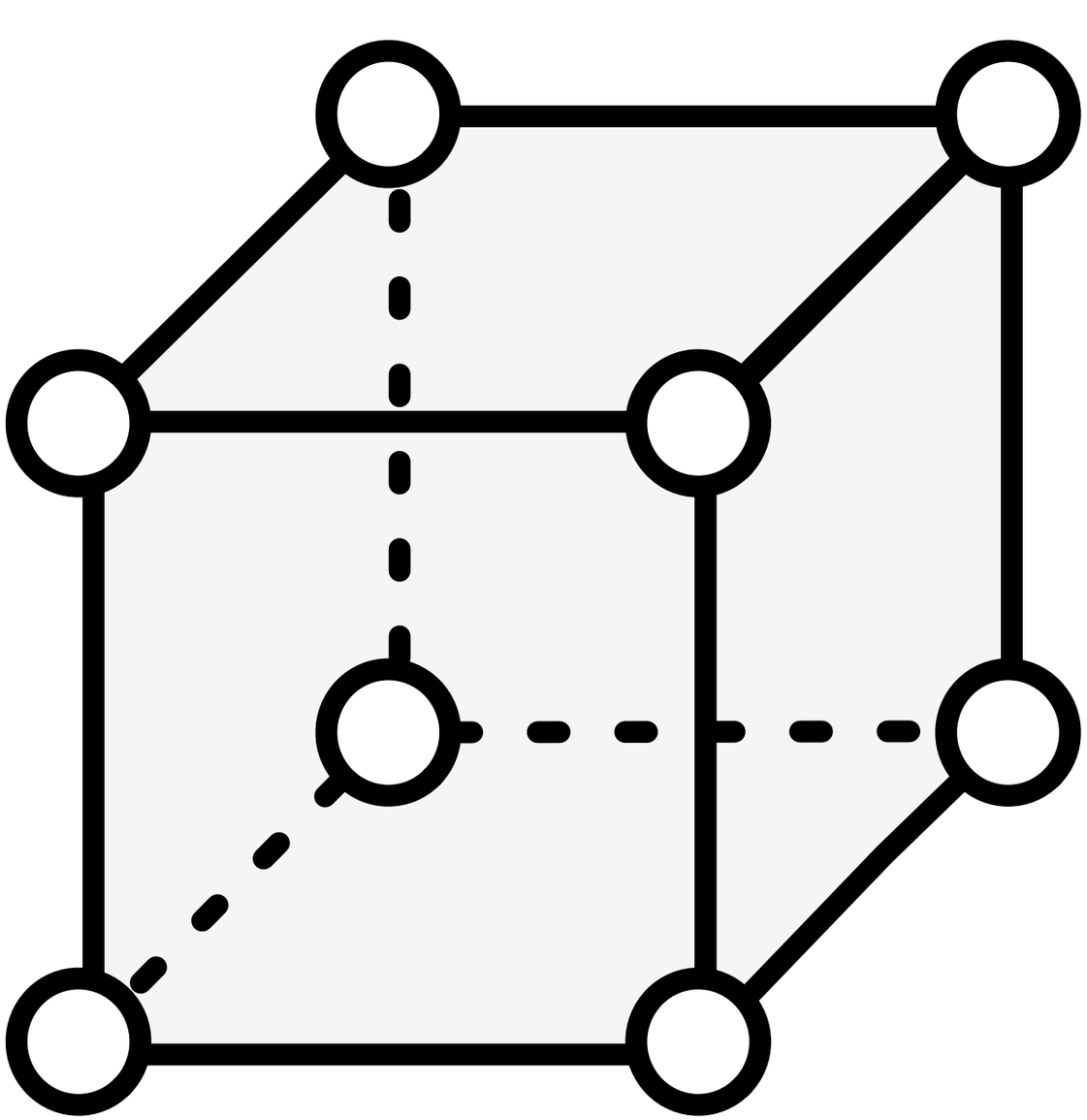
3D Contouring

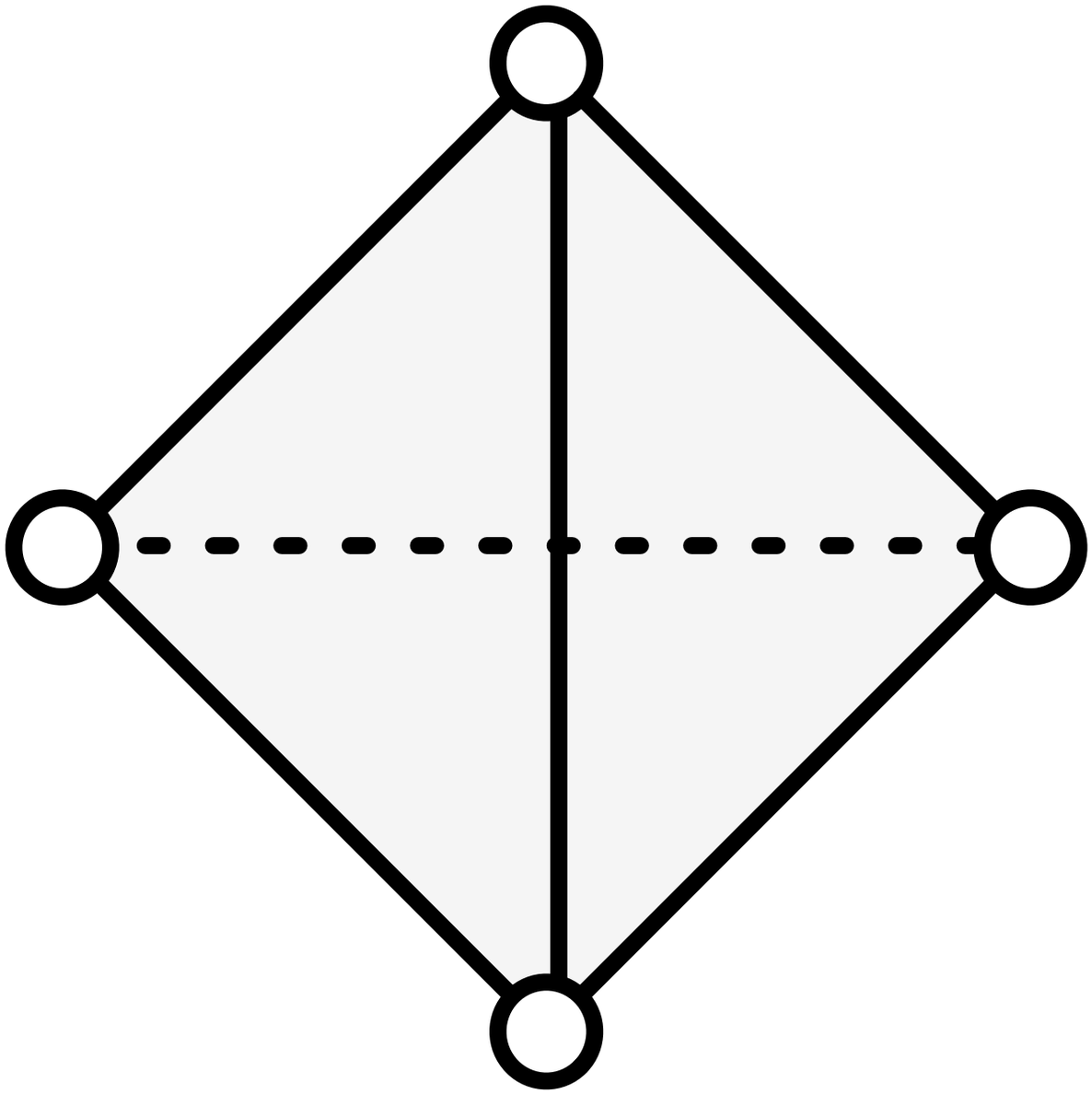


3D Contouring

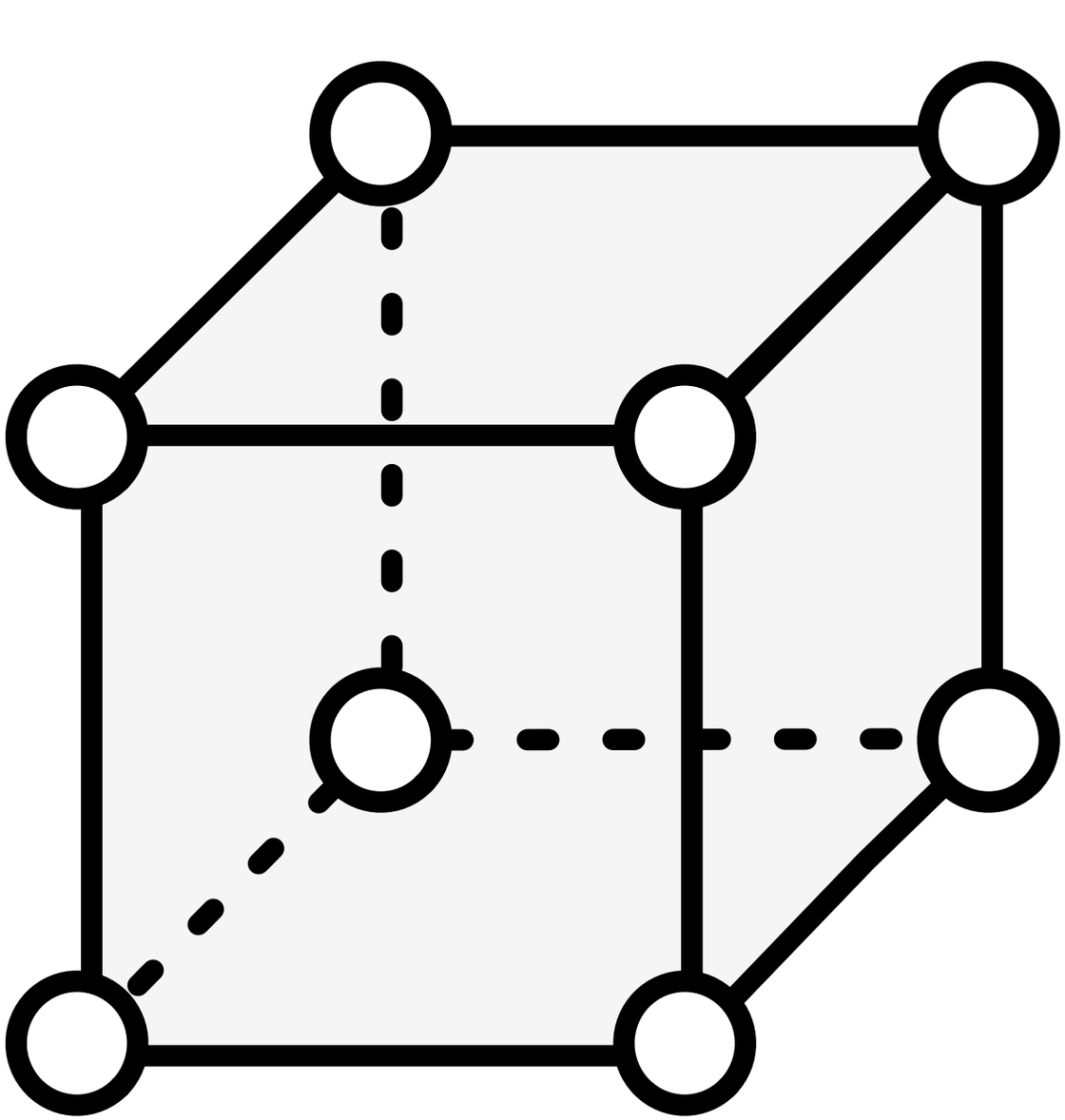


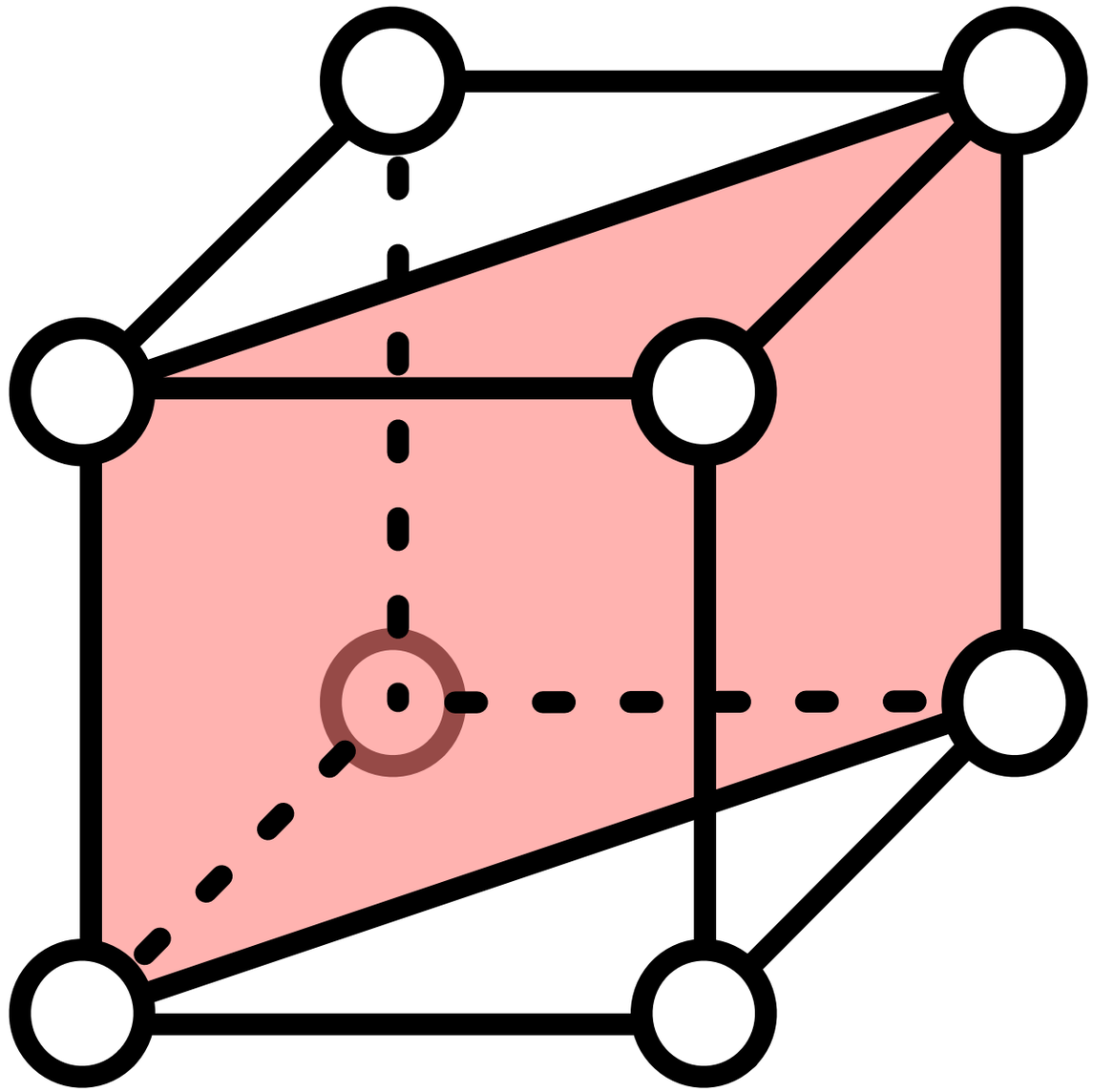
Splitting 3D space into simple shapes



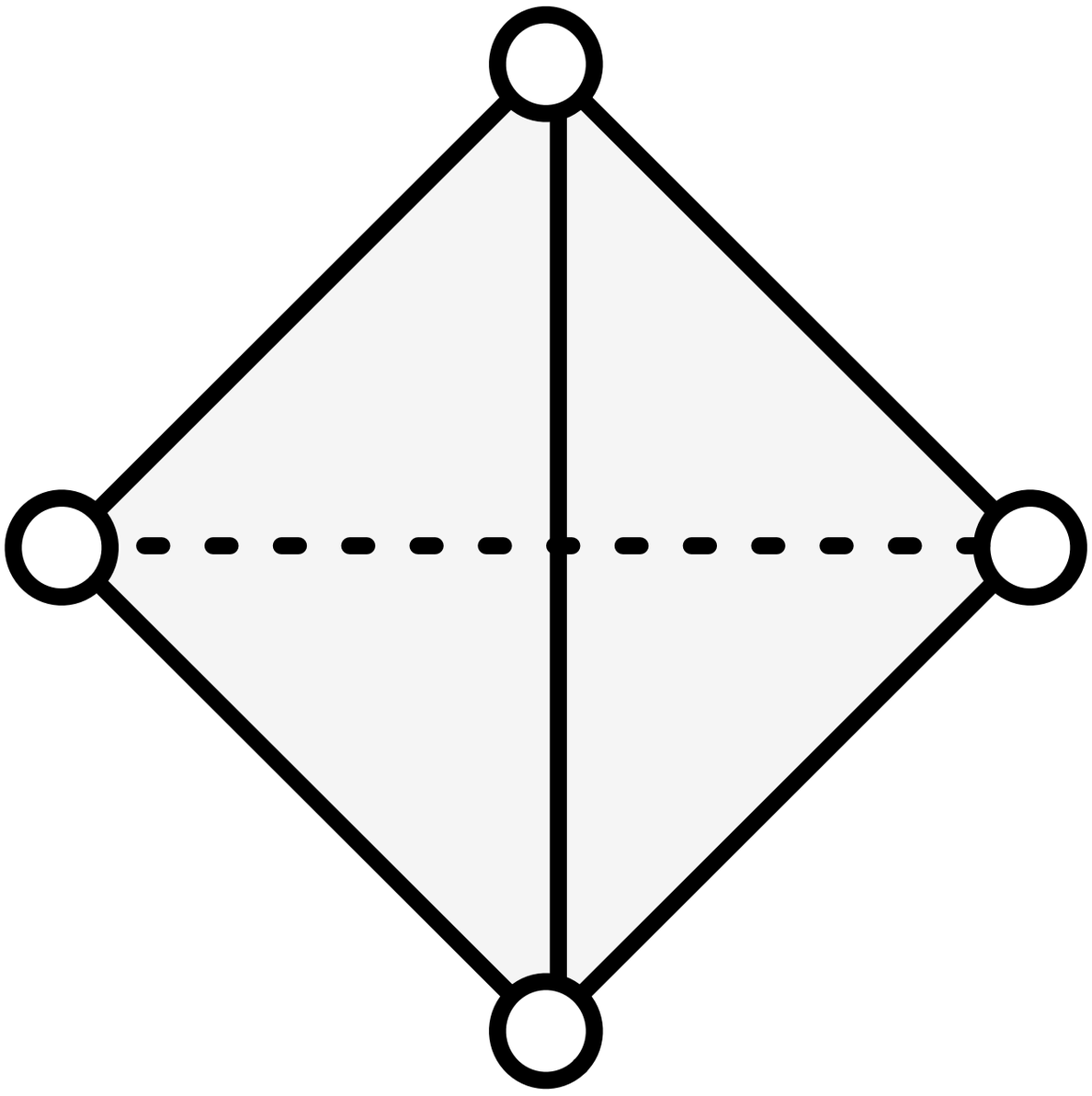


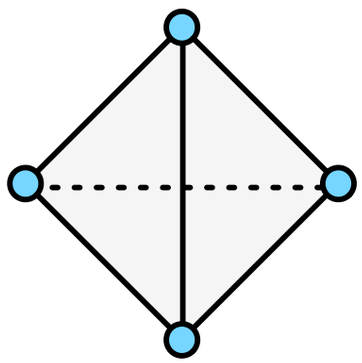
Cube into tetrahedra

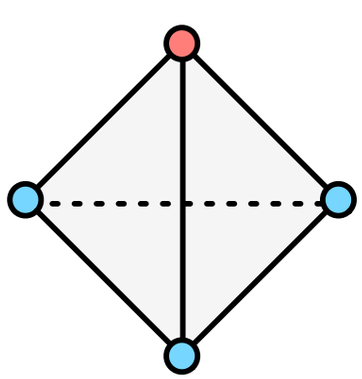


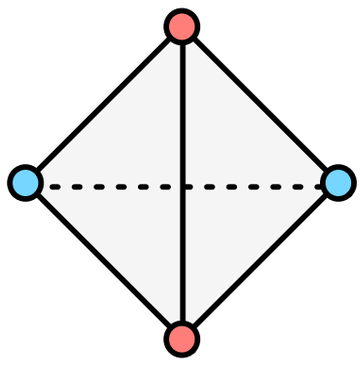


Marching Tetrahedra



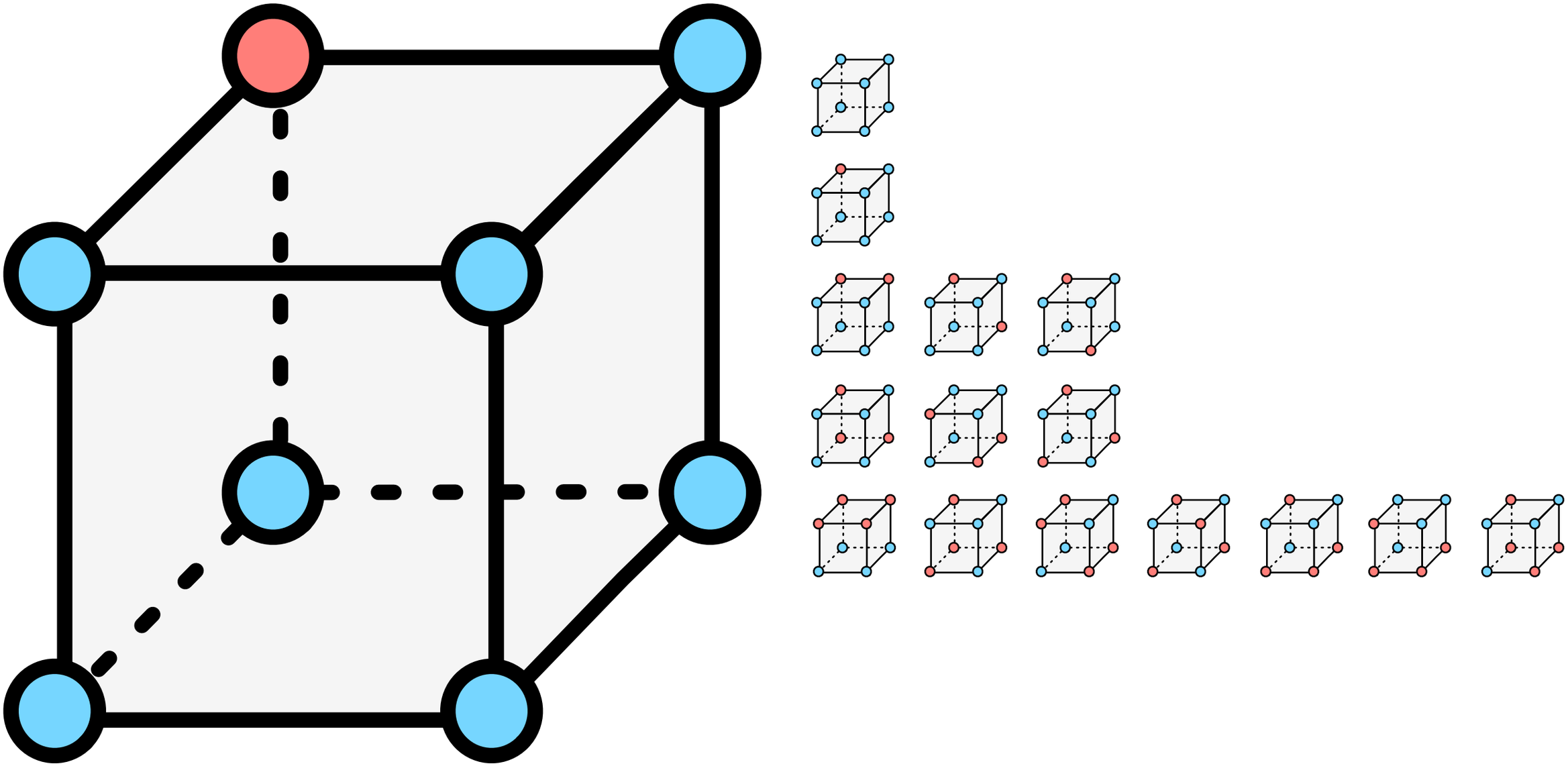




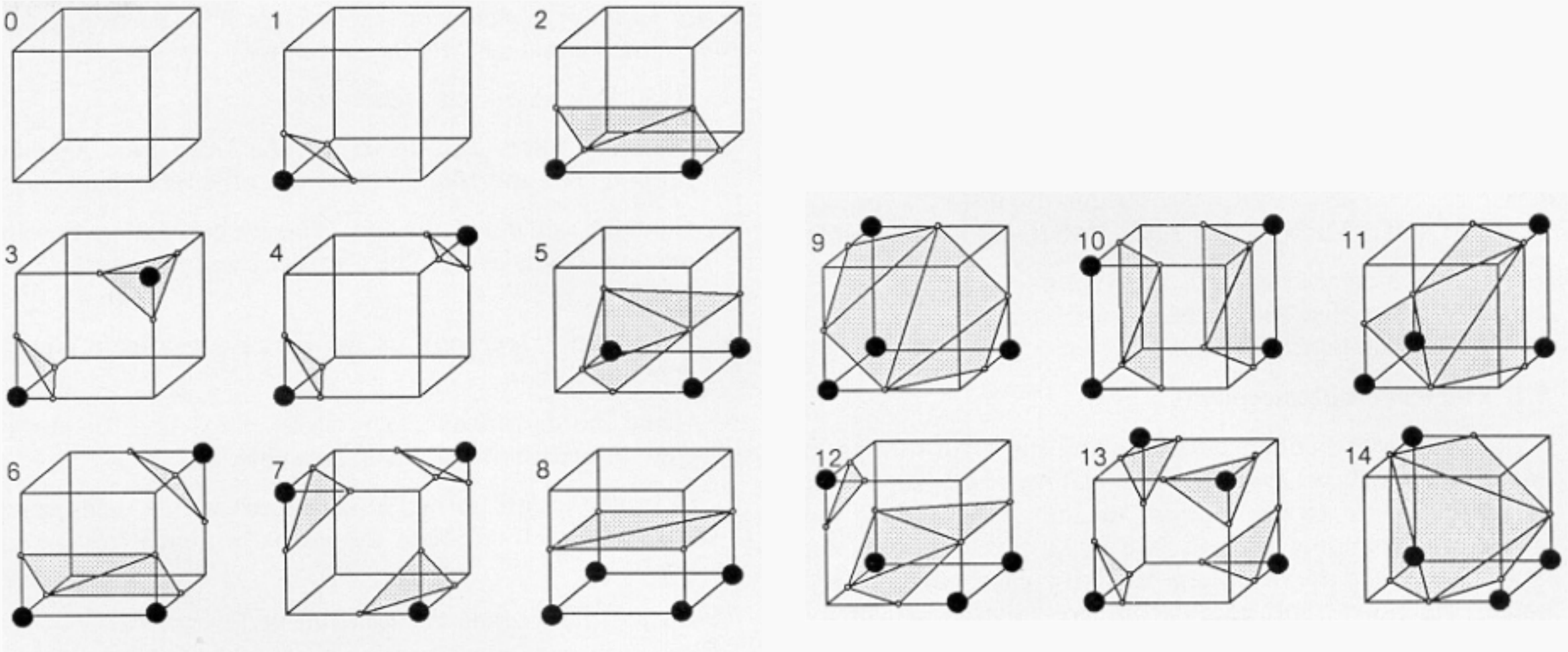


3 cases, “obvious”

3D Contouring



3D Contouring



n-D Contouring

Isosurfacing in Higher Dimensions

Praveen Bhaniramka Rephael Wenger Roger Crawfis

Computer & Information Science
The Ohio State University
Columbus, Ohio

Abstract

Visualization algorithms have seen substantial improvements in the past several years. However, very few algorithms have been developed for directly studying data in dimensions higher than three. Most algorithms require a sampling in three-dimensions before applying any visualization algorithms. This sampling typically ignores vital features that may be present when examined in oblique cross-sections, and places an undo burden on system resources when animation through additional dimensions is desired. For time-varying data of large data sets, smooth animation is desired at interactive rates. This paper provides a fast marching-

the marching cubes algorithm first build a table of these 2^8 cases and then use this table to determine the structure of the intersection of the surface with each cube. The actual location of the surface within the cube depends upon linear interpolations of the values of the cube vertices along the edges intersected by the isosurface.

By exploiting symmetry, Lorensen and Cline reduced the 2^8 cases to fourteen. They analyzed these fourteen cases by hand, constructing a triangulated surface for each case. Montani, Scateni and Scopigno added more cases to resolve certain ambiguities and inconsistencies in Lorensen and Cline's original algorithm [12]. This algorithm is commonly known as the Modified Marching

n-D Contouring

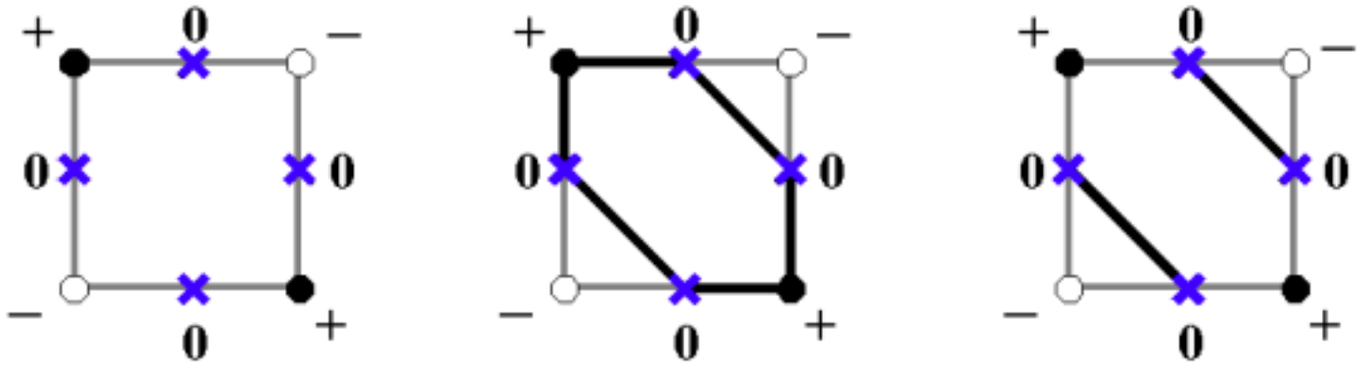
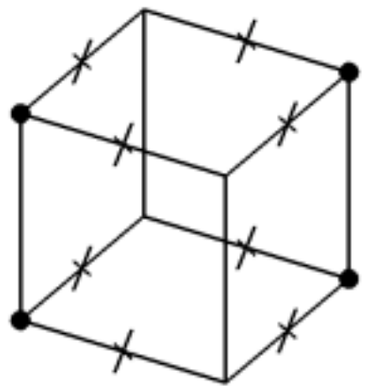
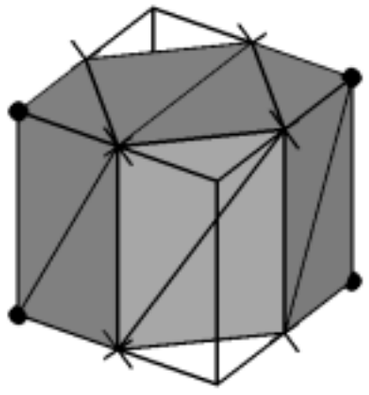


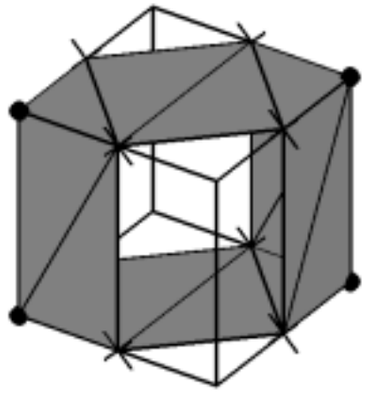
Figure 1a. Two-dimensional example of the algorithm.



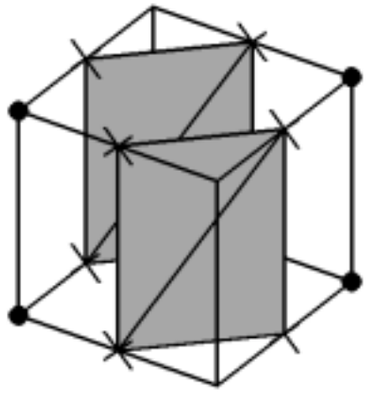
Input vertices



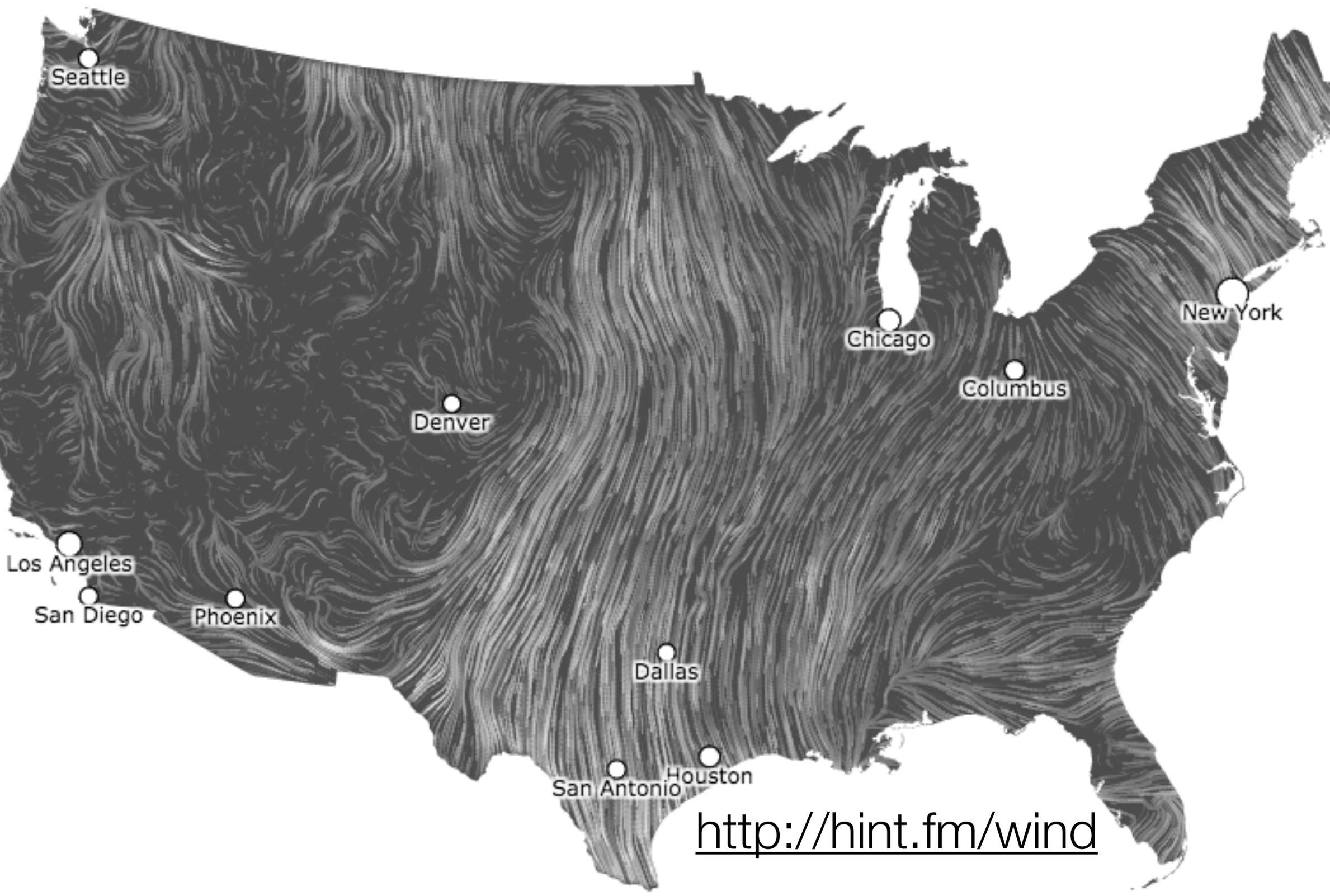
Convex Hull



Coplanar Faces

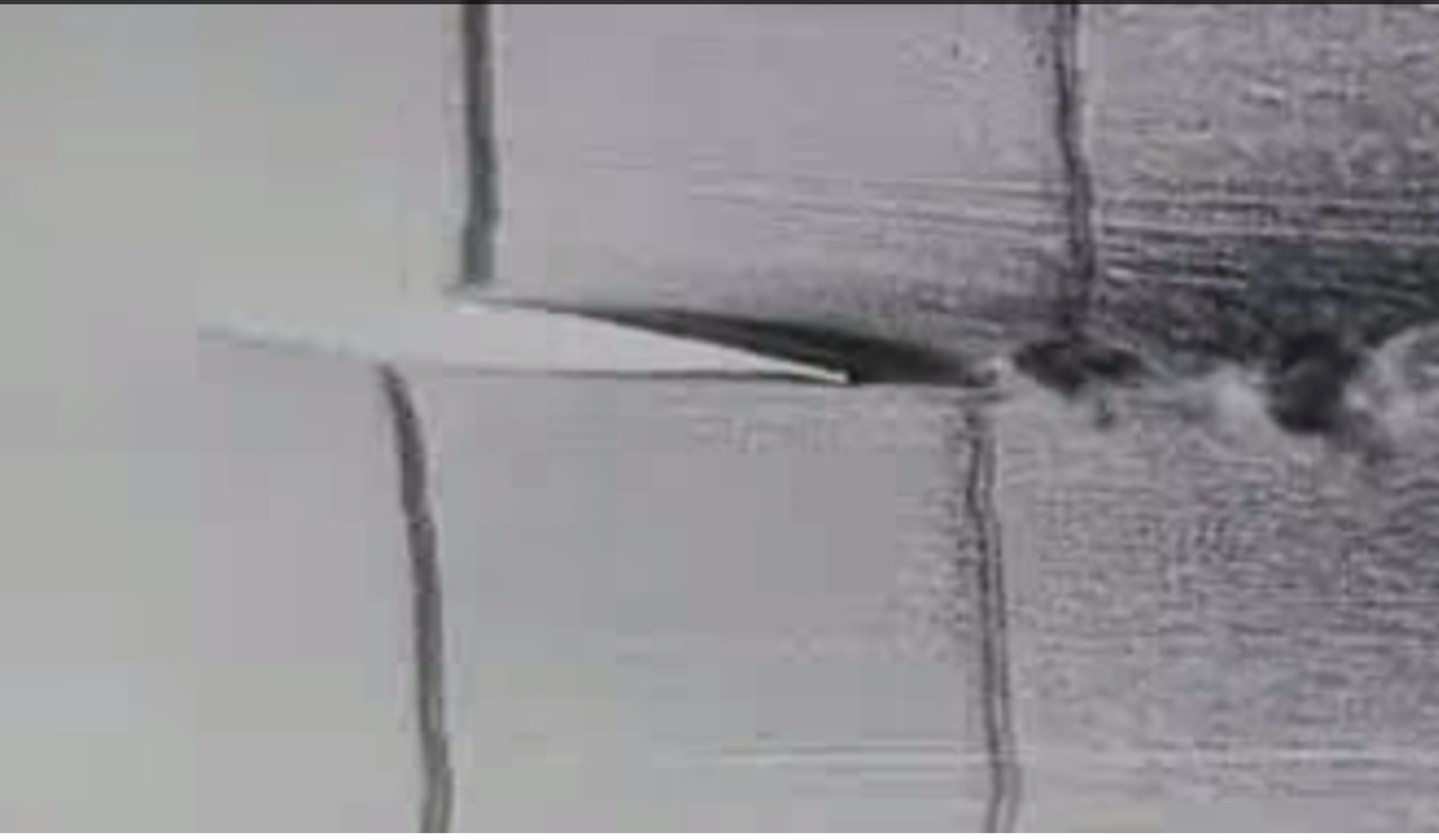


Resulting Isosurface



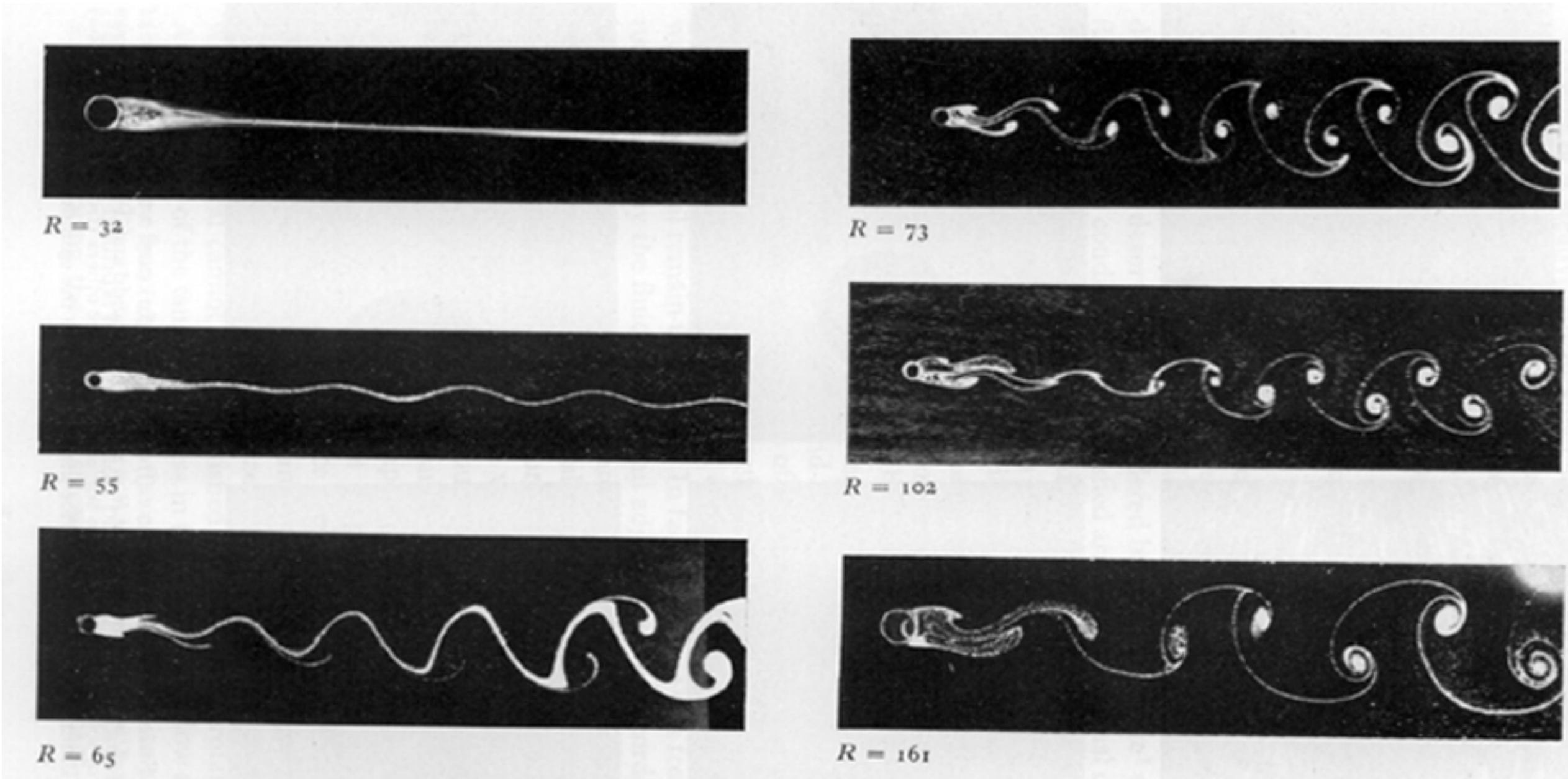
Spatial Data: Vector Fields

https://www.youtube.com/watch?v=nuQyKGuXJOs



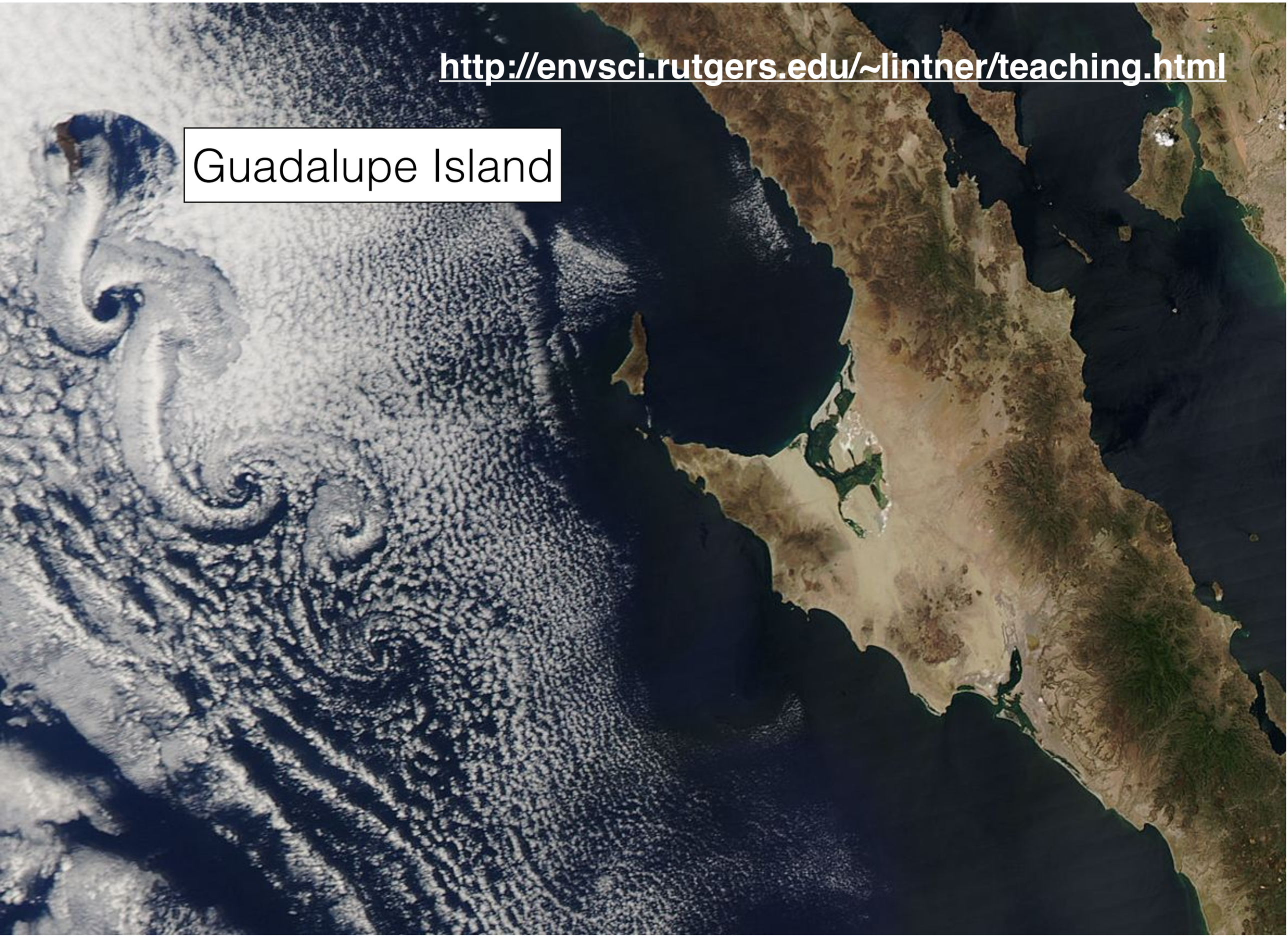
Spatial Data: Vector Fields

Experimental Flow Vis



von Kármán vortex street, depending on Reynolds number

http://envsci.rutgers.edu/~lintner/teaching.html

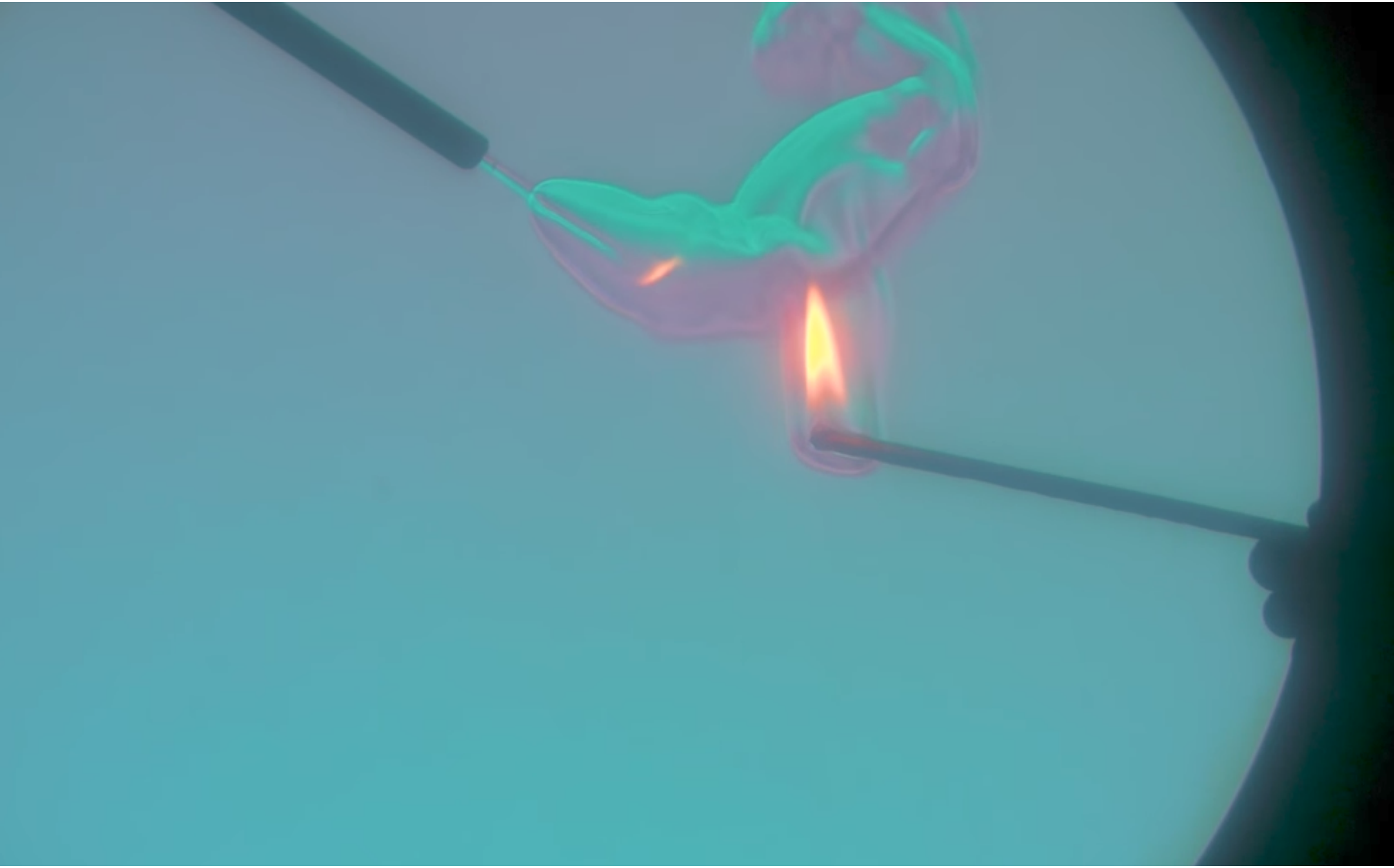


https://www.flickr.com/photos/pauln82/6639913813

Schlieren (Knife-Edge) Photography

https://www.youtube.com/watch?v=4tgOyU34D44

Schlieren (Knife-Edge) Photography



https://www.youtube.com/watch?v=K7pQsR8WFSo

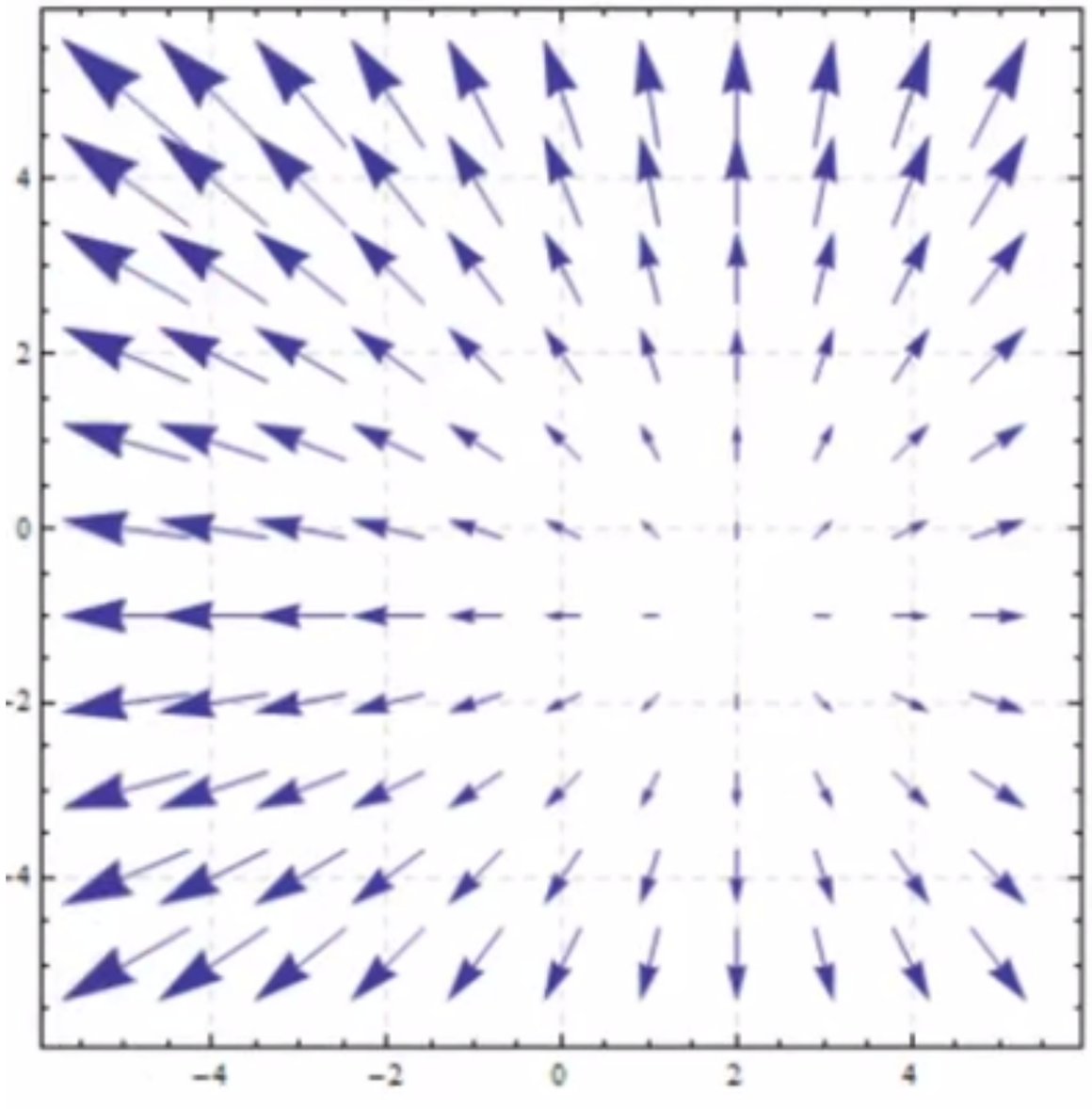
Mathematics of Vector Fields

$$v : R^n \rightarrow R^n$$

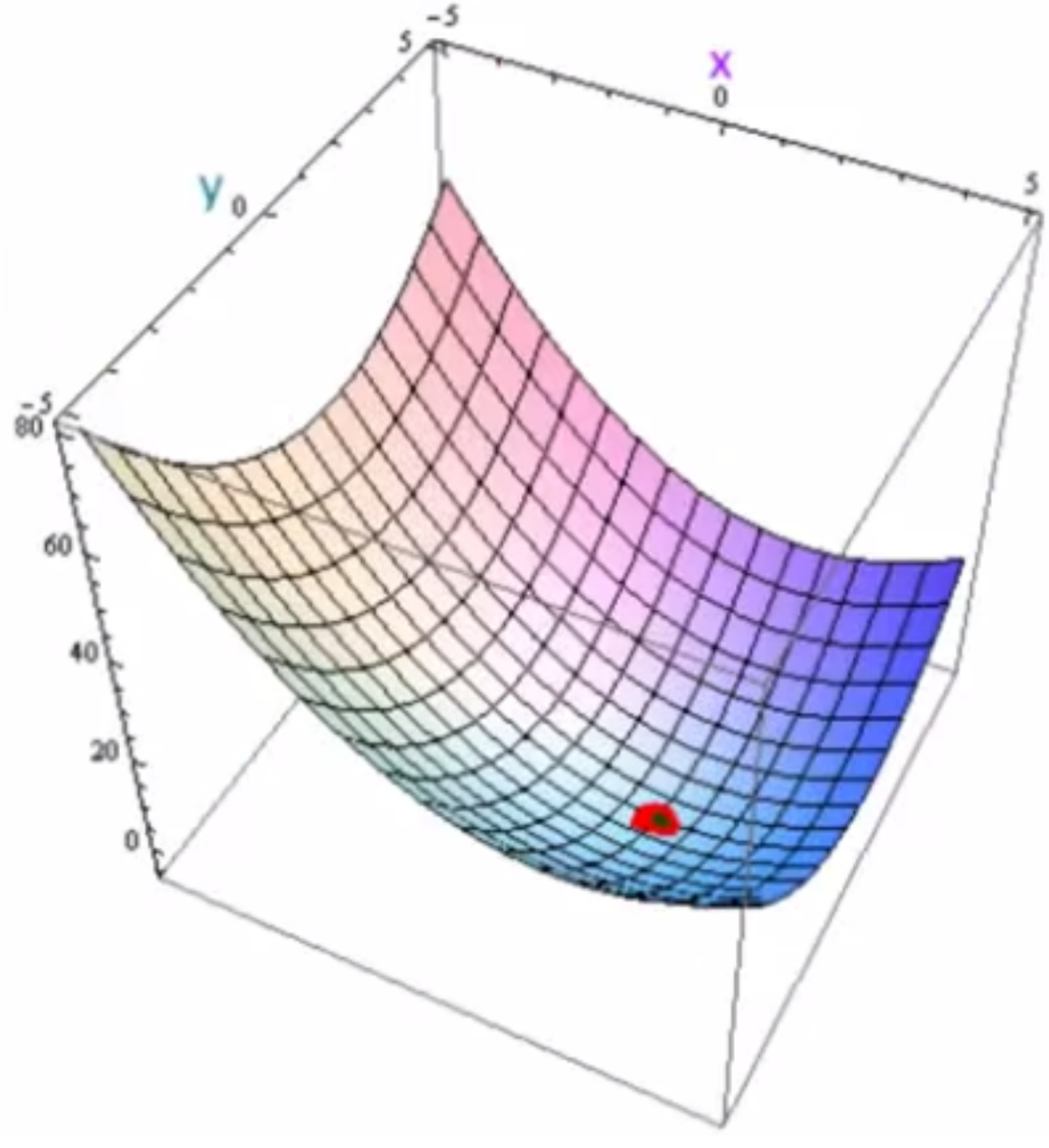
Function from vectors to vectors

A simple vector field: the gradient

The gradient field $< 2x\text{-}4 , 2y\text{+}2 >$

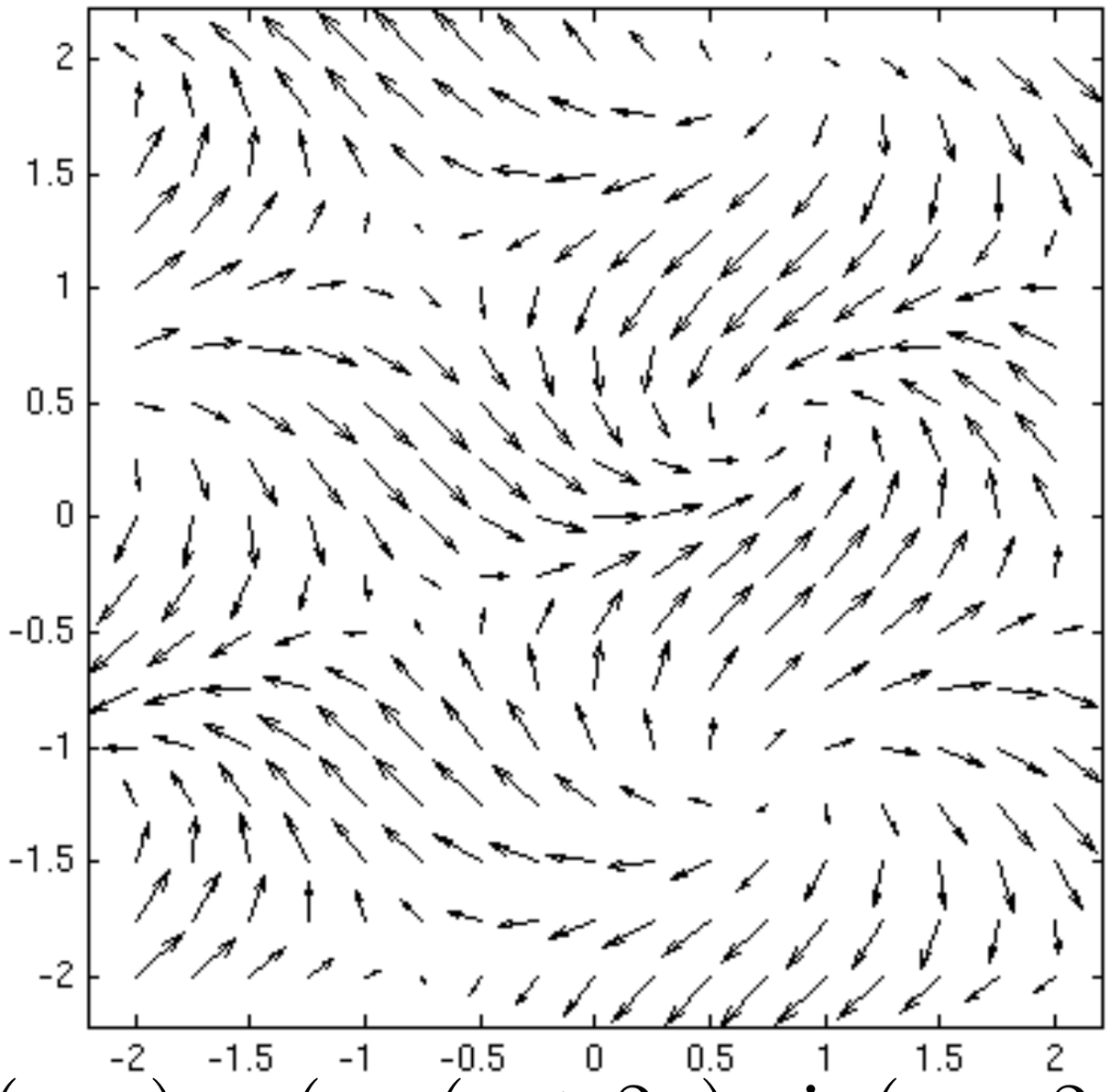


of the function $f = x^2 - 4x + y^2 + 2y$.



https://www.youtube.com/watch?v=v0_LlyVquF8

Vector fields can be more complicated

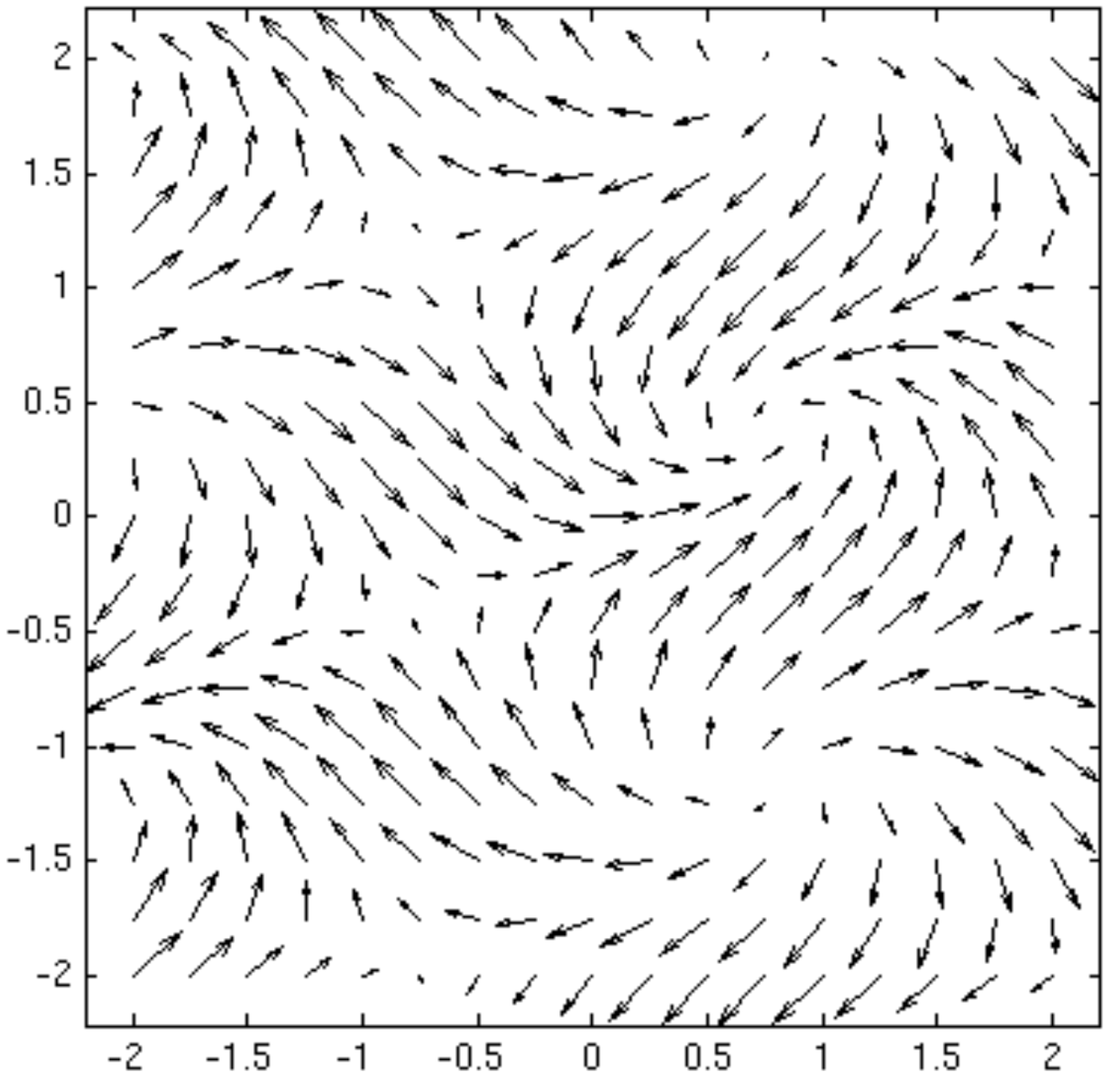


$$v(x, y) = (\cos(x + 2y), \sin(x - 2y))$$

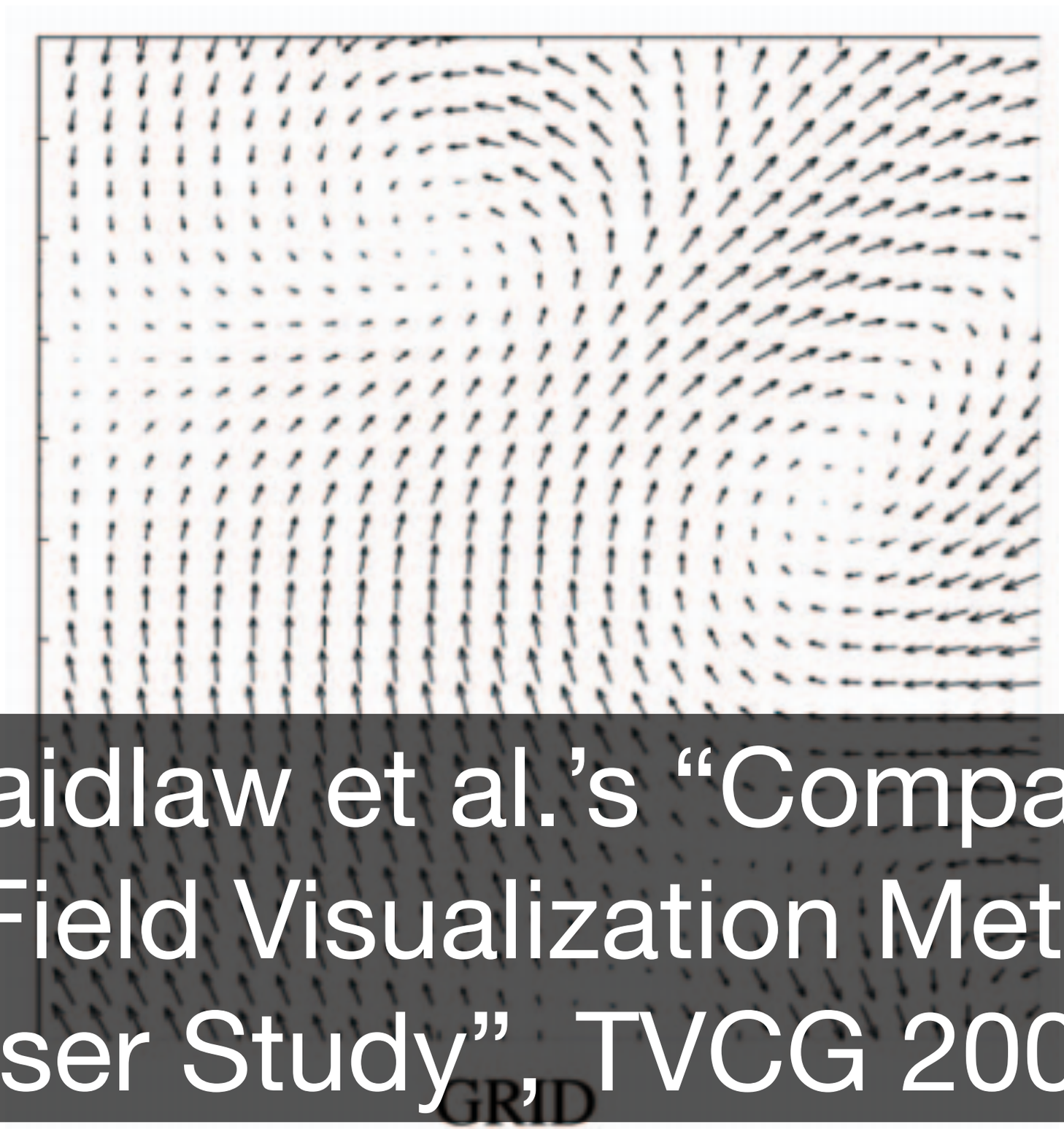
http://www.math.umd.edu/~petersd/241/html/ex27b.html

Glyph Based Techniques

Hedgehog Plot: Not Very Good

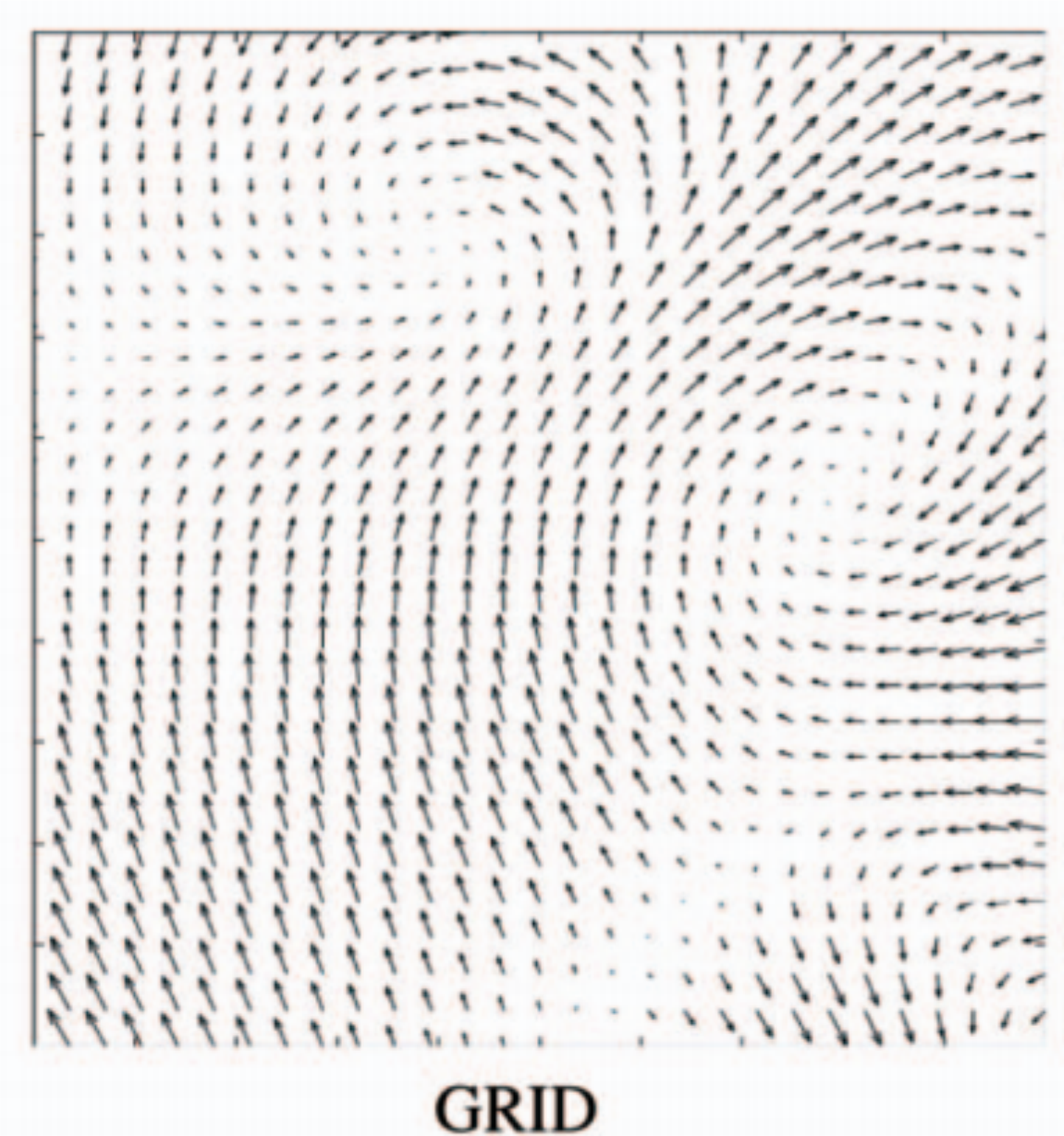


Hedgehog Plot: Not Very Good

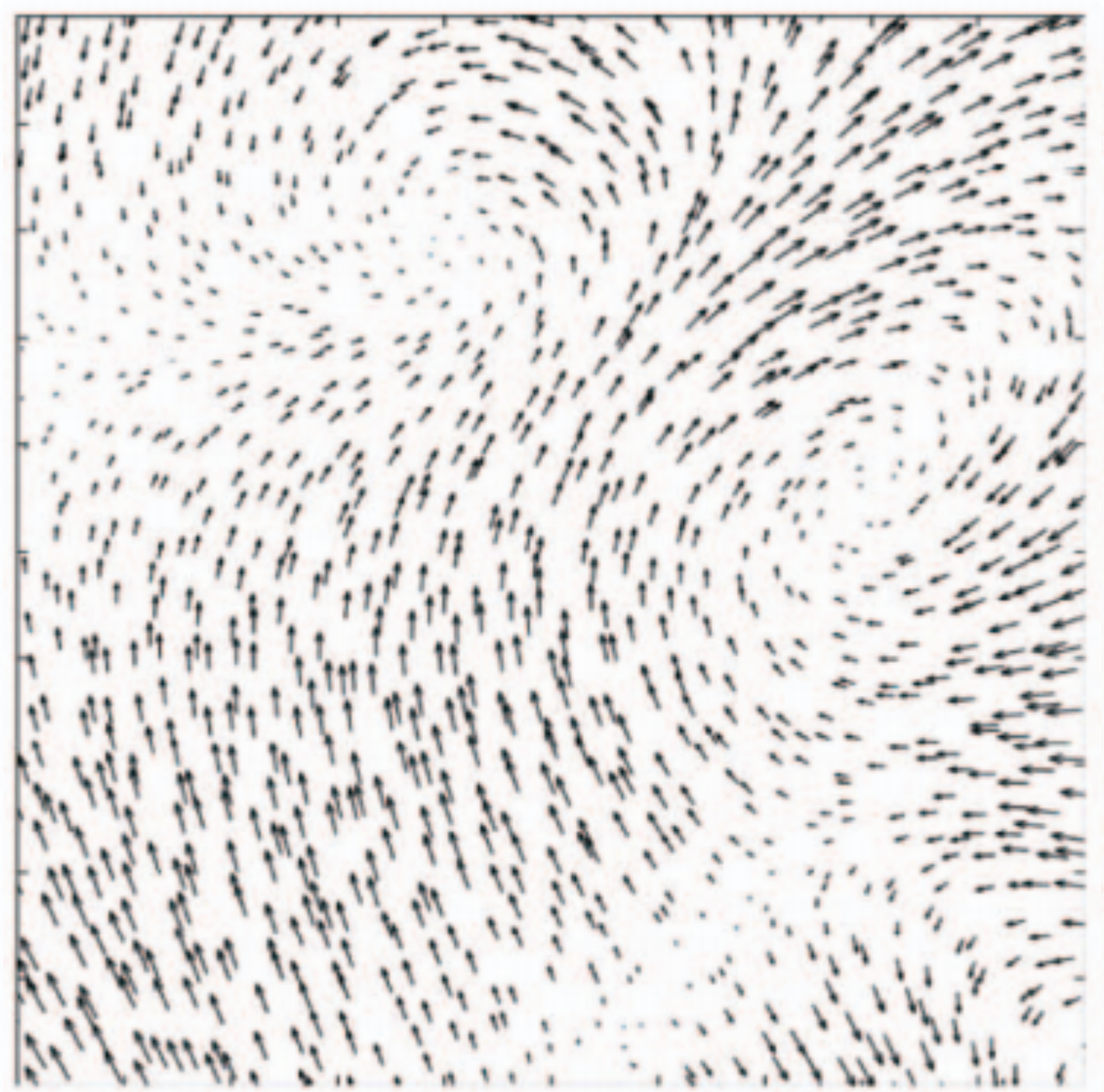


From Laidlaw et al.'s "Comparing 2D Vector Field Visualization Methods: A User Study", TVCG 2005

Hedgehog Plot: Not Very Good



Jittered Hedgehog Plot: Better



JIT

Space-filling scaled glyphs

LIT

Streamline-Guided Placement

OSTR

Streamline-Guided Placement

OSTR

Streamlines

Streamlines

Streamlines

Curves everywhere tangent to the vector field

Curves everywhere tangent to the vector field

$$\begin{aligned} x'(t) &= v_x(x(t), y(t)) \\ y'(t) &= v_y(x(t), y(t)) \end{aligned}$$

Visualization via streamlines

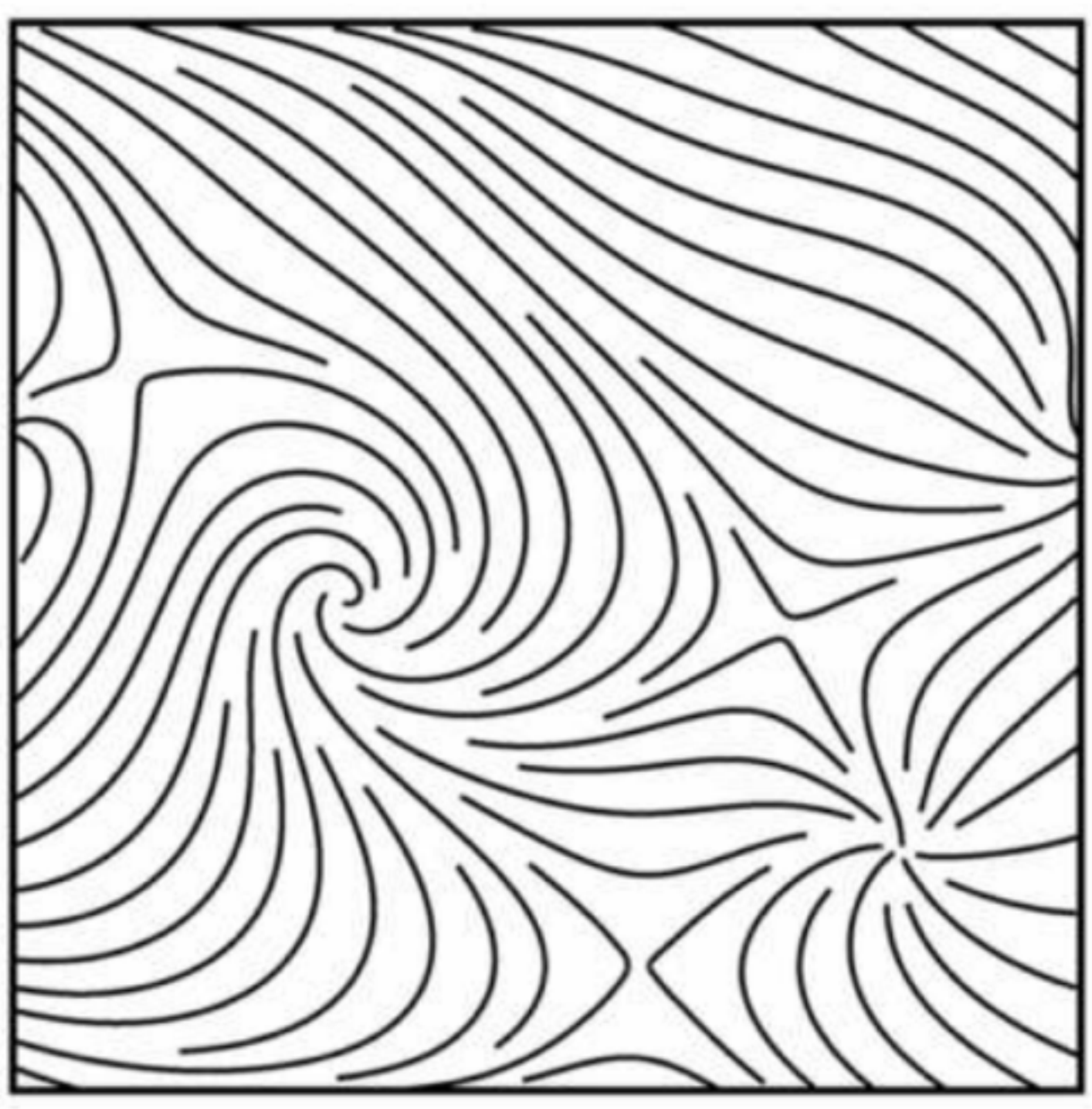
- Pick a set of seed points
- Integrate streamlines from those points

- **Which seed points?**

Uniform placement

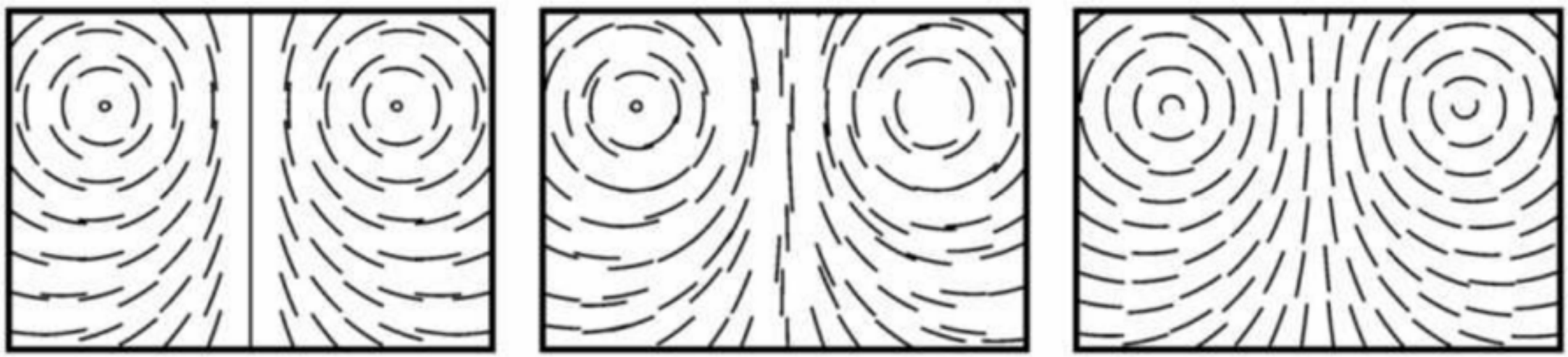
Turk and Banks, Image-Guided Streamline Placement
SIGGRAPH 1996

Density-optimized placement



Turk and Banks, Image-Guided Streamline Placement
SIGGRAPH 1996

Density-optimized placement



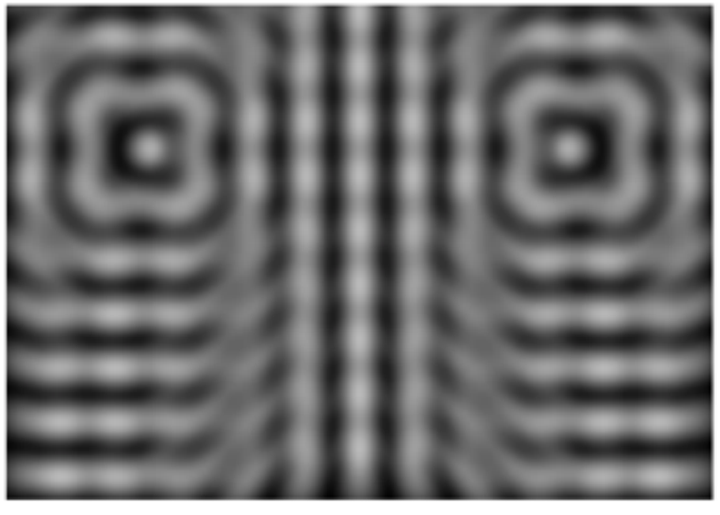


Figure 2: (a) Short streamlines with centers placed on a regular grid (top); (b) filtered version of same (bottom).

Figure 3: (a) Short streamlines with centers placed on a jittered grid (top); (b) filtered version showing bright and dark regions (bottom).

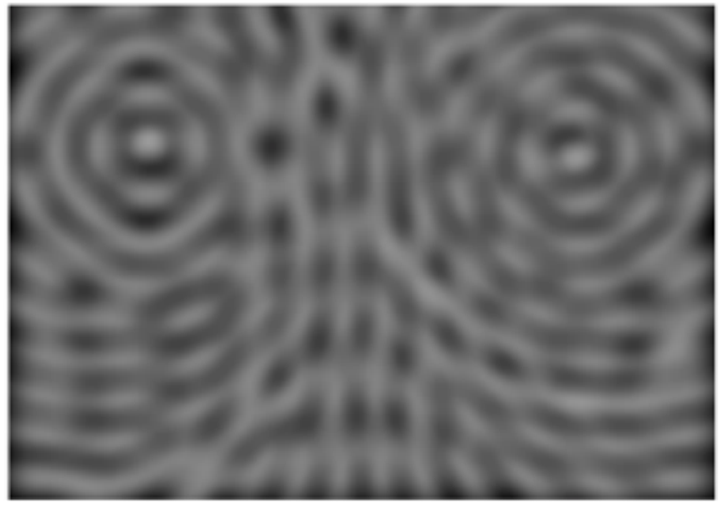


Figure 4: (a) Short streamlines placed by optimization (top); (b) filtered version showing fairly even gray value (bottom).

Turk and Banks, Image-Guided Streamline Placement
SIGGRAPH 1996

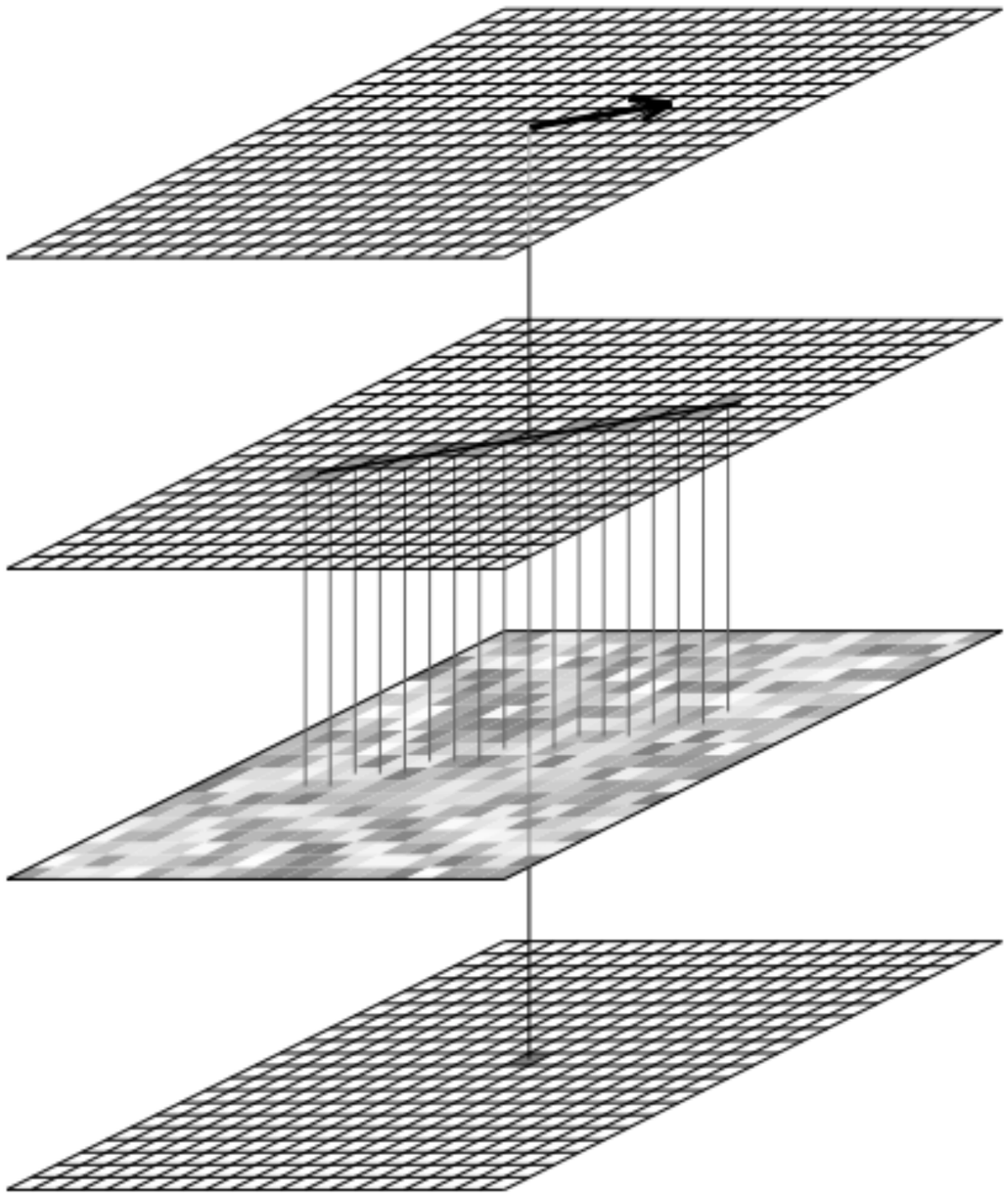
Image-Based Vector Field Visualization

Line Integral Convolution

http://www3.nd.edu/~cwang11/2dflowvis.html

Cabral and Leedom, Imaging Vector Fields using Line Integral Convolution. SIGGRAPH 1993

Line Integral Convolution



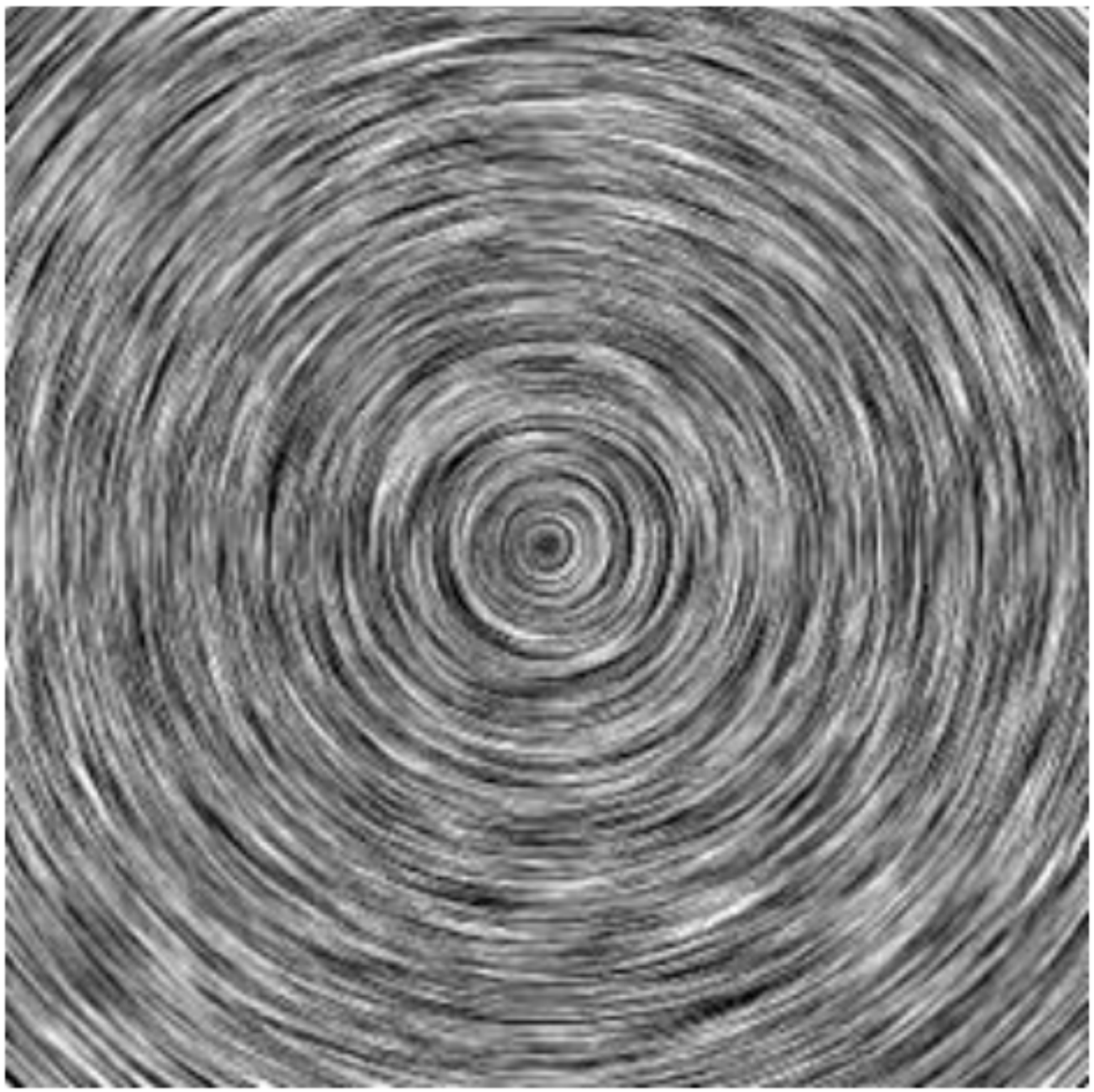
Given a
vector field

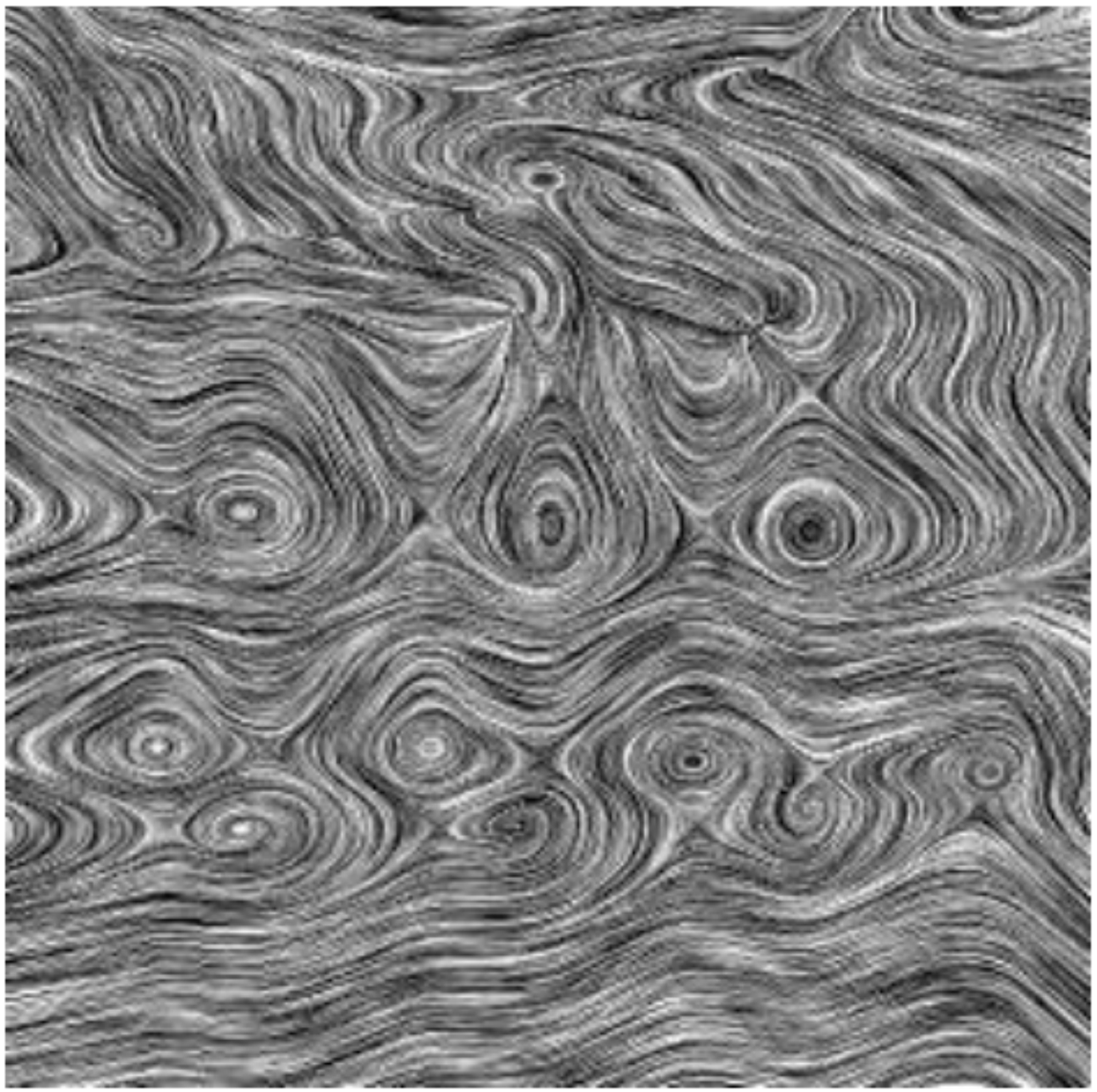
compute
streamlines

convolve
source of noise
along streamlines

Result

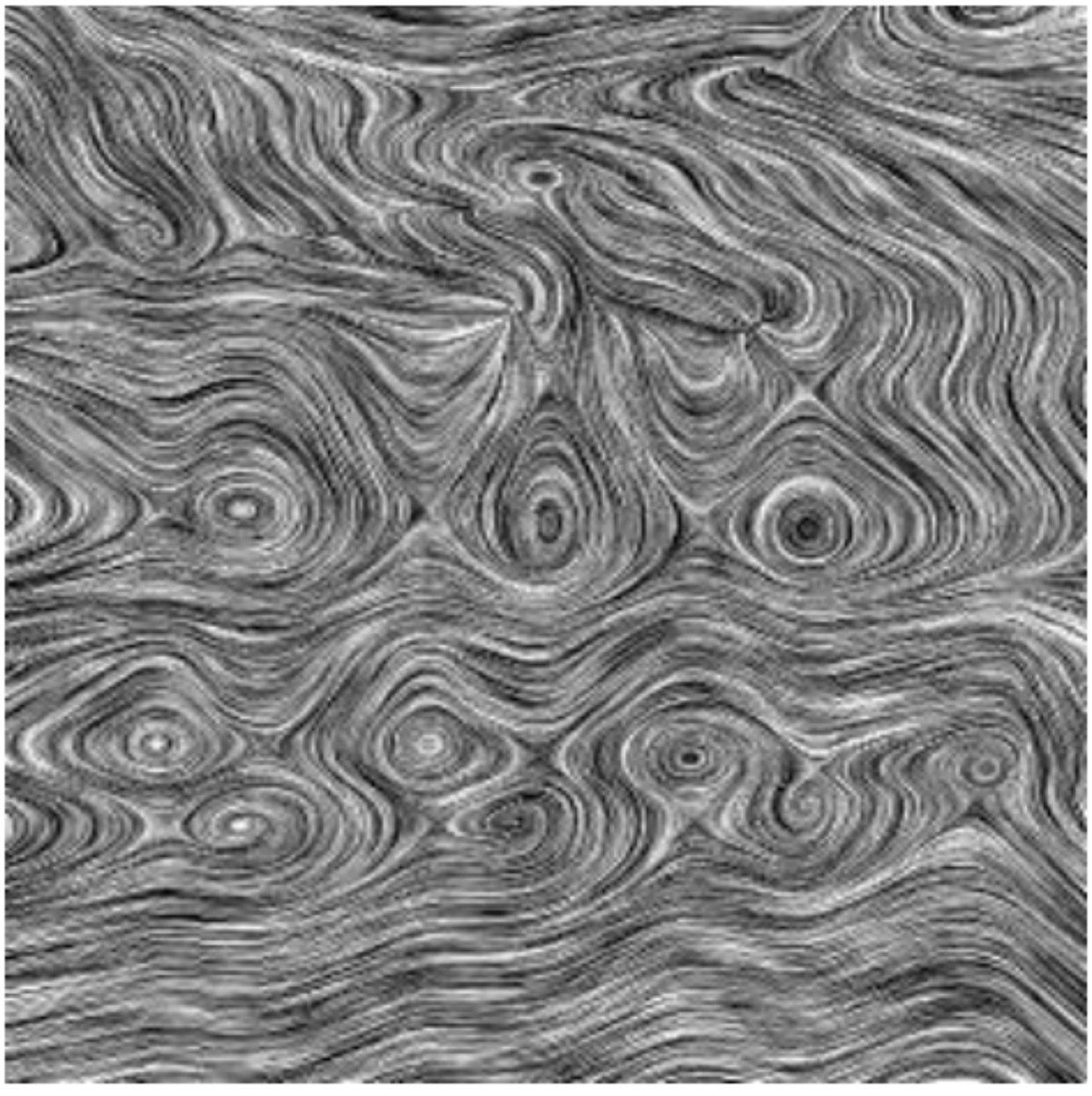
Line Integral Convolution





Advantages

- “Perfect” space usage
- Flow features are very apparent



Downsides

- No perception of velocity!
- No perception of direction!

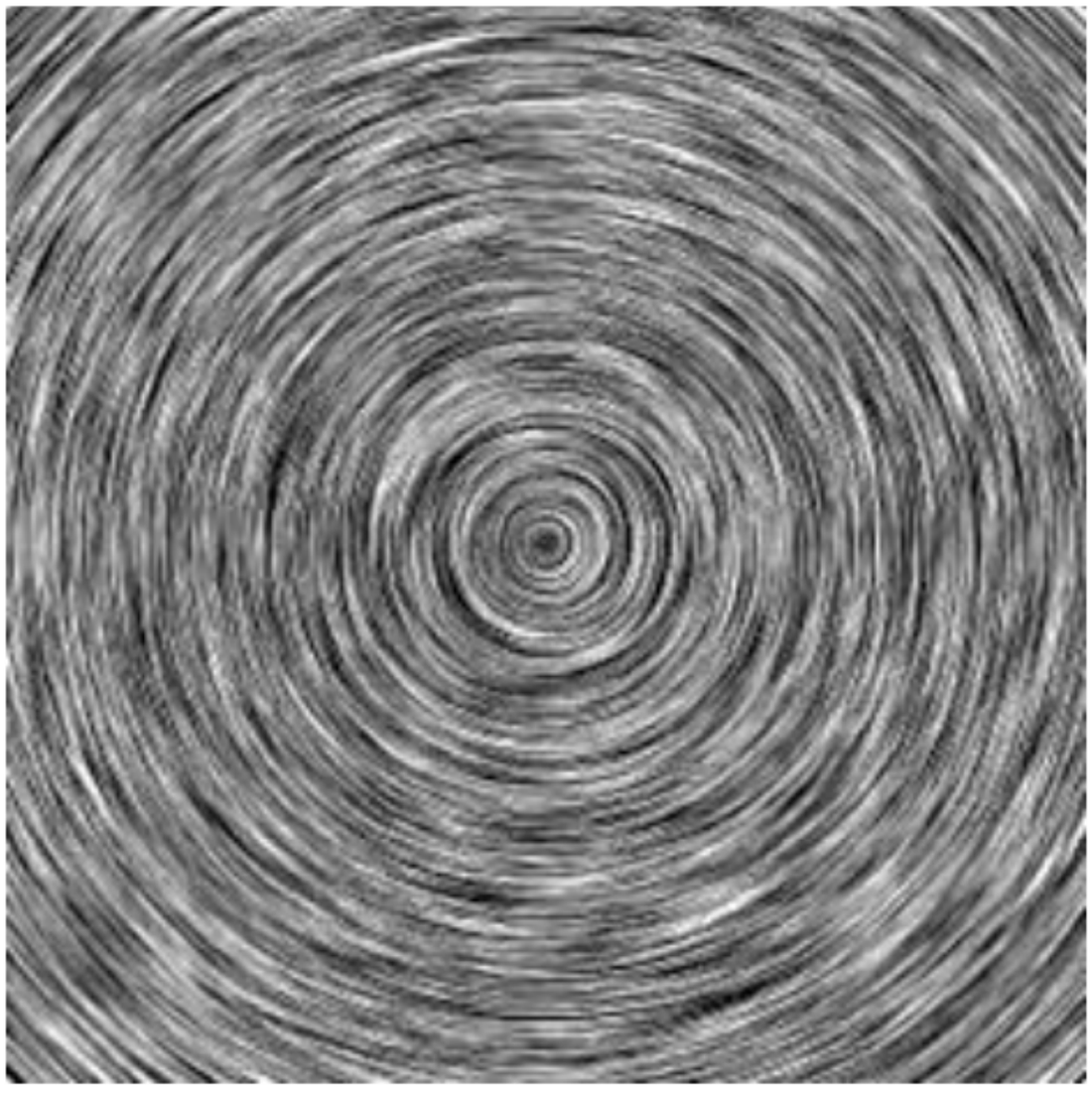


Image-Based Flow Visualization

- Adaptation of LIC for graphics cards
- Extremely fast and simple to implement
- Animation gives perception of velocity
- (But requires knowledge of computer graphics, which we're not assuming in this course)

Image-Based Flow Visualization

http://www.win.tue.nl/~vanwijk/ibfv/

Next class: methods to integrate ODEs