

# CS 247 – Scientific Visualization

## Lecture 27: Vector / Flow Visualization, Pt. 6

Markus Hadwiger, KAUST

# Reading Assignment #14 (until May 11)



## Read (required):

- Data Visualization book, Chapter 6.7
- J. van Wijk: *Image-Based Flow Visualization*,  
ACM SIGGRAPH 2002  
<http://www.win.tue.nl/~vanwijk/ibfv/ibfv.pdf>

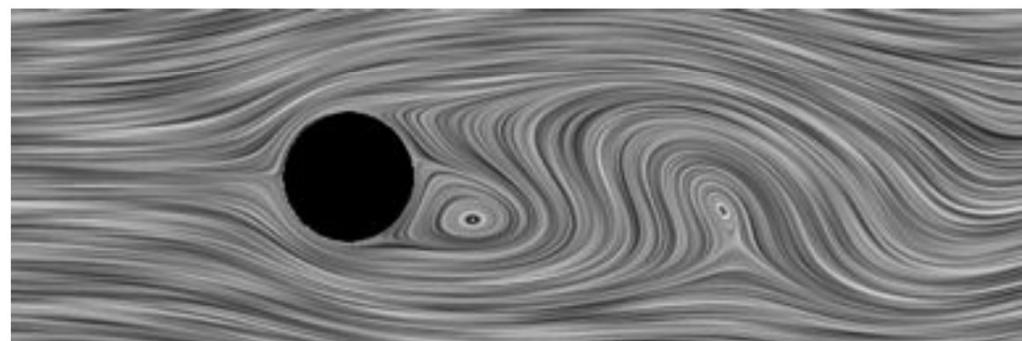
## Read (optional):

- T. Günther, A. Horvath, W. Bresky, J. Daniels, S. A. Buehler:  
*Lagrangian Coherent Structures and Vortex Formation in High Spatiotemporal-Resolution Satellite Winds of an Atmospheric Karman Vortex Street*, 2021  
<https://www.essoar.org/doi/10.1002/essoar.10506682.2>
- H. Bhatia, G. Norgard, V. Pascucci, P.-T. Bremer:  
*The Helmholtz-Hodge Decomposition – A Survey*, TVCG 19(8), 2013  
<https://doi.org/10.1109/TVCG.2012.316>
- Work through online tutorials of multi-variable partial derivatives, grad, div, curl, Laplacian:  
<https://www.khanacademy.org/math/multivariable-calculus/multivariable-derivatives>  
<https://www.youtube.com/watch?v=rB83DpBJQsE> (3Blue1Brown)
- Matrix exponentials:  
<https://www.youtube.com/watch?v=O850WBJ2ayo> (3Blue1Brown)

# Line Integral Convolution (LIC)

# Line Integral Convolution

- Line Integral Convolution (LIC)
  - Visualize dense flow fields by imaging its integral curves
  - Cover domain with a random texture (so called ‚input texture‘, usually stationary white noise)
  - Blur (convolve) the input texture along stream lines using a specified filter kernel
- Look of 2D LIC images
  - Intensity distribution along stream lines shows high correlation
  - No correlation between neighboring stream lines



# Line Integral Convolution I



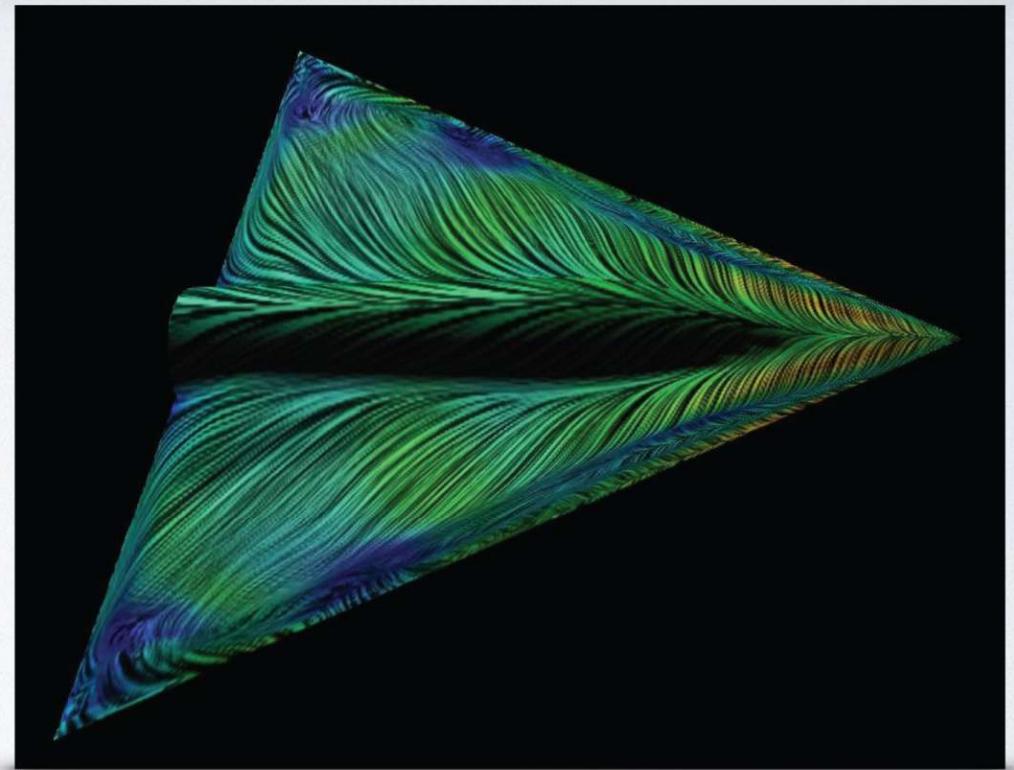
- Line Integral Convolution (LIC):
  - goal: general overview of flow
  - approach: use dense textures
  - idea: flow ↔ visual correlation



# Line Integral Convolution I



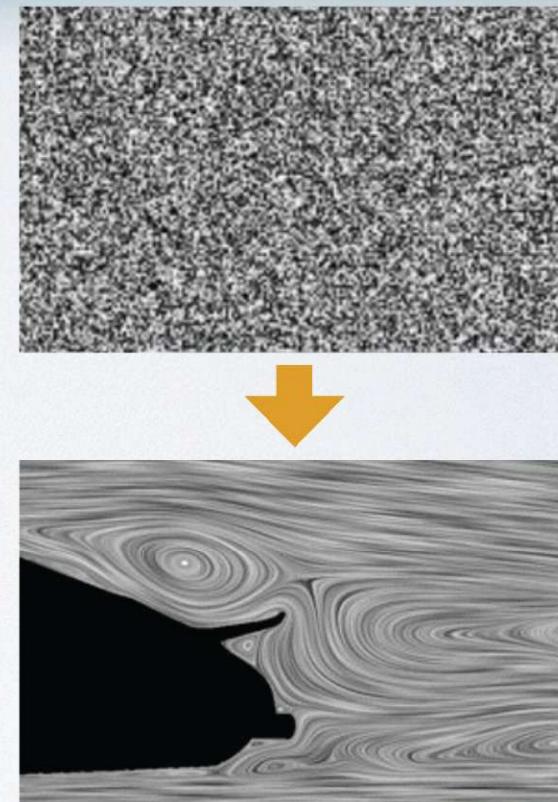
- Line Integral Convolution (LIC):
  - goal: general overview of flow
  - approach: use dense textures
  - idea: flow ↔ visual correlation



# Line Integral Convolution II



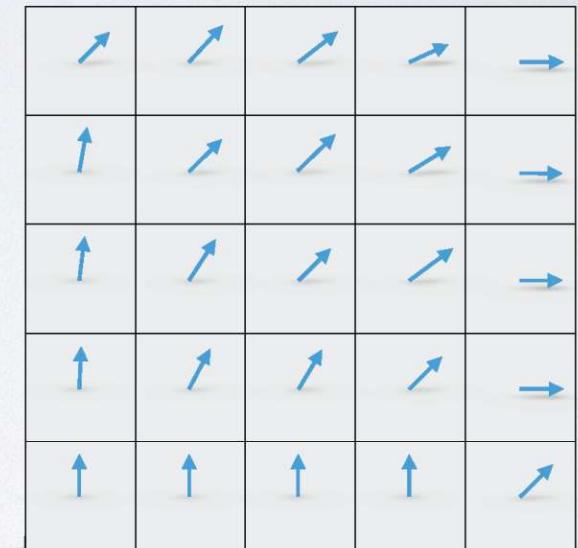
- Idea
  - global visualization technique
  - dense representation
  - start with random texture
  - smear along stream lines
- Only for stream lines!  
(steady flow, i.e. time-independent fields)





# Line Integral Convolution III

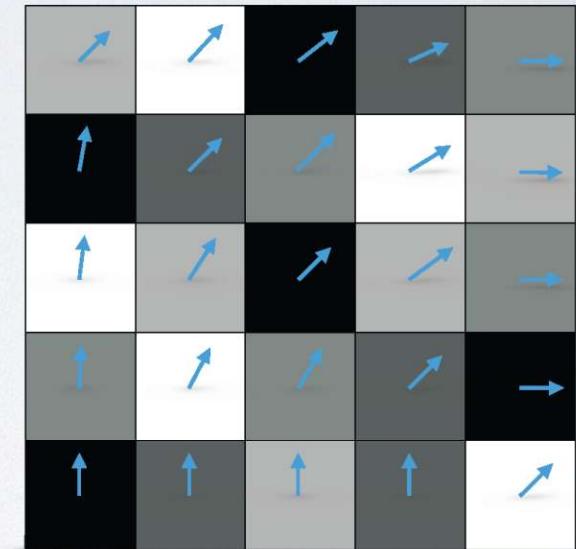
- How LIC works
  - visualize dense flow fields by imaging integral curves
  - cover domain with a random texture ('input texture', usually stationary white noise)
  - blur (convolve) the input texture along stream lines





# Line Integral Convolution III

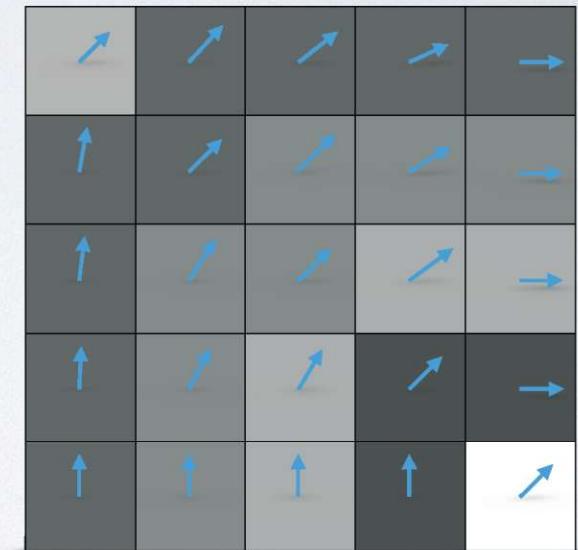
- How LIC works
  - visualize dense flow fields by imaging integral curves
  - cover domain with a random texture ('input texture', usually stationary white noise)
  - blur (convolve) the input texture along stream lines





# Line Integral Convolution III

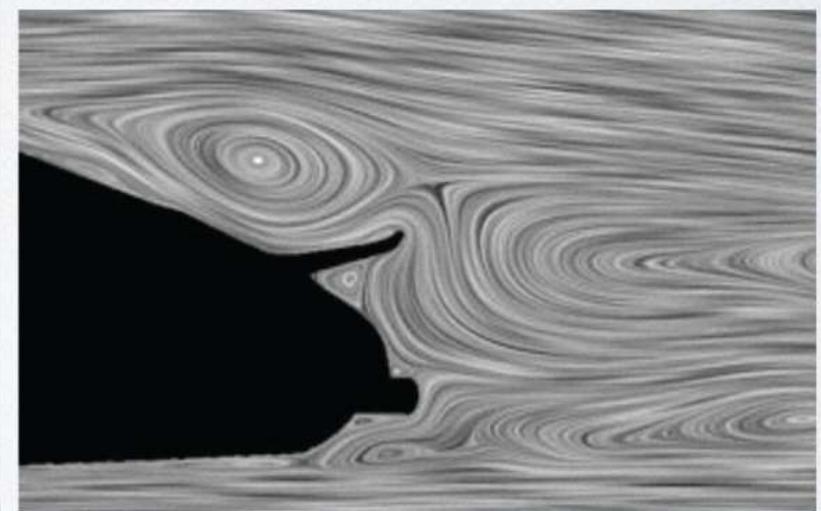
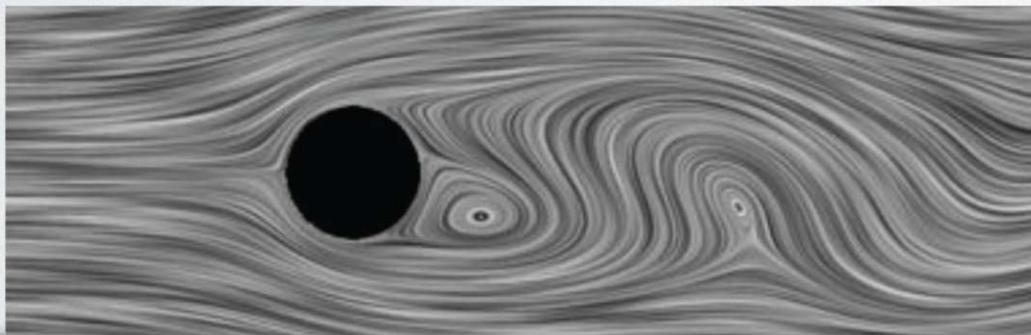
- How LIC works
  - visualize dense flow fields by imaging integral curves
  - cover domain with a random texture ('input texture', usually stationary white noise)
  - blur (convolve) the input texture along stream lines



# Line Integral Convolution IV



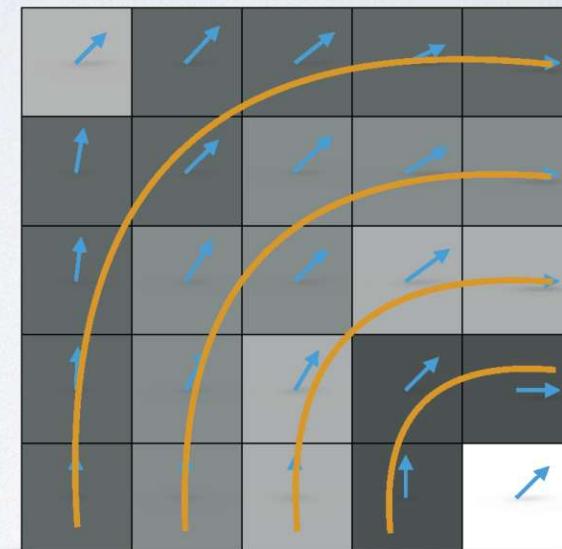
- Look of 2D LIC images
  - intensity along stream lines shows high correlation
  - no correlation between neighboring stream lines





# LIC Approach - Goal

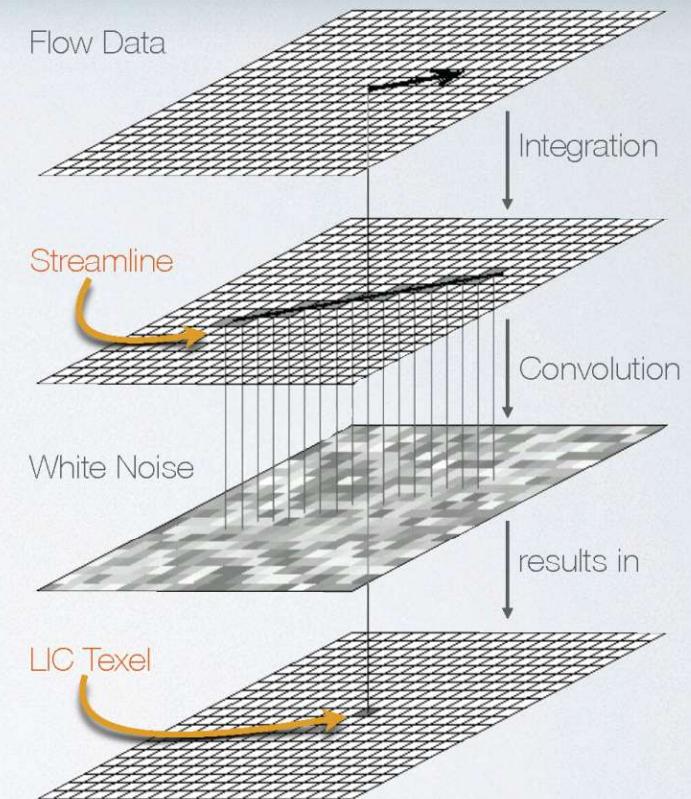
- For every texel: let the texture value
  - correlate with neighboring texture values along the flow (in flow direction)
  - not correlate with neighboring texture values across the flow (normal to flow direction)
- Result: along streamlines the texture values are correlated  $\Rightarrow$  visually coherent!





# LIC Approach - Steps

- Idea: “smear” white noise (no a priori correlations) along flow
- Calculation of a texture value:
  - follow streamline through point
  - filter white noise along streamline



# Convolution Example

## Gaussian Blur

[en.wikipedia.org/wiki/Gaussian.blur](https://en.wikipedia.org/wiki/Gaussian_blur)

Cut off filter kernel after an extent of, e.g.,  
3\*standard deviation in each direction

Example:

0.00000067	0.00002292	<b>0.00019117</b>	0.00038771	<b>0.00019117</b>	0.00002292	0.00000067
0.00002292	0.00078634	0.00655965	0.01330373	0.00655965	0.00078633	0.00002292
<b>0.00019117</b>	0.00655965	0.05472157	0.11098164	0.05472157	0.00655965	<b>0.00019117</b>
0.00038771	0.01330373	0.11098164	<b>0.22508352</b>	0.11098164	0.01330373	0.00038771
<b>0.00019117</b>	0.00655965	0.05472157	0.11098164	0.05472157	0.00655965	<b>0.00019117</b>
0.00002292	0.00078633	0.00655965	0.01330373	0.00655965	0.00078633	0.00002292
0.00000067	0.00002292	<b>0.00019117</b>	0.00038771	<b>0.00019117</b>	0.00002292	0.00000067

Note that 0.22508352 (the central one) is 1177 times larger than 0.00019117 which is just outside  $3\sigma$ .

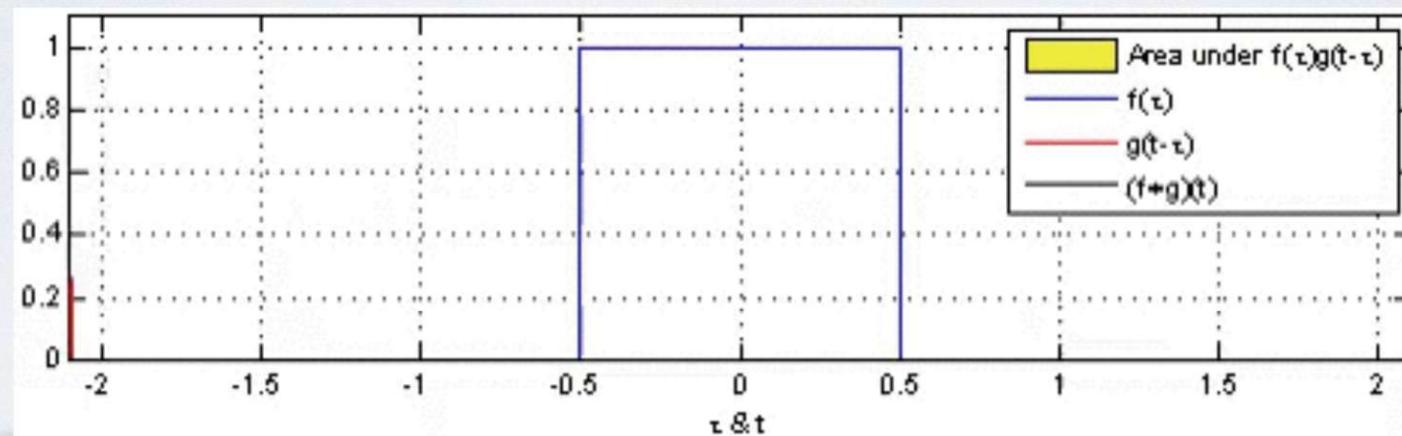
Can do multiple iterations to achieve  
larger effective filter size





# LIC Approach - 1D Convolution I

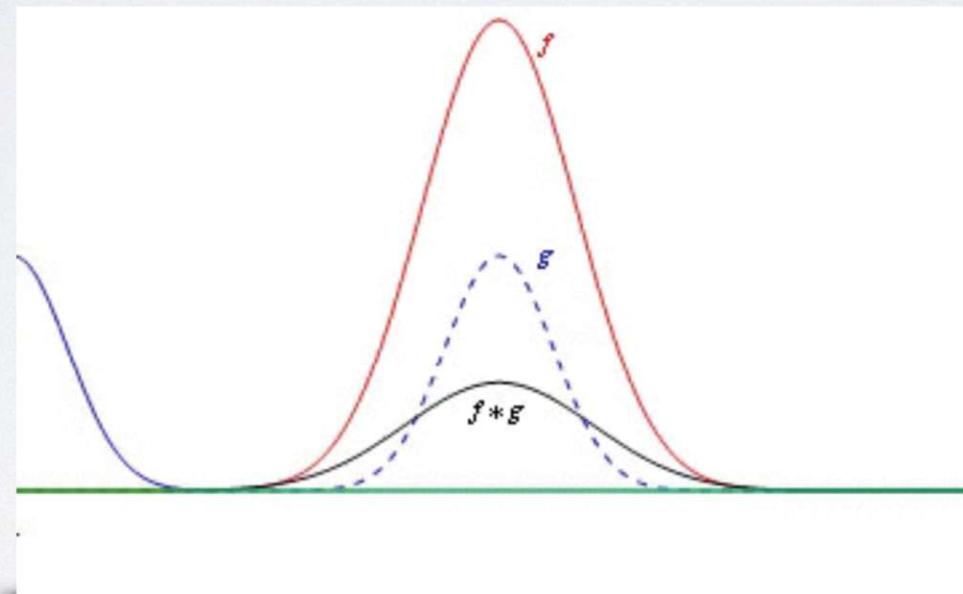
- Convolution defined as  $(f * g)(x) := \int_{\mathbb{R}^n} f(\tau)g(x - \tau)d\tau$





# LIC Approach - 1D Convolution II

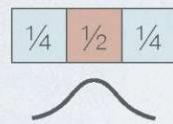
- Convolution defined as  $(f * g)(x) := \int_{\mathbb{R}^n} f(\tau)g(x - \tau)d\tau$



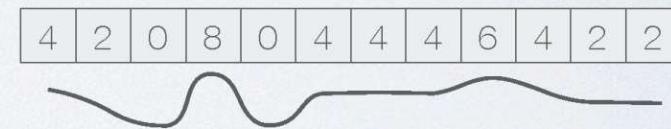
# LIC Approach - 1D Convolution III



$k(x)$  convolution kernel



$f(x)$  original signal



$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

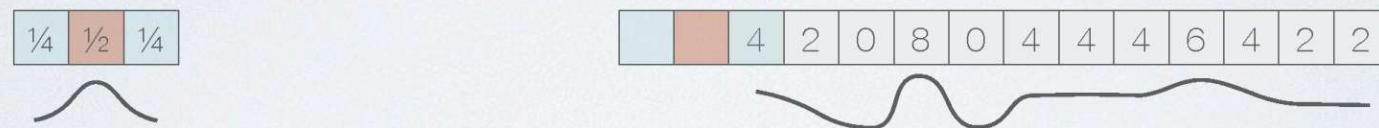
$(f * k)(x)$  smoothed signal



# LIC Approach - 1D Convolution III



$k(x)$  convolution kernel      common section L       $f(x)$  original signal



$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

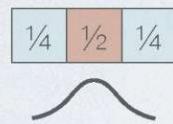
$(f * k)(x)$  smoothed signal





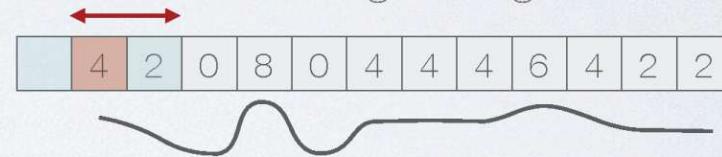
# LIC Approach - 1D Convolution III

$k(x)$  convolution kernel



common section L

$f(x)$  original signal



$$\frac{1}{4} \cdot 0 + \frac{1}{2} \cdot 4 + \frac{1}{4} \cdot 2$$

$(f * k)(x)$  smoothed signal



$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$



# LIC Approach - 1D Convolution III

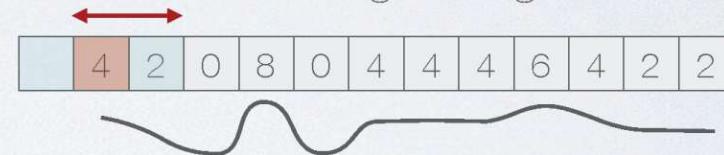
$k(x)$  convolution kernel

1/4	1/2	1/4
-----	-----	-----



common section L

$f(x)$  original signal



$$\frac{1}{4} \cdot 0 + \frac{1}{2} \cdot 4 + \frac{1}{4} \cdot 2$$

$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

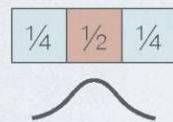
$(f * k)(x)$  smoothed signal

3									
---	--	--	--	--	--	--	--	--	--

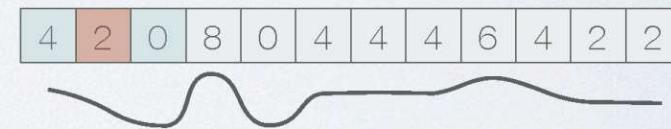


# LIC Approach - 1D Convolution III

$k(x)$  convolution kernel



$f(x)$  original signal



$$\frac{1}{4} \cdot 4 + \frac{1}{2} \cdot 2 + \frac{1}{4} \cdot 0$$

$(f * k)(x)$  smoothed signal

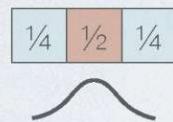


$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

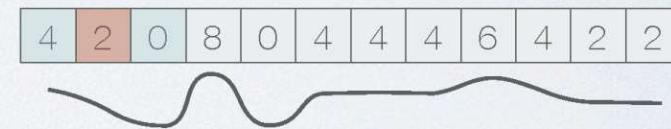


# LIC Approach - 1D Convolution III

$k(x)$  convolution kernel



$f(x)$  original signal



$$\frac{1}{4} \cdot 4 + \frac{1}{2} \cdot 2 + \frac{1}{4} \cdot 0$$

$(f * k)(x)$  smoothed signal

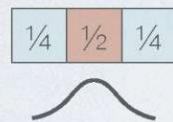


$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

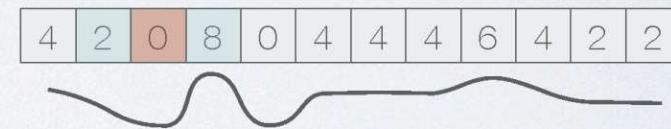


# LIC Approach - 1D Convolution III

$k(x)$  convolution kernel



$f(x)$  original signal



$$\frac{1}{4} \cdot 2 + \frac{1}{2} \cdot 0 + \frac{1}{4} \cdot 8$$

$(f * k)(x)$  smoothed signal

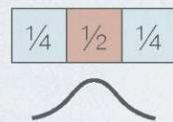


$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

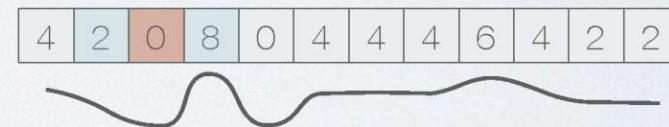


# LIC Approach - 1D Convolution III

$k(x)$  convolution kernel



$f(x)$  original signal



$$\frac{1}{4} \cdot 2 + \frac{1}{2} \cdot 0 + \frac{1}{4} \cdot 8$$

$(f * k)(x)$  smoothed signal

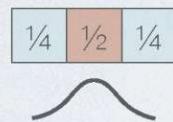


$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

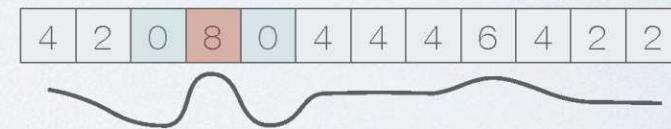
# LIC Approach - 1D Convolution III



$k(x)$  convolution kernel



$f(x)$  original signal



$$\frac{1}{4} \cdot 0 + \frac{1}{2} \cdot 8 + \frac{1}{4} \cdot 0$$

$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

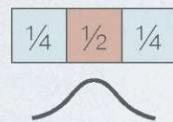
$(f * k)(x)$  smoothed signal



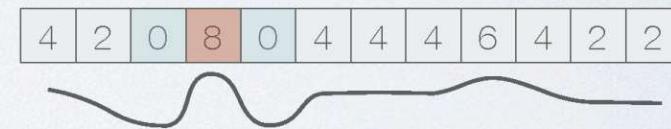


# LIC Approach - 1D Convolution III

$k(x)$  convolution kernel



$f(x)$  original signal



$$\frac{1}{4} \cdot 0 + \frac{1}{2} \cdot 8 + \frac{1}{4} \cdot 0$$

$(f * k)(x)$  smoothed signal

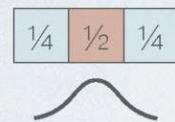


$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

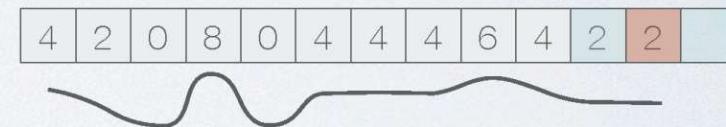
# LIC Approach - 1D Convolution III



$k(x)$  convolution kernel

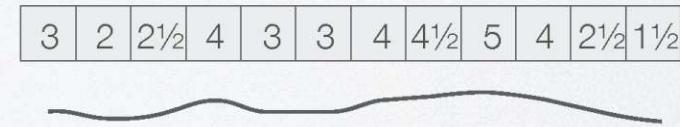


$f(x)$  original signal

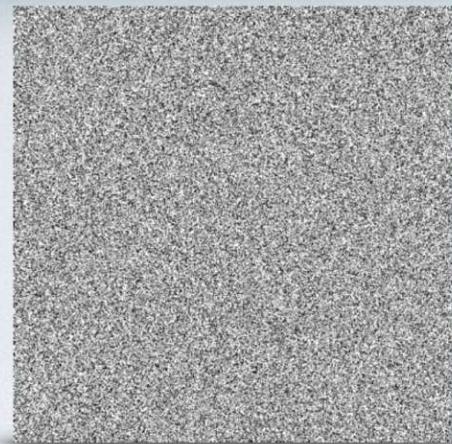


$$(f * k)(x) = \int_{-L/2}^{L/2} f(\tau)k(x - \tau)d\tau$$

$(f * k)(x)$  smoothed signal



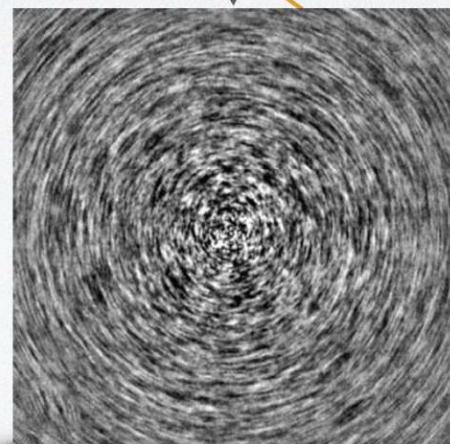
# LIC Approach - 1D Convolution IV



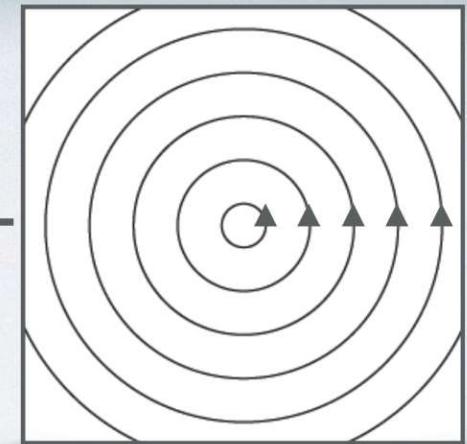
Input Noise

Convolution

$$\int T(\mathbf{x}(t+s))k(s)ds$$

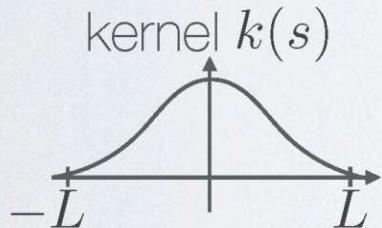


Final Image



Vector Field

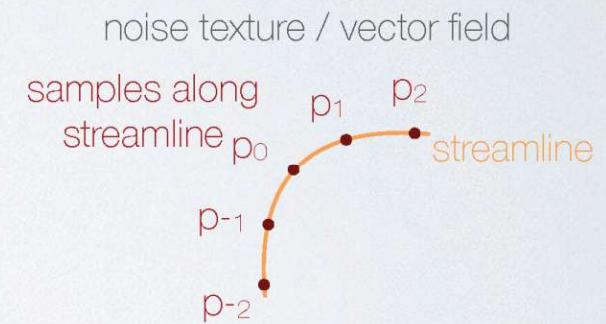
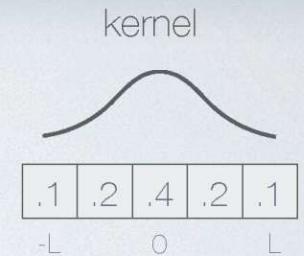
Streamline Tracing



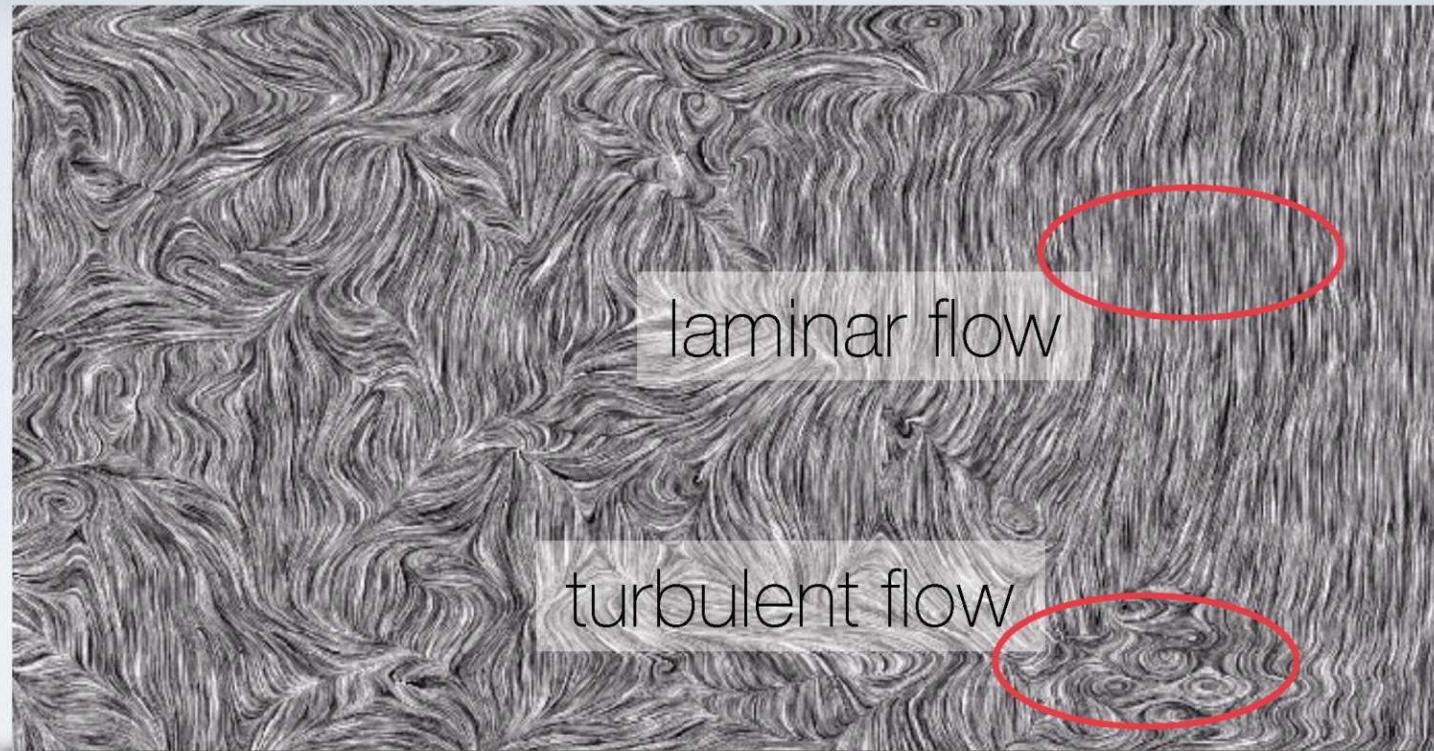
# LIC - Algorithm



```
for each pixel //perfect fit for fragment shader  
  
    t = texture( position, noise_texture );  
  
    smoothed_value = kernel_value(center) * t;  
    P+ = p- = position;  
  
    for 1 to L // loop over kernel  
  
        v+ = texture( p+, vector_texture );  
        p+ = streamlineIntegration(p+, v+);  
        smoothed_value +=  
            kernel_value * texture( p+, noise_texture );  
  
        v- = -texture( p-, vector_texture );  
        p- = streamlineIntegration(p-, v-);  
        smoothed_value +=  
            kernel_value * texture( p-, noise_texture );
```



# LIC - 2D Example





# Linear Algebra Approach (1)

- Toeplitz matrix: constant diagonals

$\mathbf{T} := (t_{ij})$  with  $t_{ij} := t_{i-j}$

$$\mathbf{T}^{N \times N} := \begin{bmatrix} t_0 & t_{(-1)} & t_{(-2)} & \cdots & t_{(-(N-2))} & t_{(-(N-1))} \\ t_1 & t_0 & t_{(-1)} & \cdots & t_{(-(N-3))} & t_{(-(N-2))} \\ t_2 & t_1 & t_0 & \cdots & t_{(-(N-4))} & t_{(-(N-3))} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ t_{N-2} & t_{N-3} & t_{N-4} & \cdots & t_0 & t_{(-1)} \\ t_{N-1} & t_{N-2} & t_{N-3} & \cdots & t_1 & t_0 \end{bmatrix}$$



# Linear Algebra Approach (2)

- Circulant matrix: special case of Toeplitz matrix

$\mathbf{C} := (c_{ij})$  where  $c_{ij} := c_{(i-j) \bmod N}$

$$\mathbf{C}^{N \times N} := \begin{bmatrix} c_0 & c_{N-1} & c_{N-2} & \dots & c_2 & c_1 \\ c_1 & c_0 & c_{N-1} & \dots & c_3 & c_2 \\ c_2 & c_1 & c_0 & \dots & c_4 & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{N-2} & c_{N-3} & c_{N-4} & \dots & c_0 & c_{N-1} \\ c_{N-1} & c_{N-2} & c_{N-3} & \dots & c_1 & c_0 \end{bmatrix}$$

- Periodic convolution: multiply  $\mathbf{C}$  with (periodic) signal in column vector
- The Fourier transform *diagonalizes* circulant matrices

# Thank you.

## Thanks for material

- Helwig Hauser
- Eduard Gröller
- Daniel Weiskopf
- Torsten Möller
- Ronny Peikert
- Philipp Muigg
- Christof Rezk-Salama