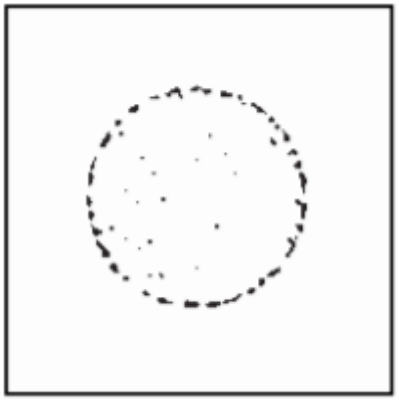
Coherent Line Drawing

HENRY KANG, SEUNGYOUG LEE, CHARLES K. CHUI

Introduction



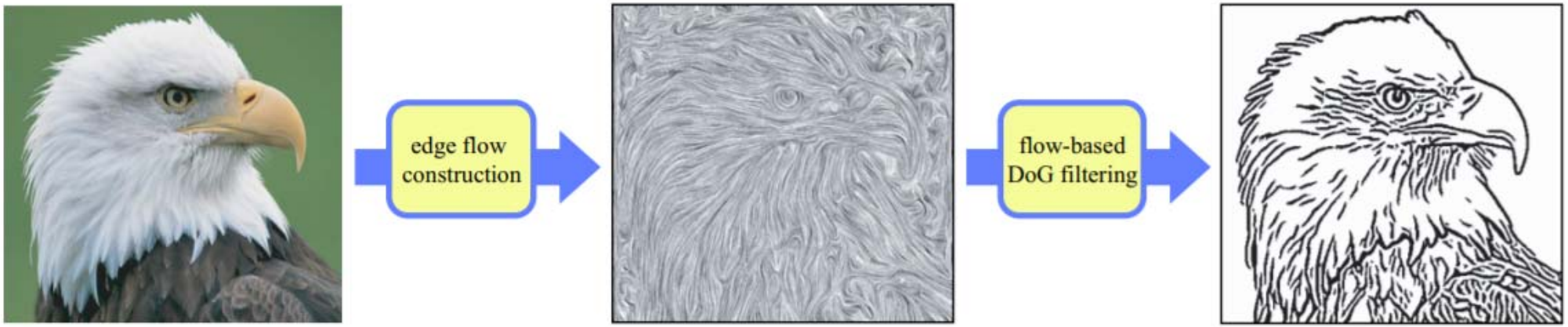
- Automatically generates a line drawing from a photograph
- Extract a set of coherent, smooth and stylistic lines
- Lines are used to convey the shape of an object

input

output

Process Overview

- Edge flow construction: smooth, feature-preserving local edge flow
- Flow-based DoG filtering: produces the line illustration



(a) Input image (b) Edge Tangent Flow (c) Line drawing

Flow Construction – Edge Tangent Flow

- Use Sobel to get the image gradient

$$\mathbf{g}(\mathbf{x}) = \nabla I(\mathbf{x})$$

- Edge Tangent: A vector perpendicular to the image gradient
  t(x) = rotate(g(x), 90)

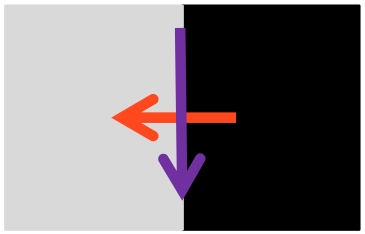
Sobel Filter

-1	0	1
-2	0	2
-1	0	1

Horizontal

-1	0	0
-2	0	0
-1	0	0

Result < 0



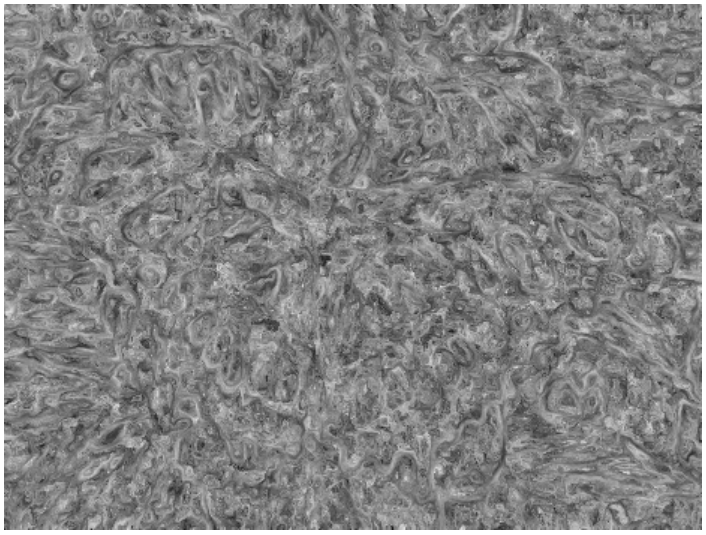
1	2	1
0	0	0
-1	-2	-1

Vertical

1	0	1
0	0	0
-1	0	-1

Result = 0

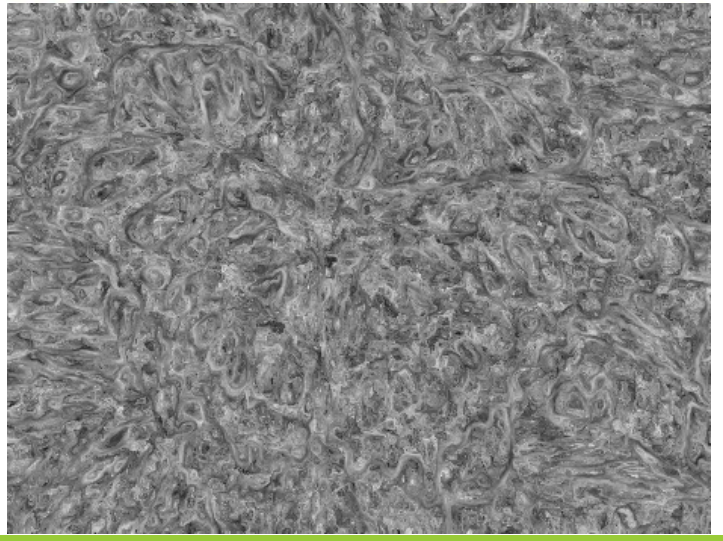
Edge Tangent Result



ETF Construction Filter

- Smooth the vector field
- Weak edges follow the neighboring dominant ones
- Preserve sharp corners
- EFT construction filter:

$$\mathbf{t}^{new}(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{y} \in \Omega(\mathbf{x})} \phi(\mathbf{x}, \mathbf{y}) \mathbf{t}^{cur}(\mathbf{y}) w_s(\mathbf{x}, \mathbf{y}) w_m(\mathbf{x}, \mathbf{y}) w_d(\mathbf{x}, \mathbf{y})$$



ETF Construction Filter

- $\Omega(\mathbf{x})$: The neighborhood of x
- k : normalizing term
- Spatial weight function w_s: (in a specific distance)

$$w_s(\mathbf{x},\mathbf{y}) = \begin{cases} 1 & \text{if } ||\mathbf{x}-\mathbf{y}|| < r, \\ 0 & \text{otherwise.} \end{cases}$$

- Magnitude weight function w_m : (follow the dominant vector)

$$w_m(\mathbf{x},\mathbf{y}) = \frac{1}{2}\left(1+\tanh[\eta\cdot(\hat{g}(\mathbf{y})-\hat{g}(\mathbf{x}))]\right)$$

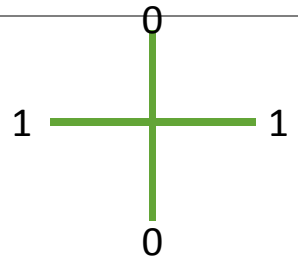
- $\hat{g}(\mathbf{z})$: the normalized gradient magnitude at z

$$\mathbf{t}^{new}(\mathbf{x}) = \frac{1}{k}\sum_{\mathbf{y}\in\Omega(\mathbf{x})} \phi(\mathbf{x},\mathbf{y})\mathbf{t}^{cur}(\mathbf{y})w_s(\mathbf{x},\mathbf{y})w_m(\mathbf{x},\mathbf{y})w_d(\mathbf{x},\mathbf{y})$$

ETF Construction Filter

- Direction weight function w_d :

$$w_d(\mathbf{x}, \mathbf{y}) = |\mathbf{t}^{cur}(\mathbf{x}) \cdot \mathbf{t}^{cur}(\mathbf{y})|$$



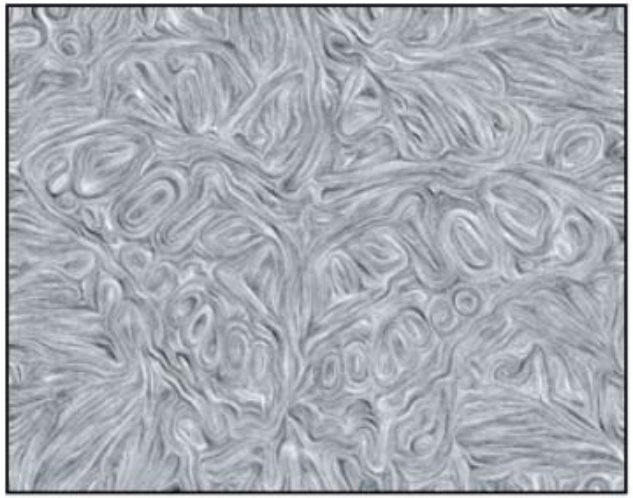
- Sign function $\phi(\mathbf{x}, \mathbf{y})$:

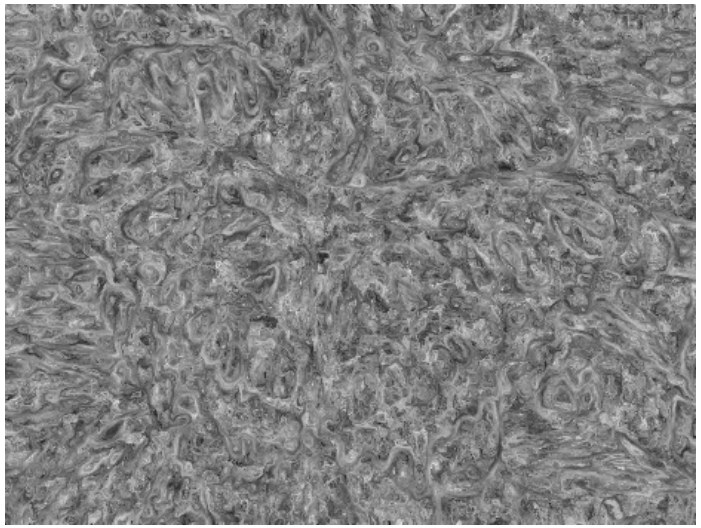
$$\phi(\mathbf{x}, \mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{t}^{cur}(\mathbf{x}) \cdot \mathbf{t}^{cur}(\mathbf{y}) > 0, \\ -1 & \text{otherwise.} \end{cases}$$

- Typically iterate 2~3 times

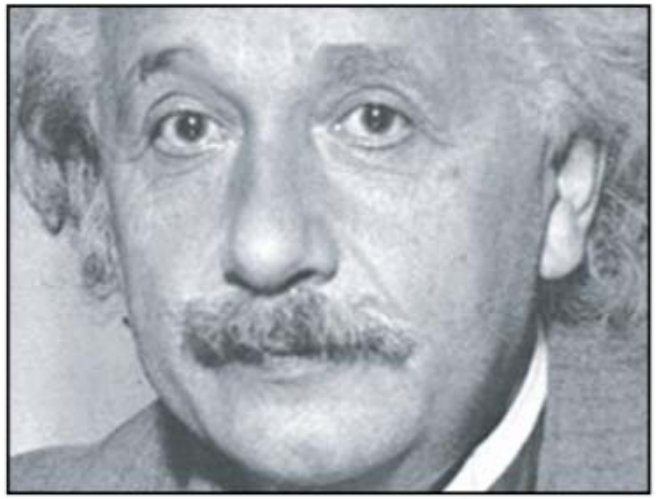
$$\mathbf{t}^{new}(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{y} \in \Omega(\mathbf{x})} \phi(\mathbf{x}, \mathbf{y}) \mathbf{t}^{cur}(\mathbf{y}) w_s(\mathbf{x}, \mathbf{y}) w_m(\mathbf{x}, \mathbf{y}) w_d(\mathbf{x}, \mathbf{y})$$

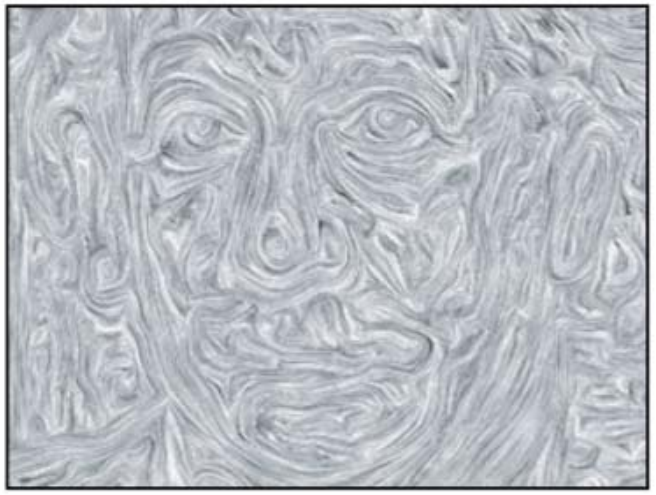
ETF Result





ETF Result



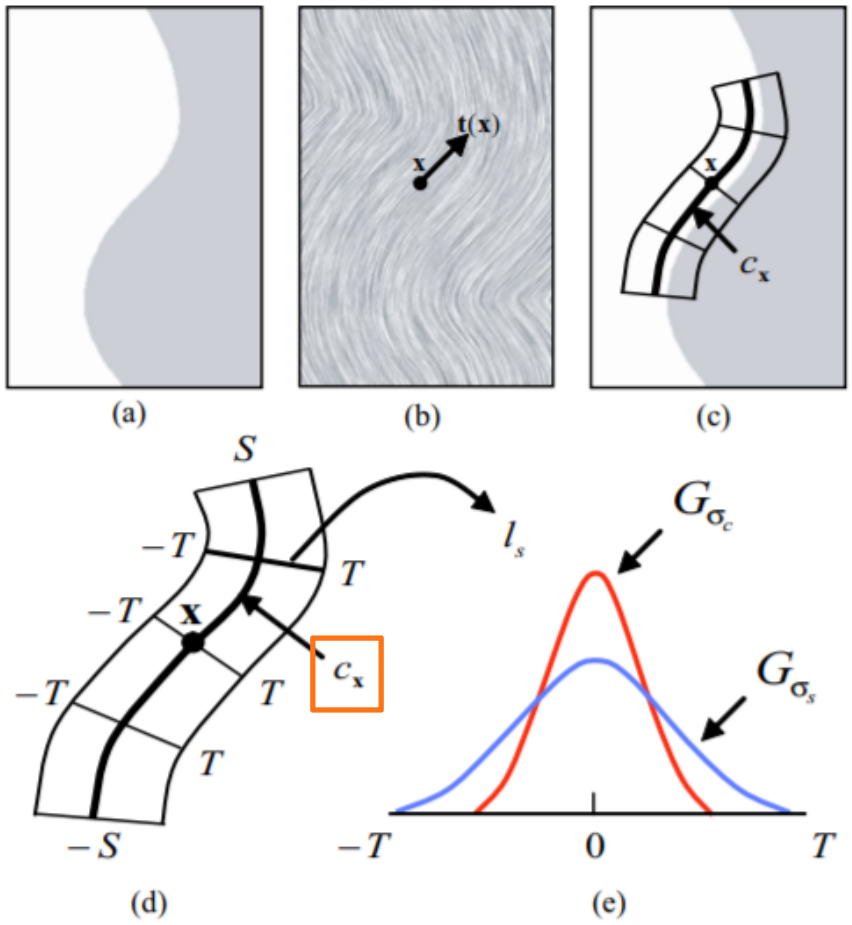


Line Construction

- Accumulate along $c_{\mathbf{x}}$ to get the value at x
- The value of every point on $c_{\mathbf{x}}$ is defined as:

$$F(s) = \int_{-T}^{T} I(l_s(t))f(t)dt$$

- $c_{\mathbf{x}}(s)$: the integral curve at x
  s is an arc-length parameter
- l_s : centered at $c_{\mathbf{x}}(s)$, and parallel to the gradient vector



Line Construction

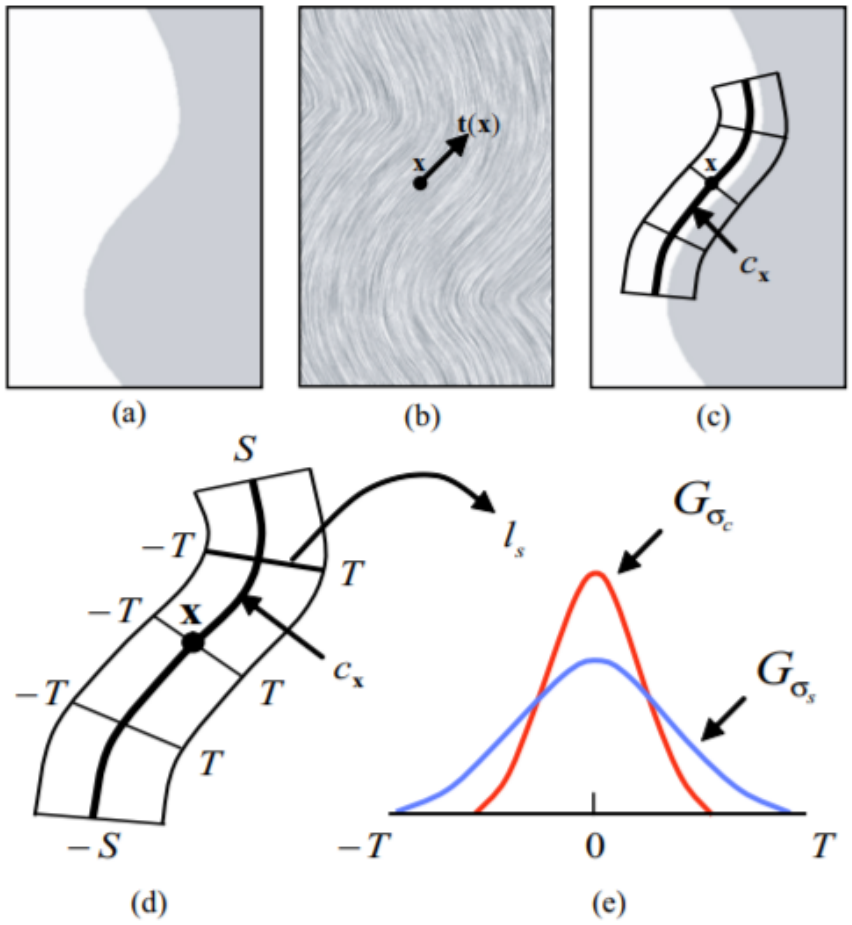
- Flow-bases Difference-of-Gaussians filtering:

$$F(s) = \int_{-T}^{T} I(l_s(t)) f(t) dt$$

- Difference-of-Gaussians filter $f(t)$:

$$f(t) = G_{\sigma_c}(t) - \rho \cdot G_{\sigma_s}(t)$$

- ρ : controls the level of noise detected
  default value 0.997
- $\sigma_s = 1.6\sigma_c$
  make to shape of f close to Laplacian-of-Gaussian



Flow-bases Difference-of-Gaussians filtering

- Feature enhancement algorithm
- The Laplacian of Gaussian is useful for detecting edges
- The input image is the edge tangent flow

Generate Lines

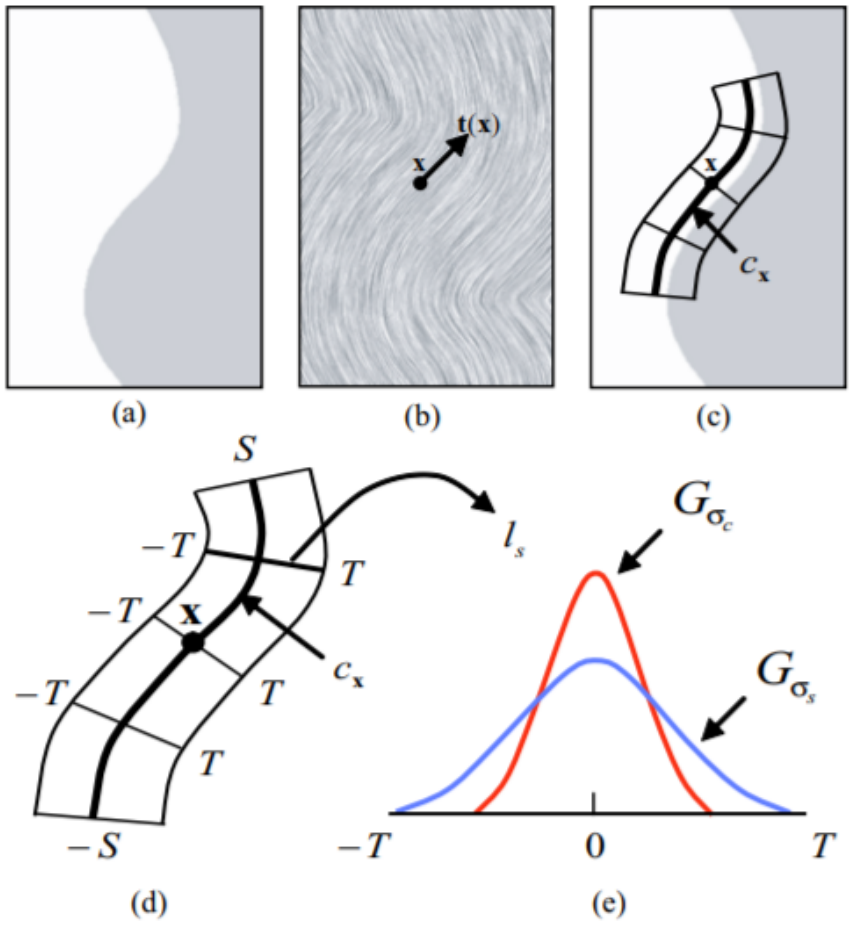
- Accumulate F(s) along $c_{\mathbf{x}}$:

$$H(\mathbf{x}) = \int_{-S}^{S} G_{\sigma_m}(s) F(s) ds$$

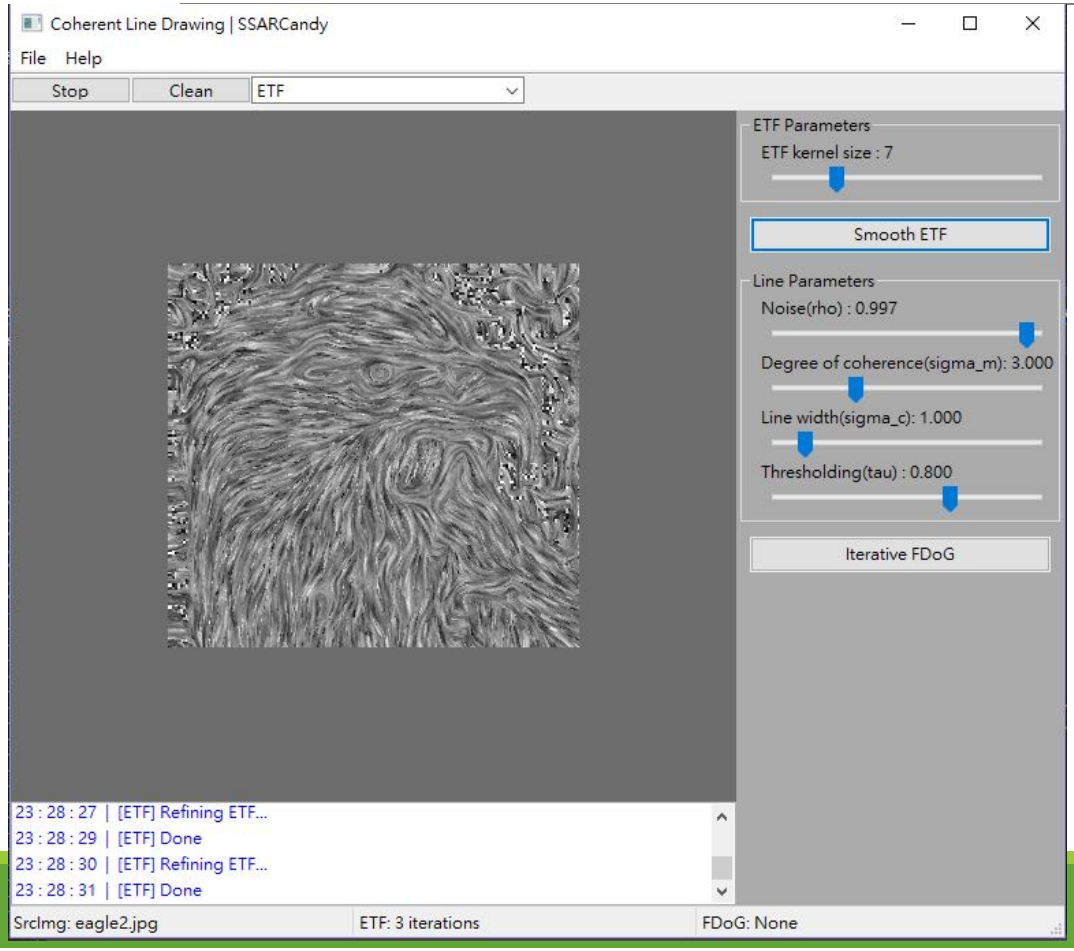
- σ_m : determines the size of S
- Convert H(x) to a black-and-white image

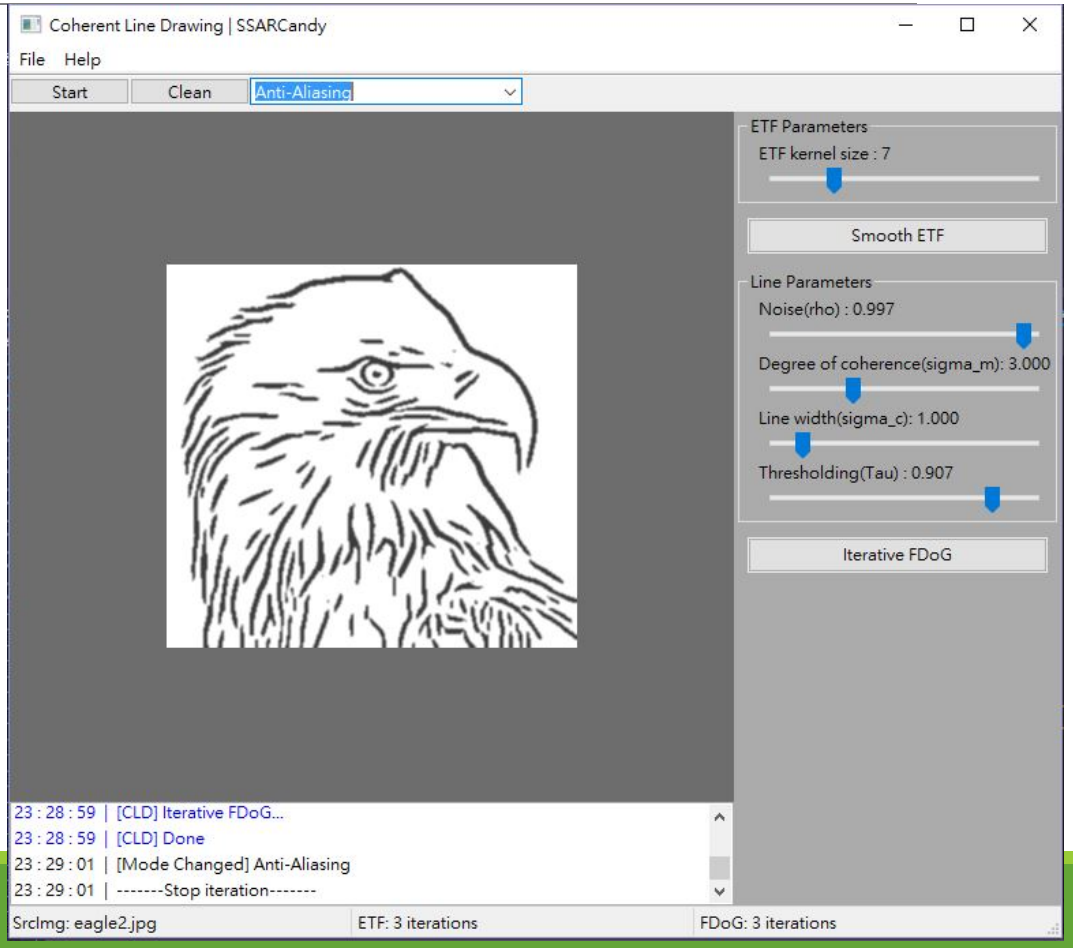
$$\tilde{H}(\mathbf{x}) = \begin{cases} 0 & \text{if } H(\mathbf{x}) < 0 \text{ and } 1 + \tanh(H(\mathbf{x})) < \tau, \\ 1 & \text{otherwise.} \end{cases}$$

- Typically iterate 2~3 times

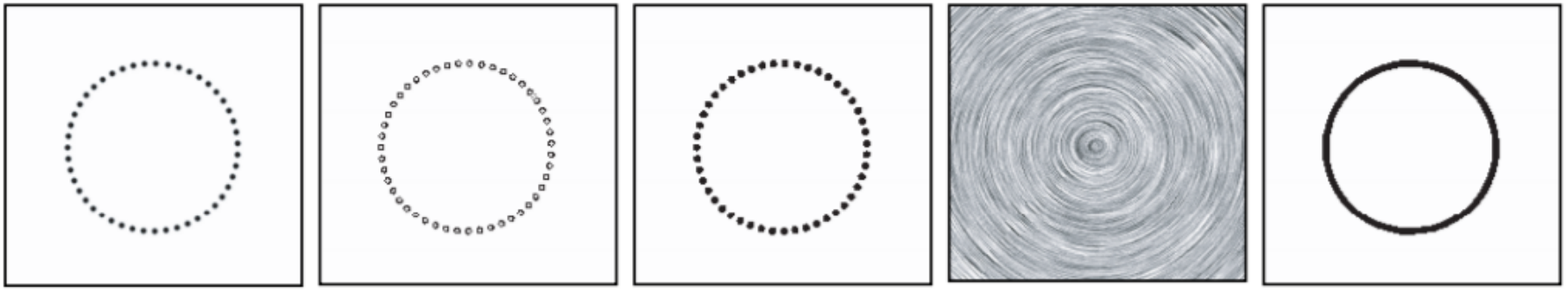


Demo

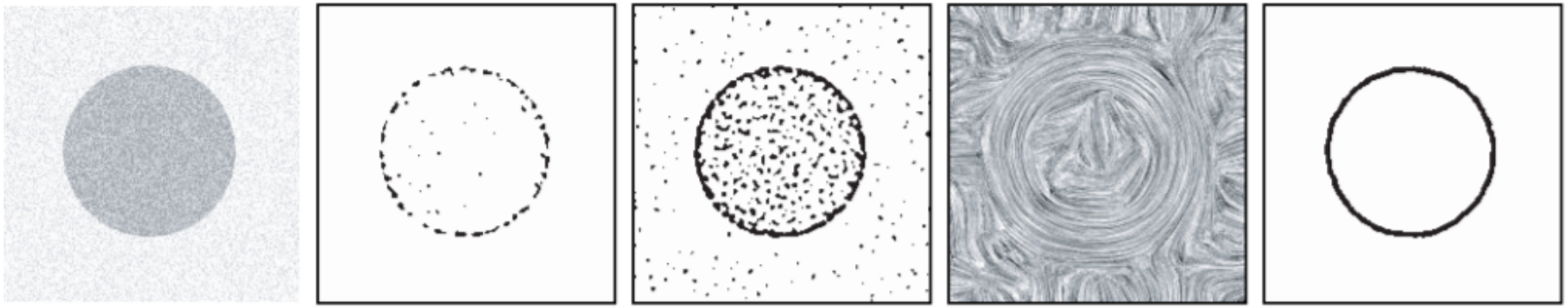

Coherent Line Drawing | SSARCandy
File Help
Stop
Clean
ETF
ETF Parameters
ETF kernel size : 7
Smooth ETF
Line Parameters
Noise(rho) : 0.997
Degree of coherence(sigma_m): 3.000
Line width(sigma_c): 1.000
Thresholding(tau) : 0.800
Iterative FDoG
23 : 28 : 27 | [ETF] Refining ETF...
23 : 28 : 29 | [ETF] Done
23 : 28 : 30 | [ETF] Refining ETF...
23 : 28 : 31 | [ETF] Done
Srclmg: eagle2.jpg
ETF: 3 iterations
FDoG: None


Coherent Line Drawing | SSARCandy
File Help
Start
Clean
Anti-Aliasing
ETF Parameters
ETF kernel size : 7
Smooth ETF
Line Parameters
Noise(rho) : 0.997
Degree of coherence(sigma_m): 3.000
Line width(sigma_c): 1.000
Thresholding(Tau) : 0.907
Iterative FDoG
23 : 28 : 59 | [CLD] Iterative FDoG...
23 : 28 : 59 | [CLD] Done
23 : 29 : 01 | [Mode Changed] Anti-Aliasing
23 : 29 : 01 | -------Stop iteration-------
Srclmg: eagle2.jpg
ETF: 3 iterations
FDoG: 3 iterations

Isolated Points and Noise

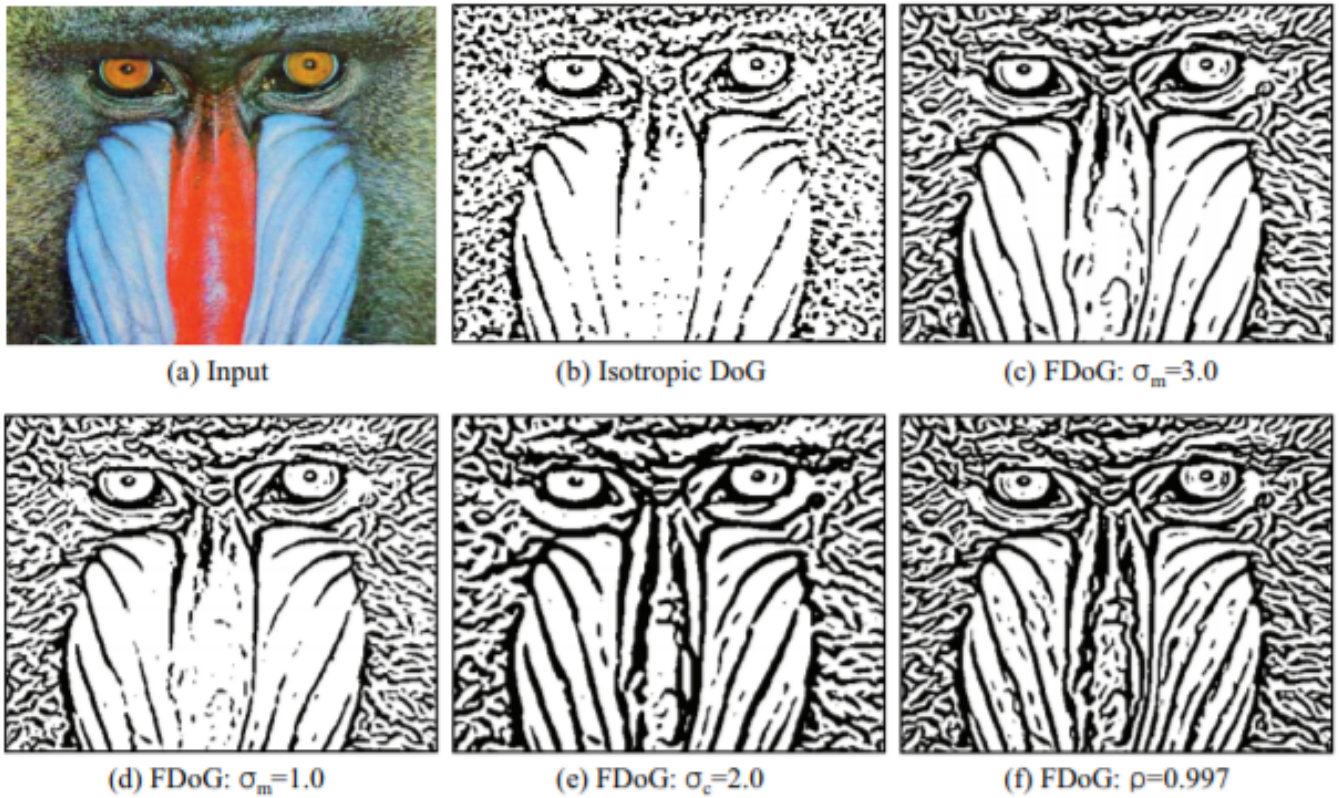


(a) Input (b) Canny (c) Isotropic DoG (d) ETF (e) FDoG



(a) Input (b) Isotropic DoG (c) Isotropic DoG (d) ETF (e) FDoG

Results of different parameters



(a) Input (b) Isotropic DoG (c) FDoG: σ_m=3.0

(d) FDoG: σ_m=1.0 (e) FDoG: σ_c=2.0 (f) FDoG: ρ=0.997

Iterations of FDoG

(a) 1st iteration (b) 2nd iteration (c) 3rd iteration

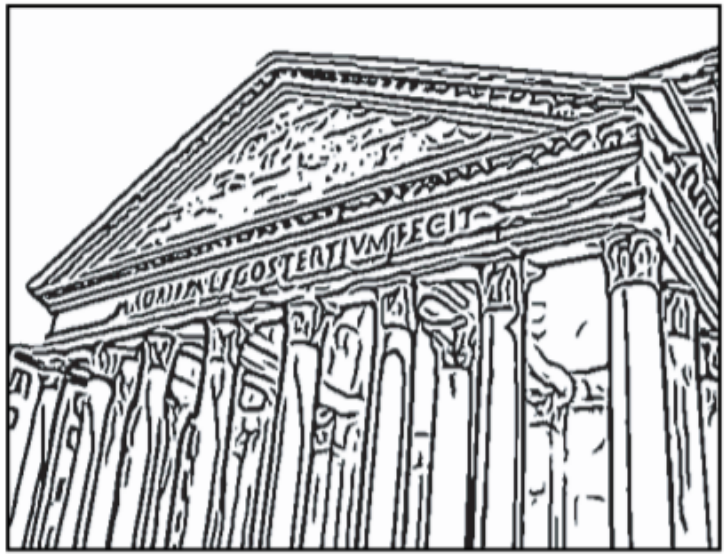
Results

(a) Lena

(b) Lighthouse

(c) Tiger

Results



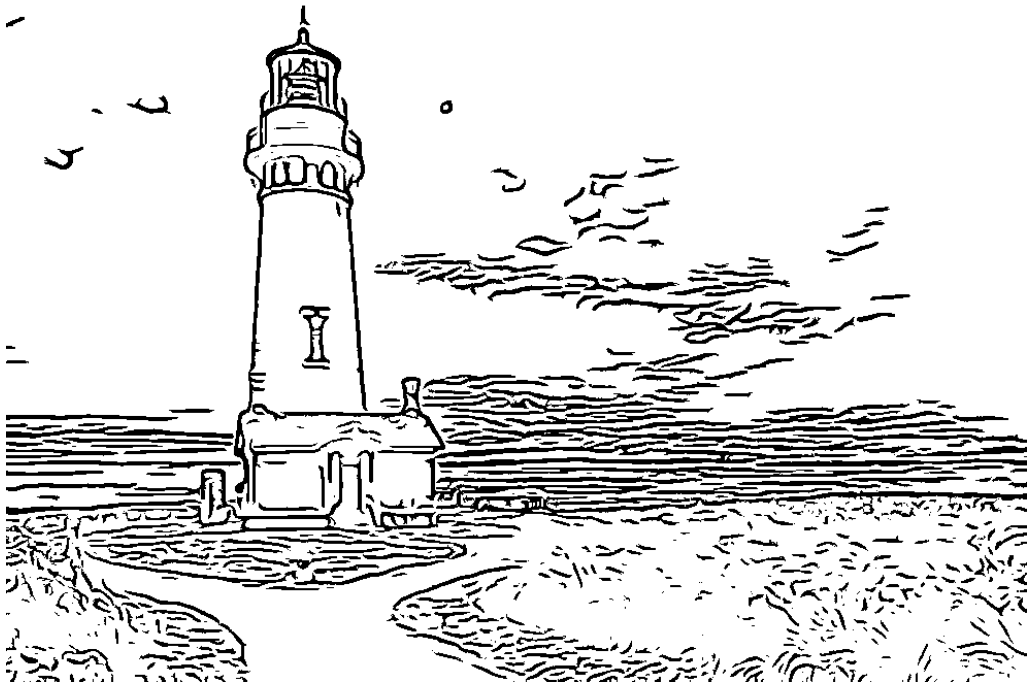
(d) Pantheon

(e) Flowers

(f) Paolina

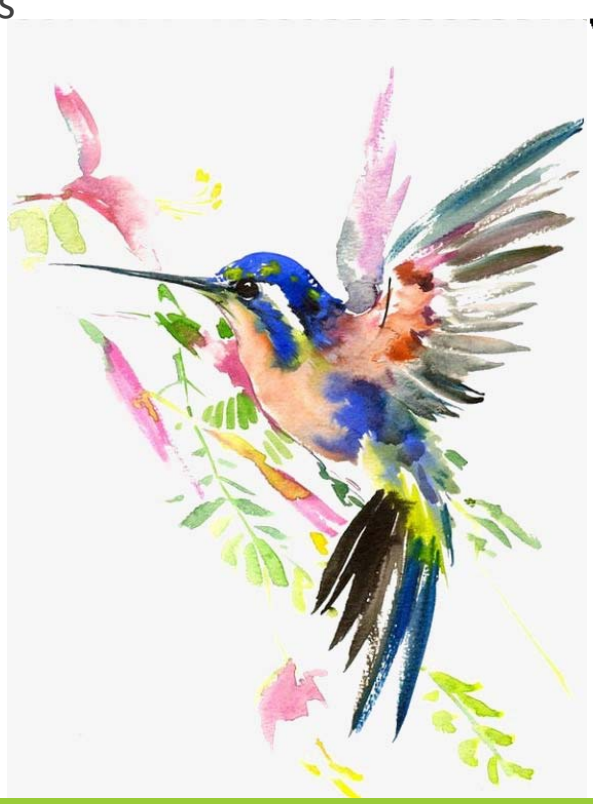
Limitations

- A high-contrast background will be filled with lines



Limitations

- The lines are formed as pixel aggregates (not strokes)
- FDoG filter operates on a local kernel thus a global-scale subjective contour may be hard to detect



Q&A

Thanks for listening!