Canny Edge Detection

Mohammad Nayeem Teli

Optimal Edge Detection: Canny

Assume:

- Linear filtering
- Additive iid Gaussian noise

Edge detector should have:

- Good Detection. Filter responds to edge, not noise.
- Good Localization: detected edge near true edge.
- Single Response: one per edge.

Optimal Edge Detection: Canny (continued)

Optimal Detector is approximately Derivative of Gaussian.

Detection/Localization trade-off

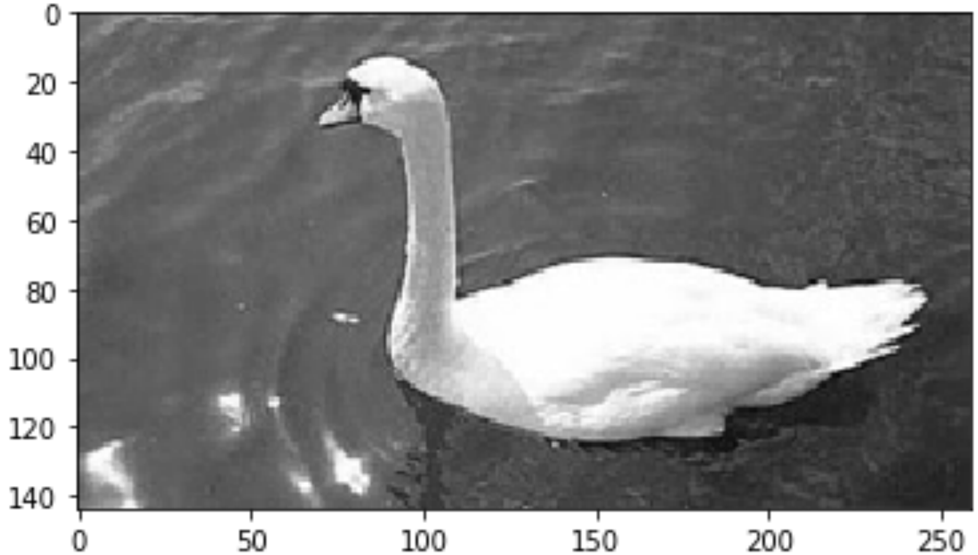
- More smoothing improves detection
- And hurts localization.

This is what you might guess from (detect change) + (remove noise)

Canny edge detector

1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient
4. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
5. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

The Canny edge detector



original image

Canny edge detector

1. Smoothing (noise reduction)

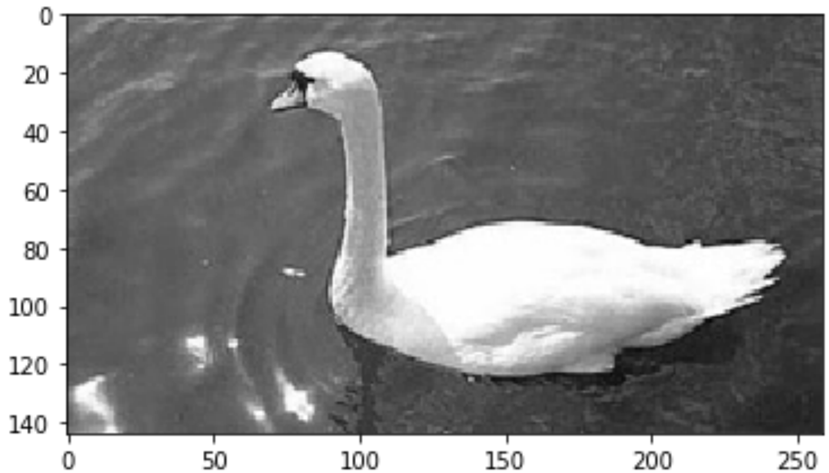
5 x 5 Gaussian kernel

$$\frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

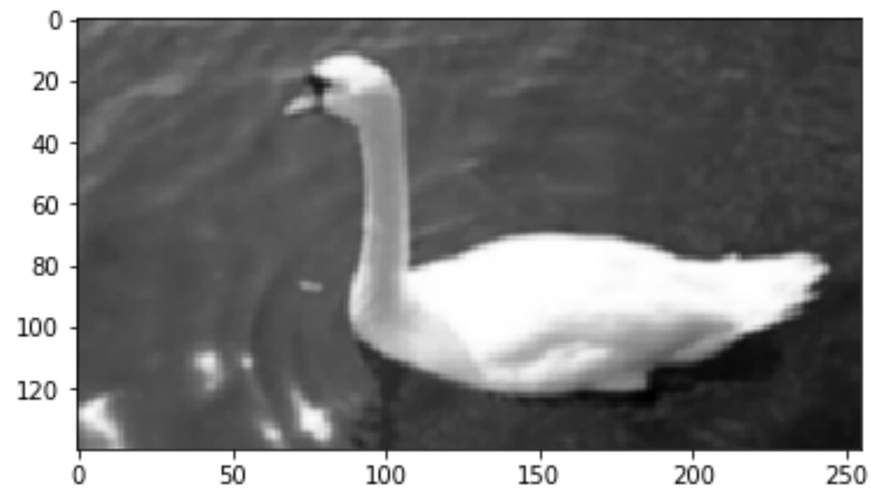
$$Filter : (2k+1) \times (2k+1) \qquad -2 \le k \le 2$$

$$x = i - (k+1); y = j - (k+1) \qquad 1 \le i, j \le 2k+1$$

The Canny edge detector



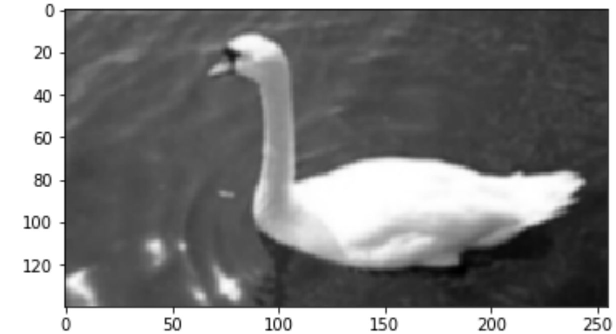
original image



smoothed image

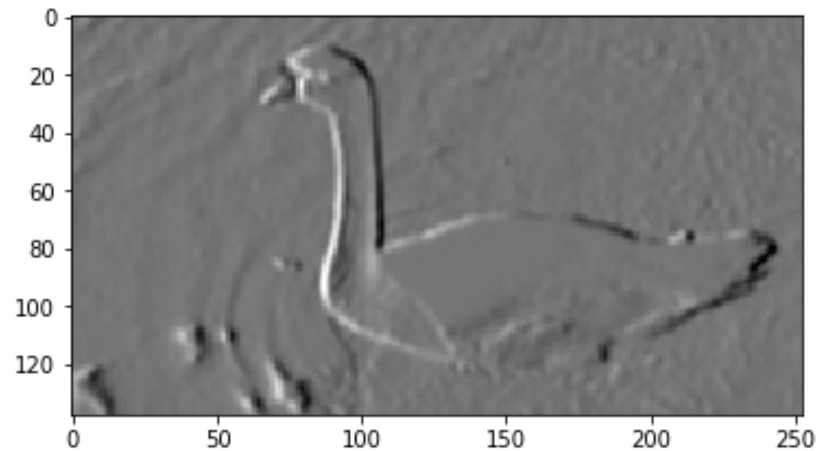
Canny edge detector

1. Smoothing (noise reduction)
2. Find derivatives (gradients)




```
[-1.,  0.,  1.]
[-2.,  0.,  2.]
[-1.,  0.,  1.]
```

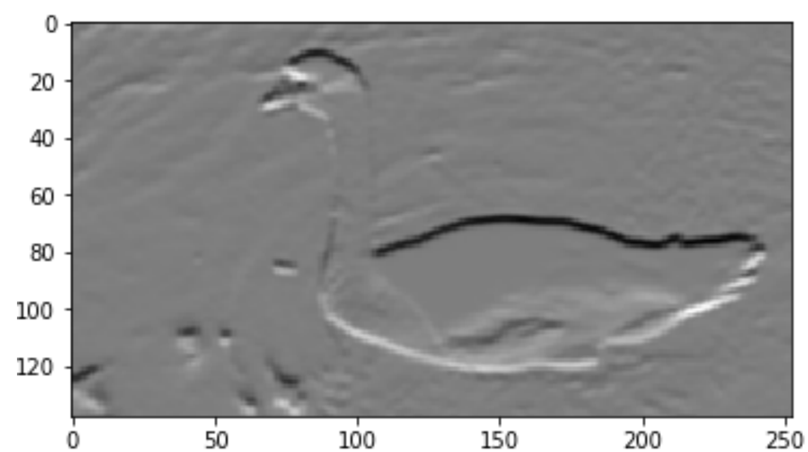
h_x



F_x

```
[ 1.,  2.,  1.]
[ 0.,  0.,  0.]
[-1., -2., -1.]
```

h_y

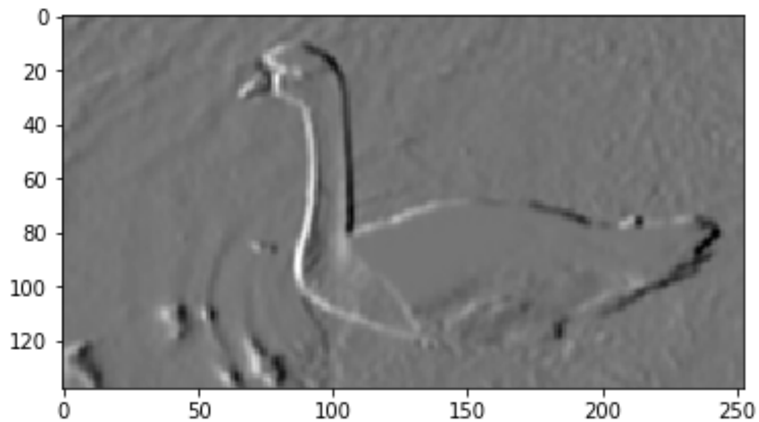


F_y

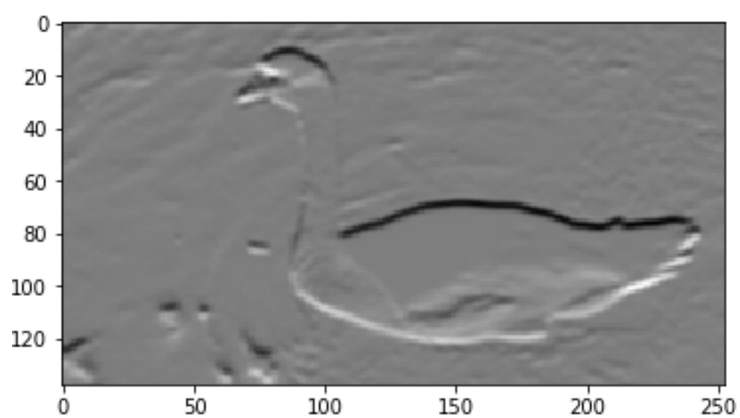
Canny edge detector

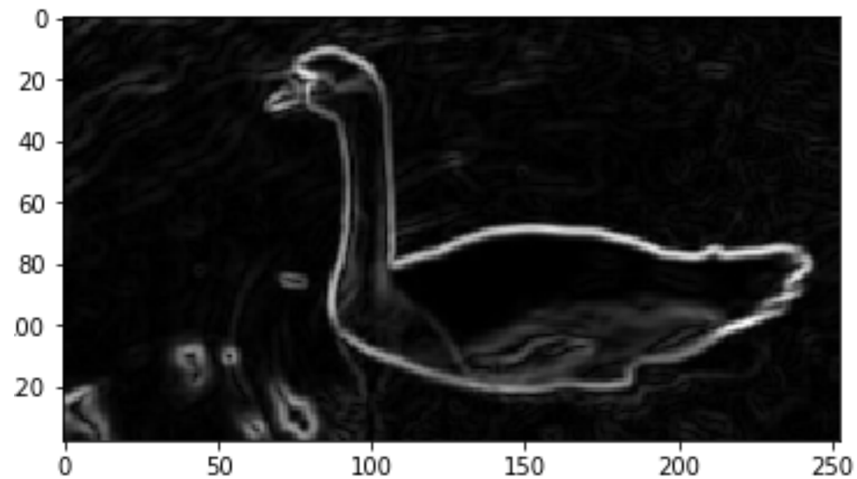
1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient

F_x



F_y



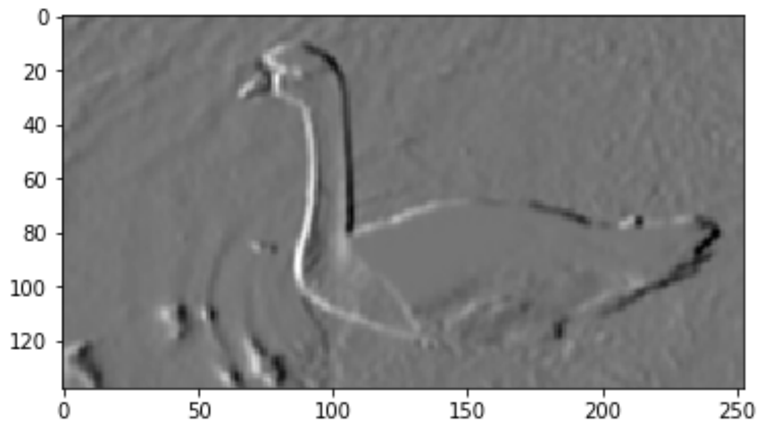


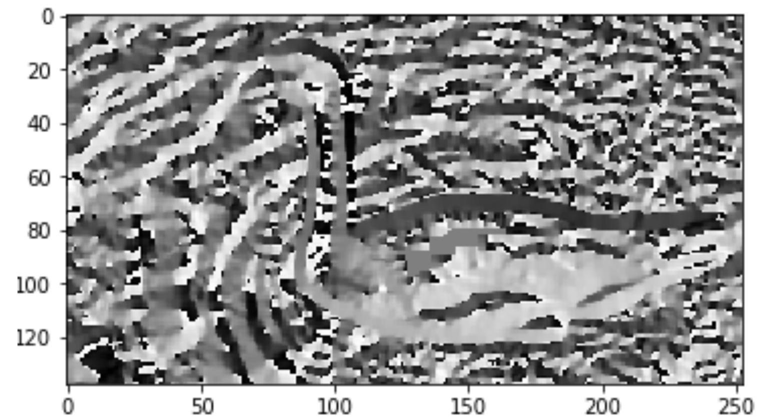
$$G = \sqrt{(F_x^2 + F_y^2)}$$

Canny edge detector

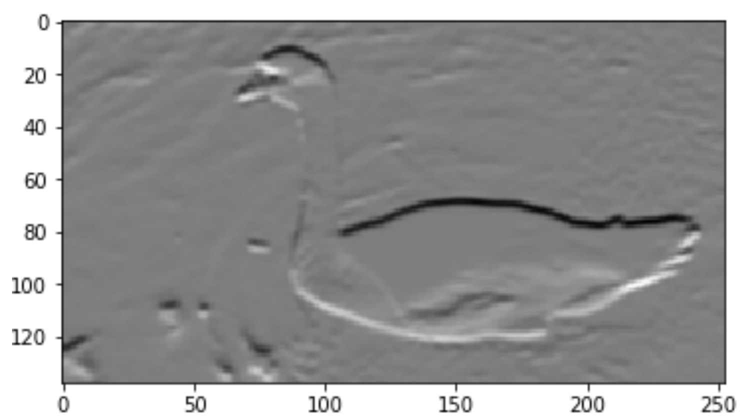
1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient

F_x



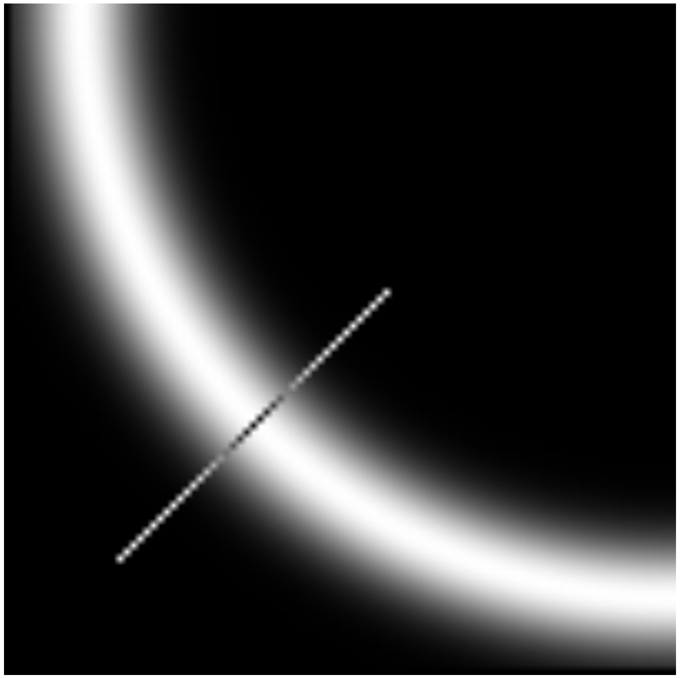


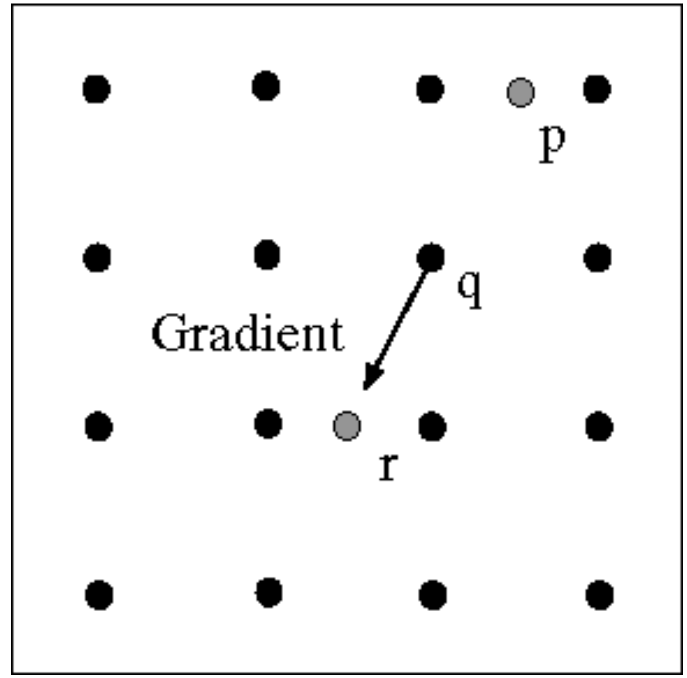
F_y



$$\theta = tan^{-1}\left(\frac{F_y}{F_x}\right)$$

Non-maximum suppression

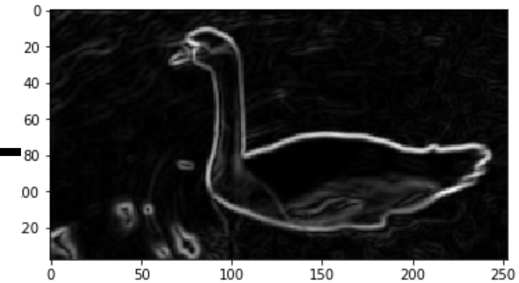


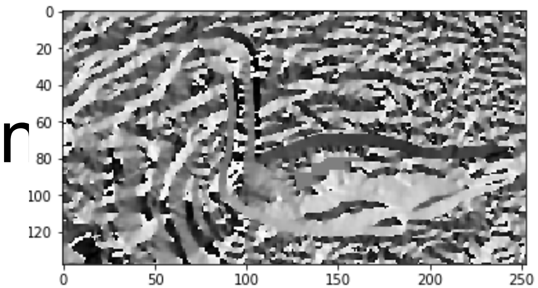


Check if pixel is local maximum along gradient direction

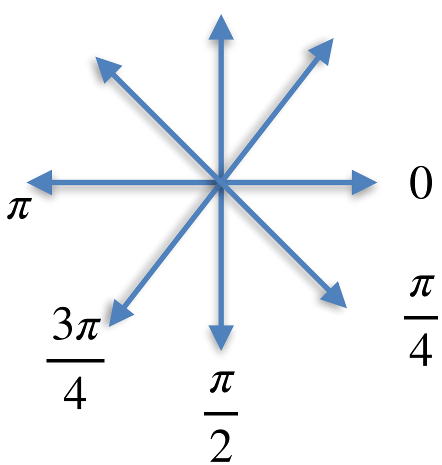
- requires checking interpolated pixels p and r

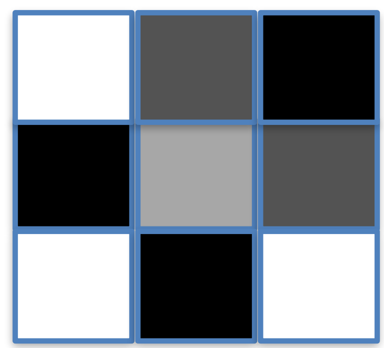
Canny edge detector



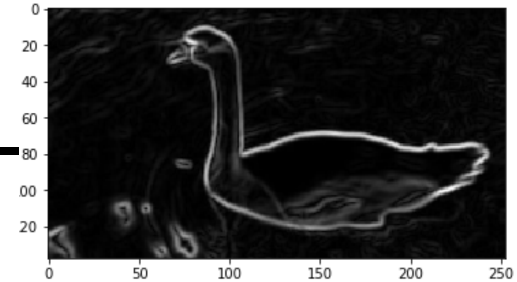


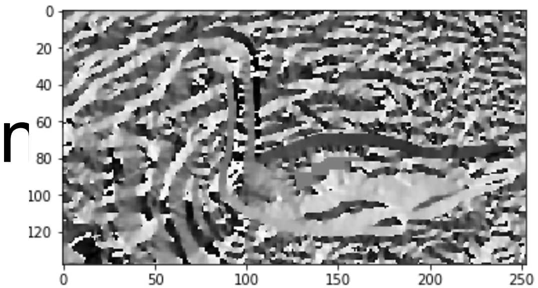
1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation
4. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width



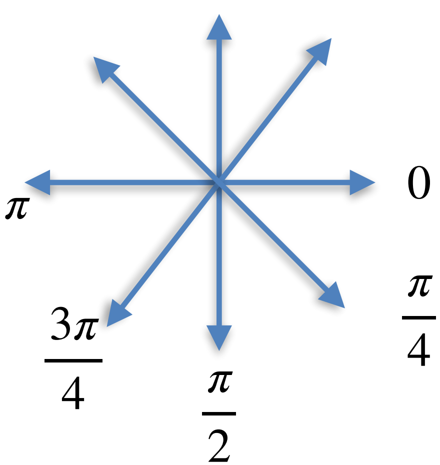


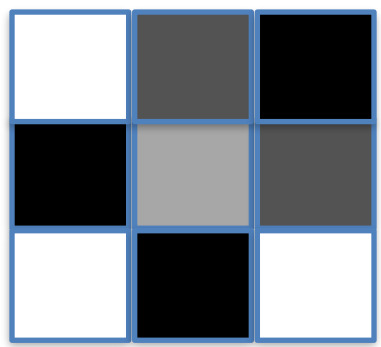
Canny edge detector

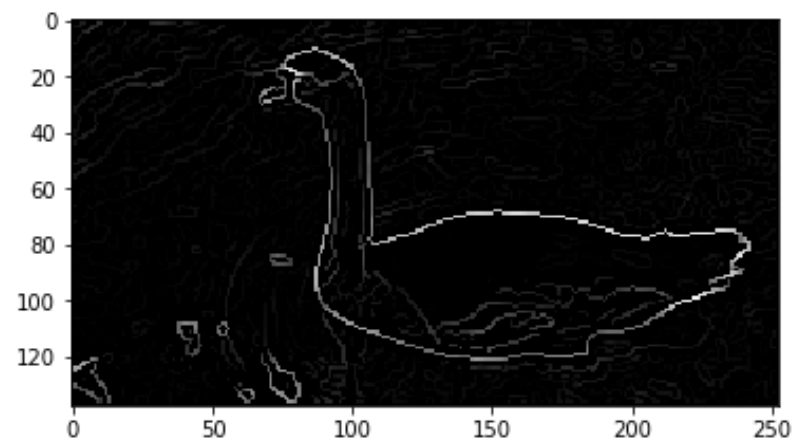




1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation
4. Non-maximum suppression:
   - Thin multi-pixel wide "ridges" down to single pixel width







Canny edge detector

1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient
4. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
5. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high

Upper threshold based on the max intensity

lower threshold based on some percentage of the upper threshold

Canny edge detector - double threshold

1. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high

Upper threshold based on the max intensity

lower threshold based on some percentage of the upper threshold

Example:

Upper threshold - 90% of max
lower threshold - 35%

<= lower threshold	lower threshold < intensity < upper threshold	>= upper threshold
irrelevant	weak	strong

Canny edge detector - double threshold

1. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high

Upper threshold based on the max intensity

lower threshold based on some percentage of the upper threshold

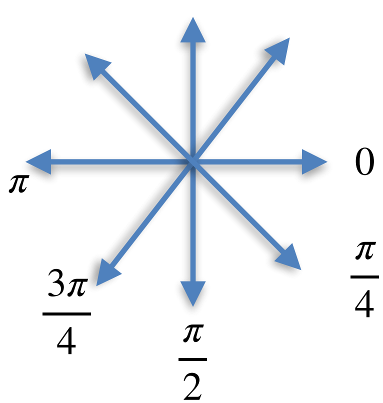
Example:

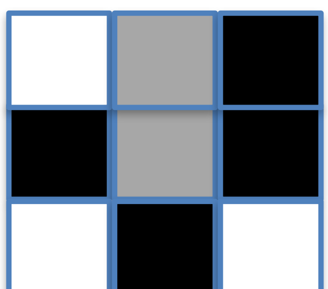
Upper threshold - 90% of max
lower threshold - 35%

<= lower threshold	lower threshold < intensity < upper threshold	>= upper threshold
irrelevant = 0	weak = low threshold	strong= 255

Canny edge detector - Hysteresis

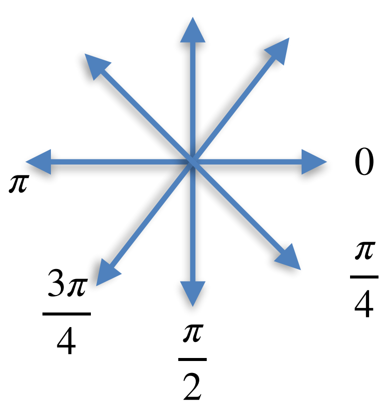
1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient
4. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
5. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - replace with the strong edge if any of the neighboring pixels is strong, else make it irrelevant.

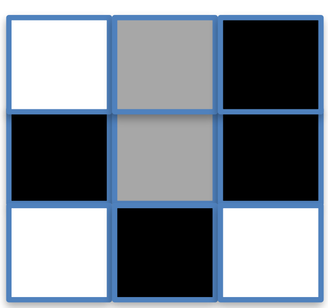


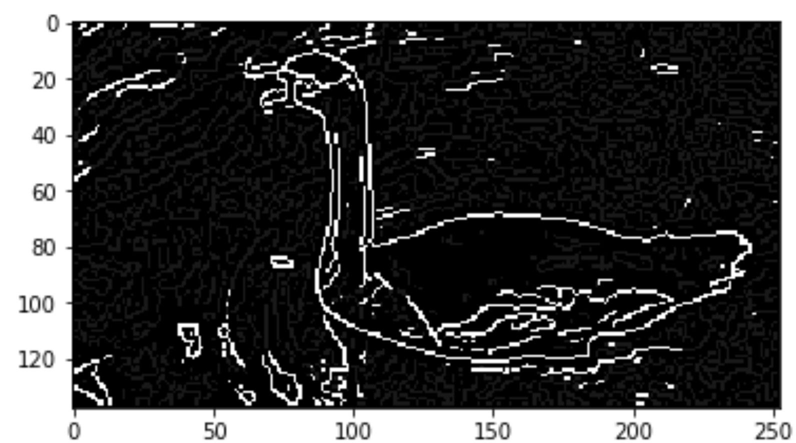


Canny edge detector - Hysteresis

1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient
4. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
5. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - replace with the strong edge if any of the neighboring pixels is strong, else make it irrelevant.







Canny Edge Detection (Example)

Original image

gap is gone

Strong + connected weak edges

Strong edges only

Weak edges

courtesy of G. Loy

Effect of σ (Gaussian kernel size)

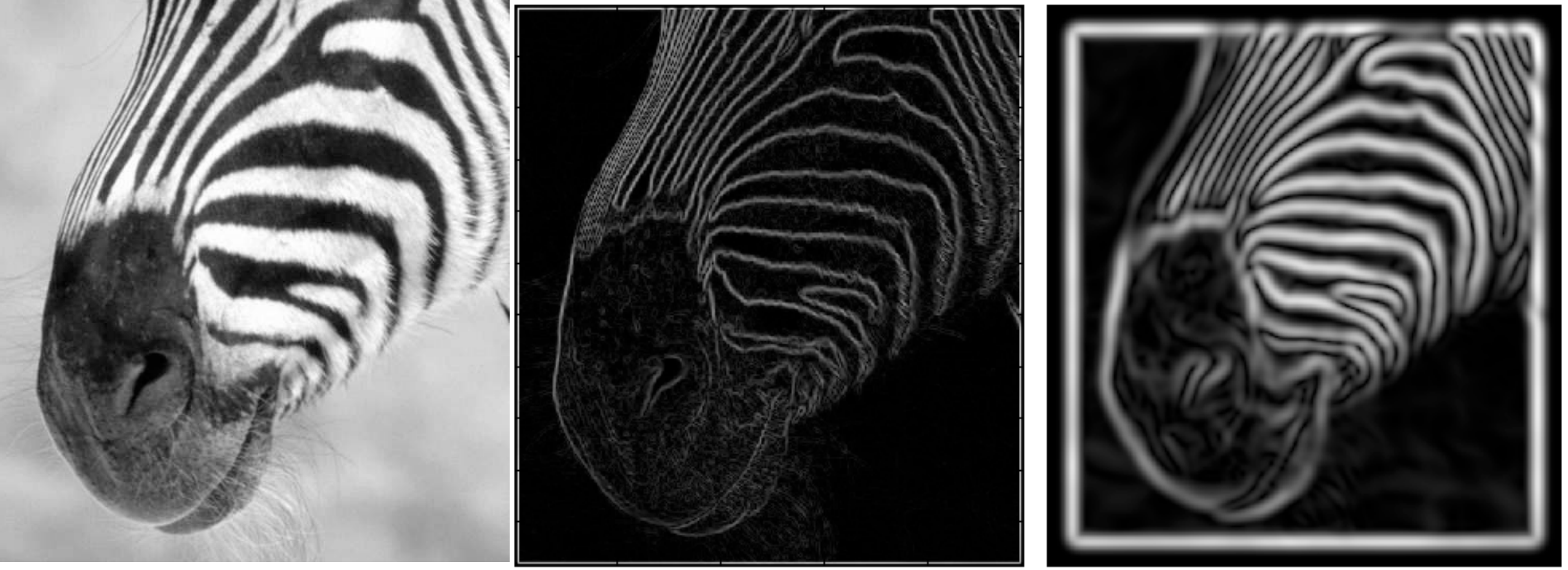
original

Canny with $\sigma = 1$

Canny with $\sigma = 2$

The choice of σ depends on desired behavior

- large σ detects large scale edges
- small σ detects fine features



Scale

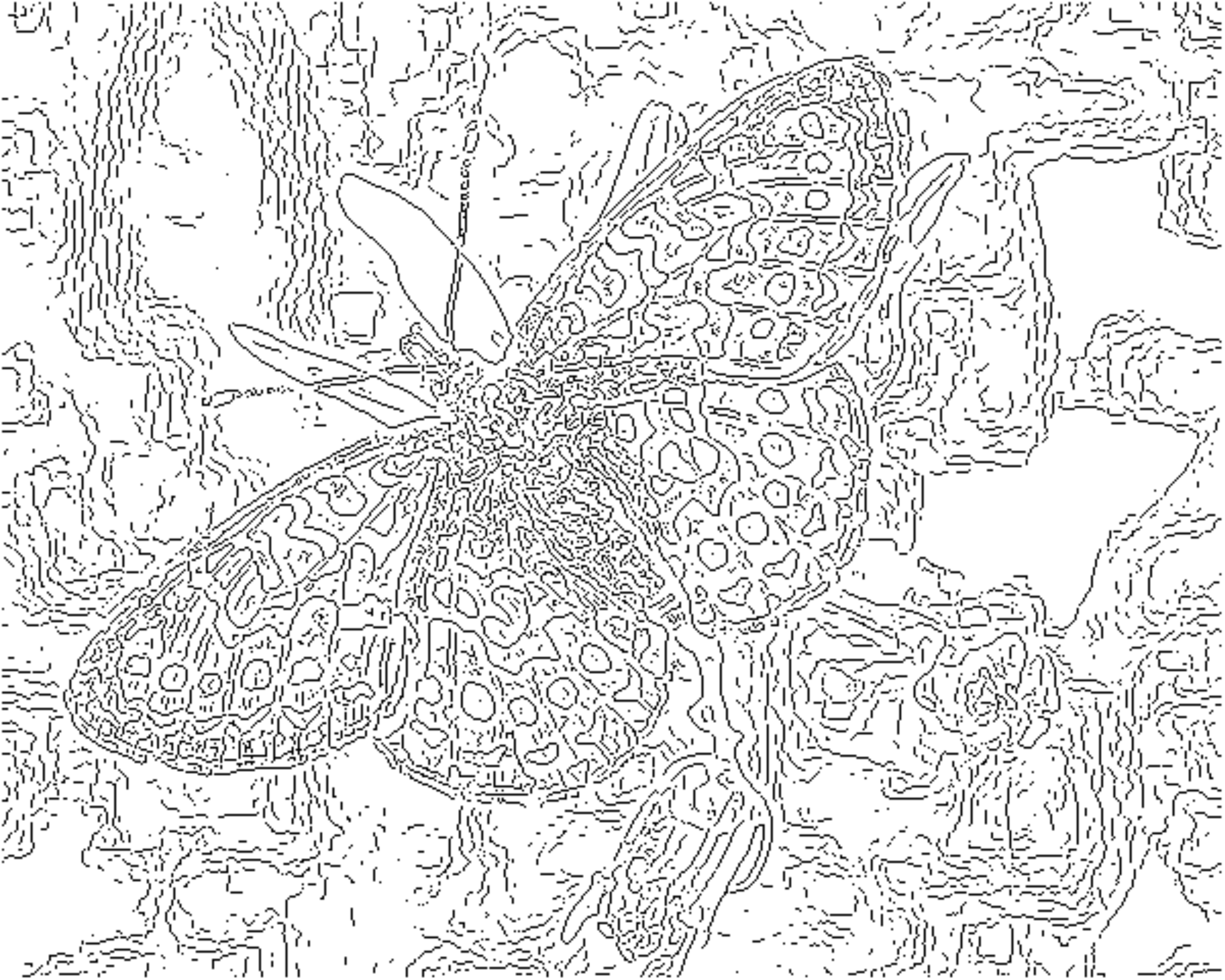
Smoothing

Eliminates noise edges.

Makes edges smoother.

Removes fine detail.

(Forsyth & Ponce)



fine scale
high
threshold

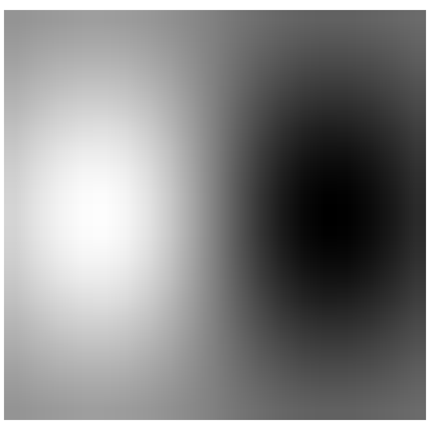
coarse scale, high threshold

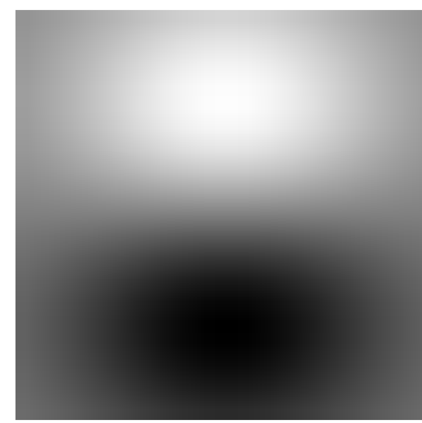
coarse
scale
low
threshold

Filters are templates

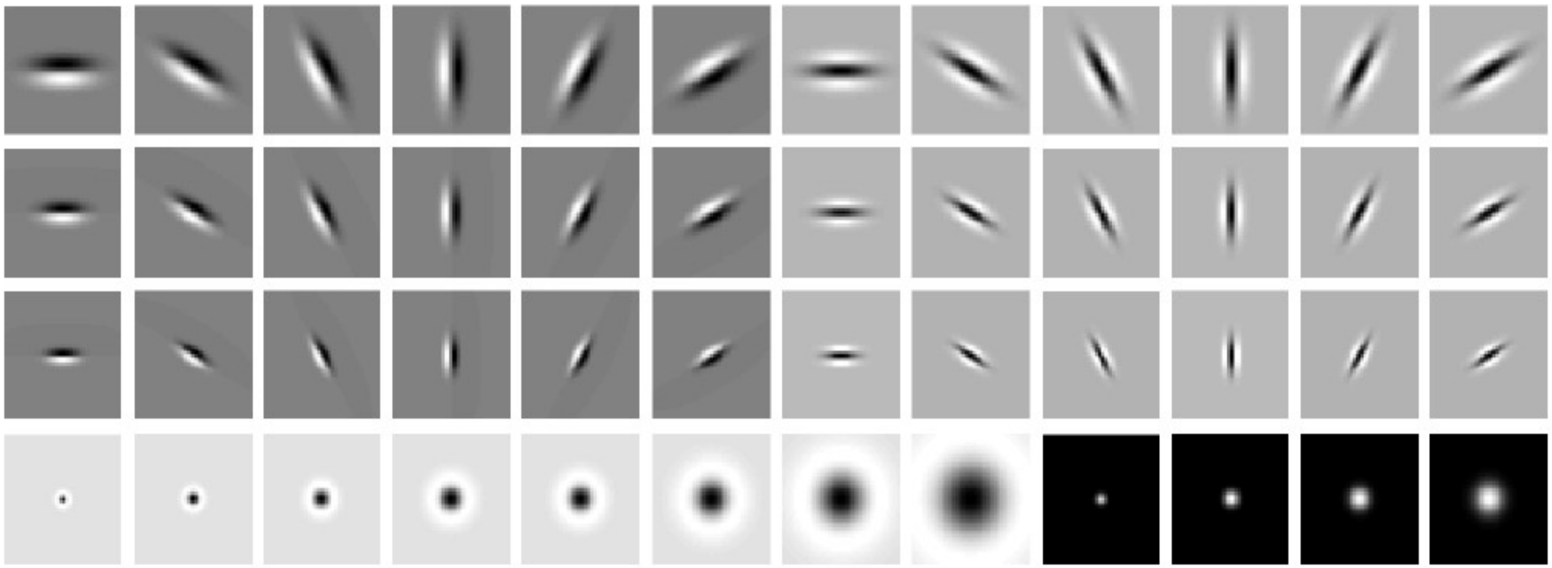
- Applying a filter at some point can be seen as taking a dot-product between the image and some vector
- Filtering the image is a set of dot products

- Insight
  - filters look like the effects they are intended to find
  - filters find effects they look like





Filter Bank



Leung & Malik, Representing and Recognizing the Visual Apperance using 3D Textons, IJCV 2001

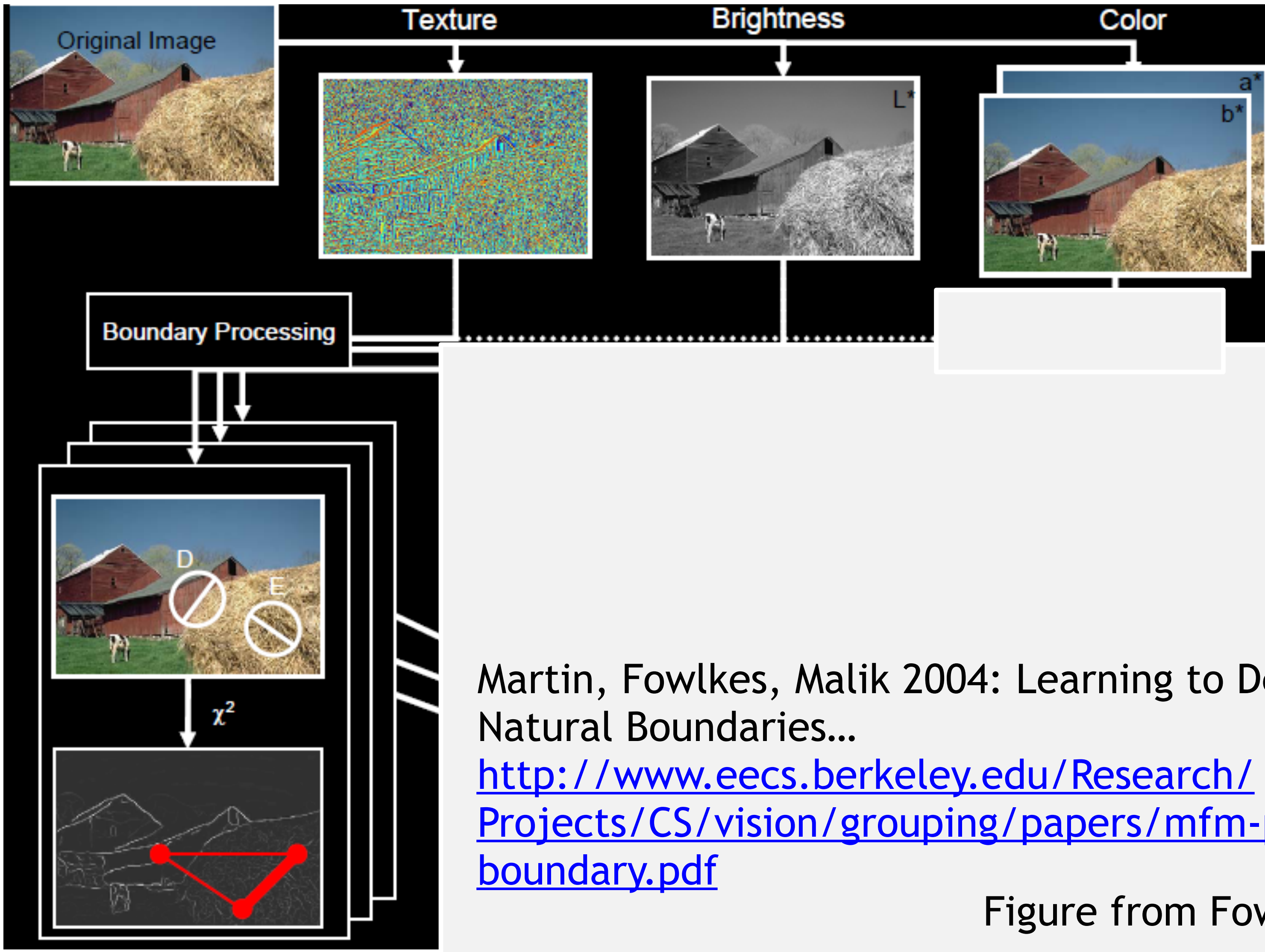
Learning to detect boundaries

image | human segmentation | gradient magnitude

Berkeley segmentation database:

http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

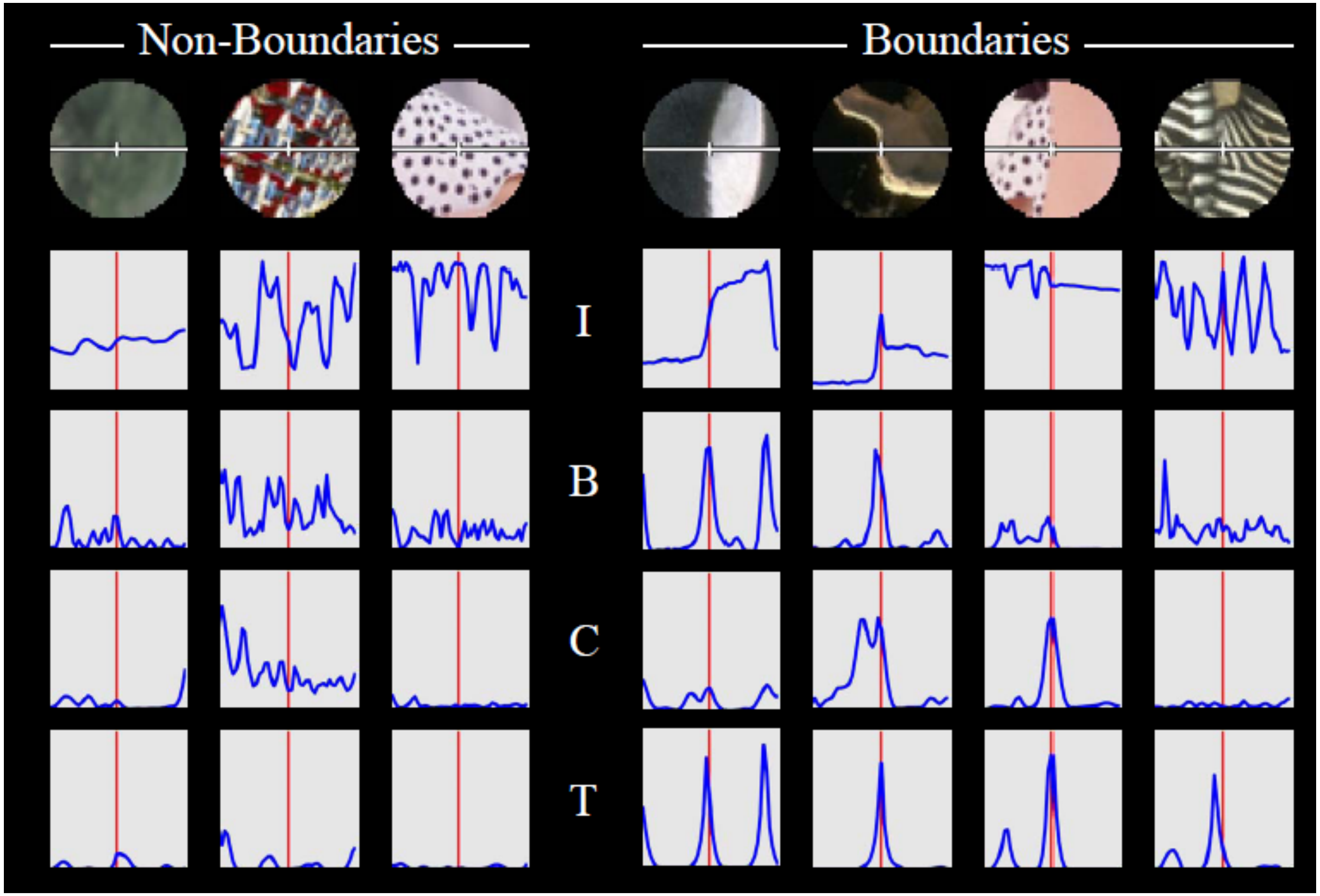
pB boundary detector



Martin, Fowlkes, Malik 2004: Learning to Detection Natural Boundaries…
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/papers/mfm-pami-boundary.pdf

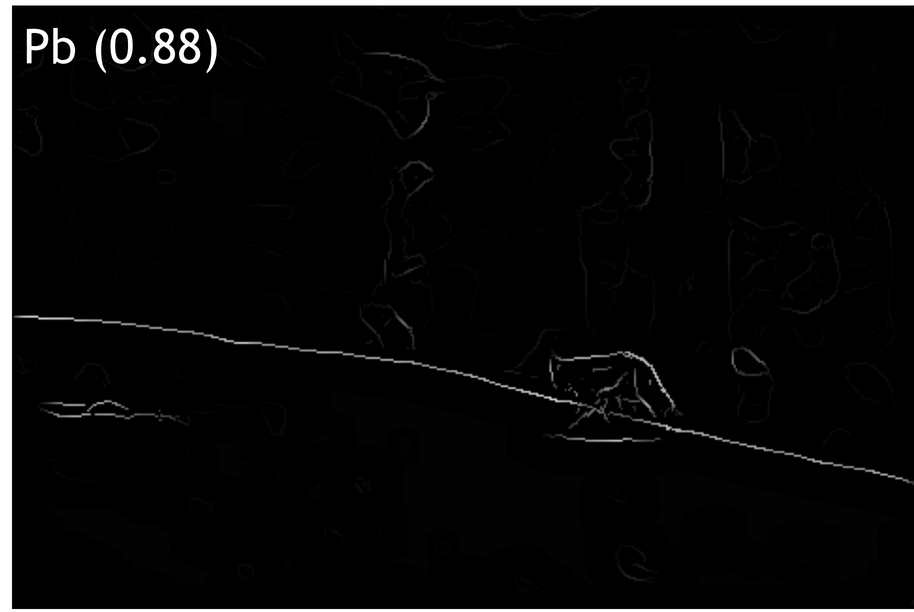
Figure from Fowlkes

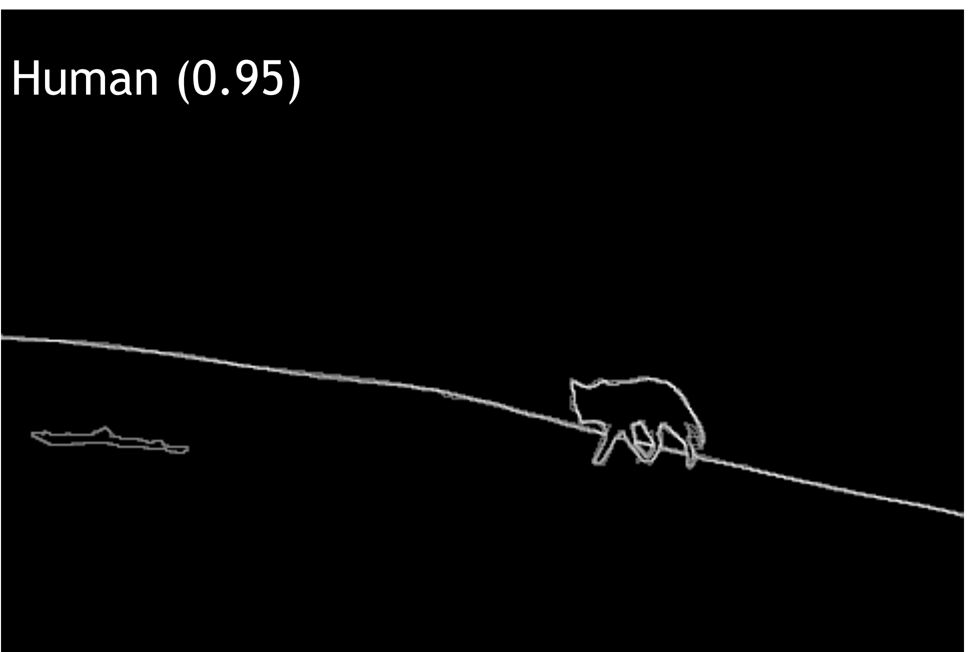
pB Boundary Detector



- Estimate Posterior probability of boundary passing through centre point based on local patch based features
- Using a Supervised Learning based framework

Results





Results