CS-565 Computer Vision

Nazar Khan

Department of Computer Science
University of the Punjab

5. Derivative Filtering & Edge Detection

Smoothing before Derivative Filtering

- Noise affects *every* signal in the real world.
- If there is noise in the signal, its effect on the derivative will be amplified. For example, derivative computed from two noisy pixels can have twice the noise.
- Always smooth a signal before computing derivatives.
- This requires two convolutions – one for smoothing and one for derivatives.
- We can exploit associativity of convolution

$$(I * S) * D = I * (S * D)$$

  to obtain a cheaper solution.
  - Apply the small derivative kernel on the small smoothing kernel. Then apply resulting kernel on the large image.

Derivative of Gaussian Filter

- For the case of Gaussian smoothing followed by derivative filtering, we can instead differentiate the smoothing kernel and then convolve with the result.
- This yields the *Derivative of Gaussian* filter.

$$1D: \quad \frac{\partial g_\sigma(x)}{\partial x} = \frac{-x}{\sqrt{2\pi}\sigma^3} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

$$2D: \quad \frac{\partial G_\sigma(x,y)}{\partial x} = \frac{-x}{2\pi\sigma^4} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

  where all means have been set to 0.
- This filter computes gradients at smoothing scale σ.
- To make a 5×1 mask, evaluate the $1D$ Derivative of Gaussian at $x = -2, -1, 0, 1, 2$.

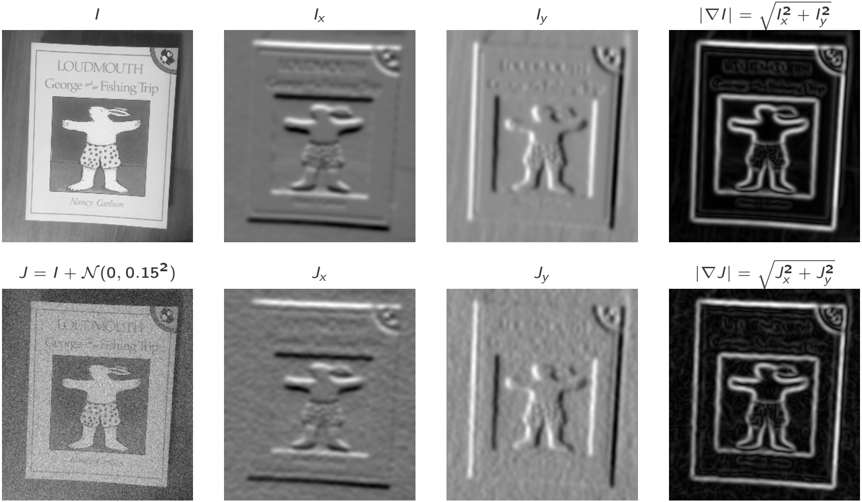


Figure: Convolving with derivative of Gaussian smooths out noise and computes derivatives. Results without noise are similar to results with noise. This demonstrates robustness to noise. Smoothing scale for all results was $\sigma = 3$. Author: N. Khan (2018)

Sobel Filter

- *Sobel operator* is a 2D filter that applies an integer valued smoothing filter before applying a derivative filter.
- Once again, the filters can be convolved with each other first to obtain a single derivative filter that is robust to noise.

x-direction

$$\frac{1}{2}\begin{array}{|c|}\hline 1 \\ \hline \mathbf{0} \\ \hline -1 \\ \hline \end{array} * \frac{1}{4}\begin{array}{|c|c|c|}\hline 1 & \mathbf{2} & 1 \\ \hline \end{array} = \frac{1}{8}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & \mathbf{0} & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

y-direction

$$\frac{1}{2}\begin{array}{|c|c|c|}\hline 1 & \mathbf{0} & -1 \\ \hline \end{array} * \frac{1}{4}\begin{array}{|c|}\hline 1 \\ \hline \mathbf{2} \\ \hline 1 \\ \hline \end{array} = \frac{1}{8}\begin{array}{|c|c|c|}\hline 1 & 0 & -1 \\ \hline 2 & \mathbf{0} & -2 \\ \hline 1 & 0 & -1 \\ \hline \end{array}$$

- This is an approximation of the Derivative of Gaussian filter but with coarser control over the smoothing scale σ.

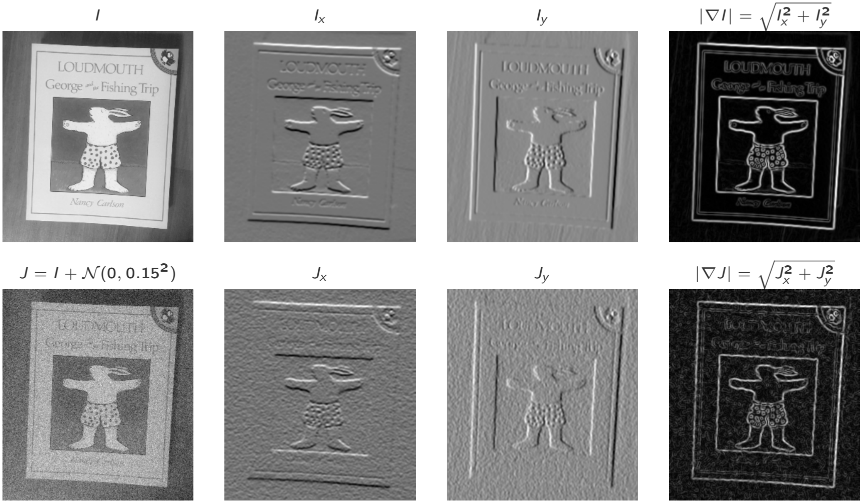


Figure: Convolving with Sobel kernels smooths out noise and computes derivatives. Results without noise are similar to results with noise. This demonstrates robustness to noise. Sobel kernel size for all results was 7×7. Author: N. Khan (2018)

A Simple Edge Detector

- Threshold gradient magnitude to obtain edge pixels.
- Percentiles can be used instead of fixed threshold.
  - The k-th percentile of a list of numbers is the value greater than k% of the numbers.
  - For example, if 70% values in a list of numbers are less than 48, then 48 is the 70-th percentile.
- For a threshold of, say, the 80-th percentile, 20% of the image pixels will be guaranteed to be marked as edge pixels.

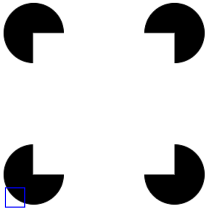
Canny Edge Detector

- Most popular edge detector.
- Three step procedure
  1. Compute derivatives of smoothed image
  2. Non-maxima suppression
  3. Hysteresis thresholding

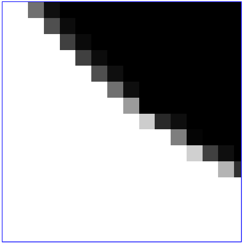
Non-maxima Suppression

- Derivative magnitudes start increasing close to an edge and decrease gradually after crossing it.
- So pixels close to an edge pixel can also exceed the detection threshold and be detected as edges.
- Solution: The actual edge location should have the highest gradient magnitude. So suppress the neighbours *across the edge*.

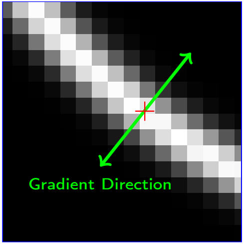
Original



Zoomed



Gradient Magnitude



Non-maxima Suppression
Quantization of Gradient Direction

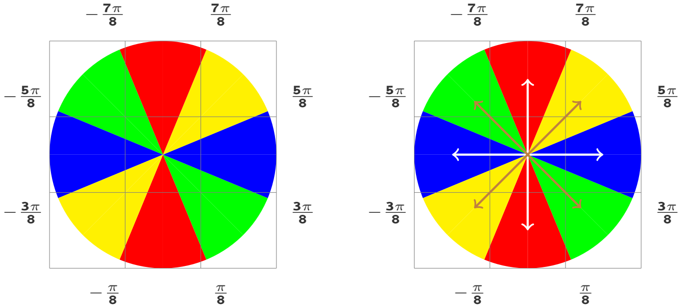


Figure: Gradient direction θ obtained via atan2 quantized into 4 orientations. Author: N. Khan (2021)

Non-maxima Suppression
Quantization of Gradient Direction

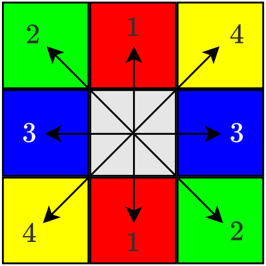


Figure: Neighboring pixels corresponding to the 4 quantized orientations. Author: N. Khan (2021)

Non-maxima Suppression

For each pixel with non-zero gradient magnitude

1. Find the quantized gradient direction.
2. If gradient magnitude of current pixel is less than either of the two neighbors in the quantized direction, set gradient magnitude of *current pixel* to 0.

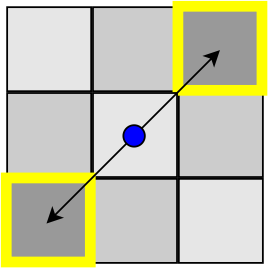


Figure: Neighbors selected for non-maxima suppression across an edge. The graylevels indicate gradient magnitude. Author: N. Khan (2021)

Non-maxima Suppression

- Non-maxima suppression ensures single-pixel thick edges.

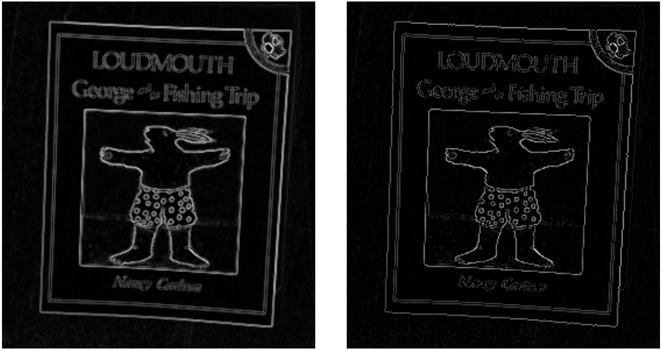


Figure: Left: Gradient magnitudes. Convolution causes magnitudes around the real edge location to be high as well. **Right**: After non-maxima suppression such fake magnitudes are suppressed. Author: N. Khan (2021)

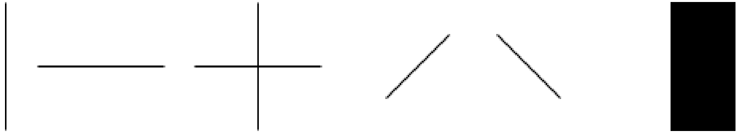
Hysteresis Thresholding

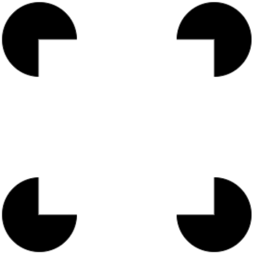
- Basic Idea: If a weak edge lies adjacent to a strong edge, include the weak edge too.
- Use pixels above upper threshold T_{high} as seed points for relevant edges.
- Recursively add all neighbours of seed points that are above the lower threshold T_{low}.

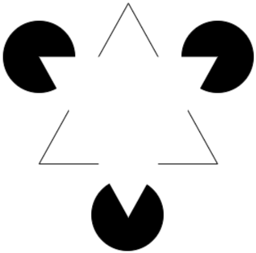
Hysteresis Thresholding

1. Scan image from left to right, top to bottom.
2. If $|\nabla I|(x, y)$ is above T_{high}
   2.1 Mark pixel (x, y) as edge.
   2.2 Look at each unvisited neighbour of pixel (x, y). If gradient magnitude of neighbour is above T_{low}, mark it as edge and *recursively* look at unvisited neighbours of (x, y).

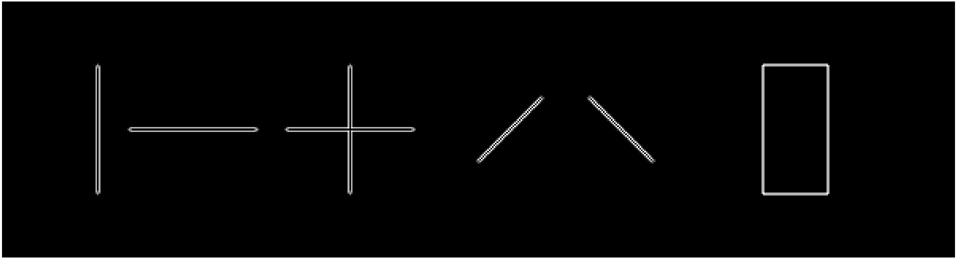
Canny Edge Detection Pipeline

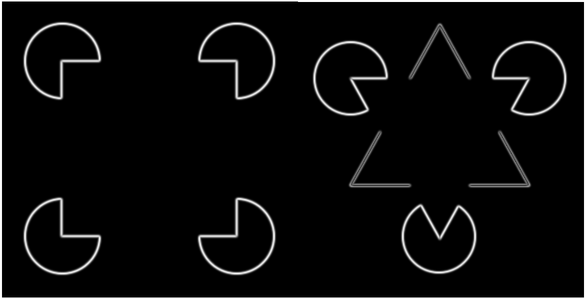




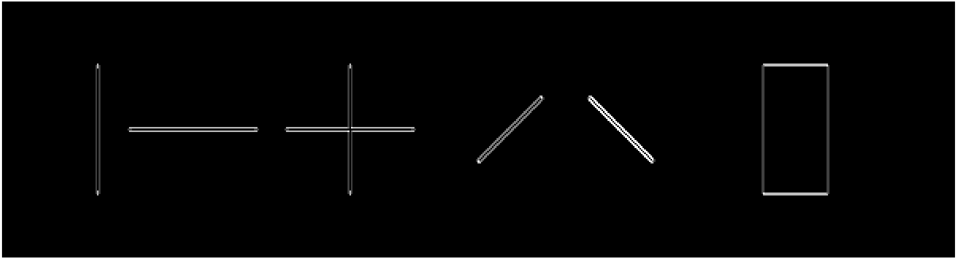


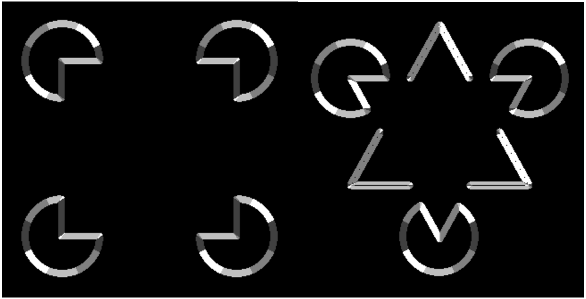
Gradient Magnitudes



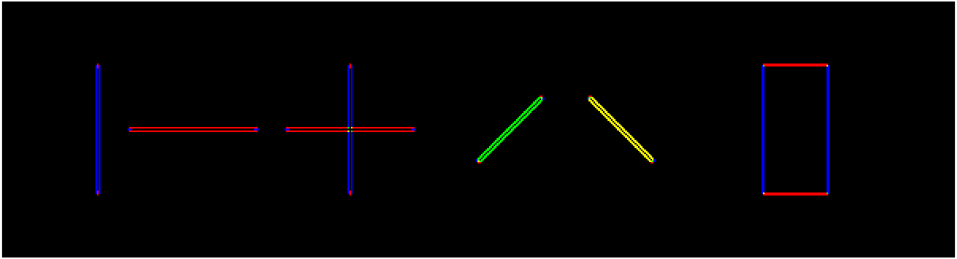


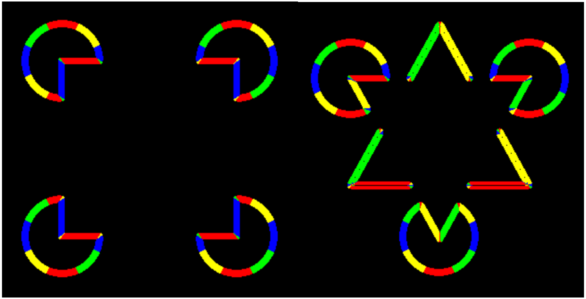
Quantized Gradient Angles



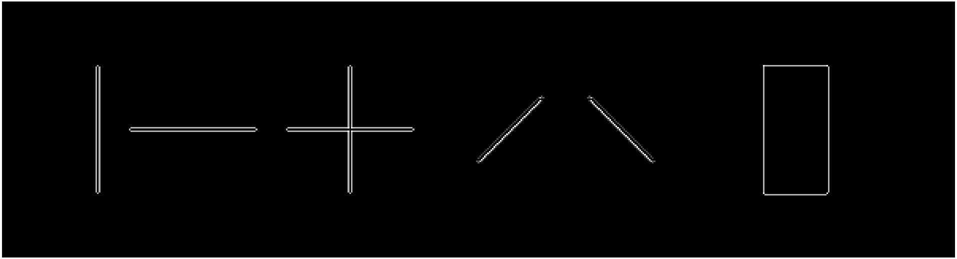


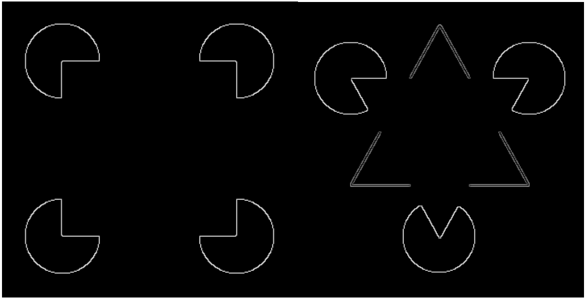
Color-coded Quantized Gradient Angles



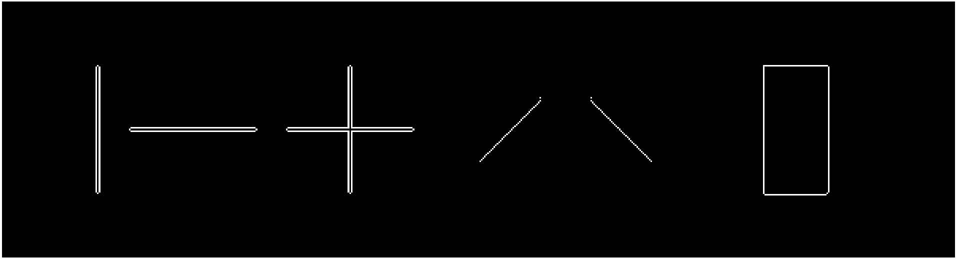


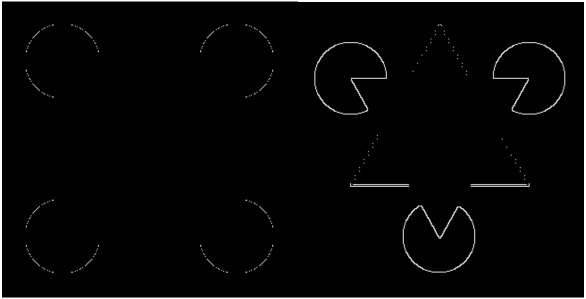
Non-maxima Suppression



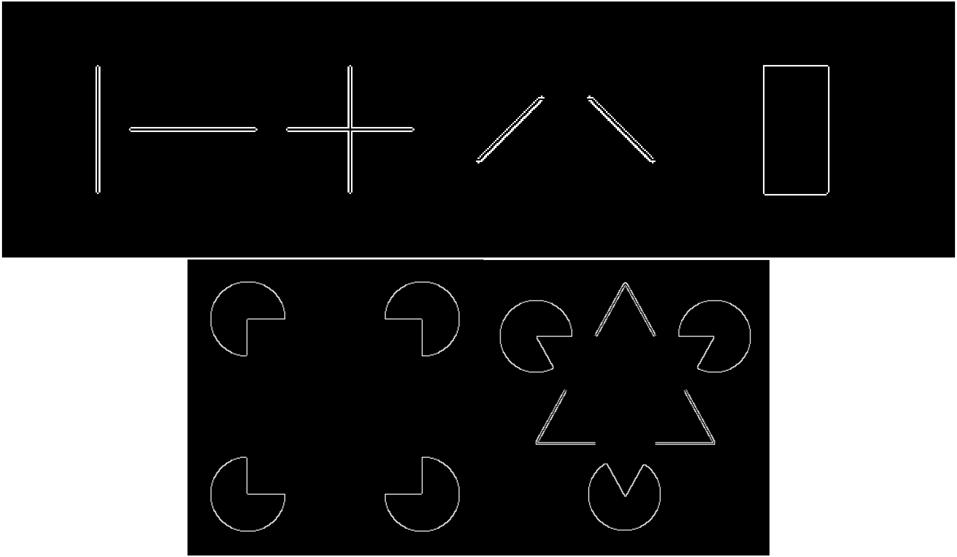


Higher Threshold

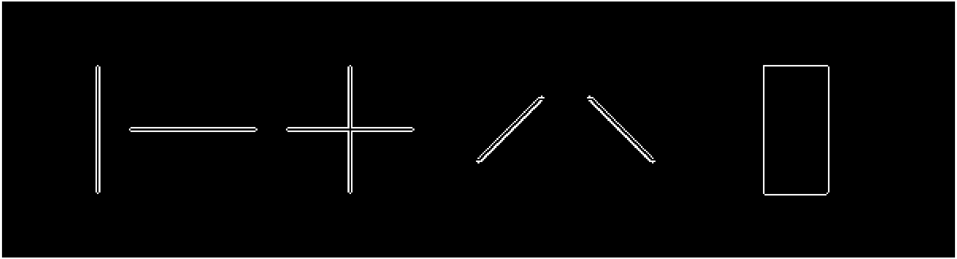


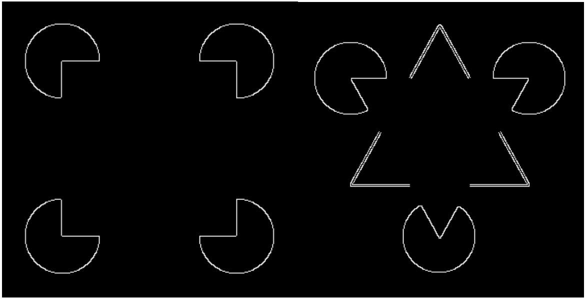


Lower Threshold



Edges





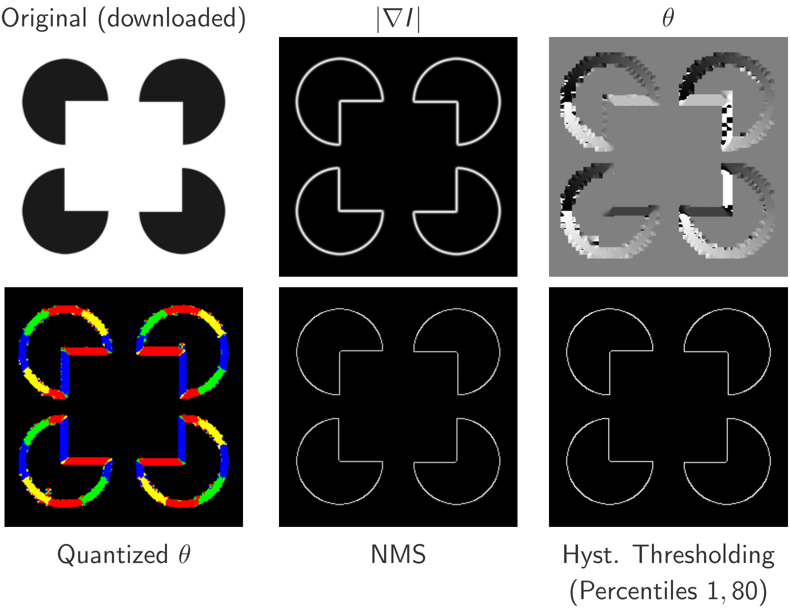


Figure: Canny edge detection pipeline. Gaussian derivatives were computed at smoothing scale $\sigma = 1$. Author: N. Khan (2021)

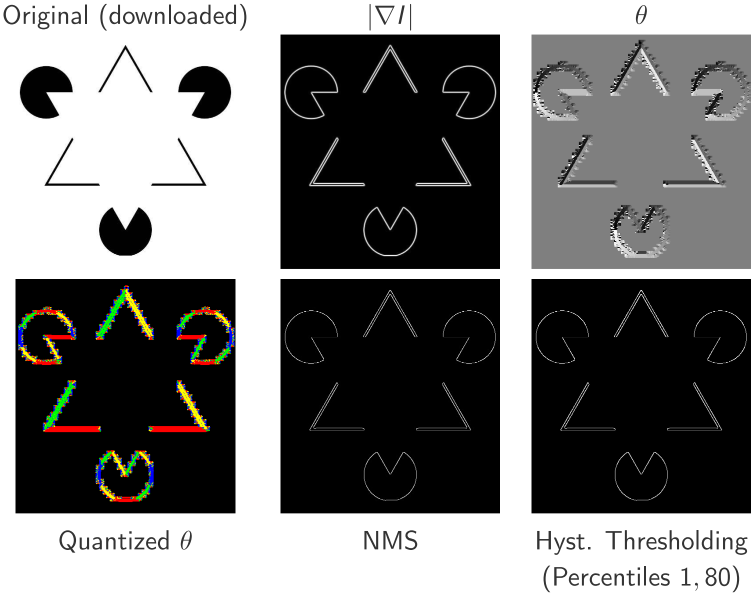


Figure: Canny edge detection pipeline. Gaussian derivatives were computed at smoothing scale $\sigma = 1$. Author: N. Khan (2021)

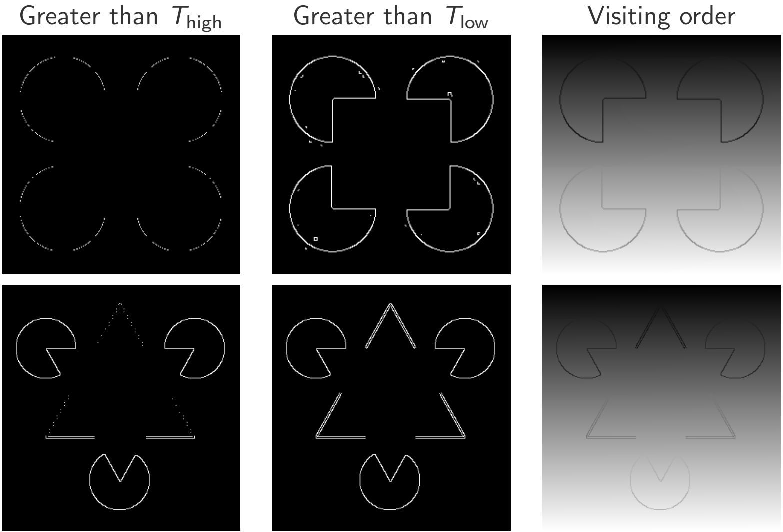


Figure: Neither the high threshold alone (**left**) nor the low threshold alone (**middle**) is sufficient for good edge detection. **Right**: Order of visiting pixels during hysteresis thresholding. Darker corresponds to pixels visited earlier. Author: N. Khan (2021)

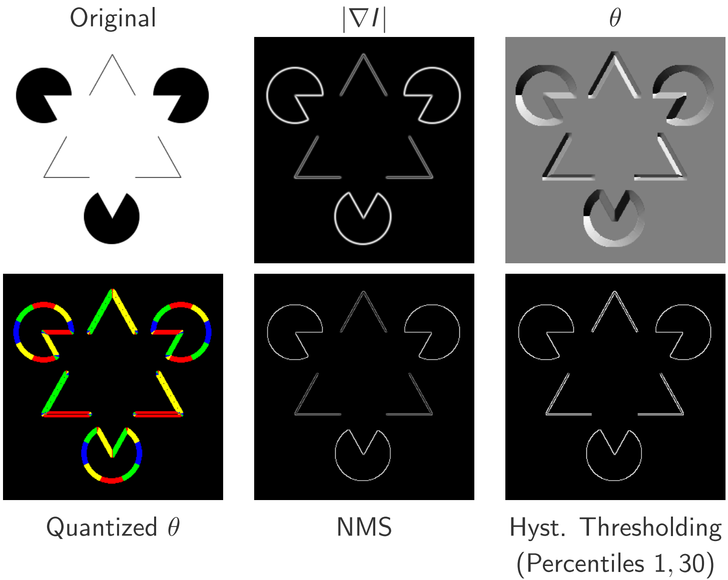


Figure: Canny edge detection pipeline. Gaussian derivatives were computed at smoothing scale $\sigma = 1$. Author: N. Khan (2021)

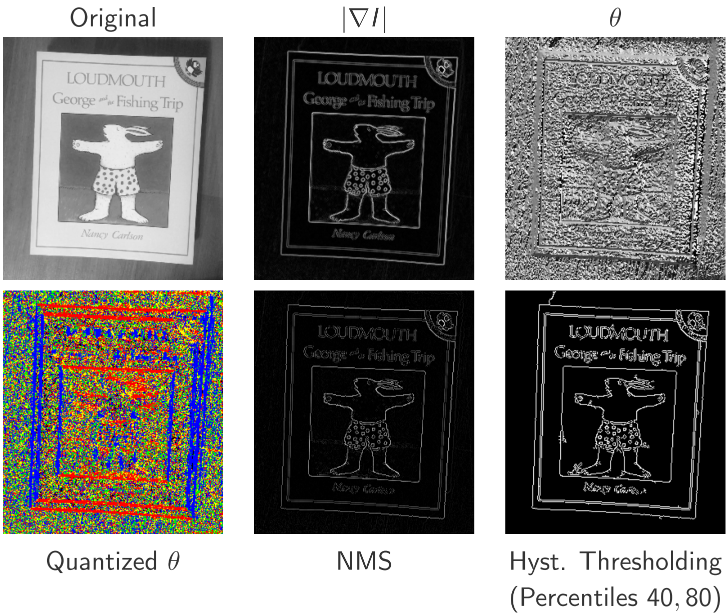
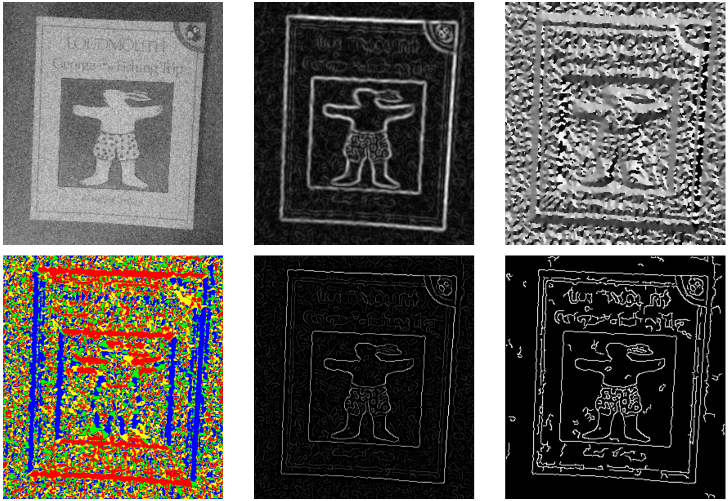


Figure: Canny edge detection pipeline. Gaussian derivatives were computed at smoothing scale $\sigma = 0.2$. Author: N. Khan (2021)

Original | $|\nabla I|$ | θ



Quantized θ | NMS | Hyst. Thresholding (Percentiles $40, 80$)

Figure: Canny edge detection on a noisy image. Gaussian derivatives were computed at smoothing scale $\sigma = 1.8$. Author: N. Khan (2021)

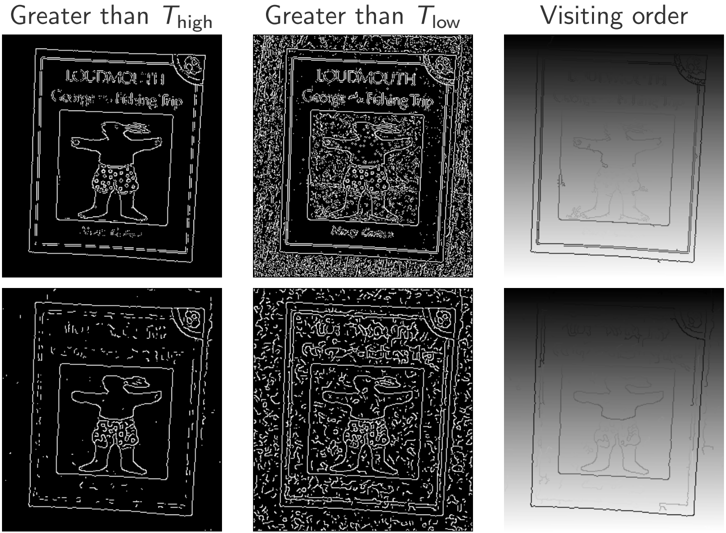


Figure: Neither the high threshold alone (**left**) nor the low threshold alone (**middle**) is sufficient for good edge detection. **Right**: Order of visiting pixels during hysteresis thresholding. Darker corresponds to pixels visited earlier. **Bottom** row is for the noisy image. Author: N. Khan (2021)