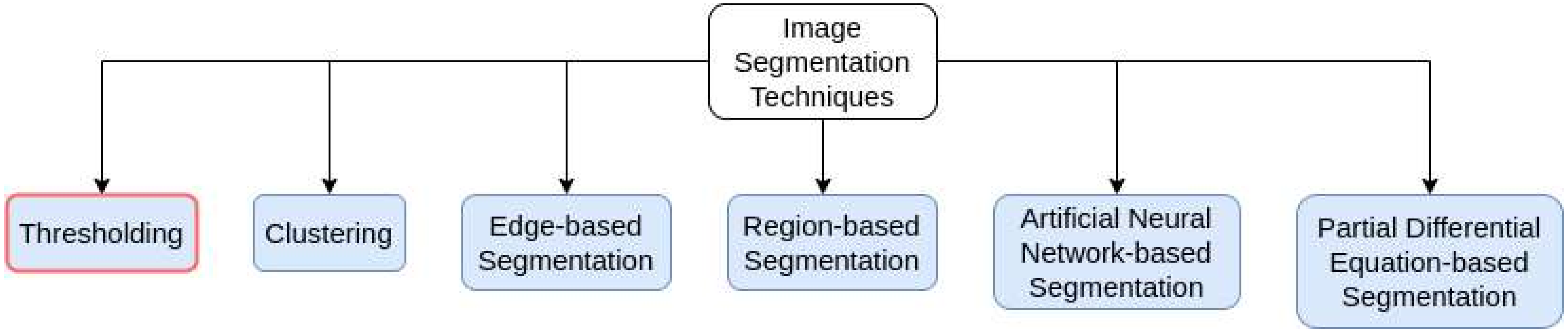
Image Segmentation

Yih-Lon Lin (林義隆)
Associate Professor,
Department of Computer Science and Information Engineering,
National Yunlin University of Science and Technology

Image Segmentation



https://learnopencv.com/otsu-thresholding-with-opencv/

Image Segmentation

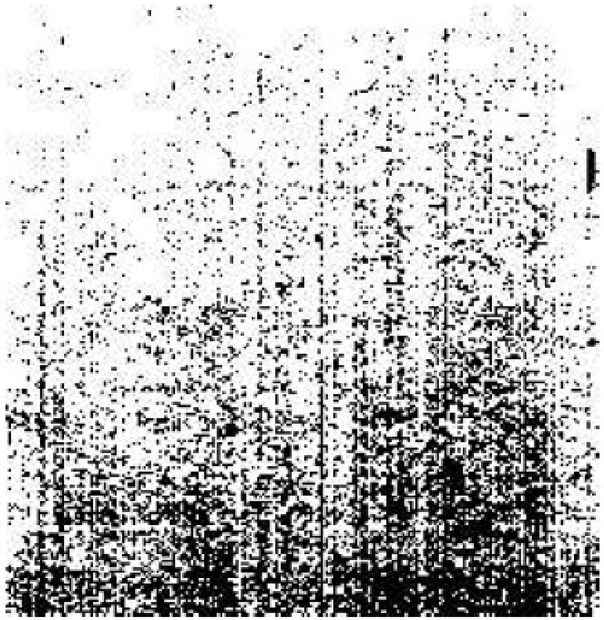
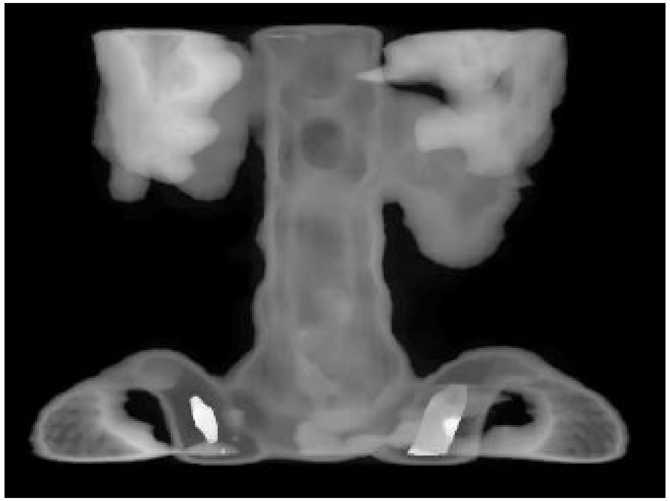


Image Segmentation



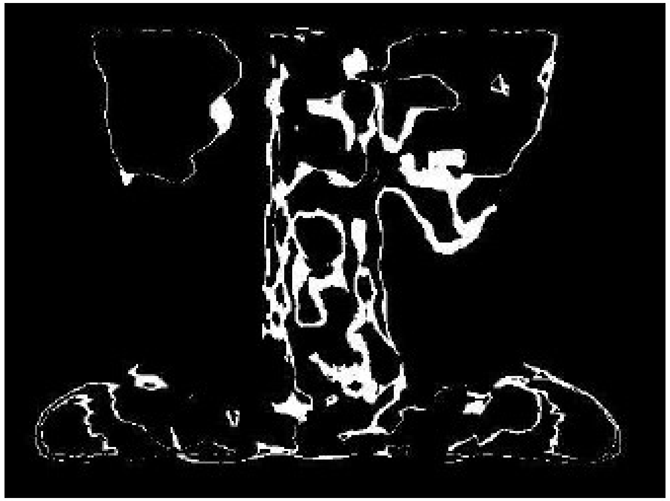
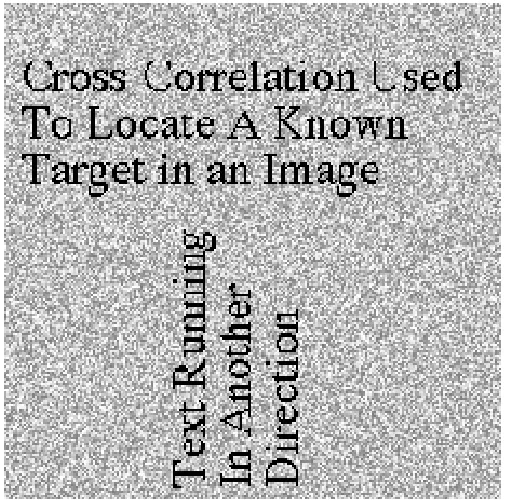


Image Segmentation



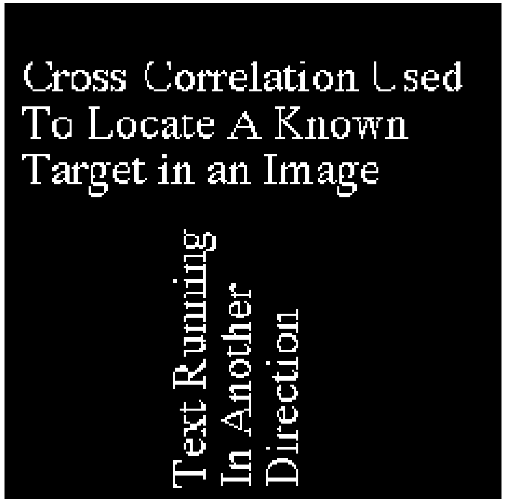
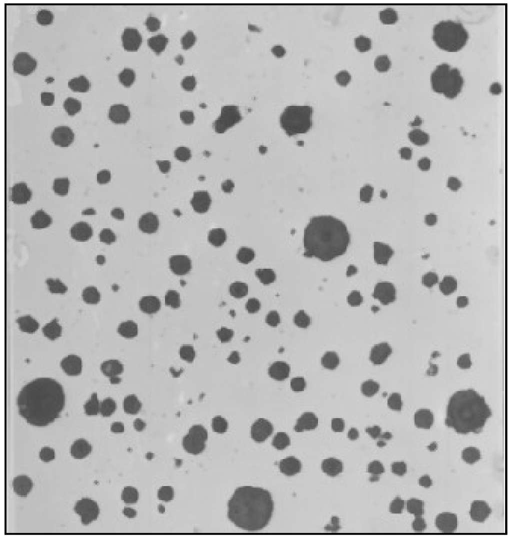
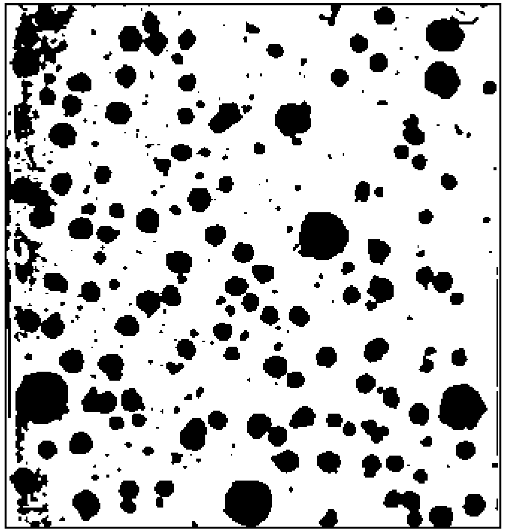


Image Segmentation



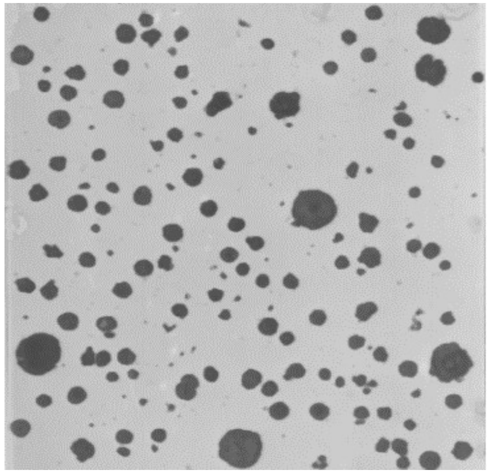
0.35

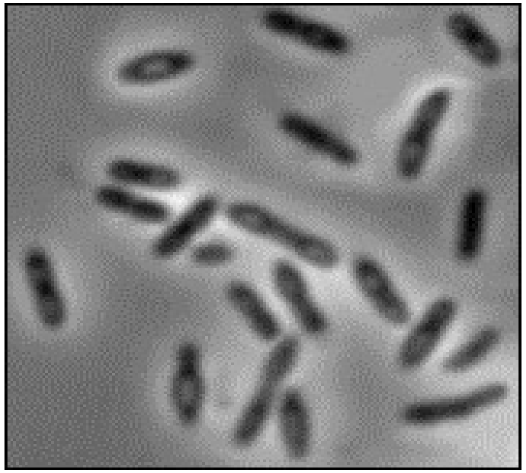


0.75

Original Image

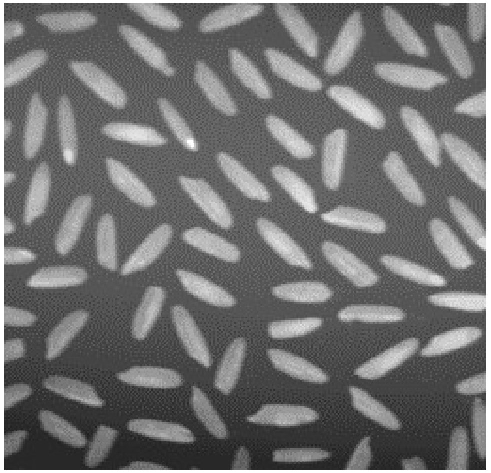
nodules1.tif





bacteria.tif

rice.tif



eight.tif

Threshold Value

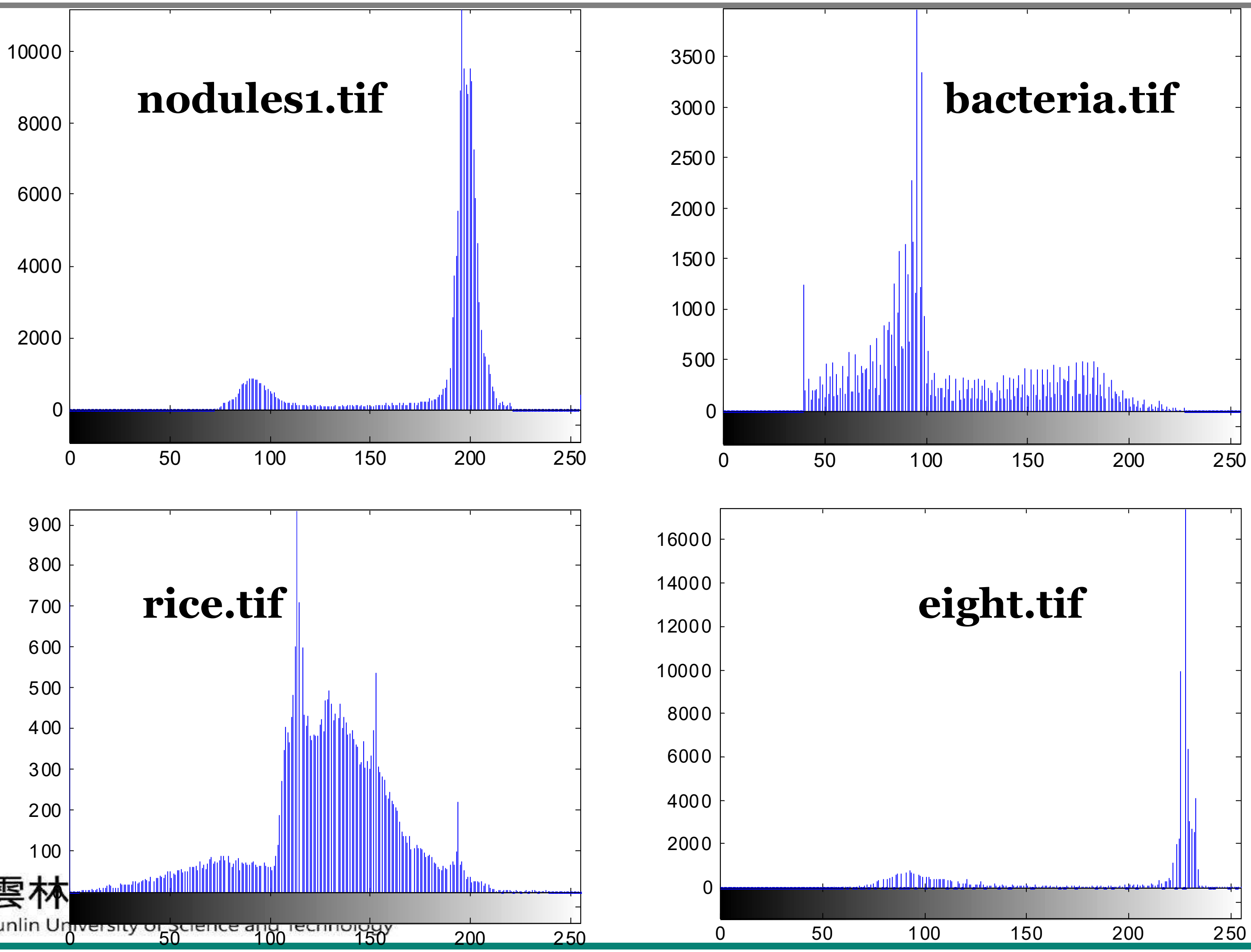
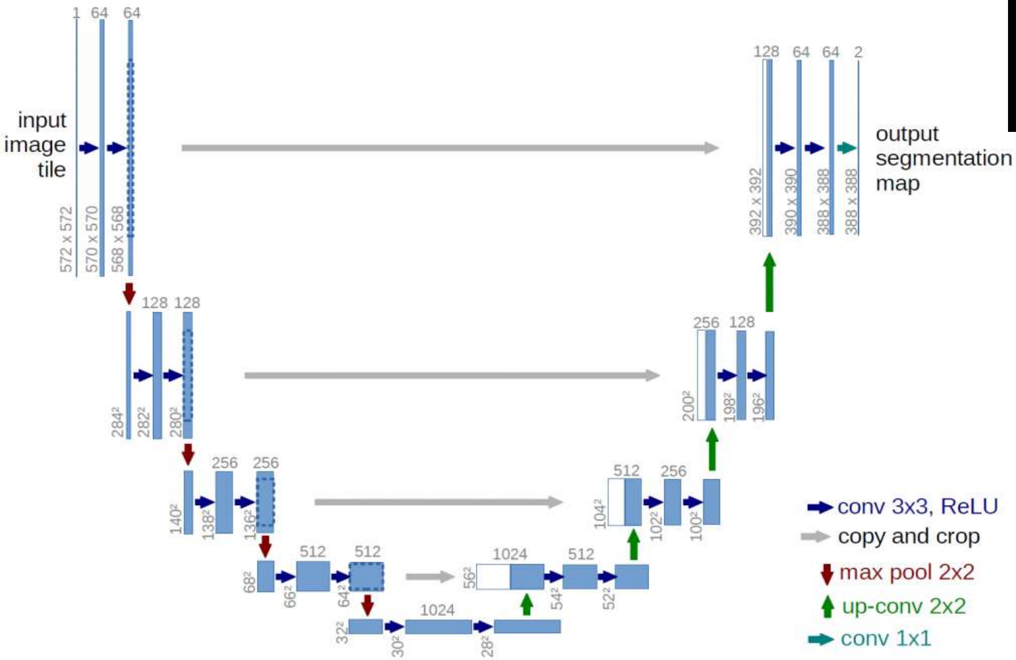
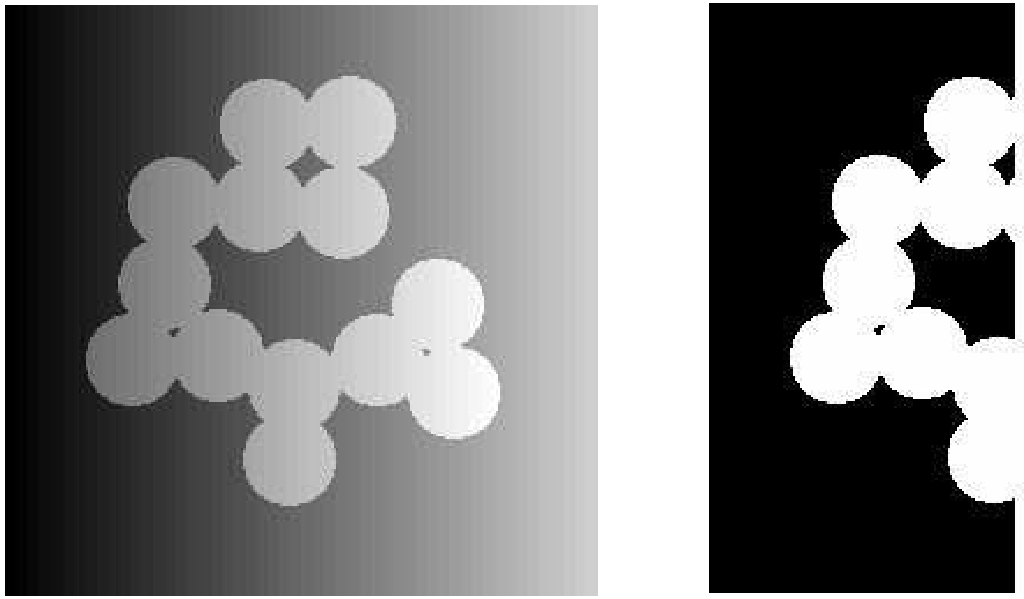


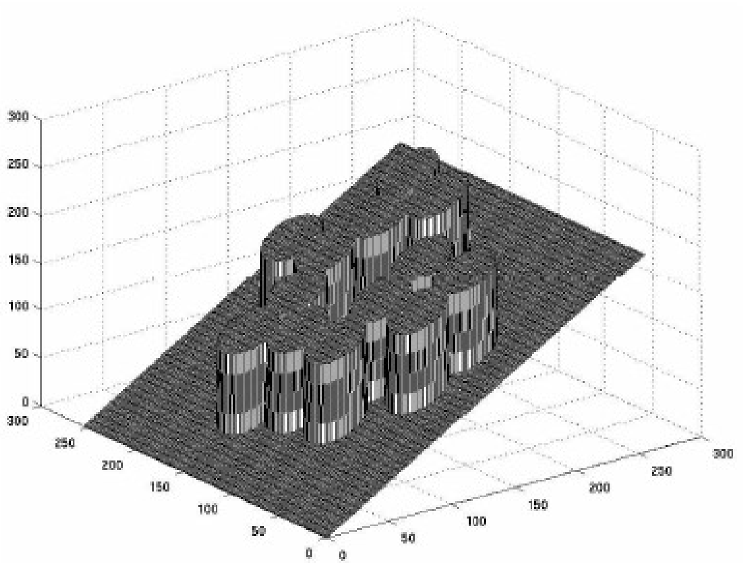
Image Segmentation



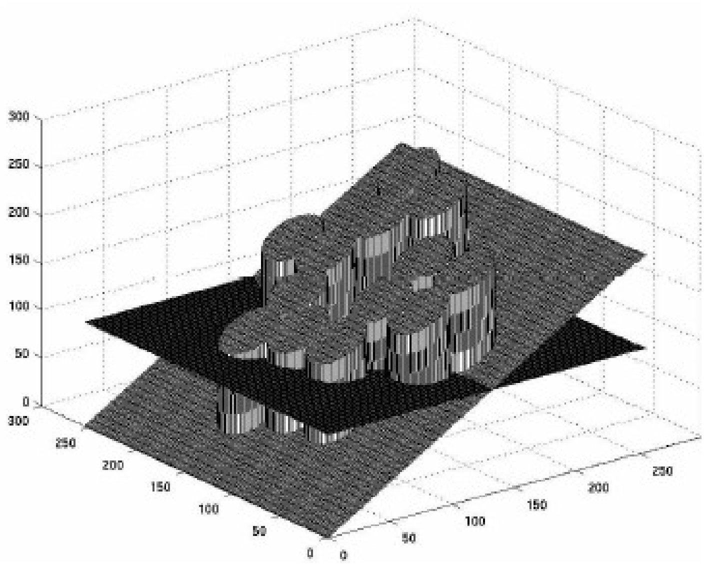
Adaptive Thresholding



Adaptive Thresholding

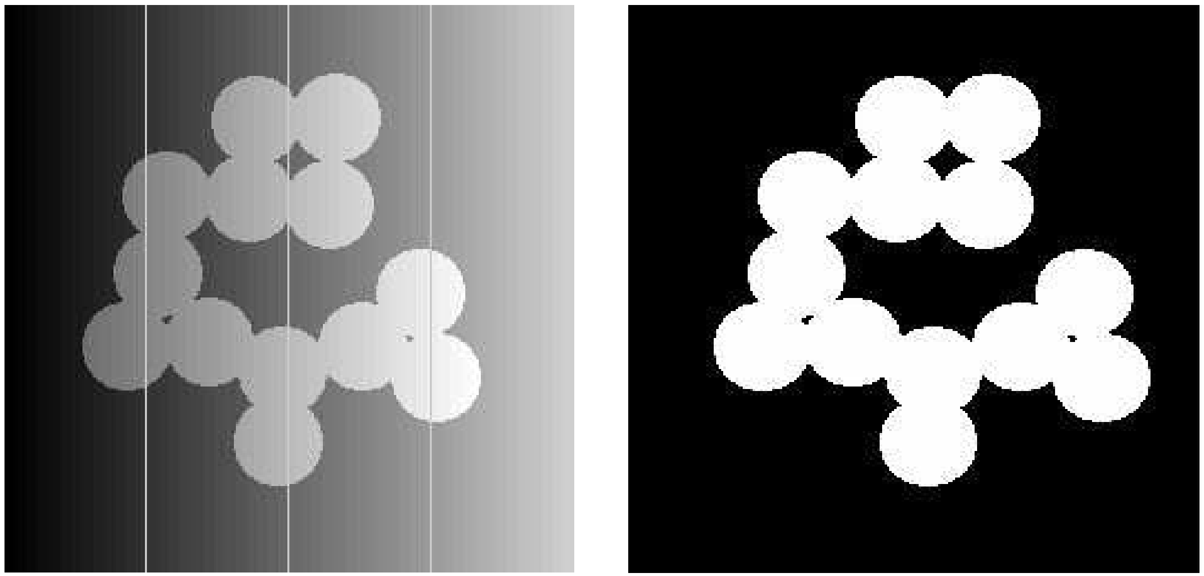


(a) The image as a function



(b) Thresholding attempt

Adaptive Thresholding



Otsu's Method

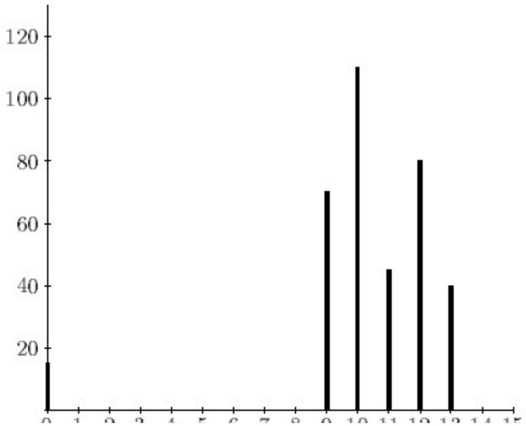
Automatic global thresholding algorithms usually have following steps.

1. Process the input image
2. Obtain image histogram (distribution of pixels)
3. Compute the threshold value T

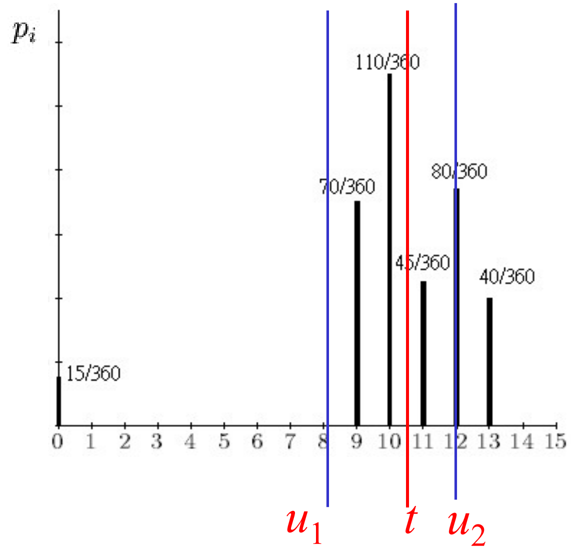
Replace image pixels into white in those regions, where saturation is greater than T and into the black in the opposite cases.

Usually, different algorithms differ in step 3

Otsu's Method



Grey level i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n_i	15	0	0	0	0	0	0	0	0	70	110	45	80	40	0	0



$$\sigma_w^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)$$

$$w_1(t) = \sum_{i=1}^{t} p(i) \qquad w_2(t) = \sum_{i=t+1}^{I} p(i)$$

$$\mu_1(t) = \sum_{i=1}^{t} \frac{ip(i)}{w_1(t)} \qquad \mu_2(t) = \sum_{i=t+1}^{I} \frac{ip(i)}{w_2(t)}$$

$$\sigma_1^2(t) = \sum_{i=1}^{t} [i - \mu_1(t)]^2 \frac{p(i)}{w_1(t)} \qquad \sigma_2^2(t) = \sum_{i=t+1}^{I} [i - \mu_2(t)]^2 \frac{p(i)}{w_2(t)}$$

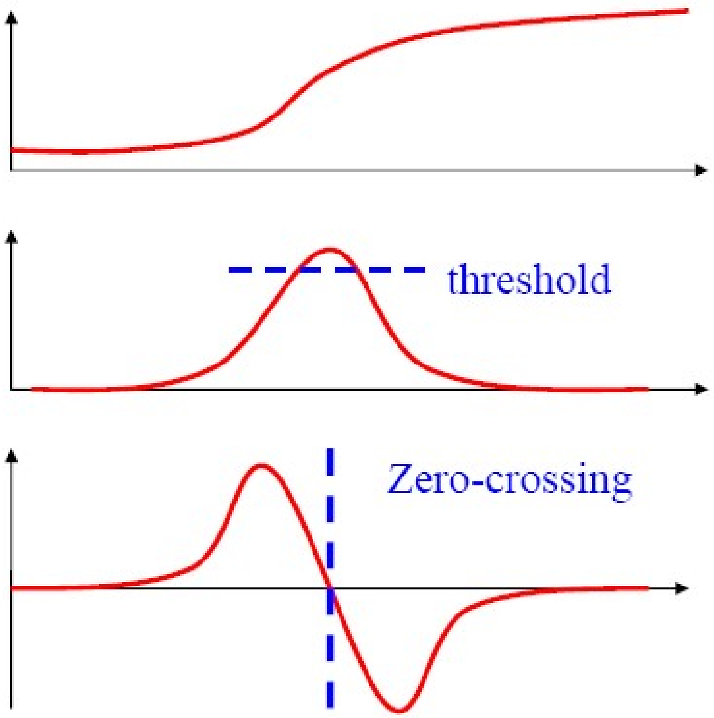
Edge Detection

- 1st derivatives operators
  - Prewitt operator
  - Sobel operator
  - Directional edge detection
- 2nd derivative operators
  - Laplacian
  - High-order finite difference approximation

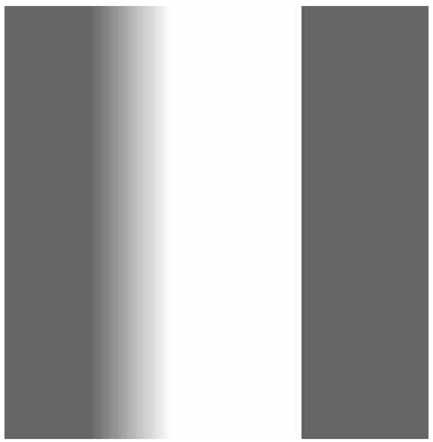
Derivative Operators

1st derivative

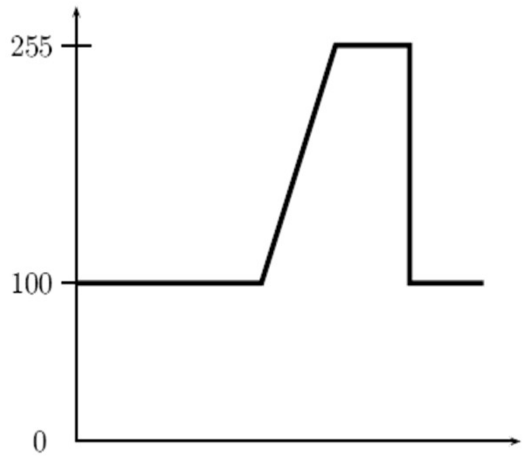
2nd derivative

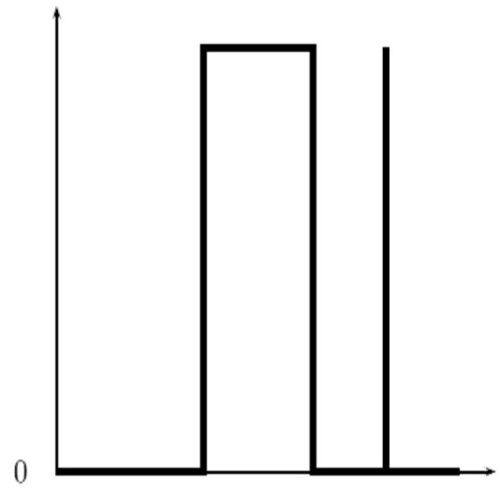


Derivative Operators



ramp edge step edge
(ideal edge)

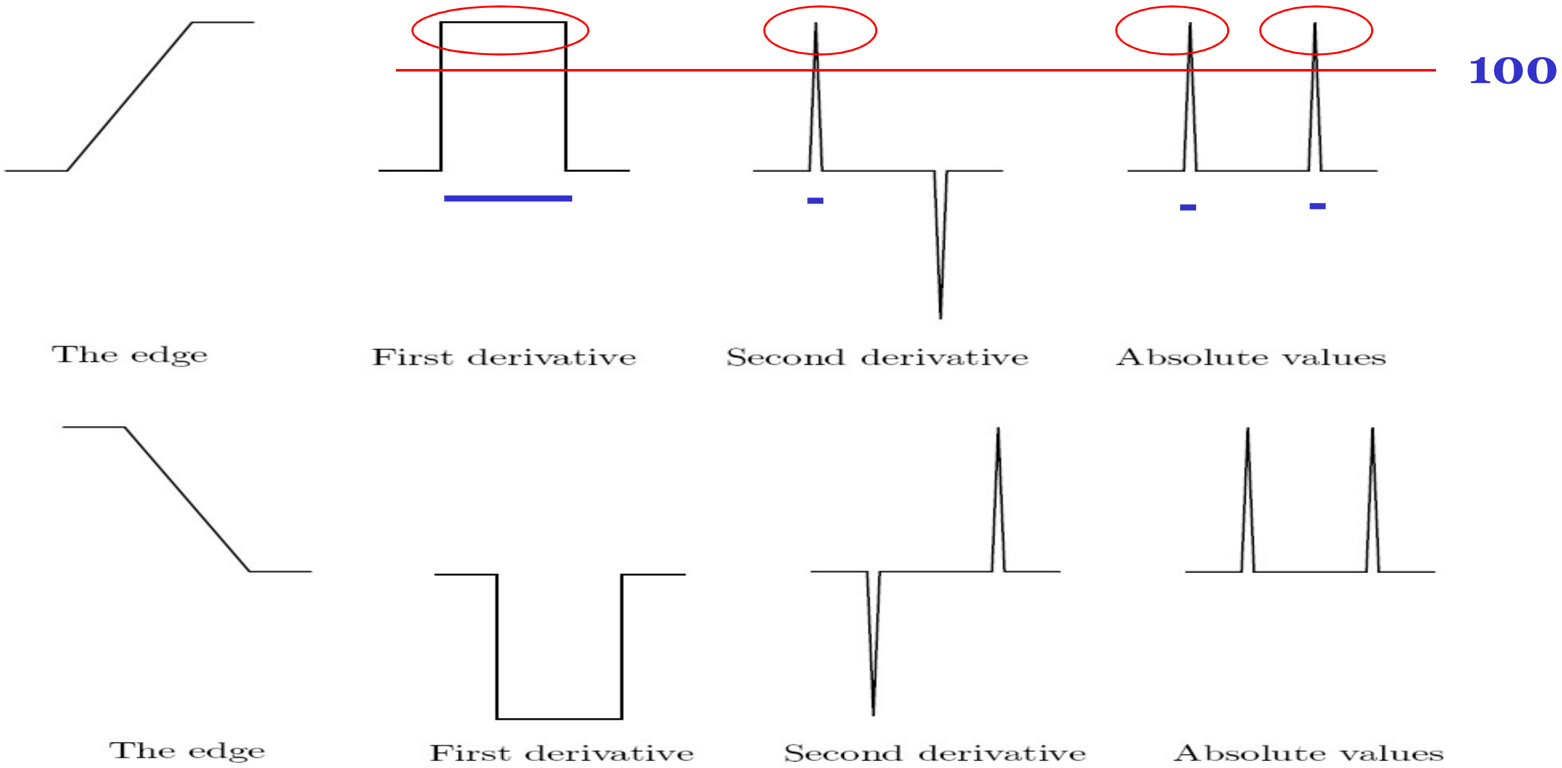




1st derivatives

Derivative Operators

- **Fundamental definitions**



The definition of the 1st derivative

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

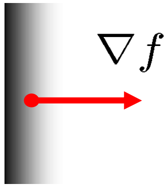
$$f(x+1) - f(x)$$

$$\lim_{h \to 0} \frac{f(x) - f(x-h)}{h}, \qquad \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h}$$

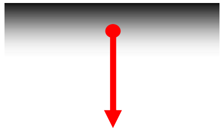
$$f(x) - f(x-1), \qquad (f(x+1) - f(x-1))/2.$$

Derivatives and Edges

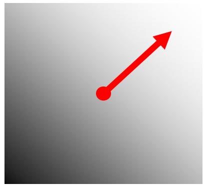
- The gradient of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$
- The gradient points in the direction of most rapid change in intensity



$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$



$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$



$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

- The gradient direction is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

- The edge strength is given by the gradient magnitude:

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Some edge detection filters

- Using the expression $f(x+1) - f(x-1)$ for the derivative, produces horizontal and vertical filters:

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

- Prewitt filters

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad P_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\sqrt{p_x^2 + p_y^2}. \qquad \max\{|p_x|, |p_y|\} \quad \text{or} \quad |p_x| + |p_y|.$$

Some edge detection filters

- Roberts cross-gradient filters

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- Sobel filters:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Second Derivatives

- The Laplacian

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\cong \left[f(x+1,y) - 2f(x,y) + f(x-1,y)\right] + \left[f(x,y+1) - 2f(x,y) + f(x,y-1)\right]$$

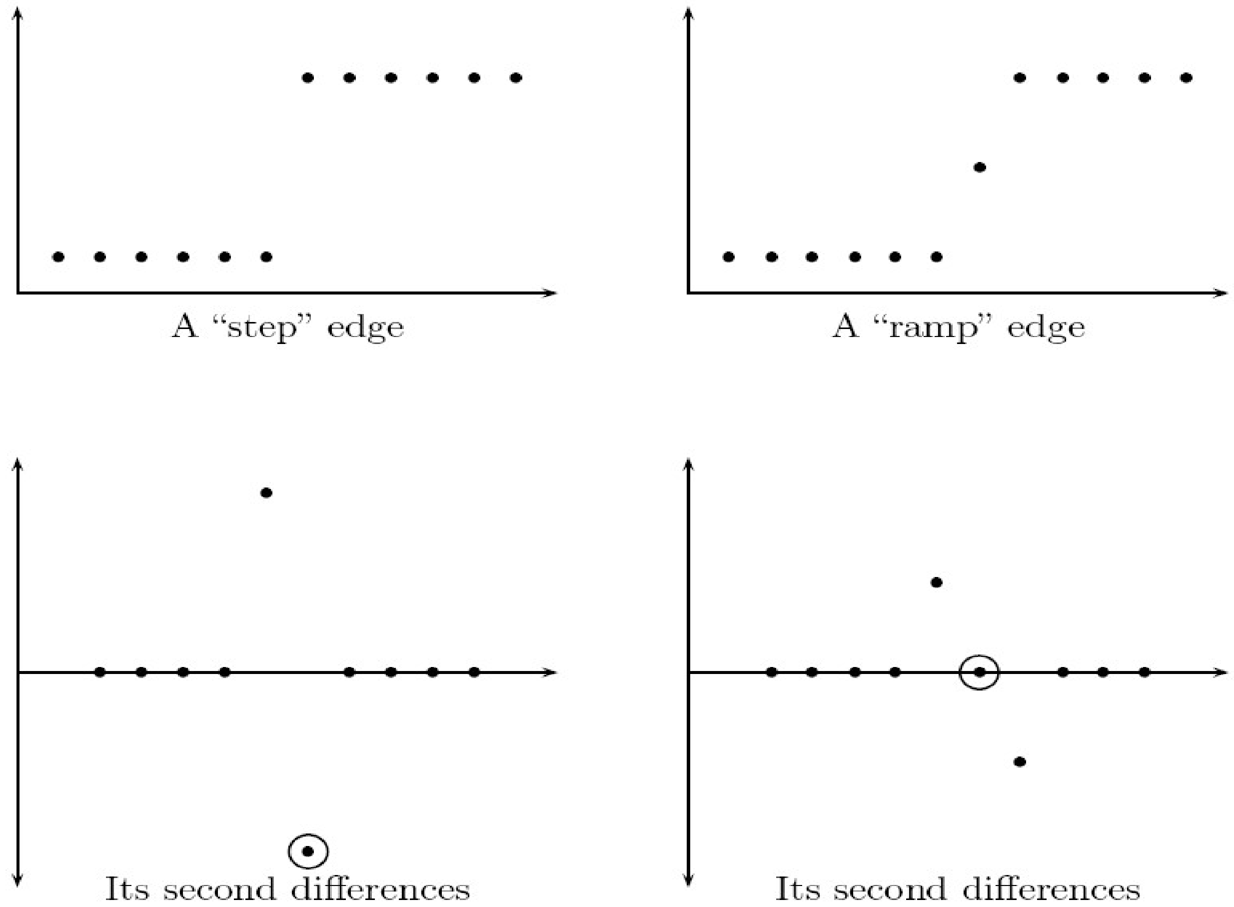
$$\nabla^2 f(x,y) \cong \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Other Laplacian masks

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

$$\frac{1}{\alpha+1}\begin{bmatrix} \alpha & 1-\alpha & \alpha \\ 1-\alpha & -4 & 1-\alpha \\ \alpha & 1-\alpha & \alpha \end{bmatrix}$$

Zero Crossings



Zero Crossings

50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	50	50	200	200	200	200	50	50
50	50	50	50	200	200	200	200	50	50
50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50

(a) A simple image

−100	−50	−50	−50	−50	−50	−50	−50	−50	−100
−50	0	150	150	150	150	150	150	0	−50
−50	150	−300	−150	−150	−150	−150	−300	150	−50
−50	150	−150	0	0	0	0	−150	150	−50
−50	150	−150	0	0	0	0	−150	150	−50
−50	150	−300	−150	0	0	0	−150	150	−50
−50	0	150	300	−150	0	0	−150	150	−50
−50	0	0	150	−300	−150	−150	−300	150	−50
−50	0	0	0	150	150	150	150	0	−50
−100	−50	−50	−50	−50	−50	−50	−50	−50	−100

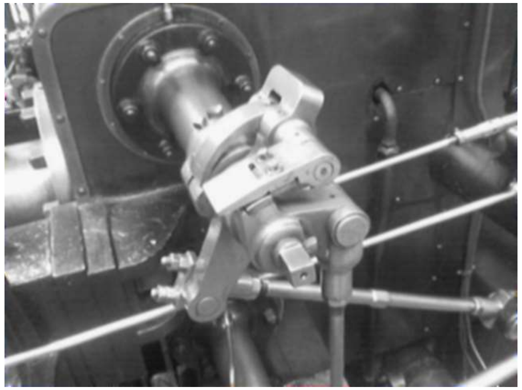
(b) After laplace filtering

Canny Edge Detection

- **Step 1: noise reduction**

  the Canny edge detector uses a filter based on the first derivative of a Gaussian.

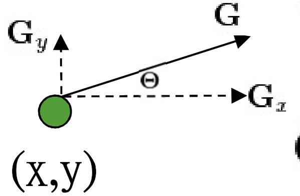
$$\mathbf{B} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * \mathbf{A} \qquad \sigma = 0.4$$



Canny Edge Detection

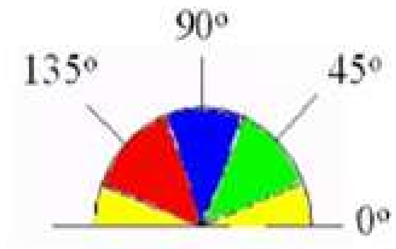
- **Step 2: Finding the intensity gradient of the image.**

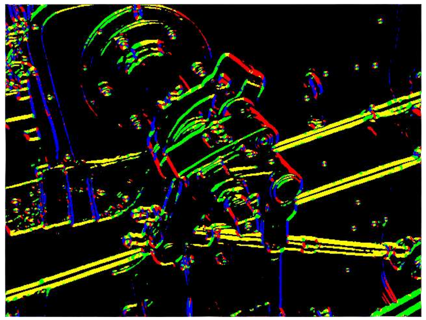
  An edge in an image may point in a variety of directions, so the Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the blurred image. The edge direction angle is **rounded** to one of four angles representing vertical, horizontal and the two diagonals (0, 45, 90 and 135 degrees for example).



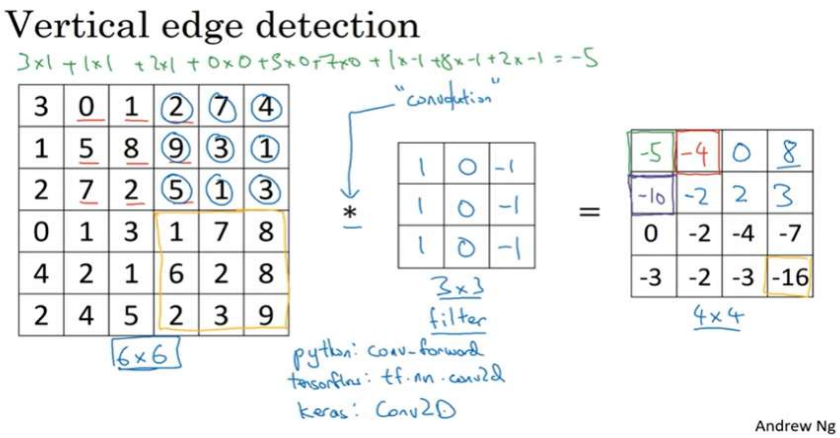
$$\mathbf{G} = \sqrt{\mathbf{G}_x^{\,2} + \mathbf{G}_y^{\,2}}$$

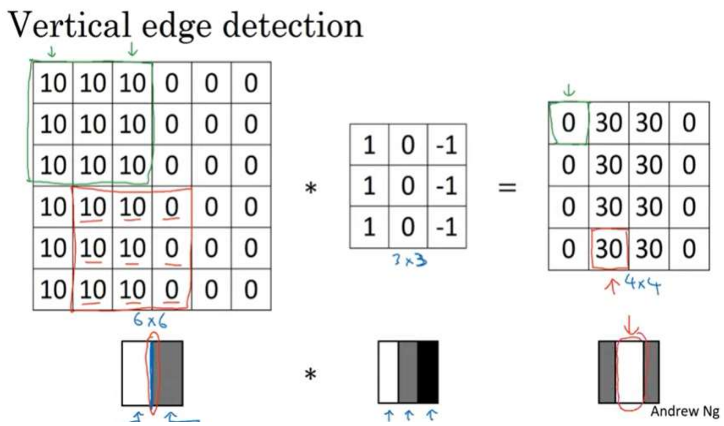
$$\mathbf{\Theta} = \operatorname{arctan}\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right)$$





Canny Edge Detection

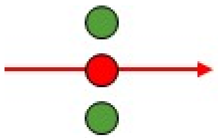




https://medium.com/@helen3909/edge-detection-%E9%82%8A%E7%B7%A3%E5%81%B5%E6%B8%AC-d5908aeb8e7c

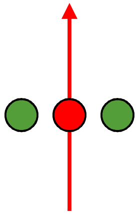
Canny Edge Detection

- Step 3:Non-maximum suppression (local maximum)

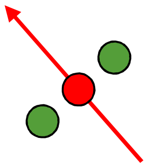


Given estimates of the image gradients, a search is then carried out to determine if the gradient magnitude assumes a local maximum in the gradient direction.

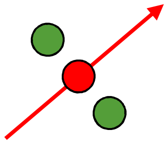
  - if the rounded angle is **zero** degrees the point will be considered to be on the edge if its intensity (gradient magnitude) is greater than the intensities in the **north and south** directions,



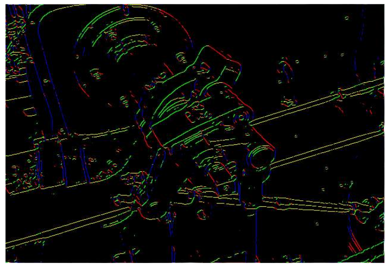
  - if the rounded angle is **90** degrees the point will be considered to be on the edge if its intensity is greater than the intensities in the **west and east** directions,



  - if the rounded angle is **135** degrees the point will be considered to be on the edge if its intensity is greater than the intensities in the **north east and south west** directions,

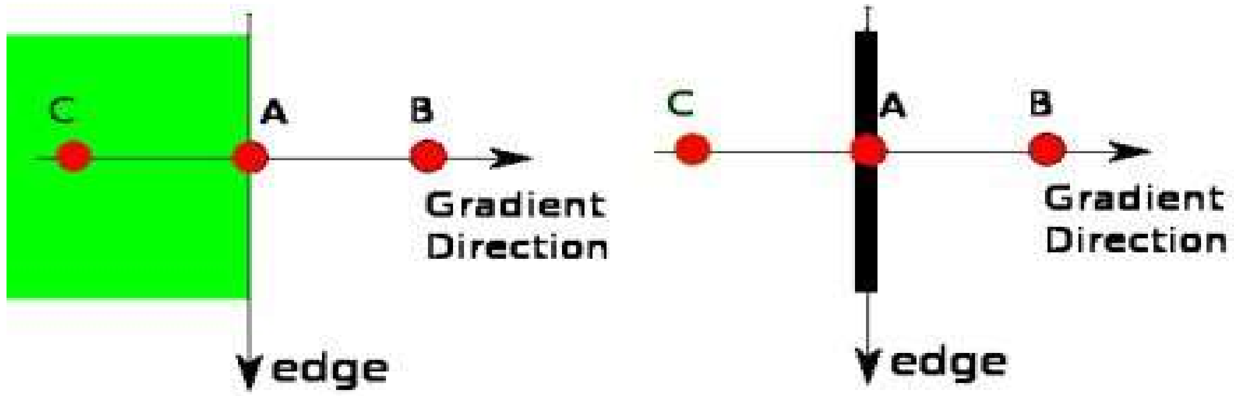


  - if the rounded angle is **45** degrees the point will be considered to be on the edge if its intensity is greater than the intensities in the **north west and south east** directions.



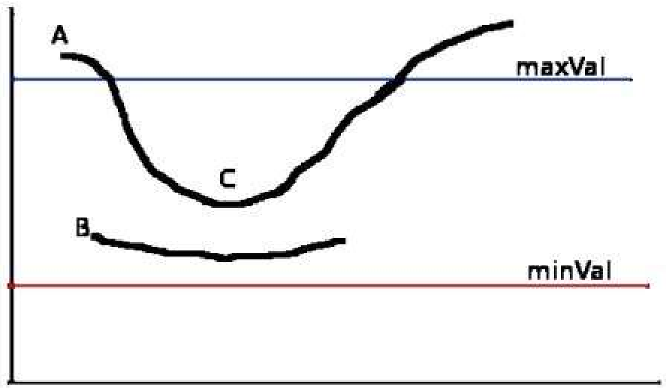
Canny Edge Detection

- Step 3:Non-maximum suppression (local maximum)



Canny Edge Detection

- Step 4: Tracing edges through the image and hysteresis thresholding
  - Thresholding with hysteresis requires two thresholds - high and low.
  - Making the assumption that important edges should be along continuous curves in the image allows us to follow a faint section of a given line and to discard a few noisy pixels that do not constitute a line but have produced large gradients.
  - Once this process is complete we have a binary image where each pixel is marked as either an edge pixel or a non-edge pixel.



Canny Edge Detection

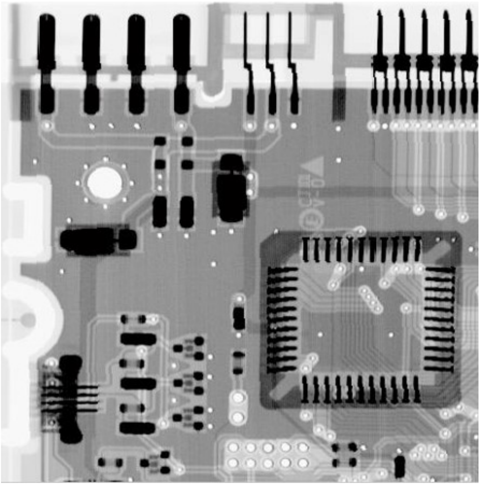
Step 1

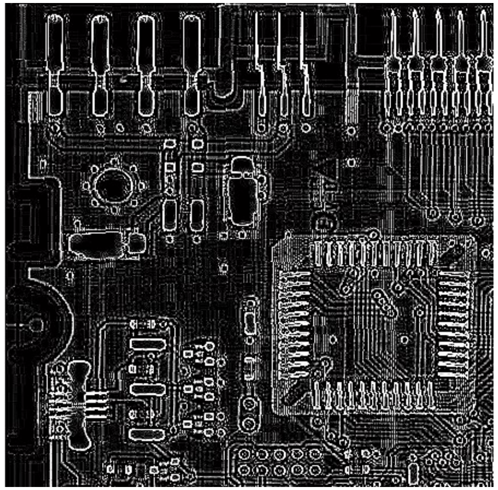
Step 2

Step 3

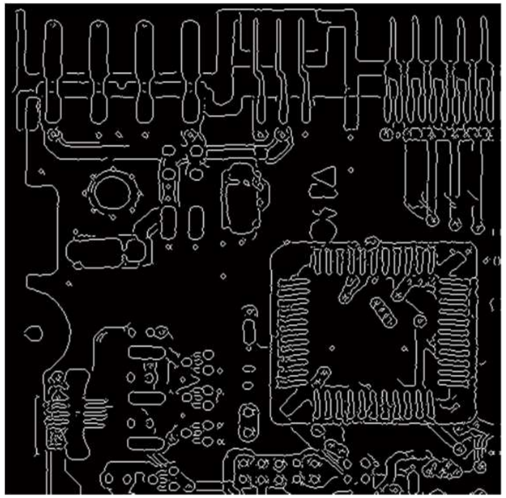
Step 4

Canny Edge Detection





Sobel



Canny Edge