Digital Image Processing

Lecture 5
Image Segmentation 2

Rahul JAIN

Spring 2022

Outline

- Edges detected by edge detector algorithms might be disconnected due to noise
- Hough transform
  - It can detect continuous straight lines even when the edges are disconnected
  - If edge points are sparse, resulting image may consist of individual points rather than straight line or curves
  - It is useful to fit a line to those points to establish a boundary between the regions
- Otsu's method
  - A thresholding technique that occupies a central position in image segmentation

Idea of Hough transform

- Hough transform is designed to find lines in images
- It can be easily varied to find other shape
- Suppose (*x*, *y*) is a point in the image

$$y = ax + b$$

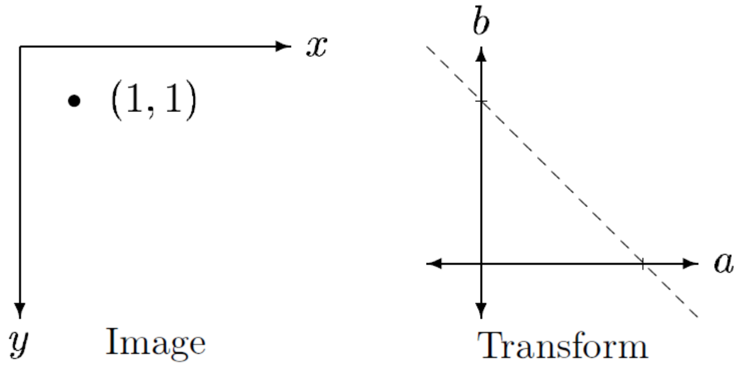
Consider (*a*, *b*) all pairs which satisfy this equation

For example (*x*, *y*) = (1, 1)

$$1 = a.1 + b$$

$$b = -a.x + y$$

$$b = -a.1 + 1$$



A point and its corresponding line in the transform

Idea of Hough transform

- We can consider *ab*-plane (parameter space) yields the equation of a single line for a fixed pair (x_i, x_i)
- Each point in the line can be mapped onto a line in the transform
- (*a*, *b*) array is the accumulator array or transform array **(*ab*-plane)**
- The point in the transform corresponding to the greatest number of intersections correspond to the strongest line in the image

$$b = -a.x + y$$

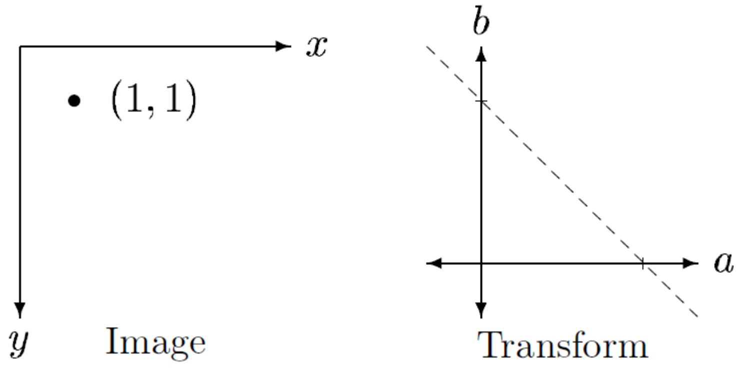
$(1,0)$: $b = -a$

$(1,1)$: $b = -a + 1$

$(2,1)$: $b = -2a + 1$

$(4,1)$: $b = -4a + 1$

$(3,2)$: $b = -3a + 2$



A point and its corresponding line in the transform

Idea of Hough transform

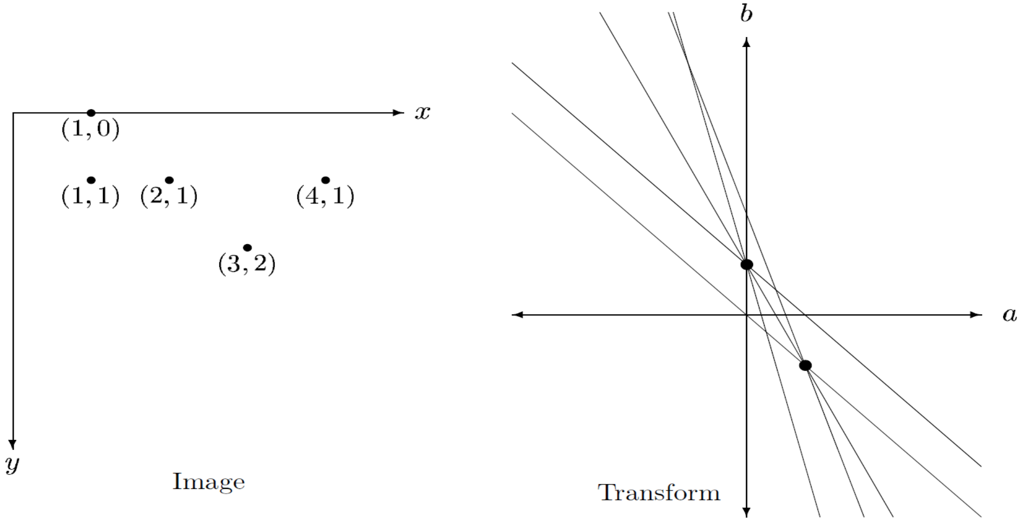


Image and its corresponding lines in the transform

$(1, 0): b = -a$

$(1, 1): b = -a + 1$

$(2, 1): b = -2a + 1$

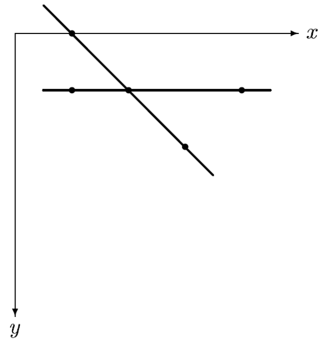
$(4, 1): b = -4a + 1$

$(3, 2): b = -3a + 2$

- The dots in the transform indicate places where there are maximum intersections of lines $(1, 0)$ and $(1, -1)$

$y = 1.x + 0$

$y = 1.x + (-1)$



Idea of Hough Transform

- Hough transform can not find vertical lines because we can not express a vertical line in the form of $y = mx + c$ where *m* represents gradient, and a vertical line has infinite gradient (*c* is the *y*–intercept, the point in which the line crosses the *y*–axis)

- We need another parameterization of lines

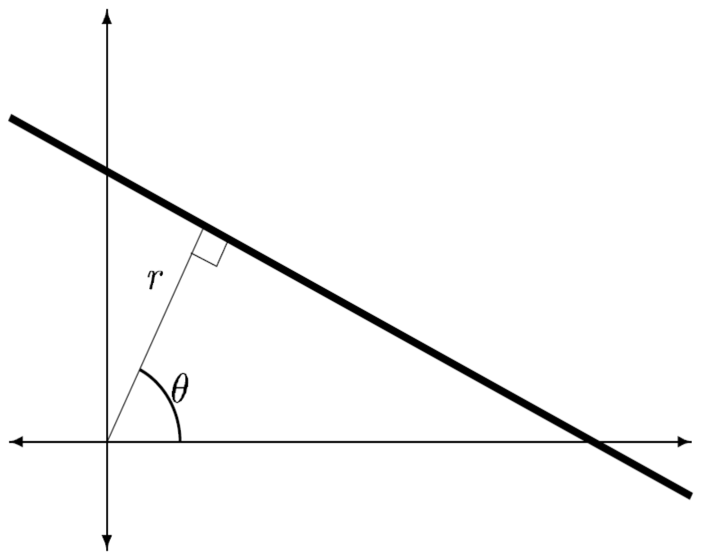
- we can describe any line in terms of two parameters

  r and θ

r is perpendicular distance from the line to origin

θ is the angle of the line's perpendicular to the *x*-axis

In this parameterization, vertical lines are simply those which have θ = 0



A line and its parameters

Idea of Hough Transform

- If we allow *r* to have negative values, we can restrict θ to the image

$$-90 < \theta < 90$$

- For any point (*p, q*) where the perpendicular to the line meets the line is

$$(p, q) = (r\cos\theta + r\sin\theta)$$

Note that the gradient of the perpendicular is

$$\tan\theta = {}^{\sin\theta}/_{\cos\theta}$$

For any point (*x*, *y*) on the line, the gradient of the line is, slope is:

$$\frac{rise}{run} = \frac{y - q}{x - p}$$

$$= \frac{y - r\sin\theta}{x - r\cos\theta}$$

Idea of Hough Transform

- Since the gradient of the line's perpendicular is $\tan\theta$, the gradient of the line itself must be:

$$-\frac{1}{\tan\theta} = -\frac{\cos\theta}{\sin\theta}$$

- Putting these two expressions for the gradient together produces:

$$\frac{y - r\sin\theta}{x - r\cos\theta} = -\frac{\cos\theta}{\sin\theta}$$

$$\mathrm{y}\sin\theta - r\,sin^2\theta = -x\cos\theta + \mathrm{r}\,cos^2\theta$$

$$\mathrm{y}\sin\theta + x\cos\theta = r\,sin^2\theta + \mathrm{r}\,cos^2\theta$$

$$\mathrm{y}\sin\theta + x\cos\theta = r\,(sin^2\theta + cos^2\theta)$$

$$(sin^2\theta + cos^2\theta) = 1$$

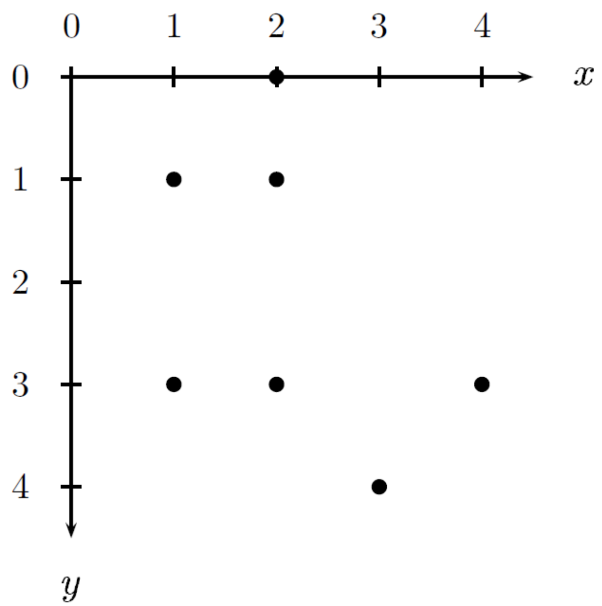
$$x\cos\theta + \mathrm{y}\sin\theta = r$$

Idea of Hough Transform

- The Hough transform can be implemented as follows:
- we start by choosing a discrete set of values of *r* and θ to use.
- For each pixel (*x*, *y*) in the image, we compute $x \cos\theta + \mathrm{y} \sin\theta$, for each value of θ
- Place the results in the appropriate position in the (*r*, θ) array
- The values of (*r*, θ) with the highest values in the array will correspond to strongest lines in the image

Idea of Hough Transform

- Consider the shown in the figure:



A small image

- We will discretize θ to use only the values

$$-45^o, 0^o, 45^o, 90^o$$

Idea of Hough Transform

- We can start by making a table containing all values $x\cos\theta + y\sin\theta$ for each point and each value of θ

(x, y)	$-45°$	$0°$	$45°$	$90°$
$(2, 0)$	1.4	2	1.4	0
$(1, 1)$	0	1	1.4	1
$(2, 1)$	0.7	2	2.1	1
$(1, 3)$	-1.4	1	2.8	3
$(2, 3)$	-0.7	2	3.5	3
$(4, 3)$	0.7	4	4.9	3
$(3, 4)$	-0.7	3	4.9	4

- The accumulator array contains the number of times each value of (r, θ) appears in the above table

	-1.4	-0.7	0	0.7	1	1.4	2	2.1	2.8	3	3.5	4	4.9
$-45°$	1	2	1	2		1							
$0°$					2		3			1		1	
$45°$						2		1	1		1		2
$90°$			1		2					3		2	

Idea of Hough Transform

- In practice this array will be very large and can be displayed as an image
- In this example the two equal largest values occur at $(r, \theta) = (2, 0^o)$ and $(r, \theta) = (3, 90^o)$

	-1.4	-0.7	0	0.7	1	1.4	2	2.1	2.8	3	3.5	4	4.9
$-45°$	1	2	1	2		1							
$0°$					2		3			1		1	
$45°$						2		1	1		1		2
$90°$			1		2					3		2	

- The lines then are

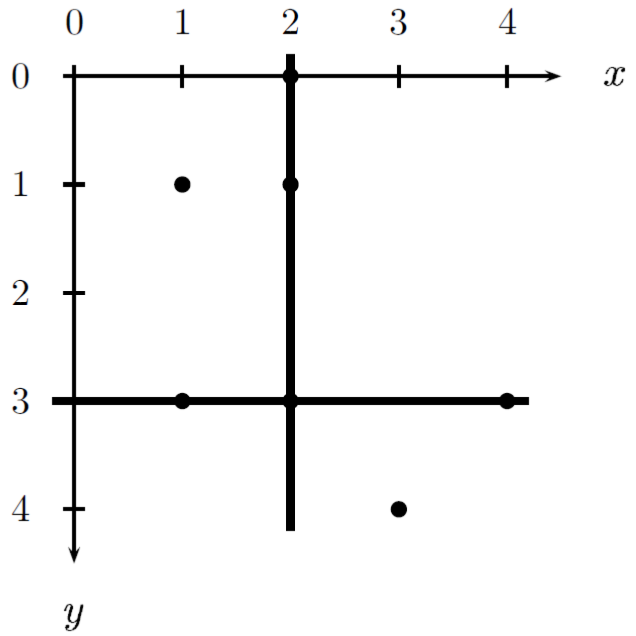
$x \cos 0 + y \sin 0 = 2$ or $x = 2$ ($\sin 0 = 0$, $\cos 0 = 1$)

and

$x \cos 90 + y \sin 90 = 3$ or $y = 3$

Idea of Hough Transform

- $x = 2$ and $y = 3$



Lines found by Hough transform

Hough transform

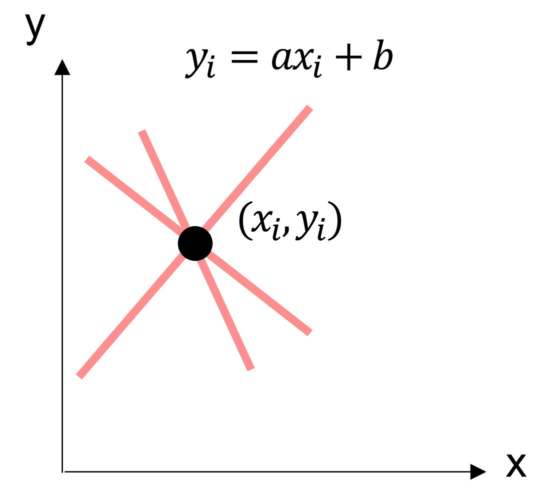
- Consider a point (x_i, y_i) in image space **(xy-plane)**
- All the lines passing through (x_i, y_i) satisfy the equation

$$y_i = ax_i + b$$

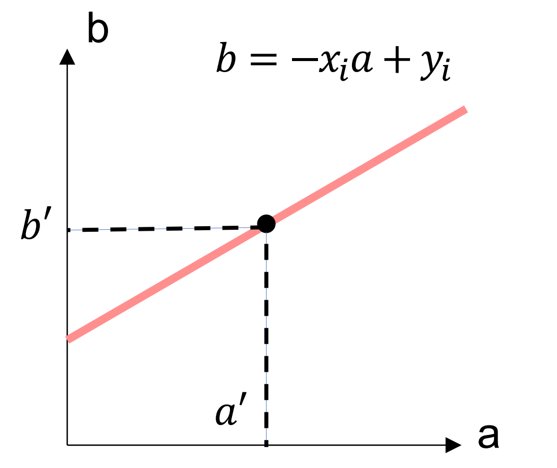
- Writing this equation in parameter space **(ab-plane)** as

$$b = -x_i a + y_i$$

the equation of a single line for fixed point (x_i, y_i)



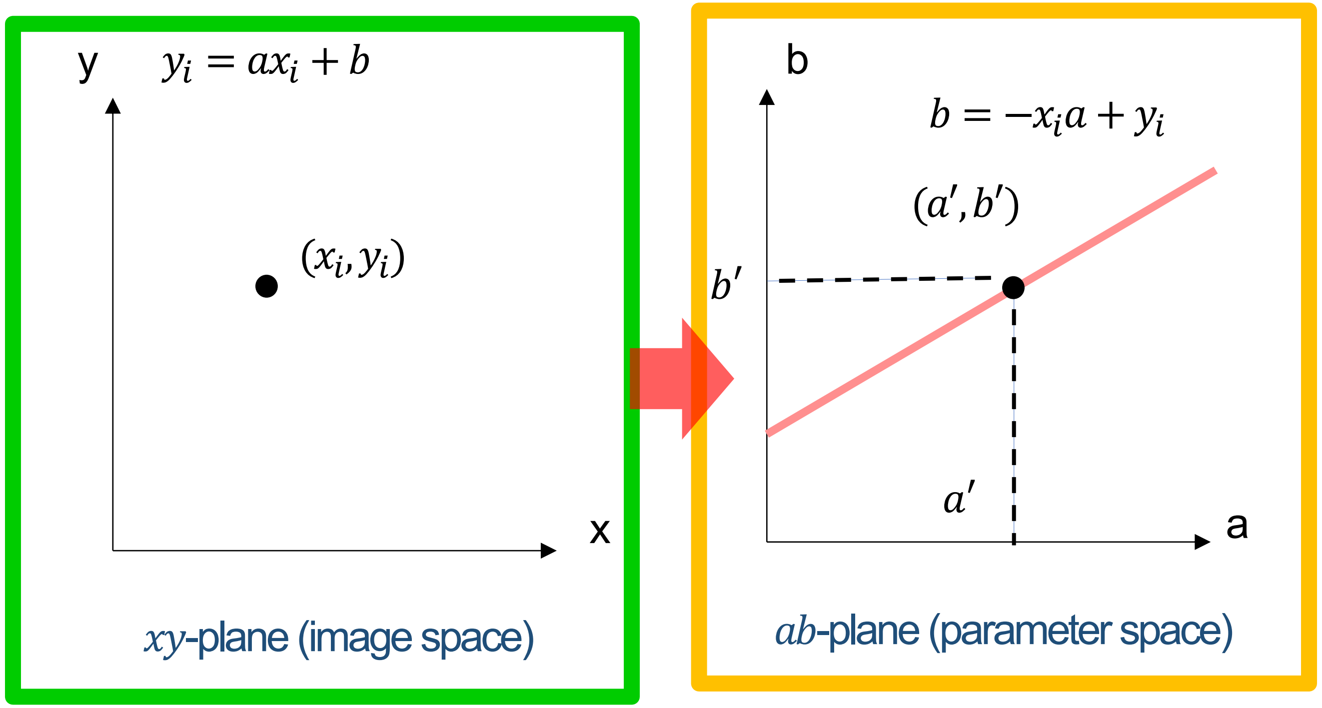
xy-plane (image space)



ab-plane (parameter space)

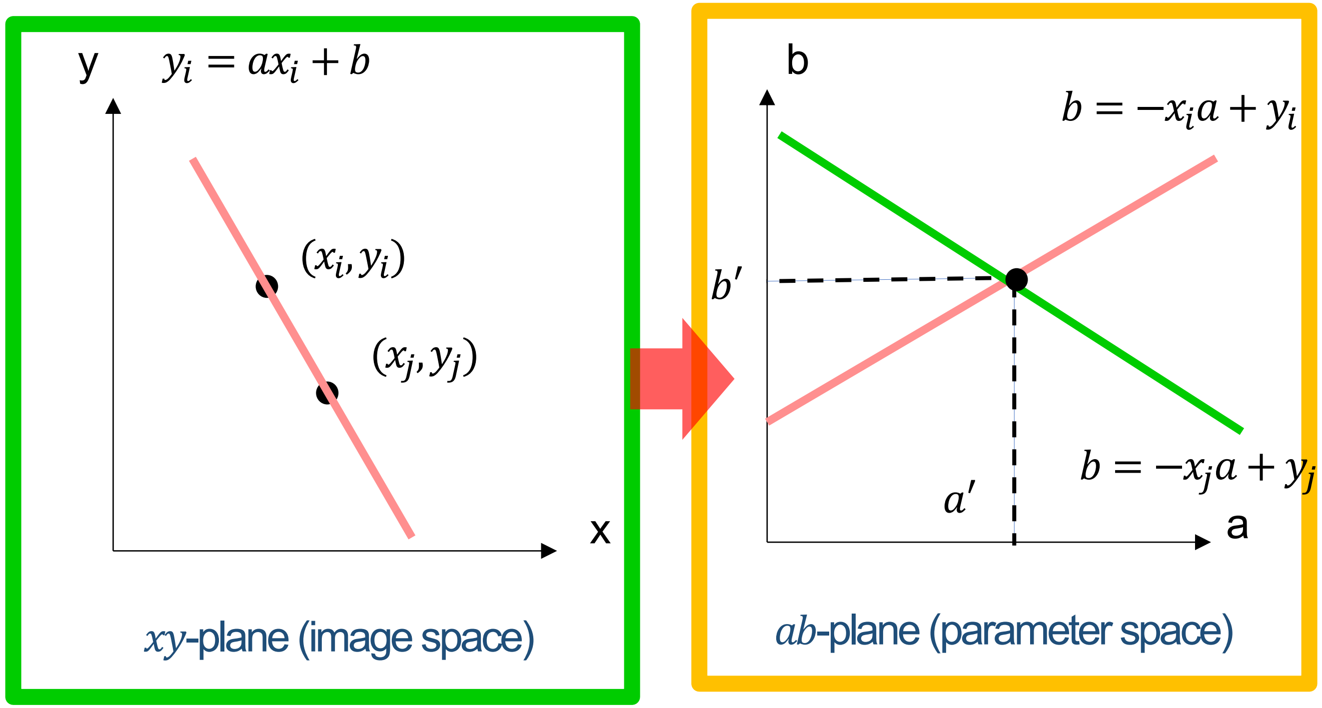
Hough transform

- A fixed point (x_i, y_i) in the image space corresponds to a single line in the parameter space



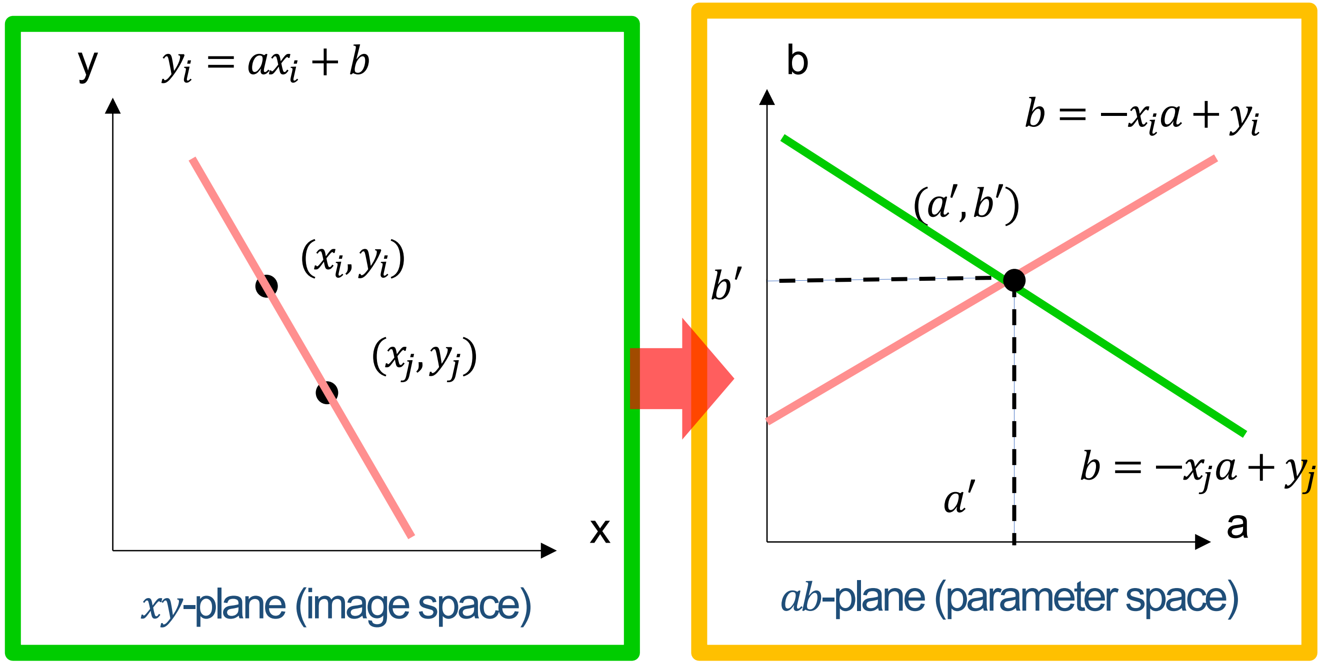
Hough transform

- A fixed point (x_i, y_i) in the image space corresponds to a single line in the parameter space
- Another point (x_j, y_j) on the same line also has a line in the parameter space associated with it



Hough transform

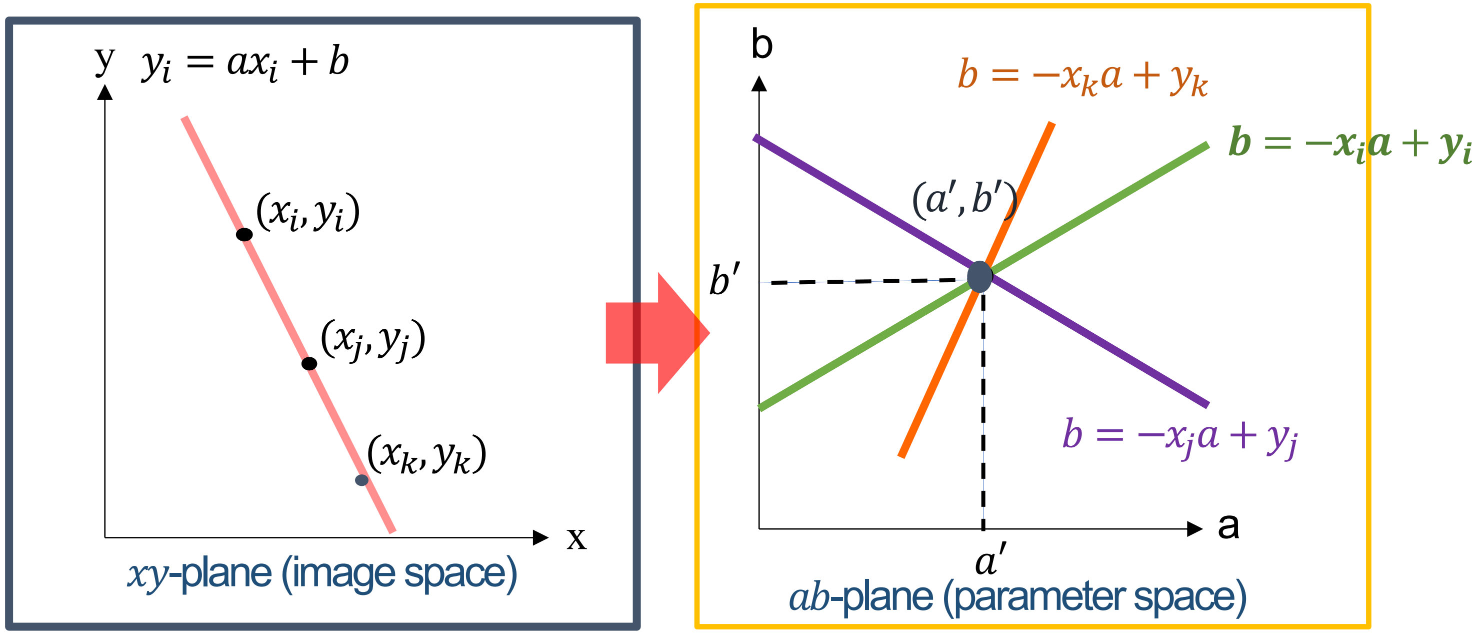
- A fixed point (x_i, y_i) in the image space corresponds to a single line in the parameter space
- Another point (x_j, y_j) on the same line also has a line in the parameter space associated with it



- The two lines intersect at some point (a', b')

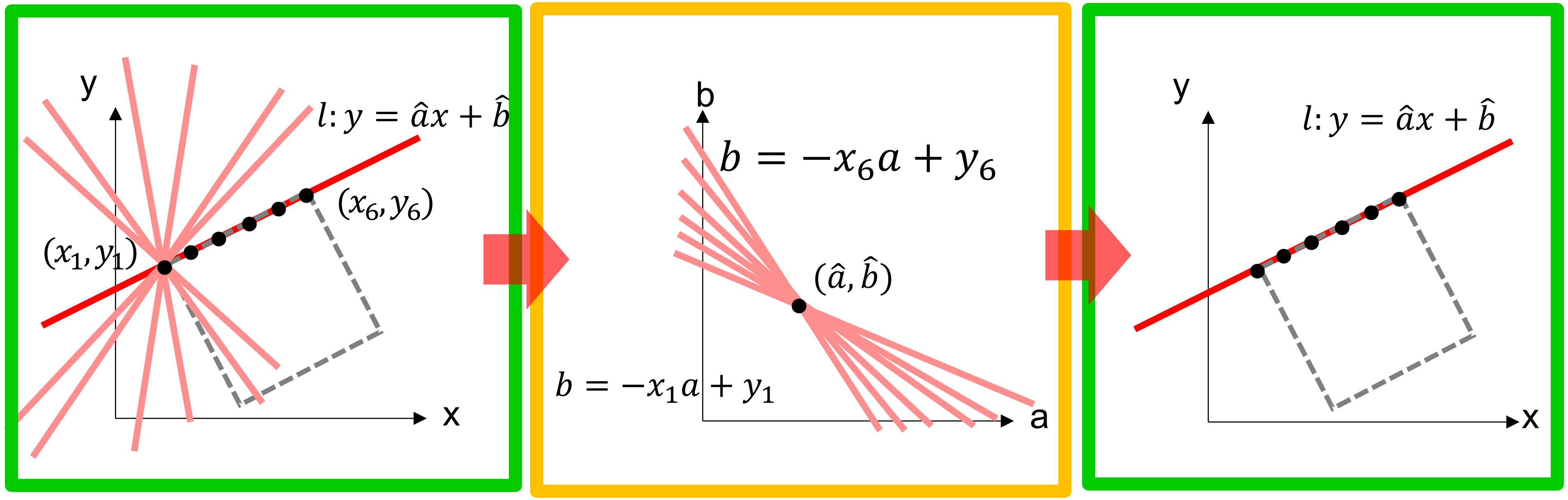
Hough transform

- All the points on the same line in the image space have lines in the parameter space that intersect at (a', b')



Hough transform

- In principle, the parameter space lines corresponding to all points in the *xy*-plane can be plotted
- the principal line in that plane can be found by identifying points in the parameter space where large numbers of parameter-space lines intersect



xy-plane (image space) *ab*-plane (parameter space) *xy*-plane (image space)

Normal Representation of a Line

- Let ρ be the length of the normal drawn from the origin to a line, which subtends an angle θ with the positive direction of x-axis.
- The normal representation of a line is defined as:

$$x \cos\theta + y \cos\theta = \rho$$

- A horizontal line has:

$\theta = 0^o$ and ρ: equal to the positive x-intercept

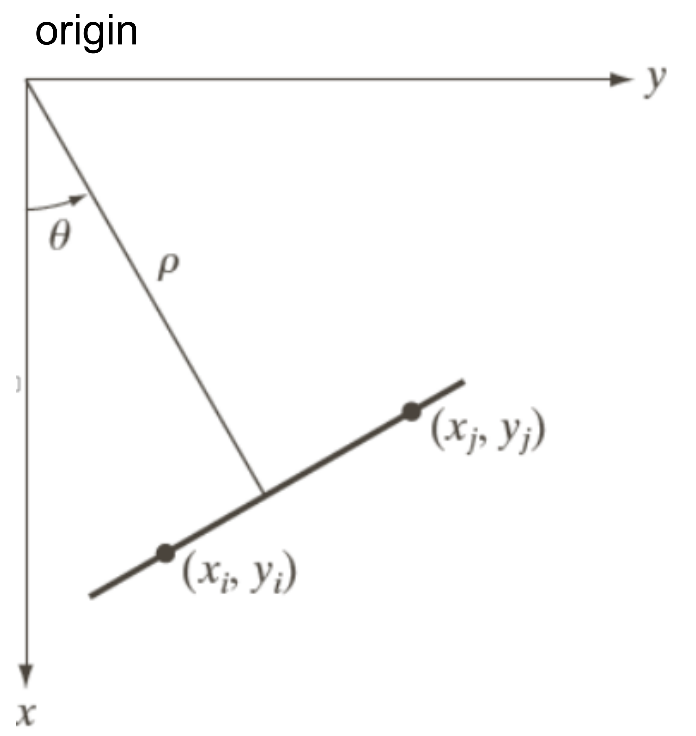
- A vertical line has

$$\theta = 90^o$$

ρ (Rho) : equal to the positive y-intercept

$$\theta = -90^o$$

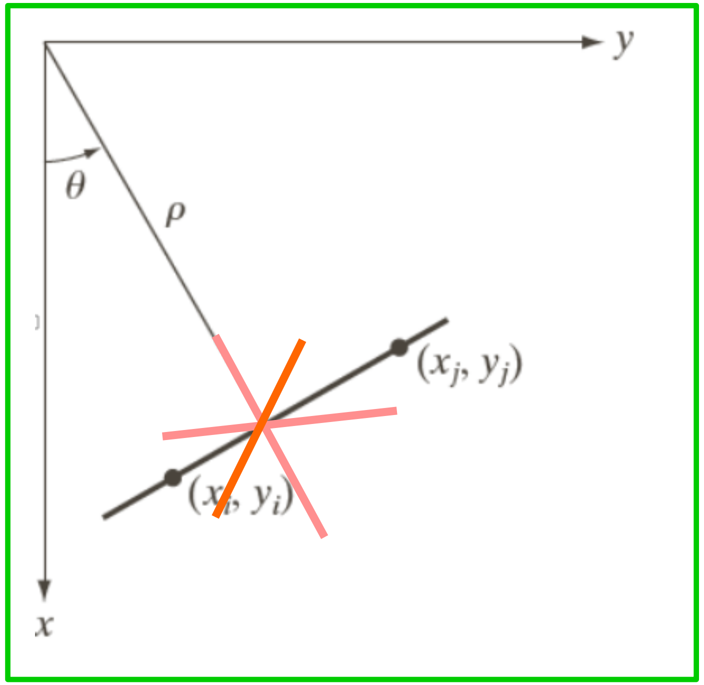
ρ: equal to the negative y-intercept



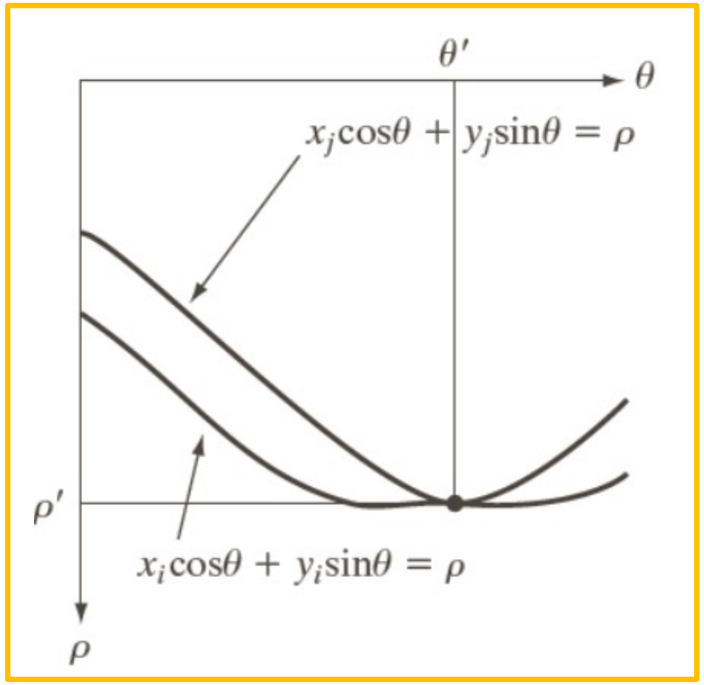
geometrical interpretation of ρ and θ

Normal Representation of a Line

- Consider the parameter space (also called **Hough space, or** $\rho\theta$-plane)
- Each sine curve represents the family of lines that pass through a particular point in the image *xy*-plane



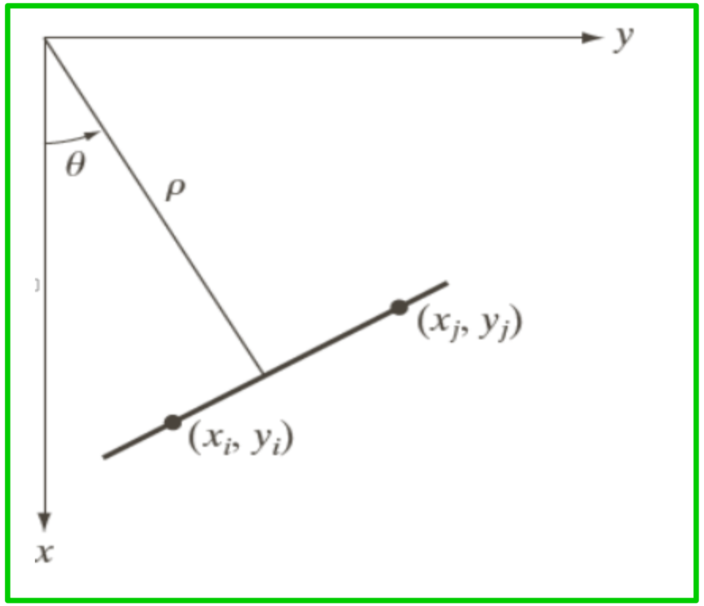
xy-plane (image space)



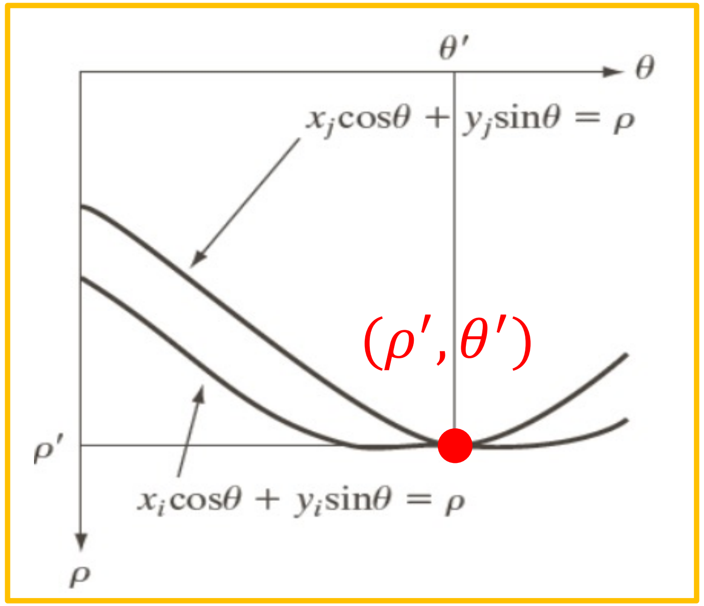
$\rho\theta$-plane (Hough space)

Normal Representation of a Line

- The points (x_i, y_i) and (x_j, y_j) in the image space correspond to two sine curves in the parameter space
- The intersection point (ρ', θ') corresponds to the line passing through both (x_i, y_i) and (x_j, y_j)



xy-plane (image space)



ρθ-plane (Hough space)

Division of Hough Space

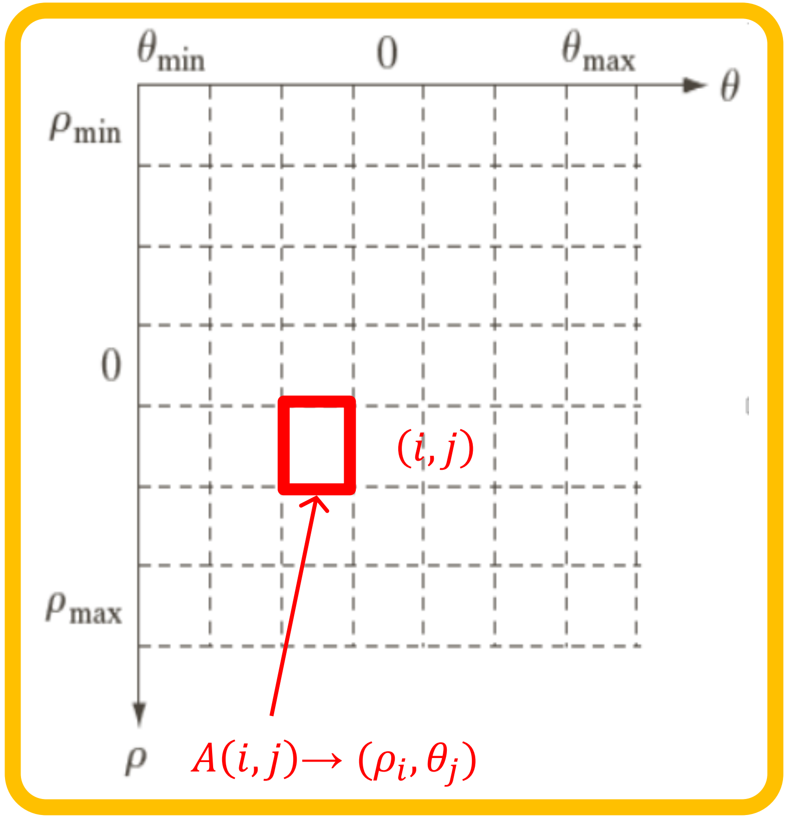
- Divide Hough space (the $\rho\theta$-plane) into **accumulator cells**
- Set the ranges of two parameters as:

$$-90^o \leq \theta \leq 90^o$$

$$-D \leq \rho \leq D$$

where D is the maximum distance between opposite corners in an image

- The cell at the coordinates (i, j) with accumulator value $A(i, j)$ corresponds to the square associated with parameter-space coordinates (ρ_i, θ_j)
- The number of subdivisions in the $\rho\theta$-plane determines the accuracy of the collinearity of the points



Accumulator cells

Algorithm of Hough Transform

Step 1. Obtain a binary edge image with filtering and thresholding (discussed in previous class)

e.g., use Sobel filters

$$f_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad f_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

The simplest thresholding:

$if\ M(i,j) \geq threshold$

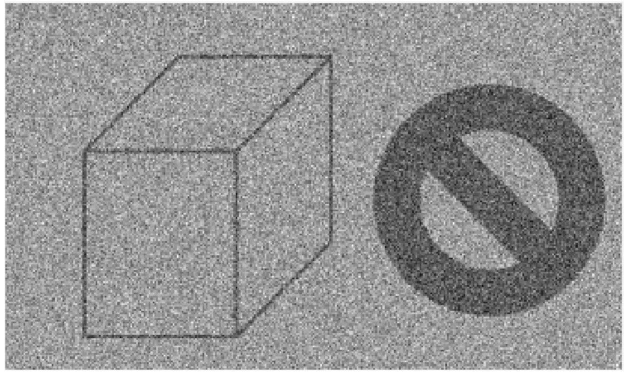
$\quad M(i,j)$=255

$else$

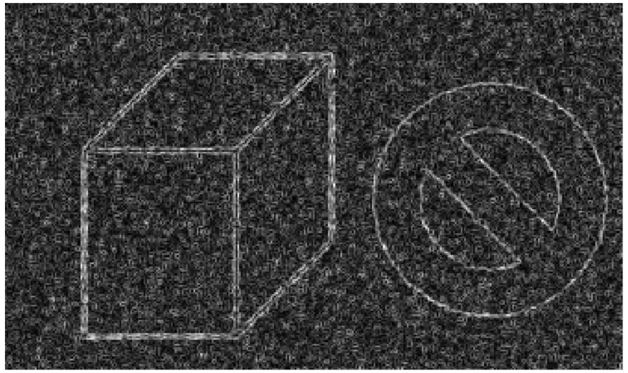
$\quad M(i,j)$=0

end

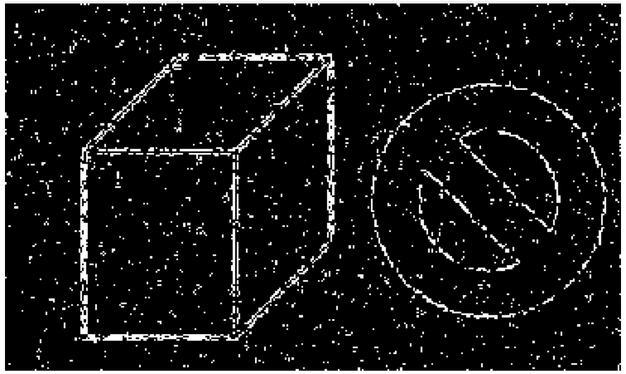
magnitude image $\quad M(i,j) = \sqrt{(g_x(i,j))^2 + (g_y(i,j))^2}$



input image (gray scale)



edge image



binary edge image $M(i,j)$

Algorithm of Hough Transform

Step 2. Transform the binary edge image into Hough space $\rho\theta$ −plane

(a) Accumulator cells are initially set to be zero ($A(i,j)$=0)

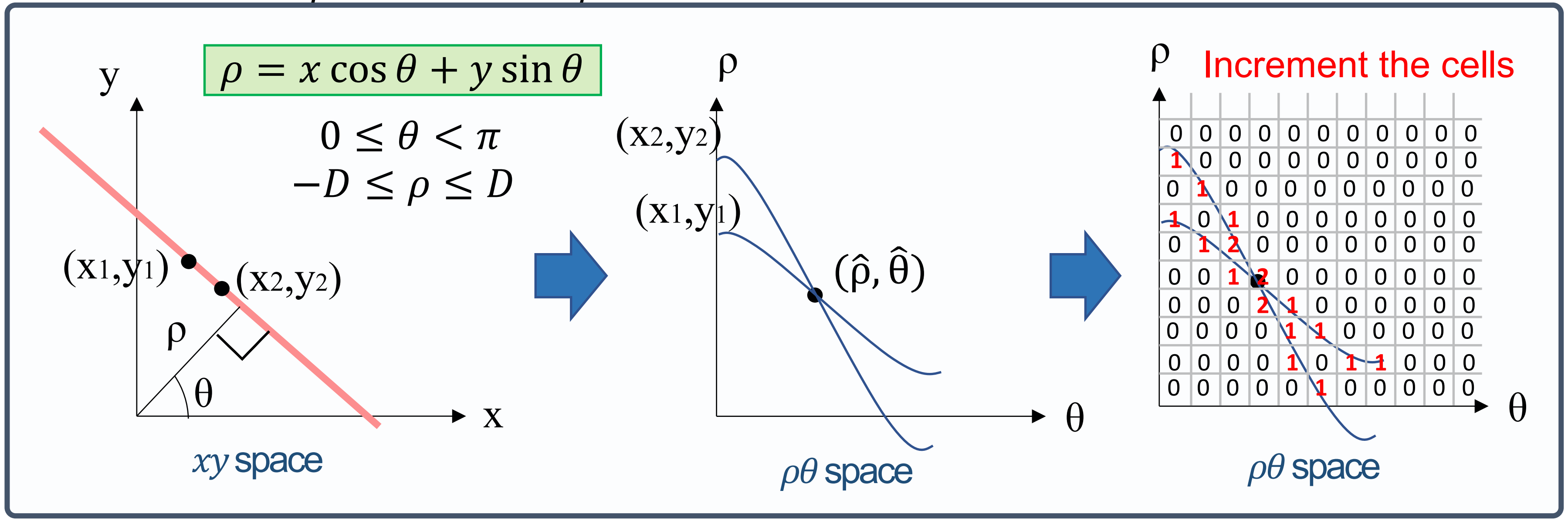
(b) For every point (x_k, y_k)=255 in the binary edge image

1) let θ be equal to each of the allowed values on the θ-axis

2) solve the equation of $x\cos\theta + y\sin\theta = \rho$ for ρ

3) round off ρ to the nearest allowed cell value on the ρ-axis

4) if a choice of θ_p results in ρ_q, let $A(p,q)$=$A(p,q)$+1

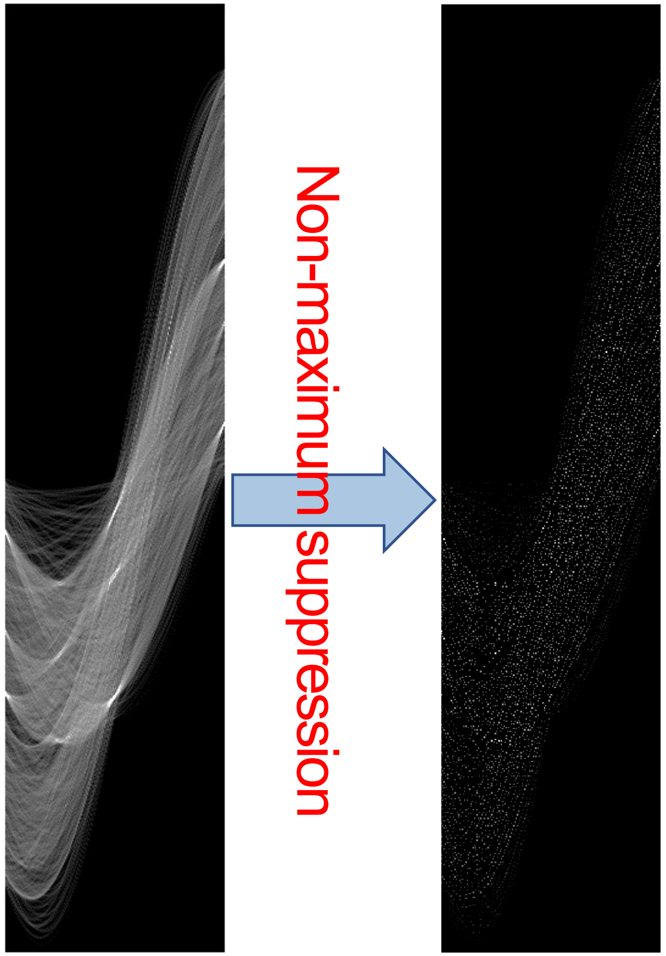


Algorithm of Hough Transform

- Step 3. Detect local maximum by non-maximum suppression

If $A(x, y)$ is less than at least one of its eight neighbors, let it be zero (suppression), otherwise keep it

30	30	70
30	90	30
20	30	30



Hough space images

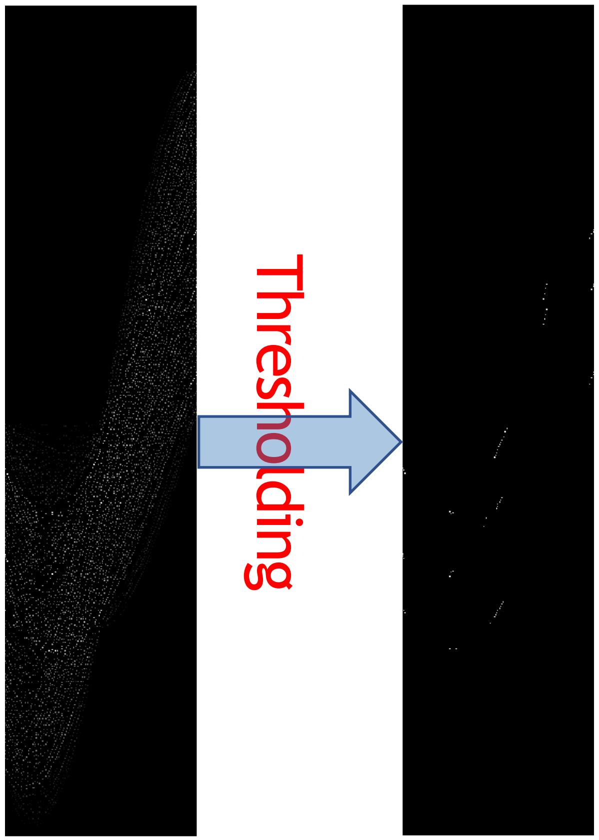
Algorithm of Hough Transform

Step 4. Do thresholding

$$
\begin{aligned}
&if\ A(i,j) \geq threshold \\
&\qquad\qquad A(i,j)=1 \\
&else \\
&\qquad\qquad A(i,j)=0 \\
&end
\end{aligned}
$$

The value of the threshold is determined according to the length (pixels) of lines wanted to be detected

e.g., suppose ρ is calculated in unit of 1 pixel, a line with a length of 10 pixels in image space corresponds to a maximum value with frequency of 10 in Hough space



Hough space images

Algorithm of Hough Transform

- Step 5. Perform inverse Hough transform using threshold values of (θ, ρ) and draw the lines for varying values of x and y

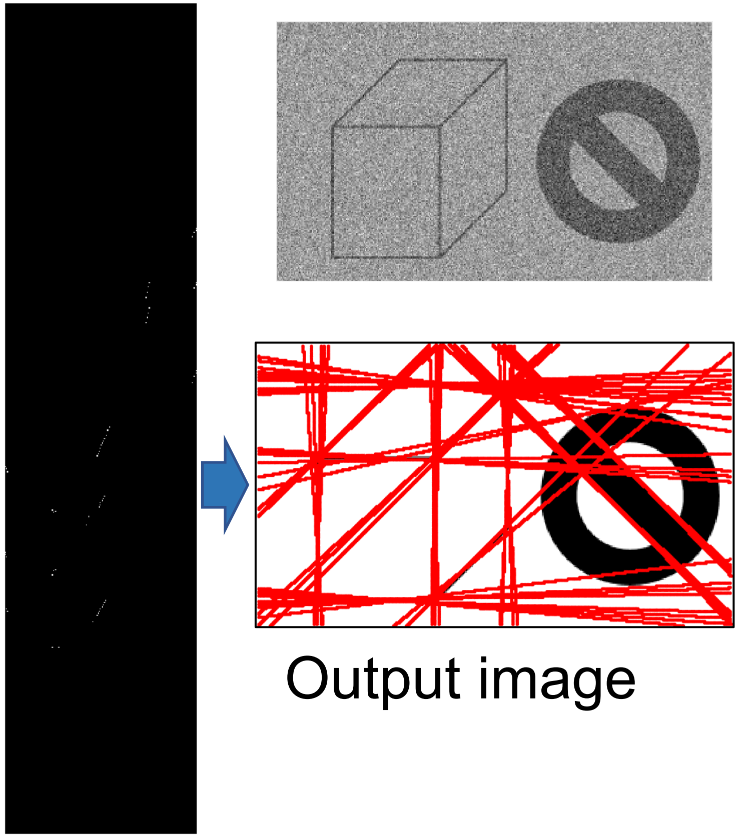
Hough transform:

$$x \cos\theta + y \sin\theta = \rho$$

Hough reverse transform:

$$y = -\frac{\cos\theta}{\sin\theta}x + \frac{\rho}{\sin\theta}$$
$$x = -\frac{\sin\theta}{\cos\theta}y + \frac{\rho}{\cos\theta}$$

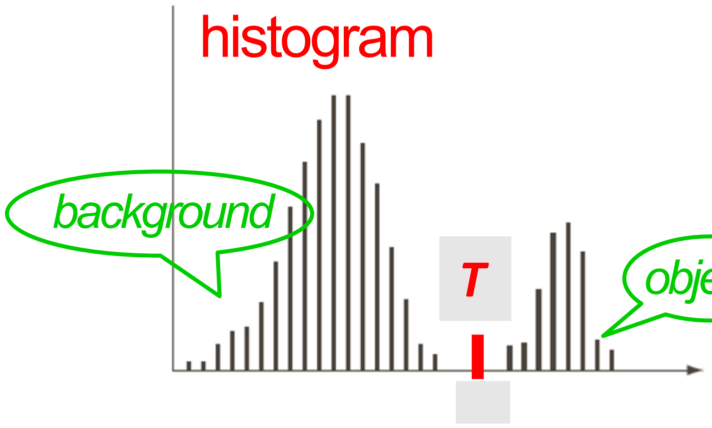
Sometimes we should choose to use these equation to avoid the problem of division by zero



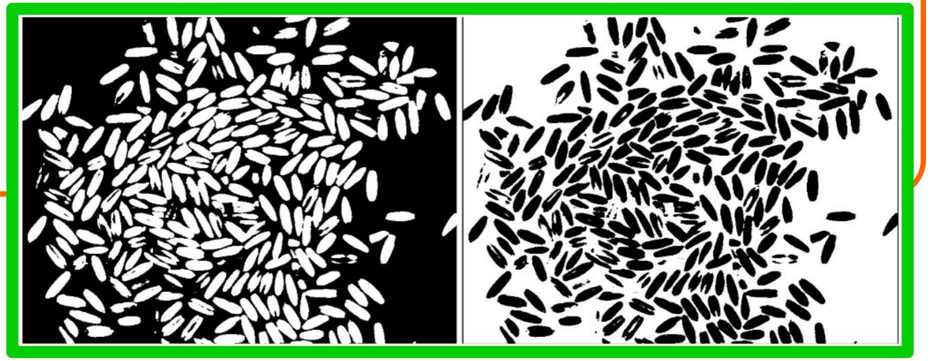
Output image

Thresholding

- We have used **thresholding** technique in many methods
- Thresholding enjoys a central position in image segmentation
- e.g., an image, $f(x,y)$, composed of light objects on a dark background $\rightarrow$ two modes histogram with a valley
- We can segment objects from the background by a threshold T placed in the valley



$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T & \text{(object point)} \\ 0 & \text{if } f(x,y) \leq T & \text{(background point)} \end{cases}$$



Noise Impact

Let's see how noise affects the histogram of an image:

(a) Image free of noise: the histogram consists of two 'spike' modes

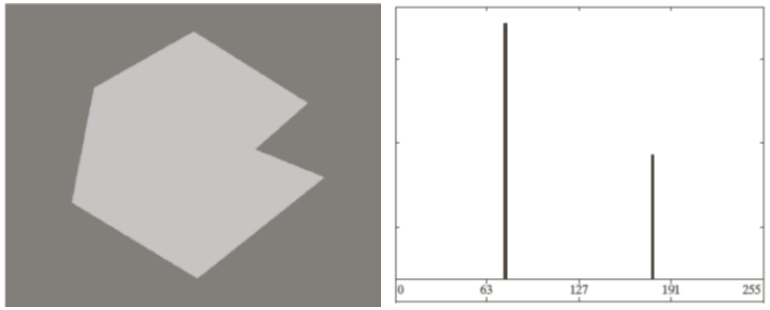
- Easy to segment by a threshold placed anywhere between the two modes

(b) Image with Gaussian noise (σ=10): the histogram modes are broader

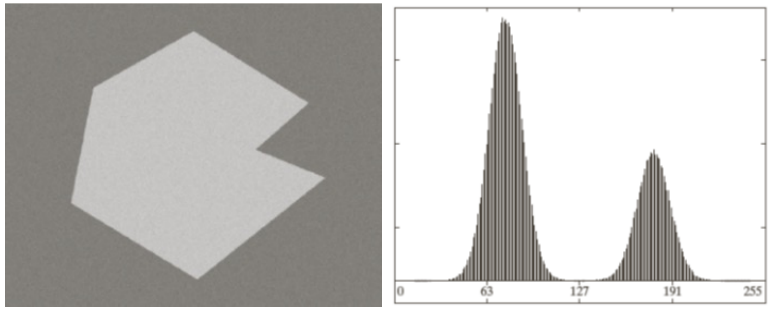
- Still easy to separate the modes

(c) Image with Gaussian noise (σ=50):

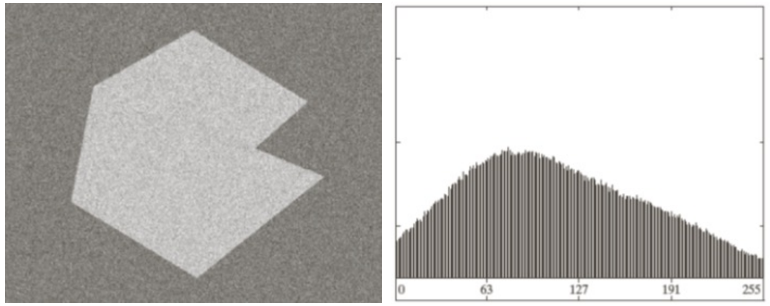
- We can not find a suitable threshold without additional processing



(a) Free of noise histogram



(b) With Gaussian noise (μ=0, σ=10) histogram



(c) With Gaussian noise (μ=0, σ=50) histogram

Basic Global Thresholding

- When the distributions of objects and background pixels are sufficiently distinct, it is possible to use a single threshold
- Thresholding for this situation is called basic global thresholding

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases}$$

- We need an algorithm for estimating automatically the threshold value T for each image

Basic Global Thresholding

Algorithm

- Select an initial estimate for the global threshold, T
- Segment the image by T to produce two groups:

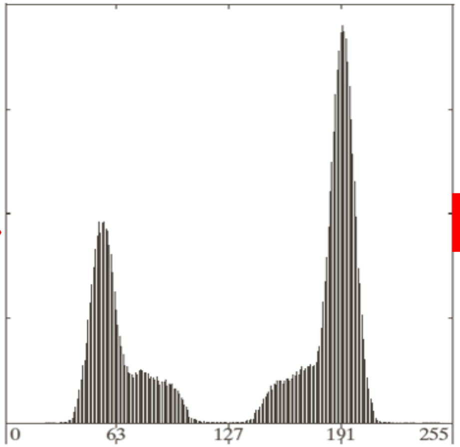
  $G1$ (pixels with intensity $> T$), and

  $G2$ (pixels with intensity $\leq T$)
- Compute the mean intensity values m_1 and m_2 for $G1$ and $G2$ respectively
- Compute a new threshold value: $T=(m_1+m_2)/2$
- Repeat steps 2-4 until the difference between values T in successive iterations is smaller than a predefined ΔT

Example

- An example: segmentation based on a threshold estimated using this algorithm

Noisy fingerprint



Histogram

Segmented result

- The original image is a noisy fingerprint
- The histogram shows a distinct valley
- Apply the algorithm:
  - Start with T=m (the average image intensity) and using ΔT=0
  - After three iterations, the algorithm resulted in T=125.4

Otsu’s Method

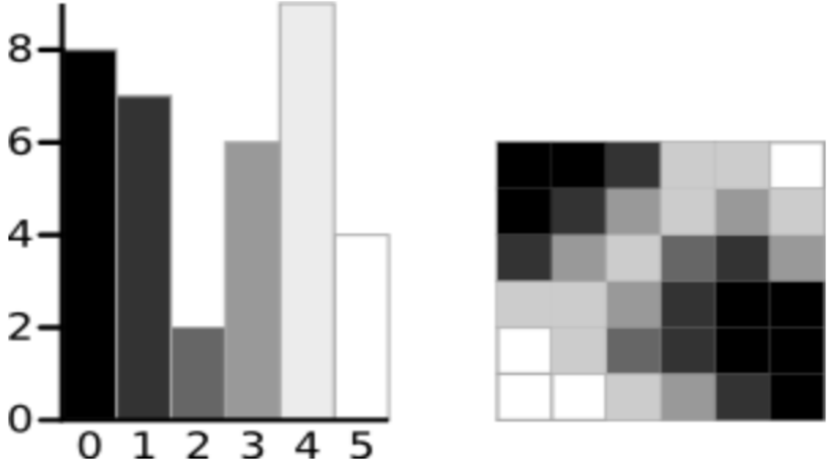
- In practice, it is very difficult to find an optimum global threshold
- Otsu’s method can automatically find an optimal threshold based on the observed distribution of pixel values
- This method maximizes the between-class variance, a well-known measure used in statistical discriminant analysis
- e.g., consider background and objects as two classes, Otsu’s method selects an optimal threshold that maximizes the variance between background and objects

Otsu’s Method

- Otsu’s method is a variance-based technique to find the threshold value where the weighted variance between the foreground and background pixels is the least
- The key idea here is to iterate through all the possible values of threshold and measure the spread of background and foreground pixels

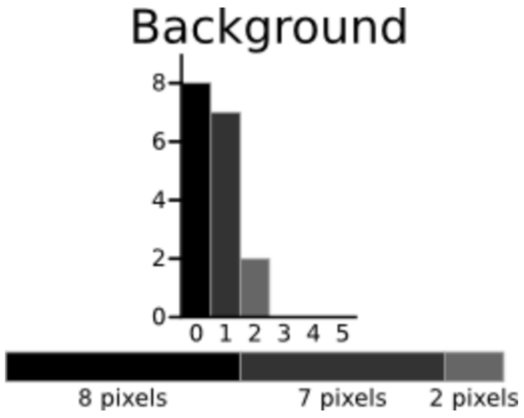
Otsu’s Method

- Suppose we have a 6x6 image and histogram for the image



A 6-level greyscale image and its histogram

Otsu’s Method



Weight w_b = (8 + 7+ 2) / 36 = 0.4722

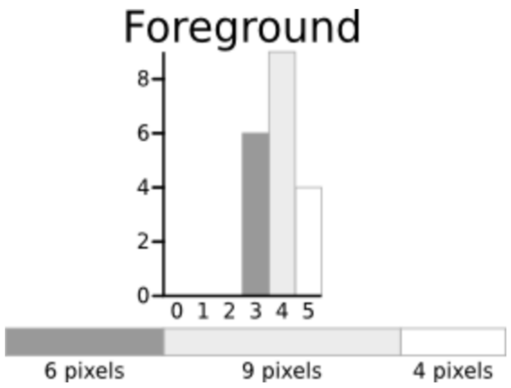
Mean μ_b = (0 x 8) + (1 x 7) + (2 x 2) / 17 = 0.6471

Variance σ_b^2 = $((0 - 0.6472)^2$ x 8) + $((1 - 0.6472)^2$ x 7) + $((2 - 0.6472)^2$ x 2) / 17

= (0.4187 x 8) + (0.1246 x 7) + (1.8304 x 2) / 17

= 0.4637

Otsu’s Method



Weight w_f = (6 + 9 + 4) / 36 = 0.5278

Mean μ_f = (3 x 6) + (4 x 9) + (5 x 4) / 19 = 3.8947

Variance σ_f^2 = $((3 - 3.8947)^2$ x 6) + $((4 - 3.8947)^2$ x 9) + $((5 - 3.8947)^2$ x 4) / 19

= (4.8033 x 6) + (0.0997 x 9) + (4.8864 x 4) / 19

= 0.5152

Otsu’s Method

- The next step is to calculate the 'Within-Class Variance’
- This is simply the sum of the two variances multiplied by their associated weights

$$\text{Within Class Variance } \sigma_w^2 = W_b\sigma_b^2 + W_f\sigma_f^2$$

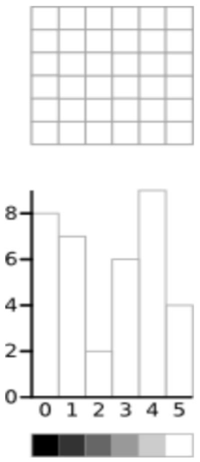
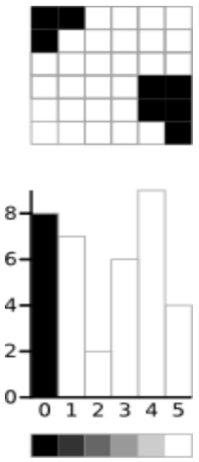
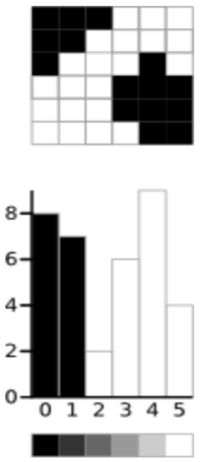
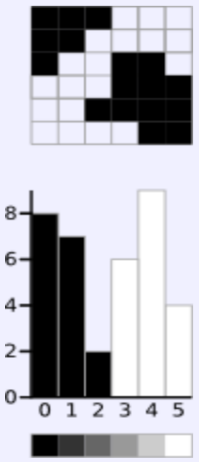
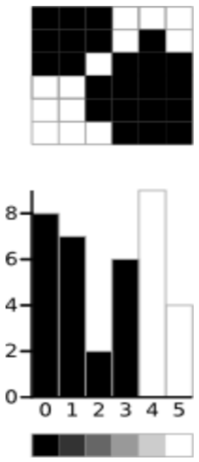
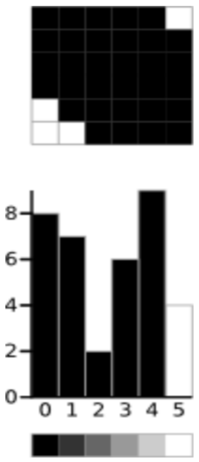
= (0.4722 x 0.4637) + (0.5278 x 0.5152)

= 0.4909

This final value is the 'sum of weighted variances' for the threshold value 3

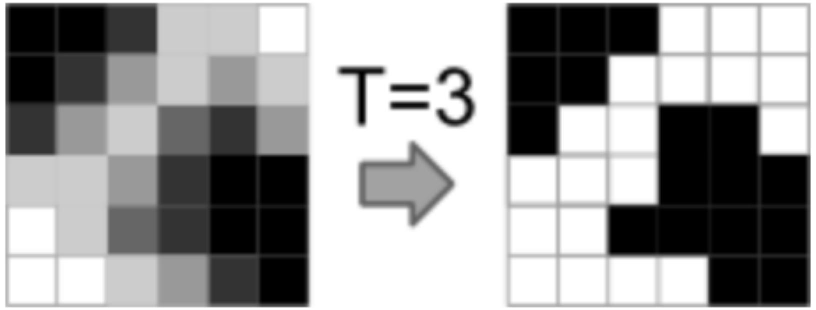
Otsu's Method

- This final value is the 'sum of weighted variances' for the threshold value 3
- This same calculation needs to be performed for all the possible threshold values 0 to 5
- The table below shows the results for these calculations

Threshold	T=0	T=1	T=2	T=3	T=4	T=5
						
Weight, Background	$W_b = 0$	$W_b = 0.222$	$W_b = 0.4167$	$W_b = 0.4722$	$W_b = 0.6389$	$W_b = 0.8889$
Mean, Background	$M_b = 0$	$M_b = 0$	$M_b = 0.4667$	$M_b = 0.6471$	$M_b = 1.2609$	$M_b = 2.0313$
Variance, Background	$\sigma^2_b = 0$	$\sigma^2_b = 0$	$\sigma^2_b = 0.2489$	$\sigma^2_b = 0.4637$	$\sigma^2_b = 1.4102$	$\sigma^2_b = 2.5303$
Weight, Foreground	$W_f = 1$	$W_f = 0.7778$	$W_f = 0.5833$	$W_f = 0.5278$	$W_f = 0.3611$	$W_f = 0.1111$
Mean, Foreground	$M_f = 2.3611$	$M_f = 3.0357$	$M_f = 3.7143$	$M_f = 3.8947$	$M_f = 4.3077$	$M_f = 5.000$
Variance, Foreground	$\sigma^2_f = 3.1196$	$\sigma^2_f = 1.9639$	$\sigma^2_f = 0.7755$	$\sigma^2_f = 0.5152$	$\sigma^2_f = 0.2130$	$\sigma^2_f = 0$
Within Class Variance	$\sigma^2_W = 3.1196$	$\sigma^2_W = 1.5268$	$\sigma^2_W = 0.5561$	$\boxed{\boldsymbol{\sigma^2_W = 0.4909}}$	$\sigma^2_W = 0.9779$	$\sigma^2_W = 2.2491$

Otsu's Method

- For the threshold equal to 3, has the lowest sum of weighted variances
- Therefore, this is the final selected threshold
- All pixels with a level less than 3 are background, all those with a level equal to or greater than 3 are foreground



Result of Otsu's Method

Otsu's Method

- You can also calculate 'between class variance'
- It can also be used for finding the best threshold and is a much better approach to use

$$Within\ Class\ Variance\ \sigma_W^2 = W_b\sigma_b^2 + W_f\sigma_f^2$$

$$Between\ Class\ Variance\ \sigma_B^2 = \sigma^2 - \sigma_W^2$$

$$= W_b(\mu_b - \mu)^2 + W_f(\mu_f - \mu)^2$$

(where $\mu = W_b\mu_b + W_f\mu_f$)

$$= W_b W_f(\mu_b - \mu_f)^2$$

Threshold	T=0	T=1	T=2	T=3	T=4	T=5
Within Class Variance	$\sigma^2_W = 3.1196$	$\sigma^2_W = 1.5268$	$\sigma^2_W = 0.5561$	$\boldsymbol{\sigma^2_W = 0.4909}$	$\sigma^2_W = 0.9779$	$\sigma^2_W = 2.2491$
Between Class Variance	$\sigma^2_B = 0$	$\sigma^2_B = 1.5928$	$\sigma^2_B = 2.5635$	$\boldsymbol{\sigma^2_B = 2.6287}$	$\sigma^2_B = 2.1417$	$\sigma^2_B = 0.8705$

Otsu's Method

Steps:

- Compute the normalized histogram of the input image, denotes the components of the histograms by $p_i, \mathrm{i} = 0,1,2, \dots, L-1$
- Compute the cumulative sums, $P_1(k)$, for *k* = 0,1,2,…,*L-1*
- Compute the cumulative means, *m*(*k*), for *k* = 0,1,2,…,*L-1*
- Compute the global mean, m_G
- Compute the between-class variance term, $\sigma_B^2(k)$, for *k* = 0,1,2,…,*L-1*
- Obtain the Otsu threshold, *k**, as the value of *k* for which $\sigma_B^2(k)$ is maximum
- Compute the global variance σ_G^2 and obtain the separability measure $\eta *$

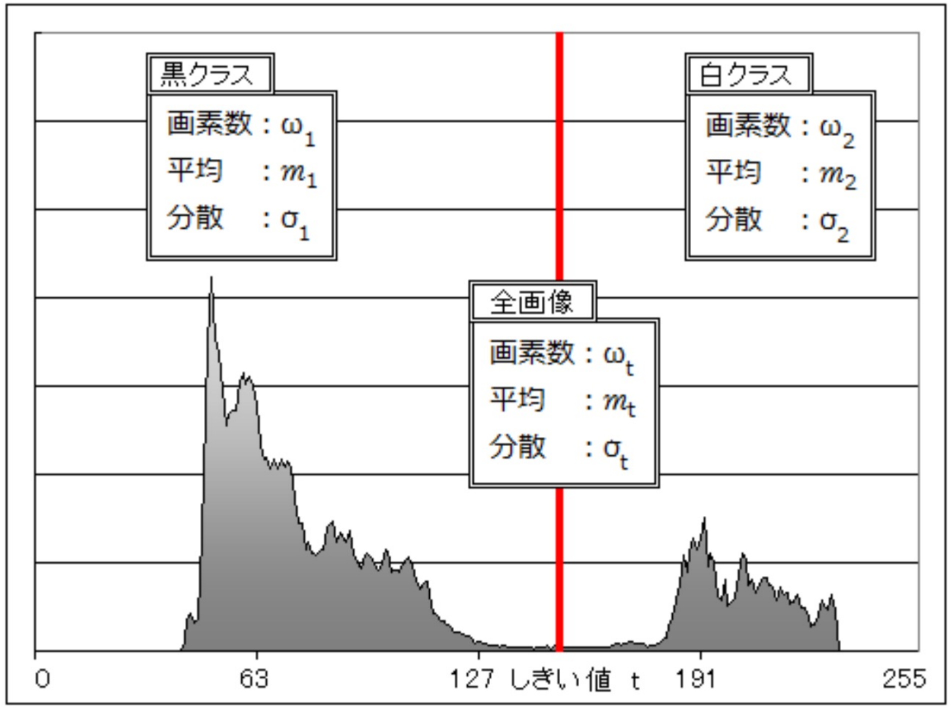
Otsu's Method

- Otsu's method is based entirely on computations performed on the histograms of an image
- It is a good way for searching the threshold at which the value of a **separation metric** is maximized
- Suppose the threshold is set to level k

Obtain two classes:

The black class: pixels' intensities are all less than or equal to k

The white class :pixels' intensities are all larger than k

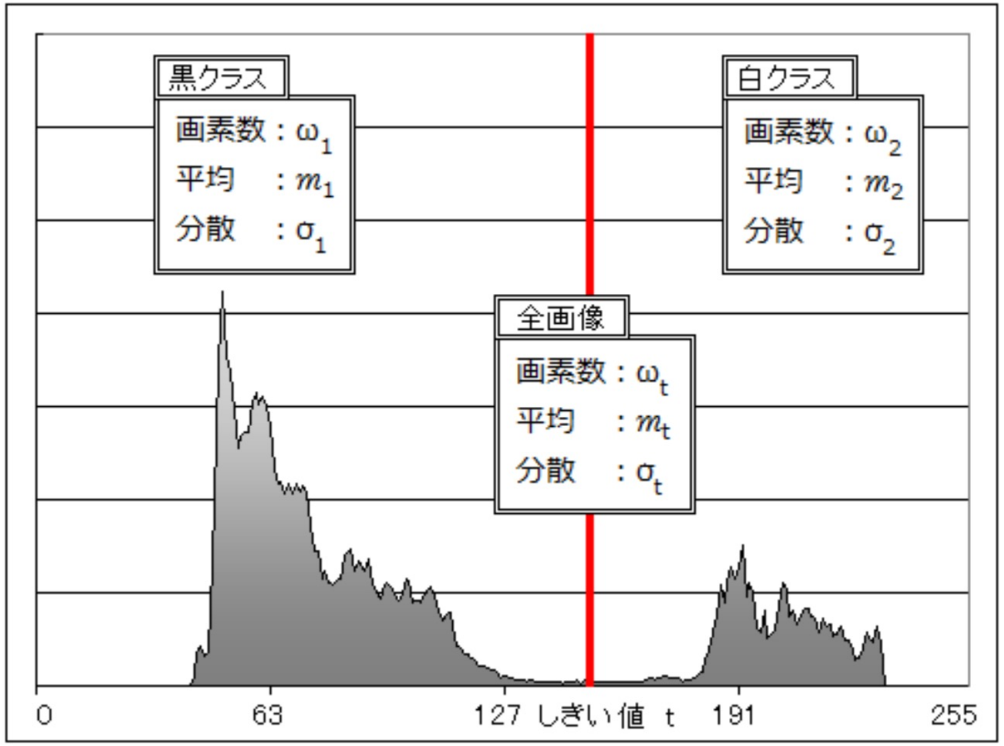


black class white class

Otsu's Method

Some definitions:

- ω_1, ω_2: the numbers of pixels assigned to the black and white classes
- m_1, m_2: the mean intensity values of pixels assigned to the black and white classes
- σ_1^2, σ_2^2: the variances of the black and white classes



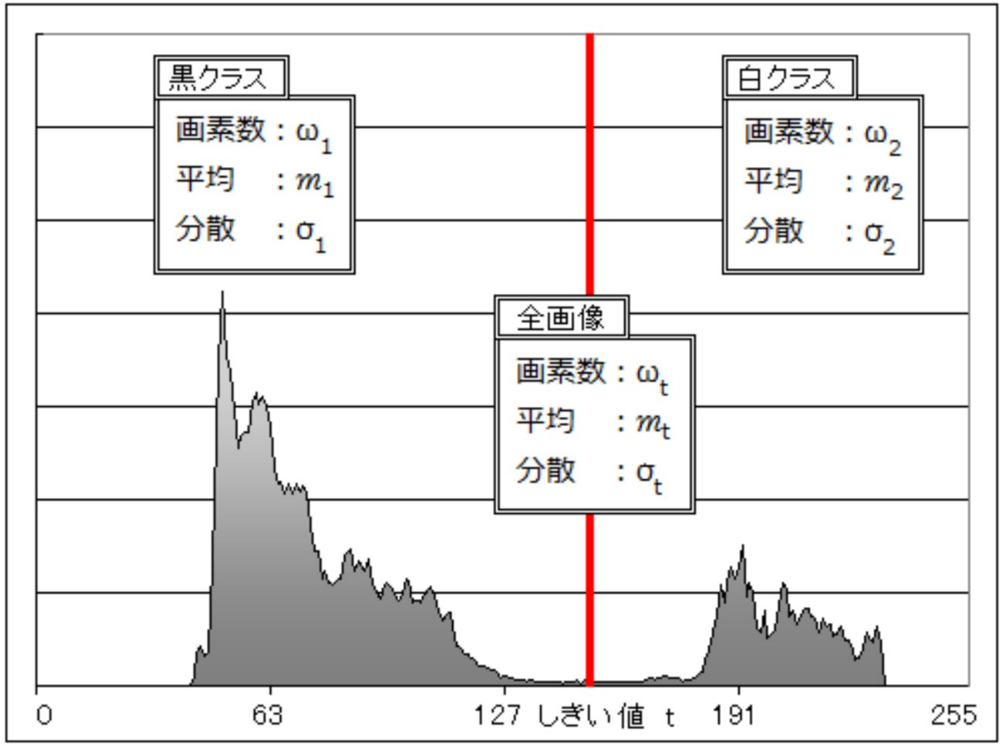
black class

white class

Otsu's Method

Definitions:

- ω_t : total number of pixels in the image
- m_t : mean intensity value of pixels in the image
- σ_t^2 : the variance of the image



black class white class

Separation Metrics

- To evaluate the "separation metric" of the threshold at level k, we use the ratio of between-class variance to within-class variance:

$$\eta = \frac{\sigma_B^2}{\sigma_W^2} = \frac{\sigma_B^2}{\sigma_t^2 - \sigma_B^2}$$

$$\boxed{\sigma_t^2 = \sigma_B^2 + \sigma_W^2}$$

- Because σ_t^2 is independent of the threshold value, maximizing η is equivalent to maximizing σ_B^2

$$\sigma_B^2 = \frac{\omega_1(m_1 - m_t)^2 + \omega_2(m_2 - m_t)^2}{\omega_1 + \omega_2} = \frac{\omega_1\omega_2(m_1 - m_2)^2}{(\omega_1 + \omega_2)^2}$$

- Eventually, we just need to maximize $\omega_1\omega_2(m_1 - m_2)^2$

Otsu's Method

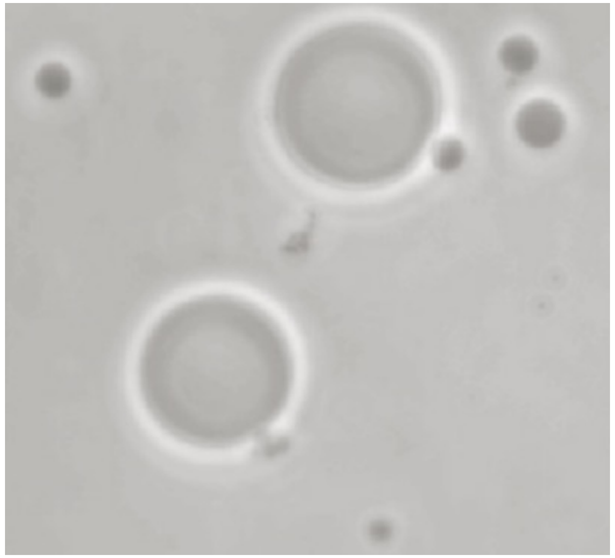
- The optimal threshold is the value, k^*, that maximizes:

$$\sigma_B^2(k*) = \max_{0 \le k \le L-1} \sigma_B^2(k)$$

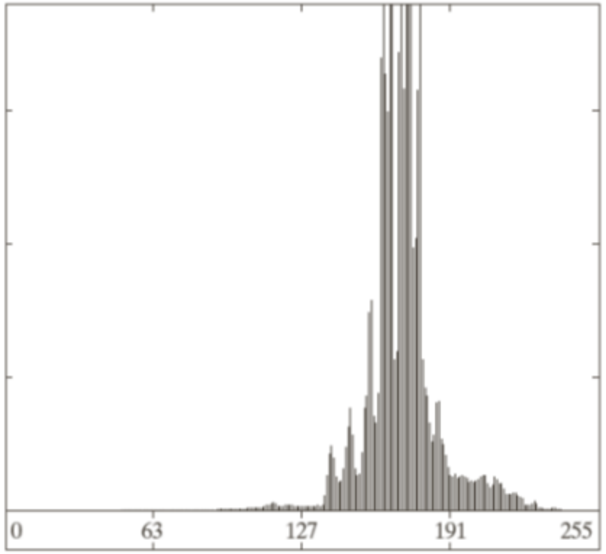
- We simply evaluate equation for all integer values of k and select the k that yields the maximum $\sigma_B^2(k)$
- All the evaluation on k can be done by using the only image histogram
- Otsu's method has the important property that it is based on computations performed on the histogram of an image

Example

- The original image: an optical microscope image of polymersome cells
- We want to segment the molecules from the background
- The histogram has no distinct valleys and the intensity difference between the background and objects is very small



Original image



histogram

Example

- The basic global thresholding algorithm failed to achieve the desired segmentation
  - The threshold value computed was 169
- The result obtained by Ostu's method is improved
  - The optimal threshold was 181 and separability measure was 0.467

The result using the basic global thresholding algorithm k=169

The result using Otsu's method k=181, η=0.467

Thank you for your attention