

# 19 - Pendeteksian Tepi (Bagian 2)

IF4073 Pemrosesan Citra Digital

Oleh: Dr. Ir. Rinaldi, M.T



Program Studi Teknik Informatika  
Sekolah Teknik Elektro dan Informatika  
Institut Teknologi Bandung  
2025

# Operator gradien lainnya

- Operator Sobel
- Operator Roberts
- Operator Prewitt
- Operator Canny

# Operator Sobel

- Tinjau pengaturan *pixel* di sekitar *pixel* (x,y):

$$\begin{bmatrix} a_0 & a_1 & a_2 \\ a_7 & (x,y) & a_3 \\ a_6 & a_5 & a_4 \end{bmatrix}$$

- Operator Sobel adalah magnitudo dari gradien yang dihitung dengan rumus

$$M = \sqrt{s_x^2 + s_y^2}$$

yang dalam hal ini, turunan parsial dihitung dengan

$$s_x = (a_2 + ca_3 + a_4) - (a_0 + ca_7 + a_6)$$

$$s_y = (a_0 + ca_1 + a_{22}) - (a_6 + ca_5 + a_4)$$

- Dengan konstanta  $c = 2$ , maka

$$s_x = (a_2 + 2a_3 + a_4) - (a_0 + 2a_7 + a_6)$$

$$s_y = (a_0 + 2a_1 + a_2) - (a_6 + 2a_5 + a_4)$$

$$\begin{bmatrix} a_0 & a_1 & a_2 \\ a_7 & (x, y) & a_3 \\ a_6 & a_5 & a_4 \end{bmatrix}$$

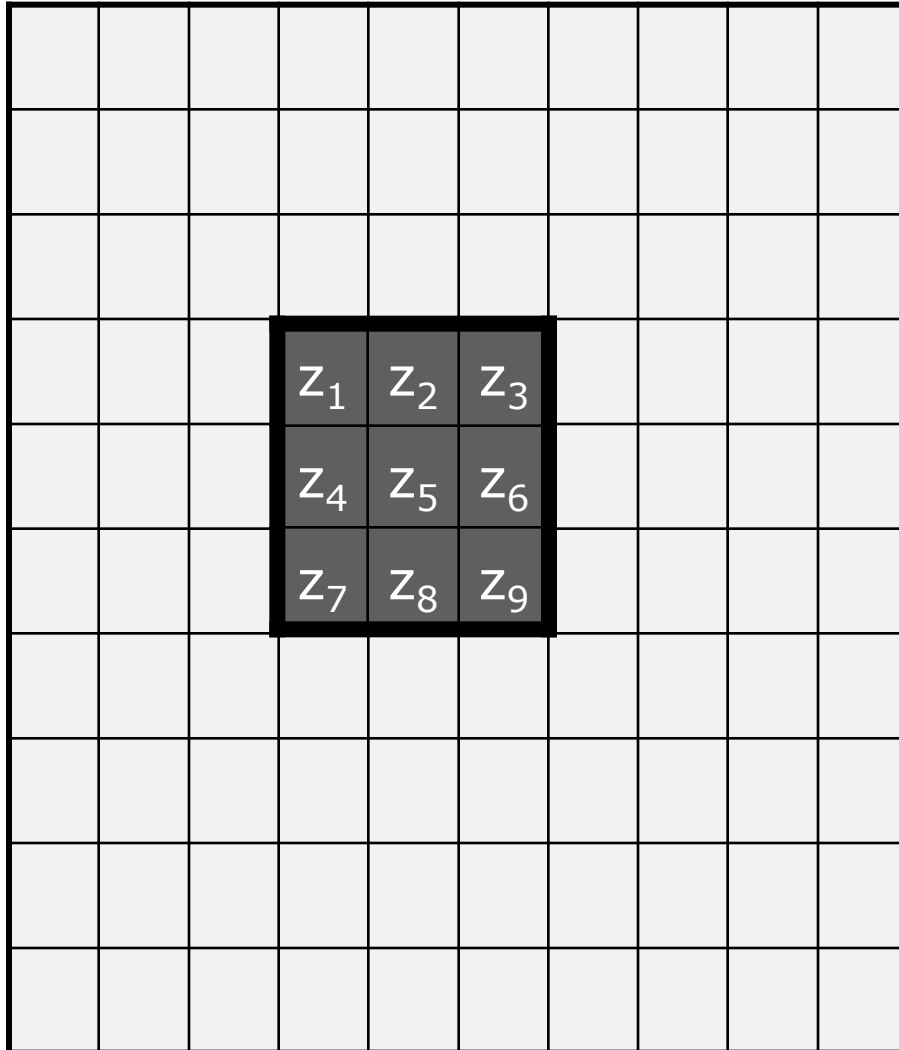
- Dalam bentuk *mask*,  $s_x$  dan  $s_y$  dapat dinyatakan sebagai

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{dan} \quad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Arah tepi dihitung dengan persamaan

$$\alpha(x, y) = \tan^{-1} \left( \frac{S_y}{S_x} \right)$$

**Catatan:** Beberapa literatur menggunakan penomoran pixel sebagai berikut sehingga matriks *mask* Sobel berbeda susunan nilainya dengan slide sebelumnya:



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

$$G_x \approx (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

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

$$G_y \approx (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

- Contoh:

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 4 | 2 | 5 | 1 |
| 2 | 1 | 6 | 4 | 2 |
| 3 | 5 | 7 | 1 | 3 |
| 4 | 2 | 5 | 7 | 1 |
| 2 | 5 | 1 | 3 | 2 |

(i) citra semula

|   |    |   |   |   |
|---|----|---|---|---|
| * | *  | * | * | * |
| * | 18 |   |   |   |
|   |    |   |   |   |
|   |    |   |   |   |
|   |    |   |   |   |

(ii) hasil konvolusi

Nilai 18 pada citra hasil konvolusi diperoleh dengan perhitungan berikut:

$$S_x = (3)(-1) + (2)(-2) + (3)(-1) + (2)(1) + (6)(2) + (7)(1) = 11$$

$$S_y = (3)(1) + (4)(2) + (2)(1) + (3)(-1) + (5)(-2) + (7)(-1) = -7$$

$$M = \sqrt{s_x^2 + s_y^2} = \sqrt{11^2 + (-7)^2} \cong |S_x| + |S_y| = |11| + |-7| = 18$$

Pada contoh ini, nilai  $M = \sqrt{s_x^2 + s_y^2}$  dihampiri dengan menghitung

$$M \cong |S_x| + |S_y|.$$



- Di bawah ini contoh lain pendeteksian tepi dengan operator Sobel, dimana hasil konvolusi diambangkan (*thresholding*) dengan  $T = 12$ .

Citra:

|   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 3 | 3 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 4 | 3 |
| 0 | 0 | 2 | 0 | 2 | 4 | 3 | 3 | 2 | 3 |
| 0 | 0 | 1 | 3 | 3 | 4 | 3 | 3 | 3 | 3 |
| 0 | 1 | 0 | 4 | 3 | 3 | 2 | 4 | 3 | 2 |
| 0 | 0 | 1 | 2 | 3 | 3 | 4 | 4 | 4 | 3 |

$|\text{gradien} - x| + |\text{gradien} - y| :$

|   |   |    |    |    |    |    |    |   |   |
|---|---|----|----|----|----|----|----|---|---|
| * | * | *  | *  | *  | *  | *  | *  | * | * |
| * | 4 | 6  | 4  | 10 | 14 | 12 | 14 | 4 | * |
| * | 6 | 8  | 10 | 20 | 16 | 12 | 6  | 0 | * |
| * | 4 | 10 | 14 | 10 | 2  | 4  | 2  | 4 | * |
| * | 2 | 12 | 12 | 2  | 2  | 4  | 6  | 8 | * |
| * | * | *  | *  | *  | *  | *  | *  | * | * |

Hasil pengambangan dengan  $T = 12$ :

|   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|
| * | * | * | * | * | * | * | * | * | * |
| * | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | * |
| * | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | * |
| * | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | * |
| * | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | * |
| * | * | * | * | * | * | * | * | * | * |

```
%Sobel
I = imread('bird.bmp');
Sx = [-1 0 1; -2 0 2; -1 0 1];
Sy = [1 2 1; 0 0 0; -1 -2 -1];
Jx = conv2(double(I), double(Sx), 'same');
Jy = conv2(double(I), double(Sy), 'same');
Jedge = sqrt(Jx.^2 + Jy.^2);
imshow(I);
figure, imshow(uint8(Jedge));
```



Citra masukan



Hasil operator Sobel

# Operator Prewitt

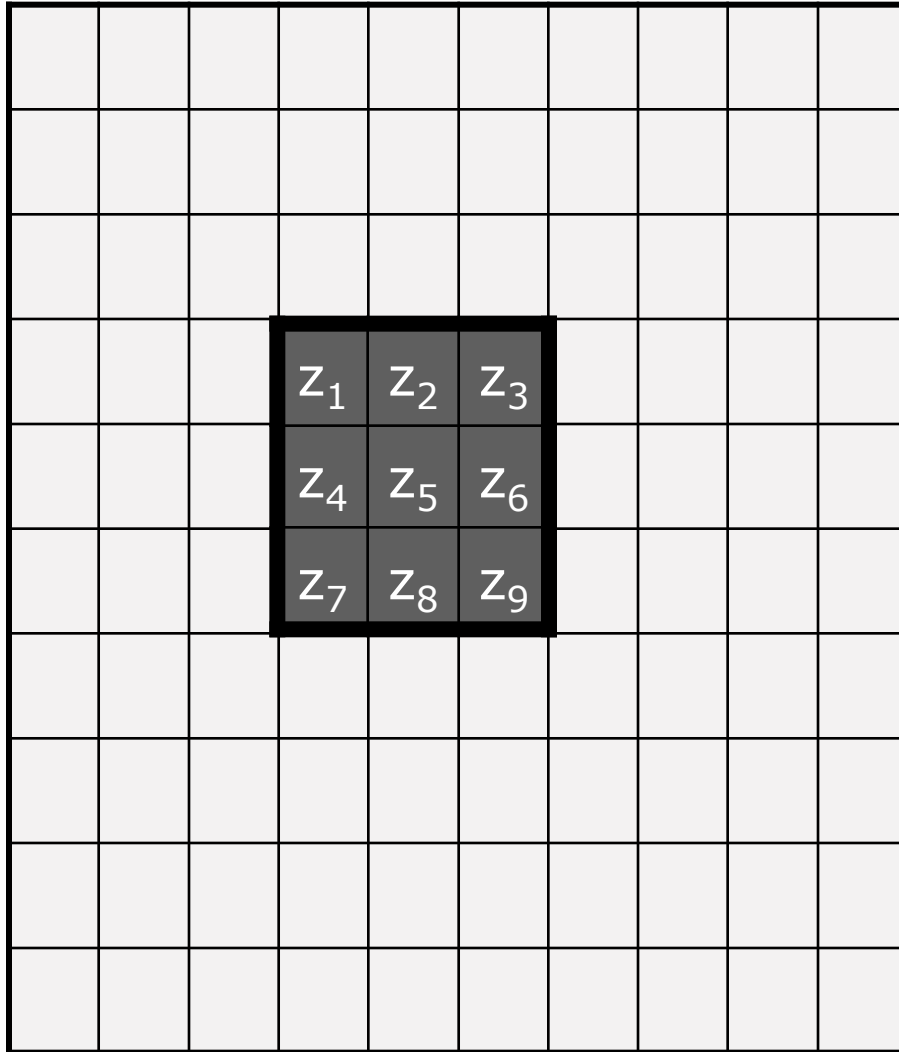
- Persamaan gradien pada operator Prewitt sama seperti operator Sobel, tetapi menggunakan nilai  $c = 1$ :

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{dan} \quad P_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

- Kekuatan tepi dan arah tepi dihitung dengan rumus:

$$G(f(x,y)) = \sqrt{P_x^2 + P_y^2} \quad \alpha(x,y) = \tan^{-1} \left( \frac{P_y}{P_x} \right)$$

**Catatan:** Beberapa literatur menggunakan penomoran pixel sebagai berikut sehingga matriks *mask* Prewitt berbeda susunan nilainya dengan slide sebelumnya:



|    |    |    |
|----|----|----|
| -1 | -1 | -1 |
| 0  | 0  | 0  |
| 1  | 1  | 1  |

$$G_x \approx (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

$$G_y \approx (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

```
I = imread('bird.bmp');  
Px = [-1 0 1; -1 0 1; -1 0 1];  
Py = [-1 -1 -1; 0 0 0; 1 1 1];  
Jx = conv2(double(I), double(Px), 'same');  
Jy = conv2(double(I), double(Py), 'same');  
Jedge = sqrt(Jx.^2 + Jy.^2);  
imshow(I);  
figure, imshow(uint8(Jedge));
```



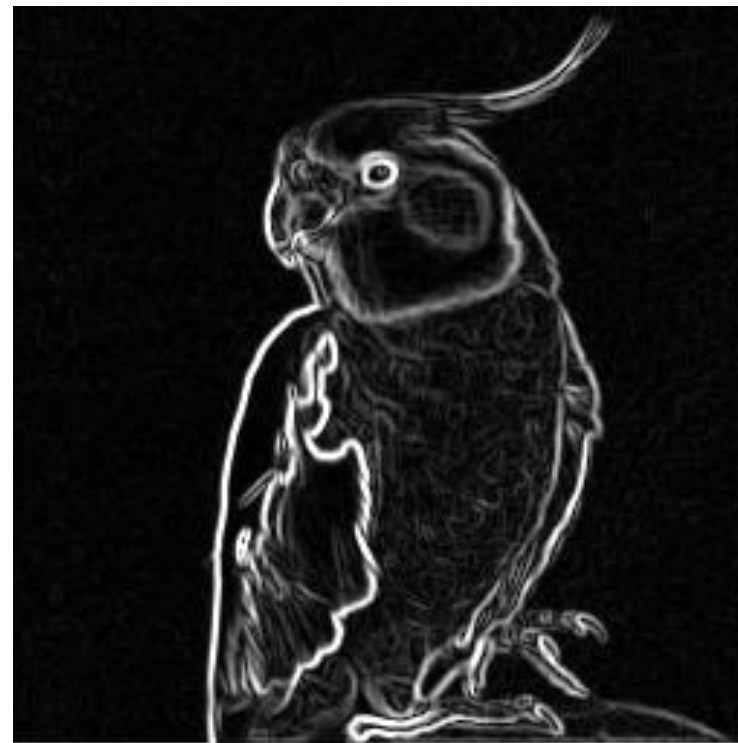
Citra masukan



Hasil operator Prewitt



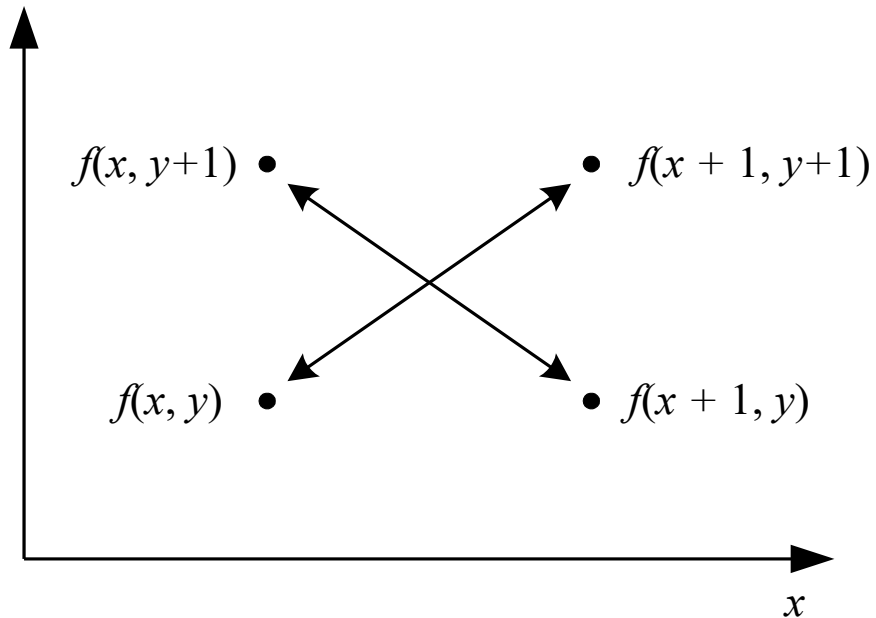
Hasil operator Sobel



Hasil operator Prewitt

# Operator Roberts

- Operator Roberts sering disebut juga operator silang



Arah tepi dihitung dengan rumus:

$$\alpha(x, y) = \frac{\pi}{4} + \tan^{-1}\left(\frac{R_-}{R_+}\right)$$

Gradien Roberts dalam arah-x dan arah-y dihitung dengan rumus:

$$R_+(x, y) = f(x+1, y+1) - f(x, y)$$

$$R_-(x, y) = f(x, y+1) - f(x+1, y)$$

Dalam bentuk *mask* konvolusi:

$$R_+ = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad \text{dan} \quad R_- = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Kekuatan tepi dihitung dengan rumus  $G[f(x, y)] = |R_+| + |R_-|$

- Contoh berikut ini memperlihatkan pendeteksian tepi dengan operator Roberts:

$$\begin{bmatrix} 3 & 4 & 2 & 5 & 1 \\ 2 & 1 & 6 & 4 & 2 \\ 3 & 5 & 7 & 1 & 3 \\ 4 & 2 & 5 & 7 & 1 \\ 2 & 5 & 1 & 3 & 2 \end{bmatrix}$$

(i) citra semula

$$\begin{bmatrix} 4 & 3 & 3 & 6 & * \\ 5 & 7 & 8 & 2 & * \\ 2 & 5 & 4 & 4 & * \\ 1 & 1 & 8 & 7 & * \\ * & * & * & * & * \end{bmatrix}$$

(ii) hasil konvolusi

Nilai 4 pada pojok kiri atas pada citra hasil konvolusi diperoleh dengan perhitungan sebagai berikut:

$$f'[0,0] = |3 - 1| + |4 - 2| = 4$$

```
I = imread('bird.bmp');  
Rx = [1 0; 0 -1];  
Ry = [0 1; -1 0];  
Jx = conv2(double(I), double(Rx), 'same');  
Jy = conv2(double(I), double(Ry), 'same');  
Jedge = sqrt(Jx.^2 + Jy.^2);  
imshow(I);  
figure, imshow(uint8(Jedge));
```



Citra masukan



Hasil operator Roberts

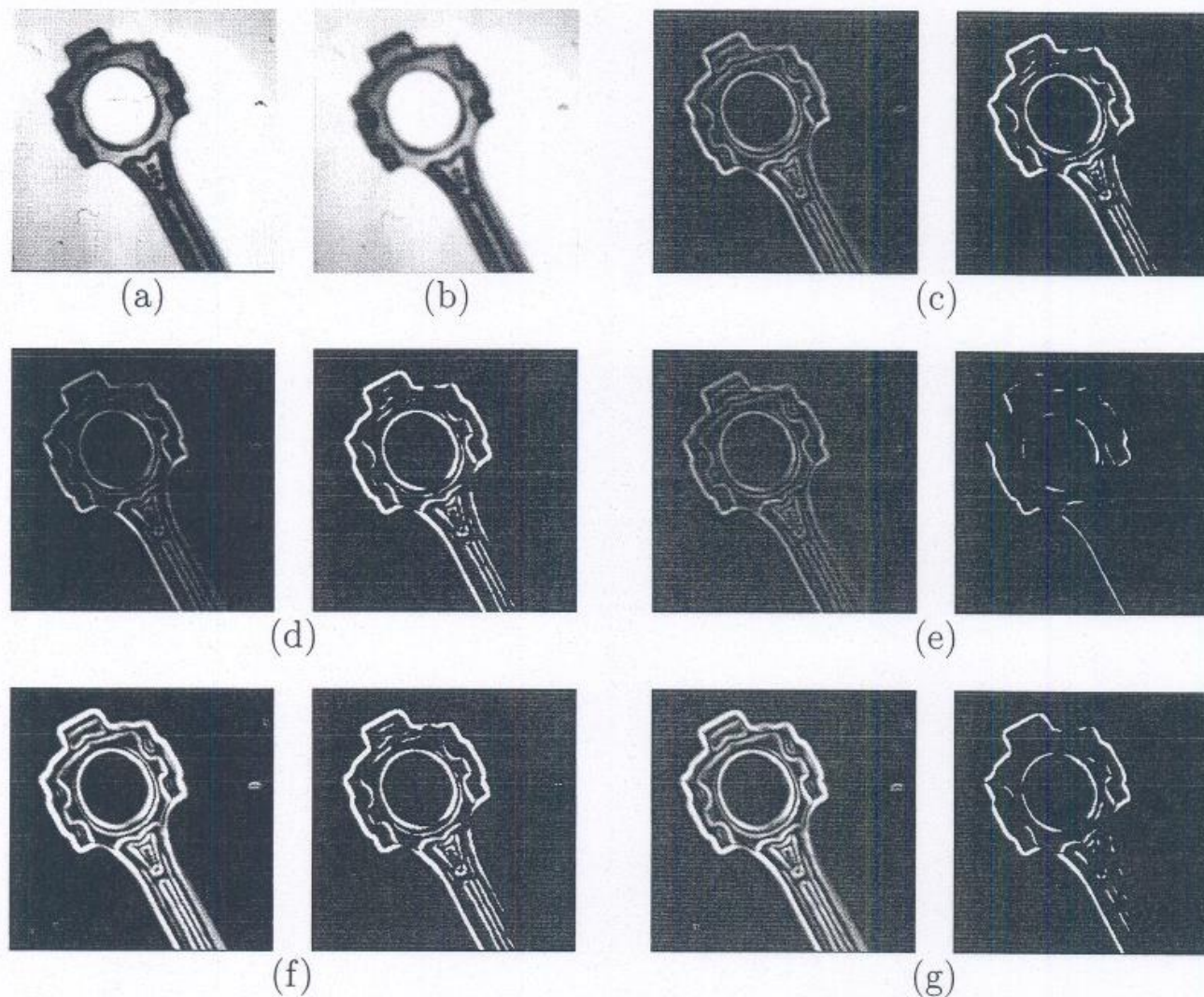


Figure 5.4: A comparison of various edge detectors. (a) Original image. (b) Filtered image. (c) Simple gradient using  $1 \times 2$  and  $2 \times 1$  masks,  $T = 32$ . (d) Gradient using  $2 \times 2$  masks,  $T = 64$ . (e) Roberts cross operator,  $T = 64$ . (f) Sobel operator,  $T = 225$ . (g) Prewitt operator,  $T = 225$ .

# Operator Canny

- Operator deteksi tepi yang terkenal karena dapat menghasilkan tepi dengan ketebalan 1 *pixel*



Langkah-langkah operator Canny:

1. Haluskan citra / menggunakan penapis Gaussian  $G$  (dengan standard deviasi  $\sigma$  yang dispesifikasikan):  $G * I$
2. Hitung gradien dan arah gradien setiap pixel dengan salah satu dari tiga operator sebelumnya (Sobel, Prewitt, Roberts)
3. Jika nilai mutlak (magnitude) gradien suatu pixel melebihi nilai ambang  $T$ , maka pixel tersebut termasuk pixel tepi.

- Operator Canny menggunakan dua nilai ambang,  $T1$  dan  $T2$  ( $T1 < T2$ ), sehingga memungkinkan deteksi dua jenis tepi: tepi kuat (*strong edges*) dan tepi lemah (*weak edges*).
- Jika magnitudo pixel di dalam citra gradien (hasil Langkah 2) melebihi nilai ambang  $T2$ , maka pixel tersebut bersesuaian dengan tepi kuat.
- Setiap pixel yang terhubung ke tepi yang kuat dan memiliki magnitudo lebih besar dari nilai ambang  $T1$ , maka pixel tersebut bersesuaian dengan tepi lemah.
- Selanjutnya dilakukan penautan tepi (*edge linking*) dengan menggabungkan tepi-tepi lemah yang terhubung dalam 8-arah dengan tepi kuat.



original image (Lena)



magnitude of the gradient with Sobel



Hasil deteksi tepi dengan operator Canny



original



Canny with  $\sigma = 1$



Canny with  $\sigma = 2$

- The choice of  $\sigma$  (in Gaussian filter) depends on desired behavior
  - large  $\sigma$  detects large scale edges
  - small  $\sigma$  detects fine features

Original  
image



Strong +  
connected  
weak edges



Strong  
edges  
only



Weak  
edges



courtesy of G. Loy

# Deteksi Tepi dengan Menggunakan Matlab

- Di dalam Matlab, selain menggunakan fungsi `conv2` untuk melakukan konvolusi dengan filter Sobel, Prewitt, Roberts, dan Canny, juga terdapat fungsi `edge` untuk mendeteksi tepi secara langsung dengan pilihan berbagai metode.

`BW = edge(I)` *returns a binary image BW containing 1s where the function finds edges in the grayscale or binary image I and 0s elsewhere. By default, edge uses the Sobel edge detection method.*

`BW = edge(I, method)` *detects edges in image I using the edge-detection algorithm specified by method. Pilihan method: Sobel (default), Prewitt, Roberts, Canny, log*

`BW = edge(I, method, threshold)` *returns all edges that are stronger than threshold.*

## Sobel



Original image



```
I = imread('boat.bmp');  
imshow(I)  
BW = edge(I, 'Sobel');  
figure, imshow(BW)
```

## Prewitt



Original image



```
I = imread('boat.bmp');  
imshow(I)  
BW = edge(I, 'Prewitt');  
figure, imshow(BW)
```

## Roberts



Original image



```
I = imread('boat.bmp');  
imshow(I)  
BW = edge(I, 'Roberts');  
figure, imshow(BW)
```

## log (Laplacian of Gaussian)



Original image



```
I = imread('boat.bmp');  
imshow(I)  
BW = edge(I, 'log');  
figure, imshow(BW)
```

## Canny

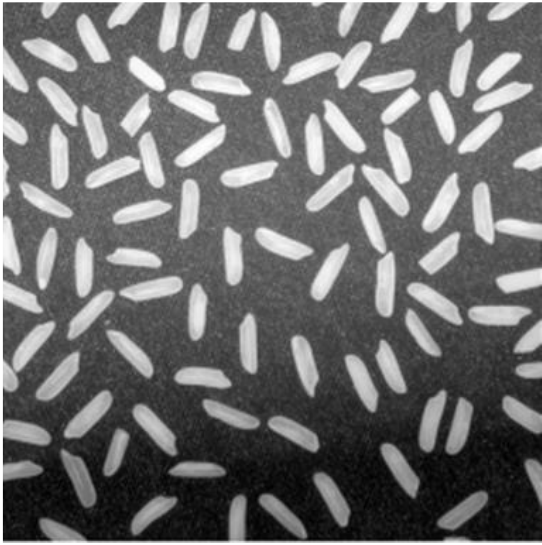


Original image

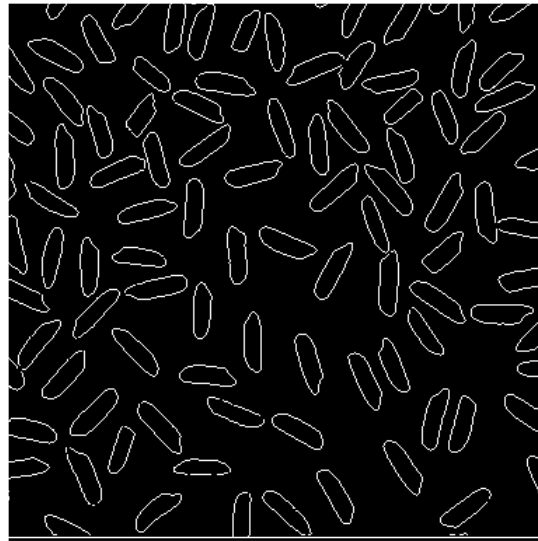
`BW = edge(I, 'Canny', threshold, sigma)` specify sigma, the standard deviation of the Gaussian filter. The default sigma is  $\sqrt{2}$ . `edge` chooses the size of the filter automatically, based on sigma.



```
I = imread('boat.bmp');  
imshow(I)  
BW = edge(I, 'Canny');  
figure, imshow(BW)
```



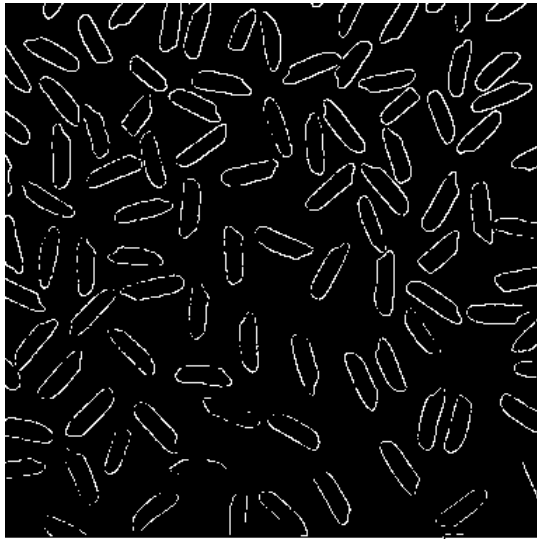
Original image



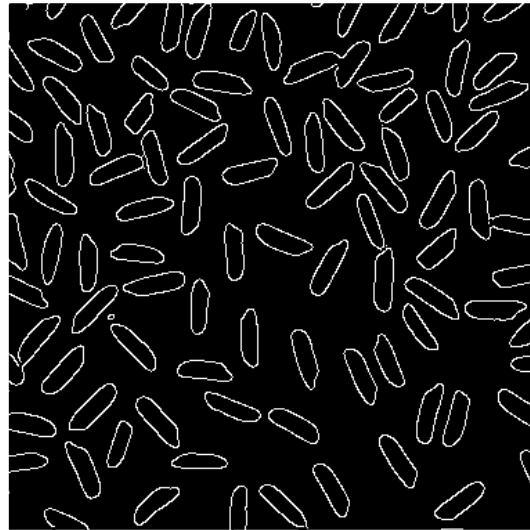
Sobel



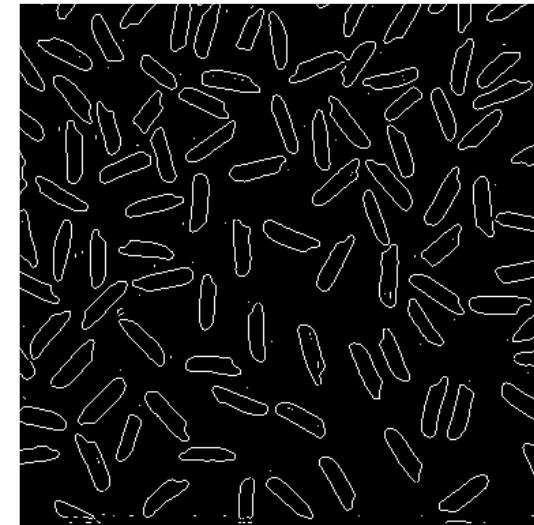
Prewitt



Roberts



Canny



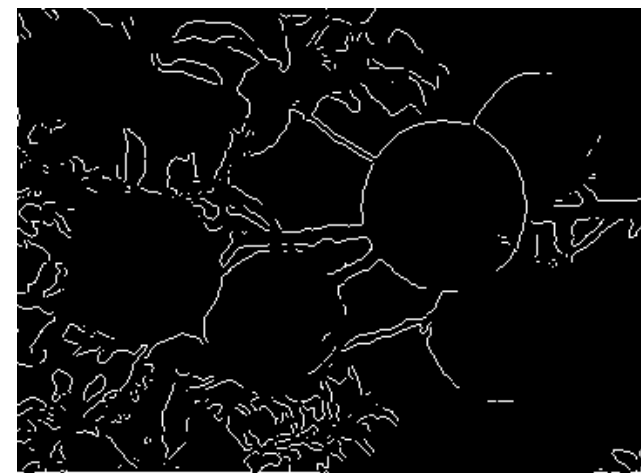
log



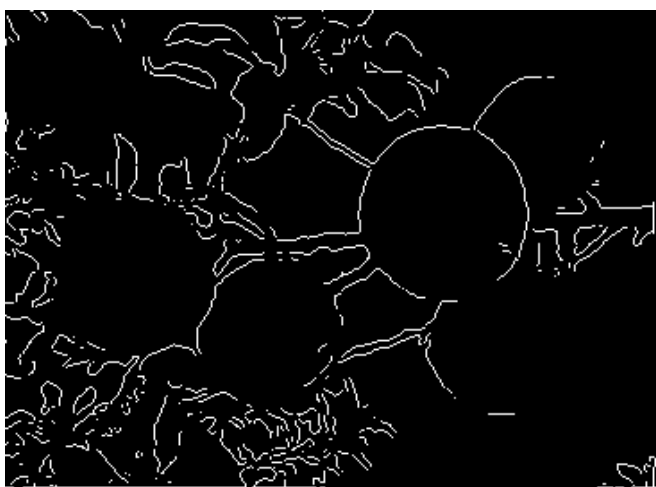
Original image



Gray image



Sobel



Prewitt



Roberts



Canny

# Deteksi tepi pada citra yang mengandung derau

```
clear all
close all
clc
Im = imread('lada-gray.bmp');
figure(1), imshow(Im); title('Original image');

Im_noise = imnoise(Im, 'salt & pepper', 0.05);
figure(2), imshow(Im_noise); title('salt & pepper');

%1 CANNY method
Im_edge_1 = edge(Im_noise, 'Canny');
figure(3), imshow(Im_edge_1); title('Edge detection using Canny');

%2 PREWITT method
Im_edge_2 = edge(Im_noise, 'prewitt');
figure(4), imshow(Im_edge_2); title('Edge detection using Prewitt');

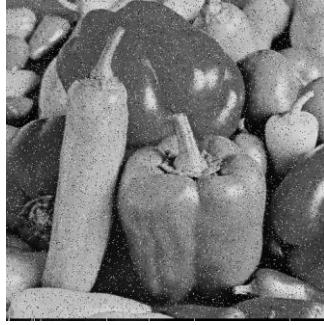
%3 ZERO CROSS method
Im_edge_3 = edge(Im_noise, 'zerocross');
figure(5), imshow(Im_edge_3); title('Edge detection using Zerocross');
```

```
%4 Roberts method
Im_edge_4 = edge(Im_noise, 'Roberts');
figure(6), imshow(Im_edge_4); title('Edge detection using roberts');

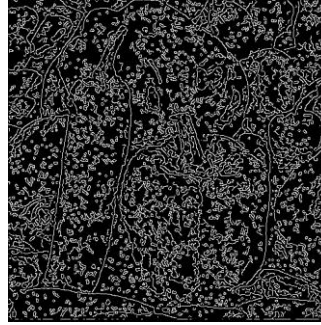
%5 Roberts method
Im_edge_5 = edge(Im_noise, 'Sobel');
figure(7), imshow(Im_edge_5); title('Edge detection using Sobel');

figure(8),
    subplot(2,3,1), imshow(Im_noise); title('Noise Image');
    subplot(2,3,2),
        imshow(Im_edge_1); title('Canny');
    subplot(2,3,3),
        imshow(Im_edge_2); title('Prewitt');
    subplot(2,3,4), imshow(Im_edge_3);
        title('Zerocross');
    subplot(2,3,5), imshow(Im_edge_4);
        title('Roberts');
    subplot(2,3,6), imshow(Im_edge_5);
        title('Sobel');
```

**Noise Image**



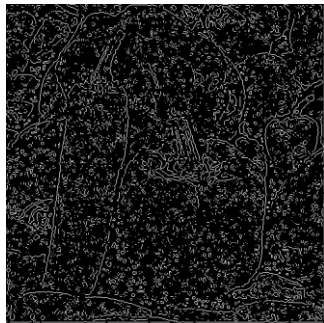
**Canny**



**Prewitt**



**Zerocross**



**Roberts**



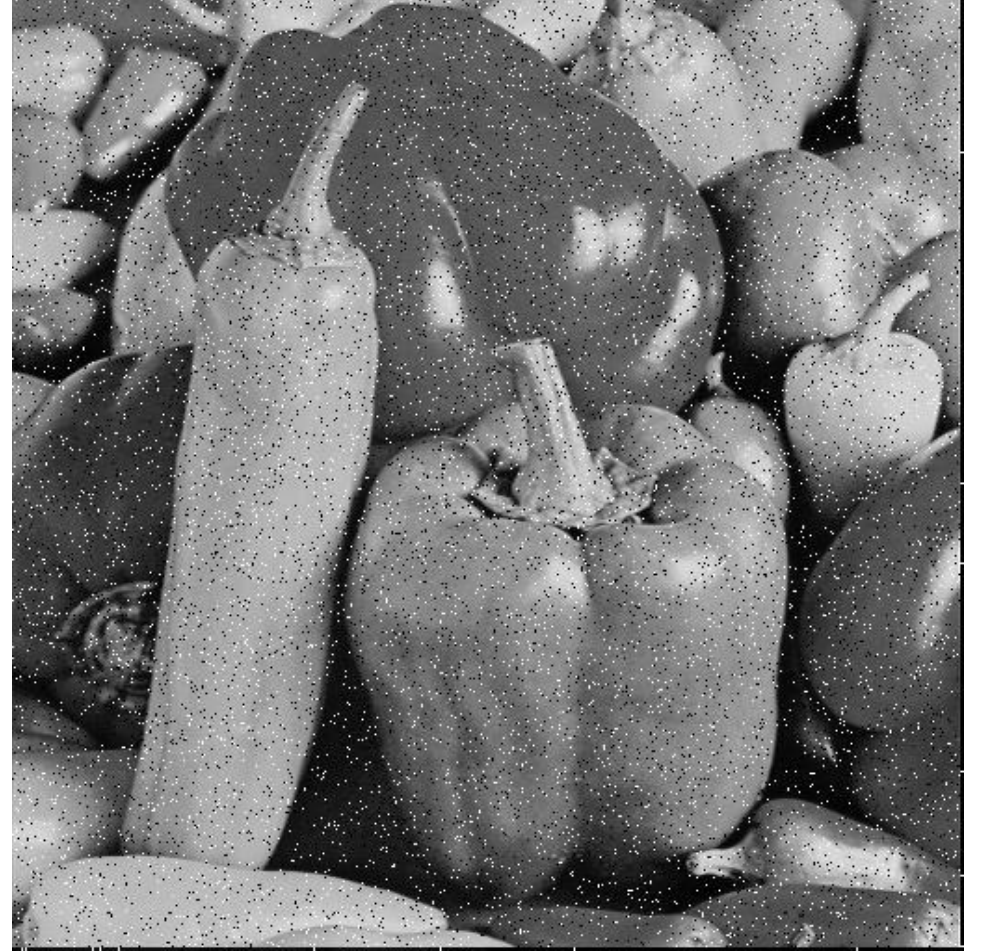
**Sobel**



**Original image**



**salt & pepper**



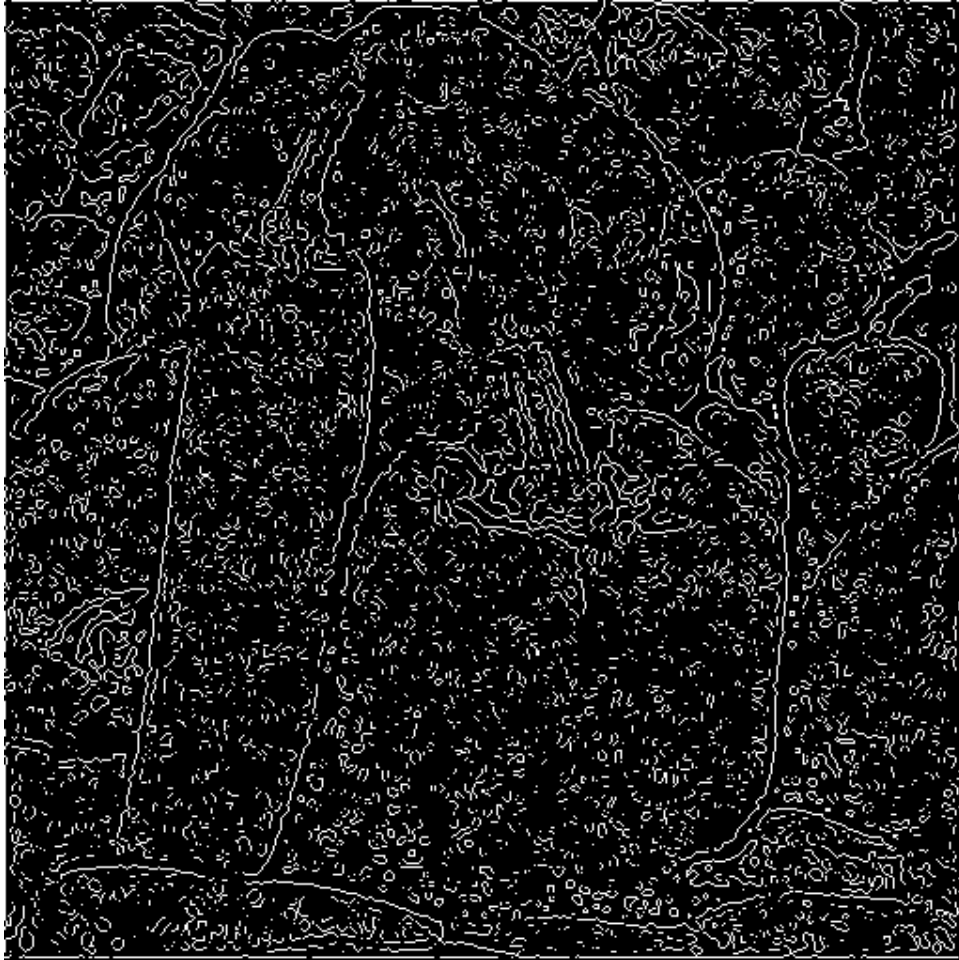
Edge detection using Canny



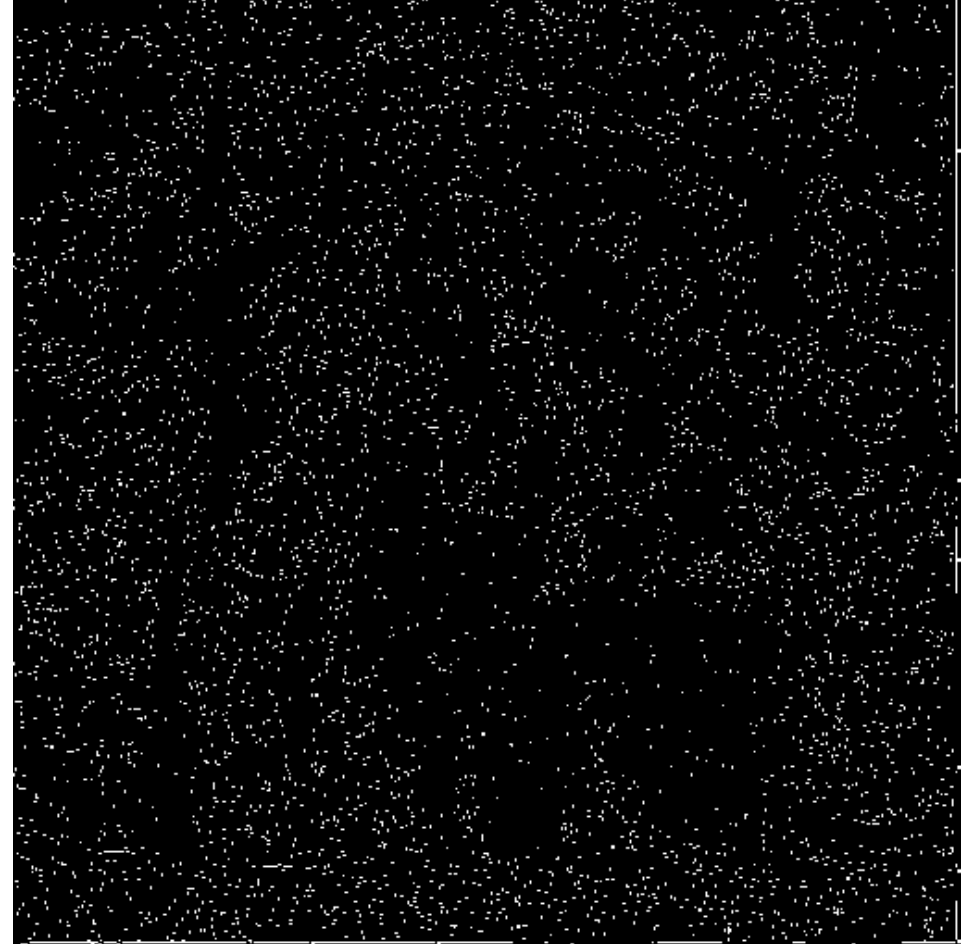
Edge detection using Prewitt



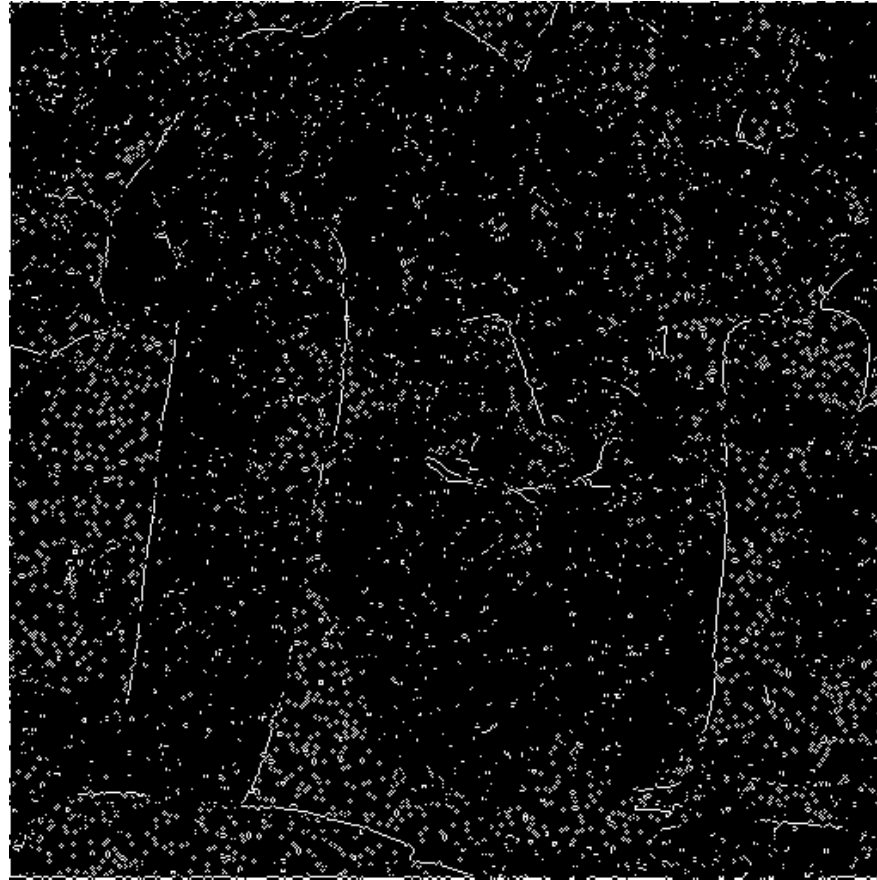
Edge detection using Zerocross



Edge detection using roberts

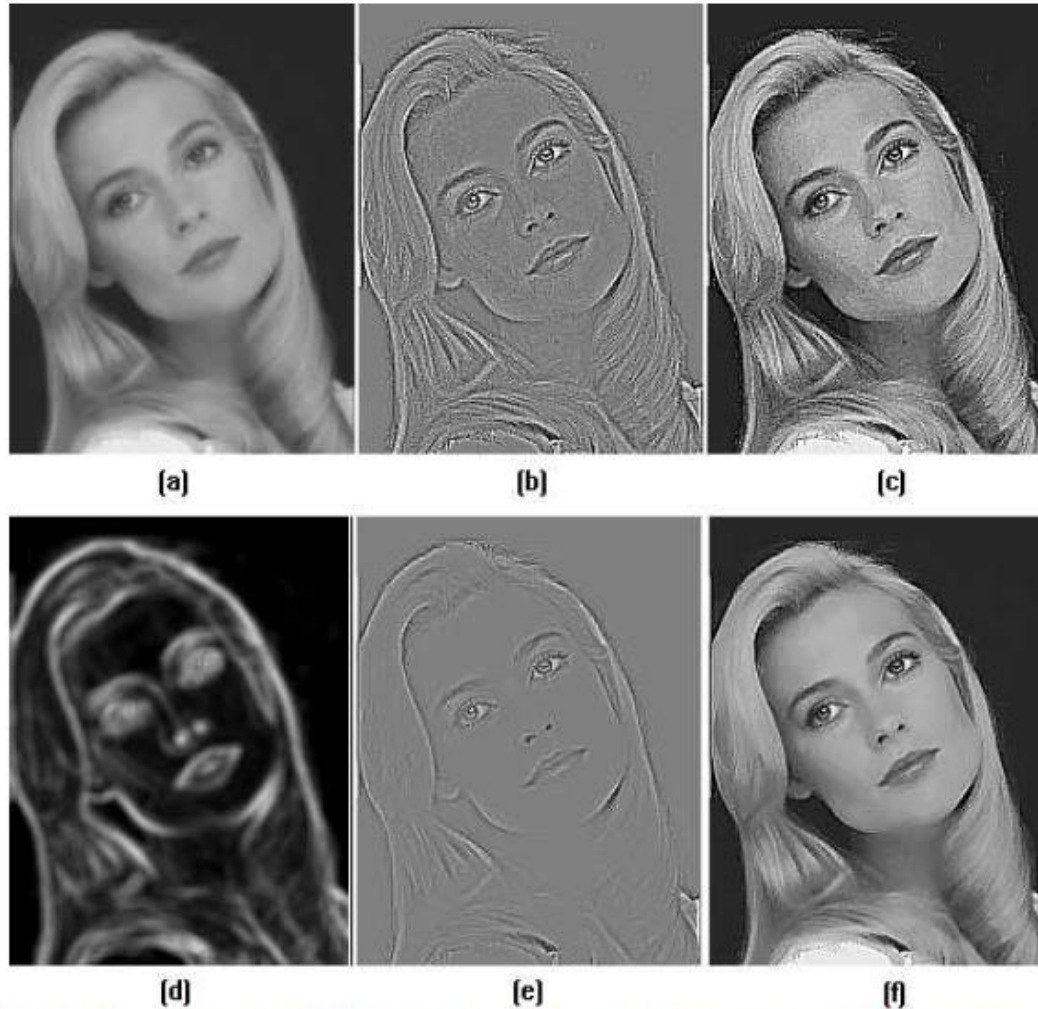


### Edge detection using Sobel



Moral dari contoh ini: untuk mendapatkan hasil deteksi tepi yang optimal, maka citra yang mengandung derau seharusnya ditapis terlebih dahulu untuk menghilangkan deraunya (image enhancement)

# Aplikasi deteksi tepi untuk *image enhancement*



**(a)** Input image; **(b)** Laplacian of (a); **(c)** Spatially invariant high-pass filtering [sum of (a) and (b)]; **(d)** Mask image [Sobel gradient of (a) smoothed by a 5x5 box filter]; **(e)** Product of (b) and (d); **(f)** Space-variant enhancement [sum of (a) and (e)].

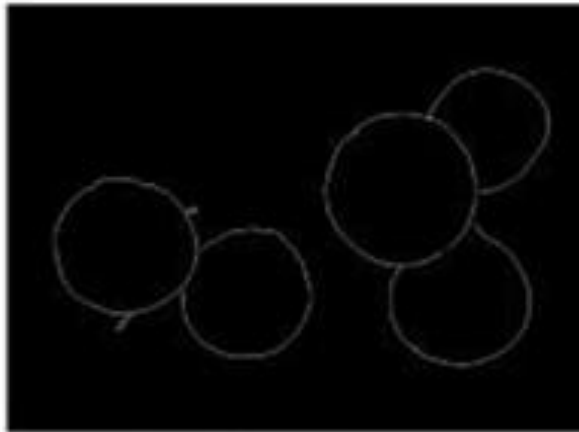
Sumber: Edge detection, Digital Image Processing, K. Pratt, Chapter 15

# Kegunaan Deteksi Tepi Untuk Segmentasi Objek

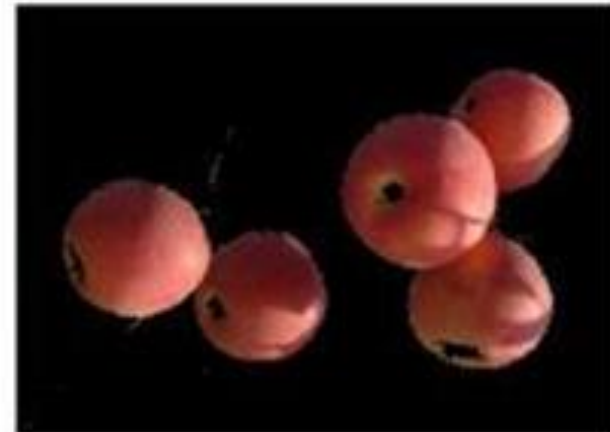
- Salah satu kegunaan deteksi tepi adalah untuk segmentasi objek, yaitu mendeteksi objek melalui bentuknya. Bentuk objek dapat diperoleh dari hasil pendeteksian tepi.
- Setelah tepi objek dideteksi, selanjutnya objek dipisahkan dari latar belakangnya, untuk kemudian digunakan dalam proses pengenalan objek (*object recognition*).



Acquired image.



Edge image.



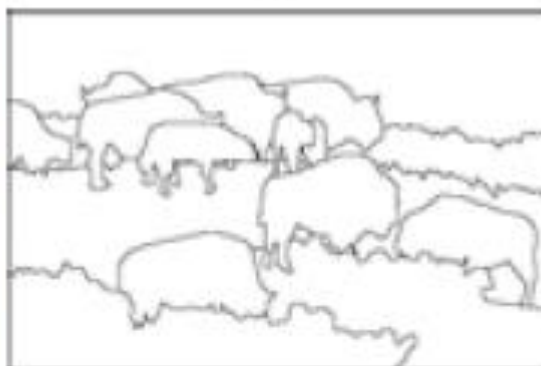
Segmentation image.

# Edge detection is just the beginning...

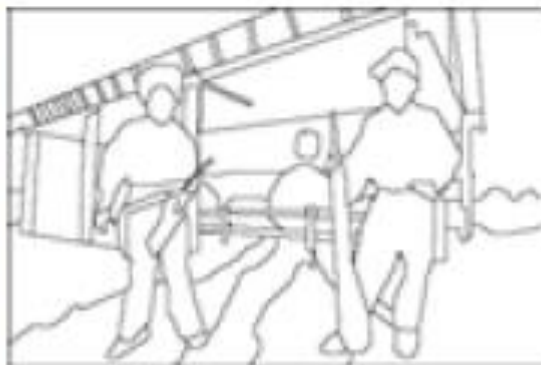
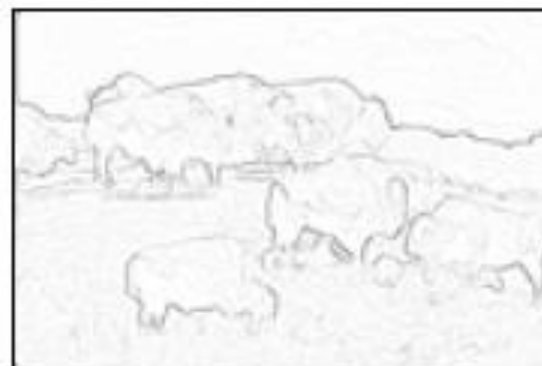
image



human segmentation



gradient magnitude



- Berkeley segmentation database:

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