Pendeteksian Tepi

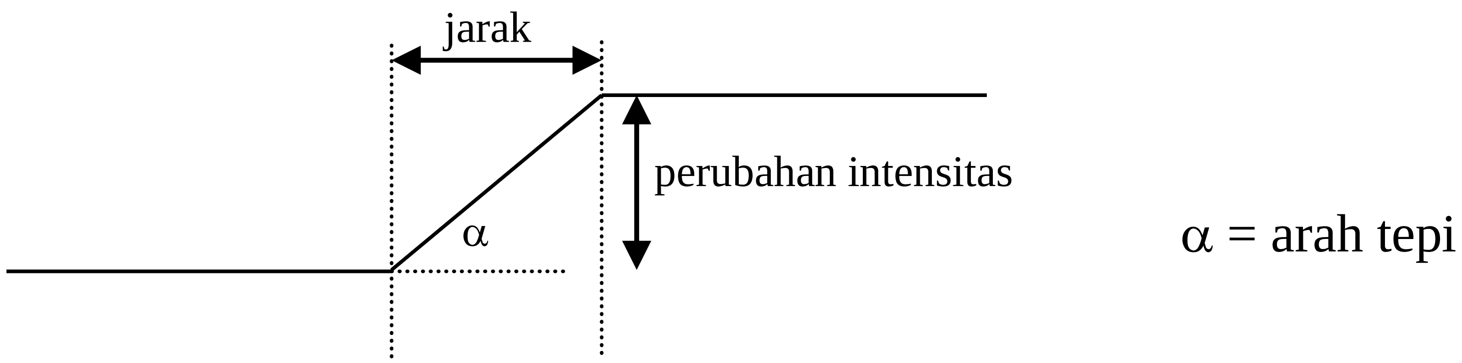
IF4073 Interpretasi dan Pengolahan Citra

Oleh: Rinaldi Munir

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2019

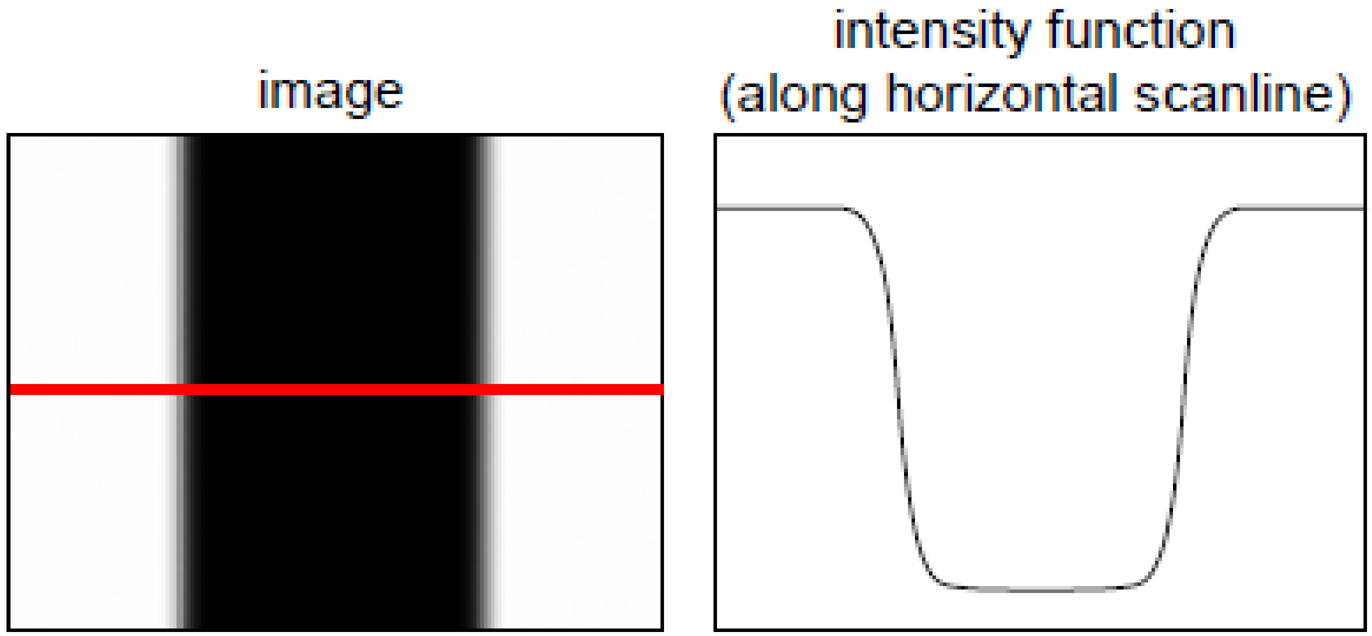
Tepi (*edge*)

- Tepi (*edge*) adalah perubahan nilai intensitas derajat keabuan yang mendadak (besar) dalam jarak yang singkat.



- Tepi memiliki arah, dan arah ini berbeda-beda pada bergantung pada perubahan intensitas

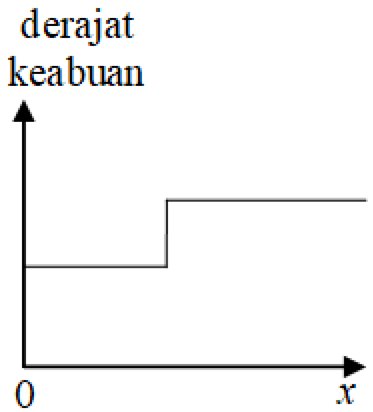
- Tepi biasanya terdapat pada batas antara dua daerah yang berbeda intensitas dengan perubahan yang sangat cepat di dalam citra.



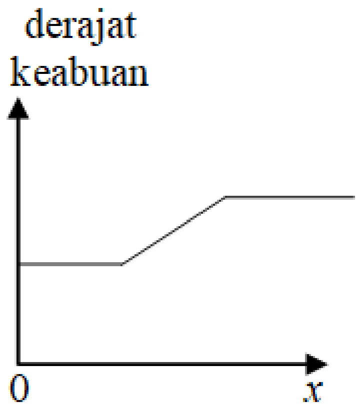
Di mana tepi?

Ini tepi

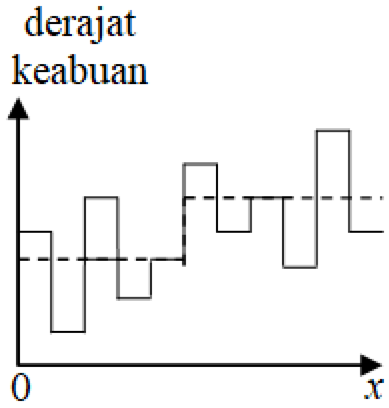
- Empat macam tepi: tepi curam (*step edge*), *tepi landau (ramp edge)*, *tepi garis (line edge), dan tepi atap (roof edge).*



(a) Tepi curam



(b) tepi landai



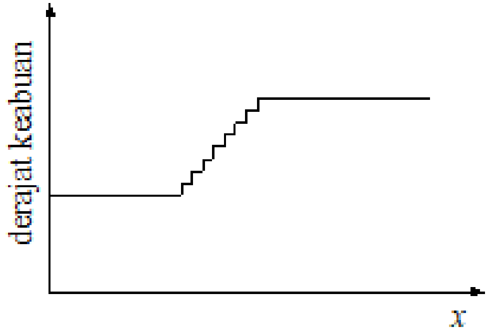
(c) tepi curam dengan derau

$$\begin{bmatrix} 4 & 4 & 4 & 8 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 & 8 \end{bmatrix}$$

Citra dengan tepi curam

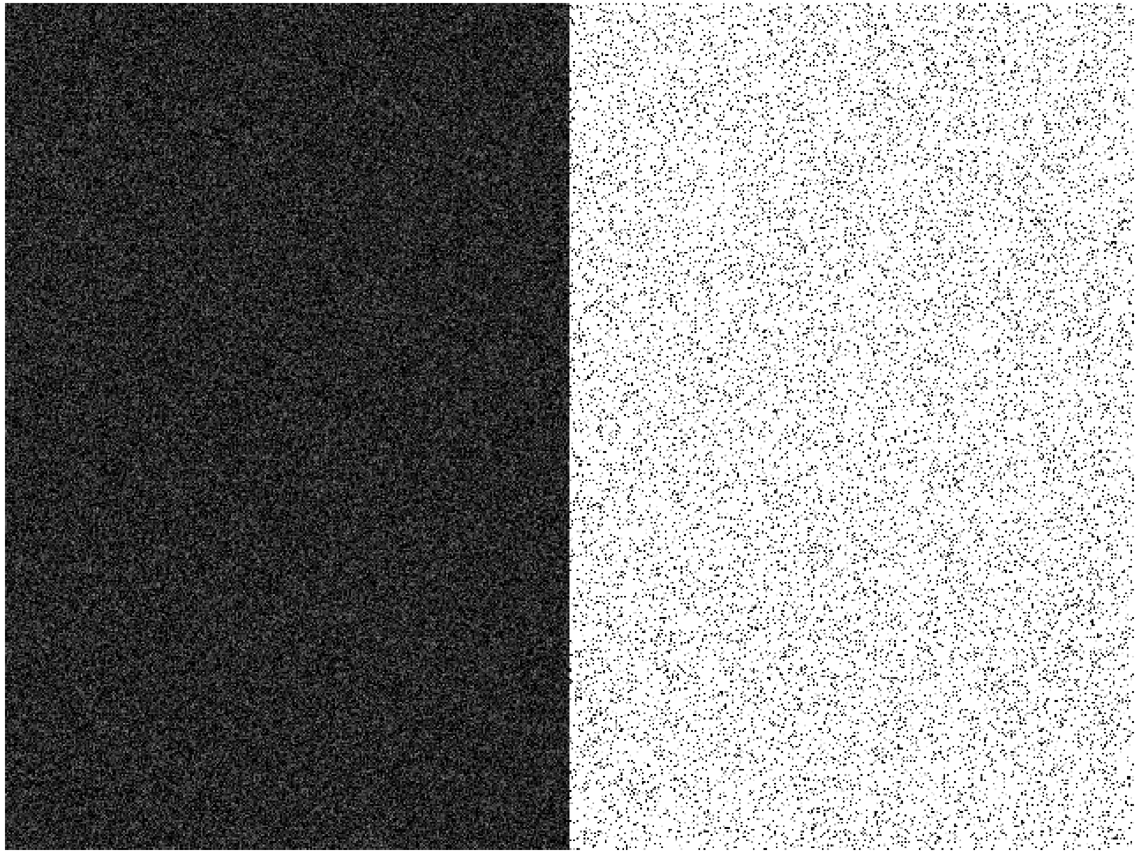
$$\begin{bmatrix} 4 & 4 & 5 & 6 & 7 & 8 & 8 & 8 \\ 4 & 4 & 5 & 6 & 7 & 8 & 8 & 8 \\ 4 & 4 & 5 & 6 & 7 & 8 & 8 & 8 \\ 4 & 4 & 5 & 6 & 7 & 8 & 8 & 8 \\ 4 & 4 & 5 & 6 & 7 & 8 & 8 & 8 \end{bmatrix}$$

Citra dengan tepi landai



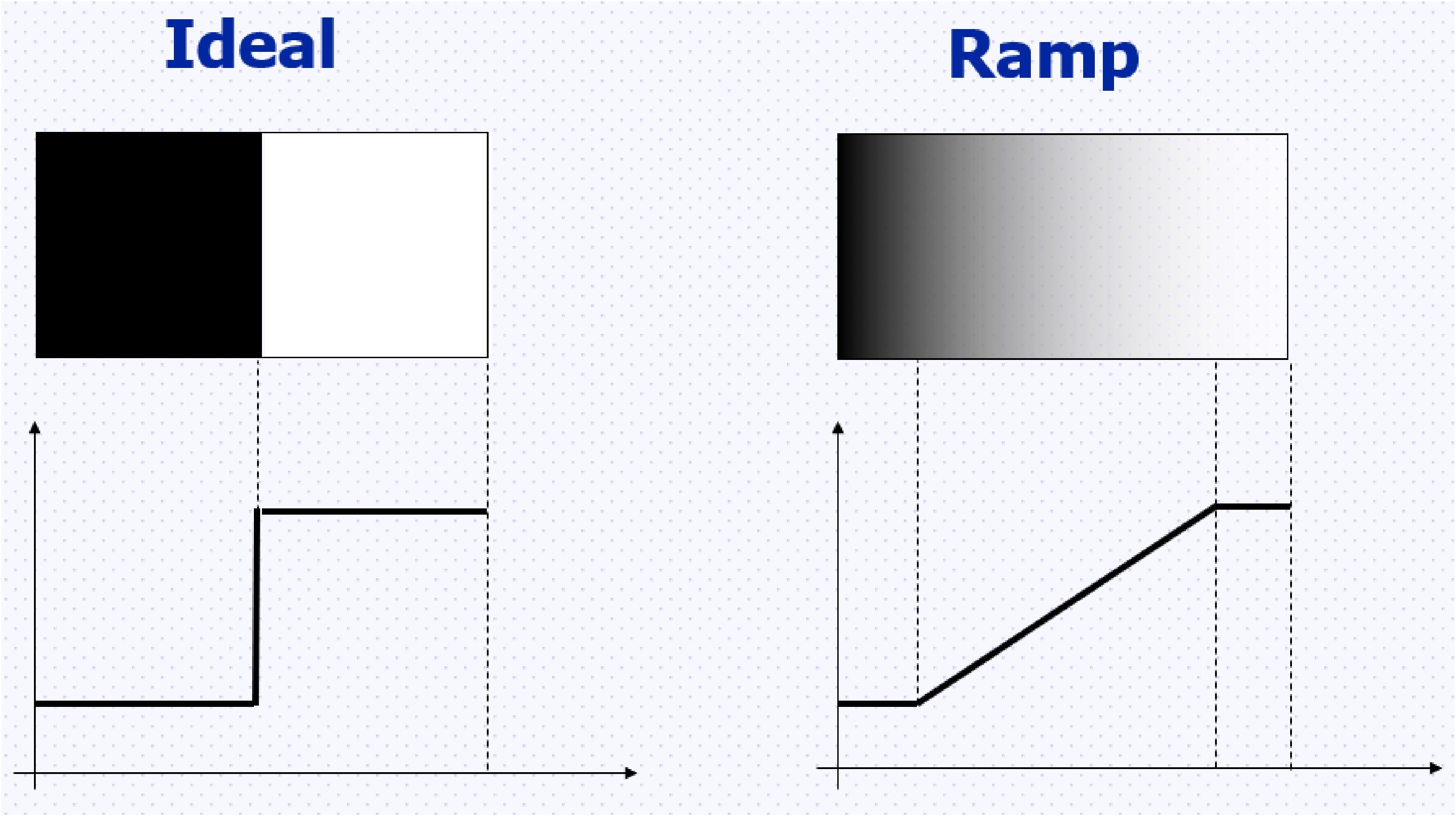
Breakdown tepi landai

Citra mengandung derau

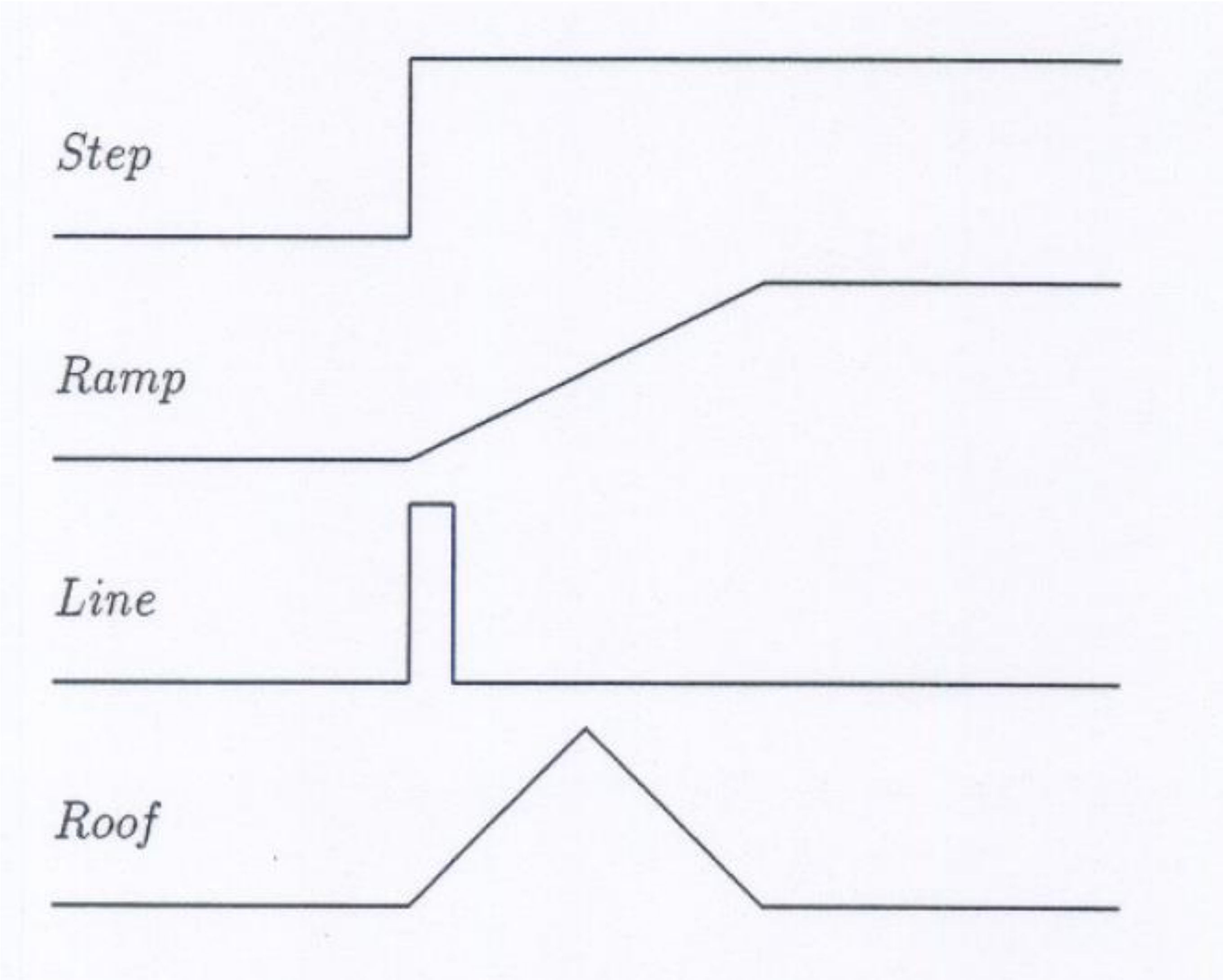


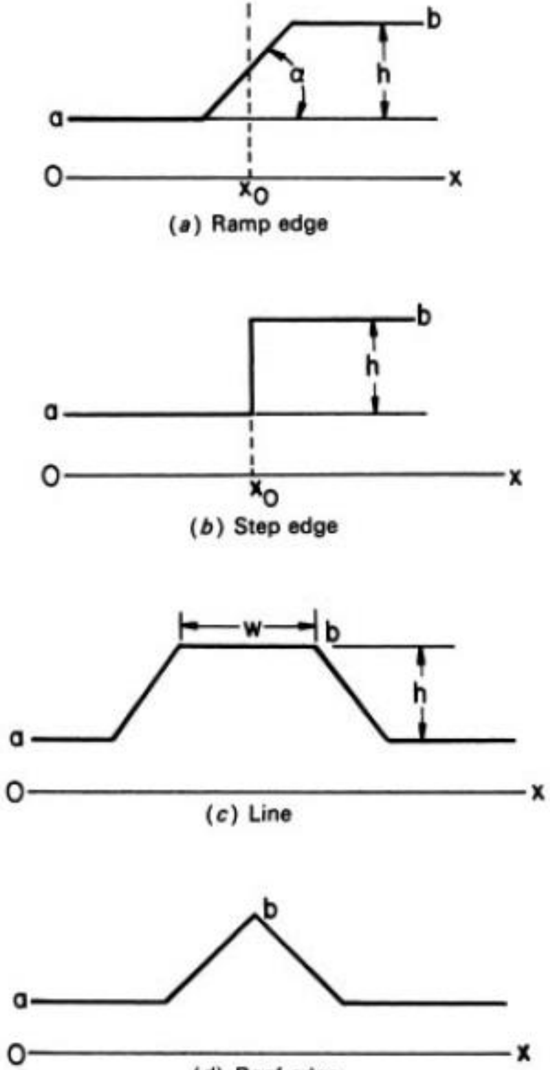
Harus bisa membedakan derau dengan tepi sebenarnya

- Idealnya sebuah tepi berbentuk curam (kemiringan 90 derajat), namun kebanyakan tepi berbentuk landai (*ramp*) dan tepi yang mengandung derau



Sumber: Texas Instruments, Edge Detection





(a) Ramp edge

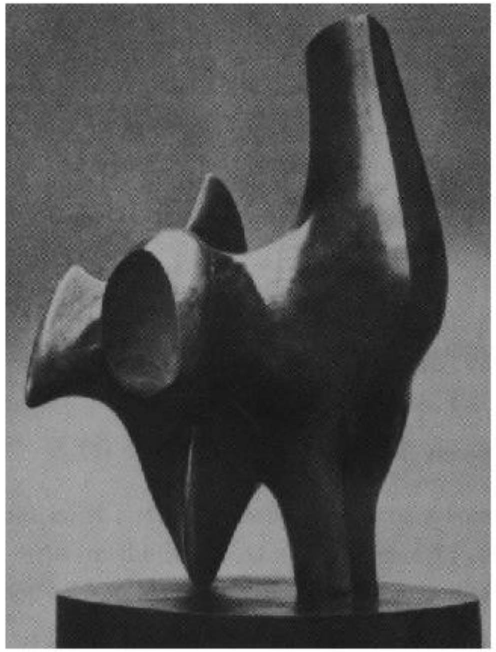
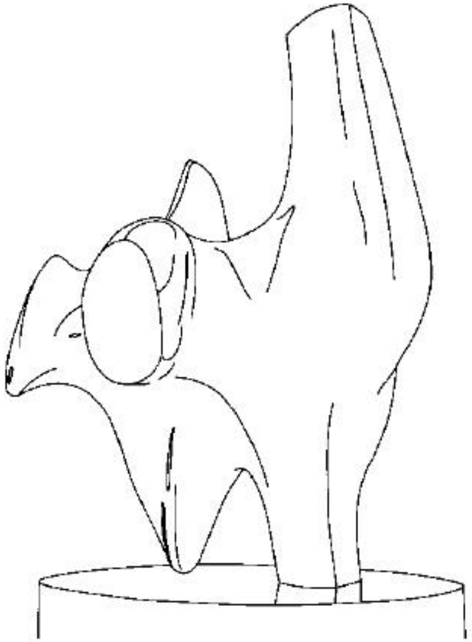
(b) Step edge

(c) Line

(d) Roof edge

Tujuan Pendeteksian Tepi

- Pendeteksian tepi bertujuan untuk meningkatkan penampakan garis batas atau objek di dalam citra.

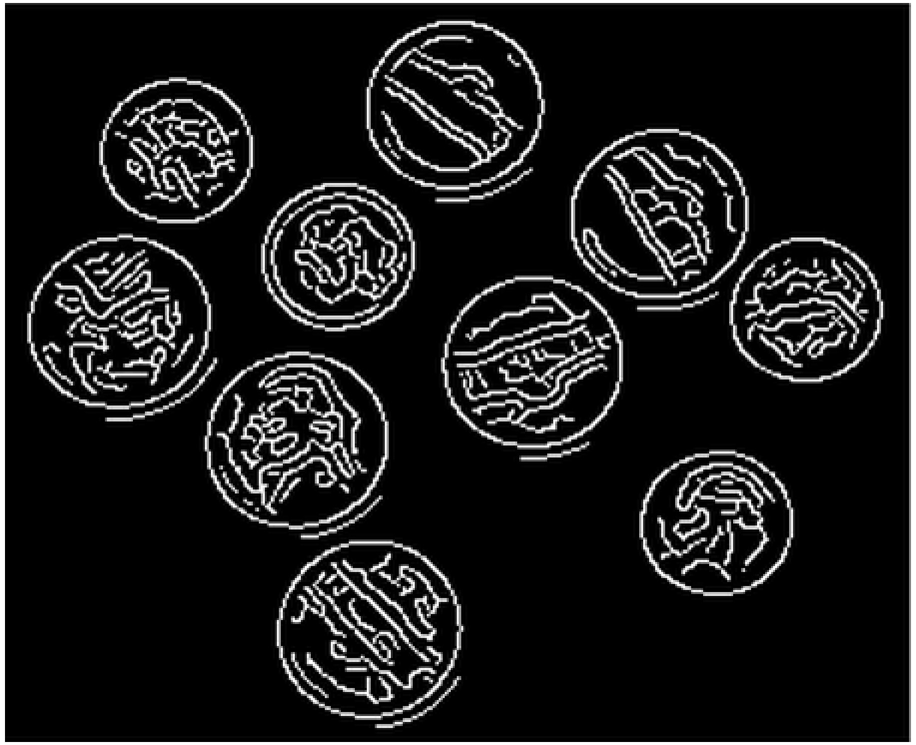



- Pendeteksian tepi mengekstraksi representasi gambar garis-garis di dalam citra.

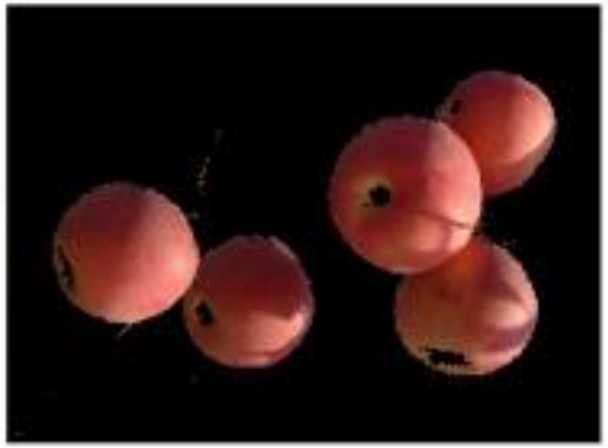
- Pendeteksian tepi berguna dalam mengenali objek di dalam citra (*image recognition*).

a) Complemented Image

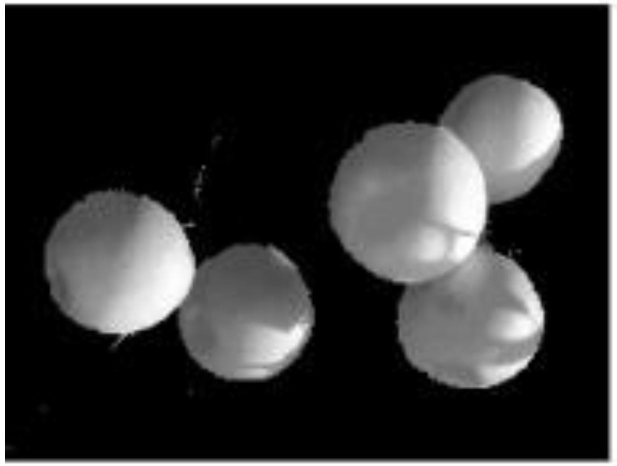
b) Edged Image



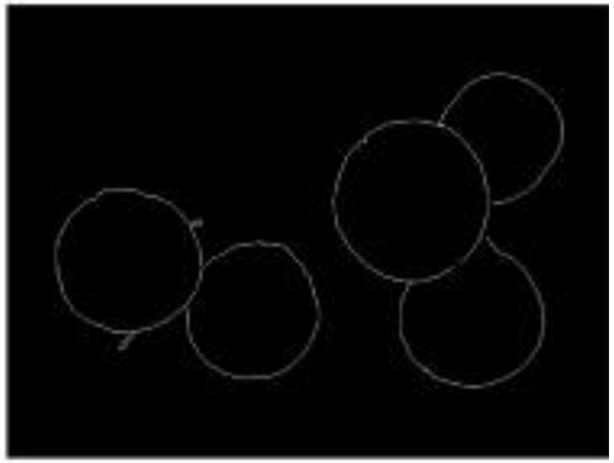
(a) Acquired image.



(b) Segmentation image.



(c) Gray image.



(d) Edge image.

- Pendeteksian tepi merupakan bagian dari **analisis citra** (*image analysis*).

- Tujuan analisis citra: mengidentifikasi parameter-parameter yang diasosiasikan dengan ciri (*feature*) dari objek di dalam citra, untuk selanjutnya parameter tersebut digunakan dalam menginterpretasi citra.

- Analisis citra pada dasarnya terdiri dari tiga tahapan:

1. Ekstrakasi ciri (*feature extraction*).

   Faktor kunci dalam mengekstraksi ciri adalah kemampuan mendeteksi keberadaan tepi (*edge*) dari objek di dalam citra.

2. Segmentasi

   Setelah tepi objek diketahui, langkah selanjutnya dalam analisis citra adalah segmentasi, yaitu mereduksi citra menjadi objek atau region, misalnya memisahkan objek-objek yang berbeda dengan mengekstraksi batas-batas objek (*boundary*).

3. Klasifikasi.

   Langkah terakhir dari analisis citra adalah klasifikasi, yaitu memetakan segmen-segmen yang berbeda ke dalam kelas objek yang berbeda pula.

Operator Gradien

- Pendeteksian tepi dapat dipahami dengan pendekatan kaklulus diferensial.

- Sebab, perubahan intensitas yang besar dalam jarak yang singkat dipandang sebagai fungsi yang memiliki kemiringan yang besar.

- Kemiringan fungsi dapat dihitung dengan turunan pertama (*gradient*)

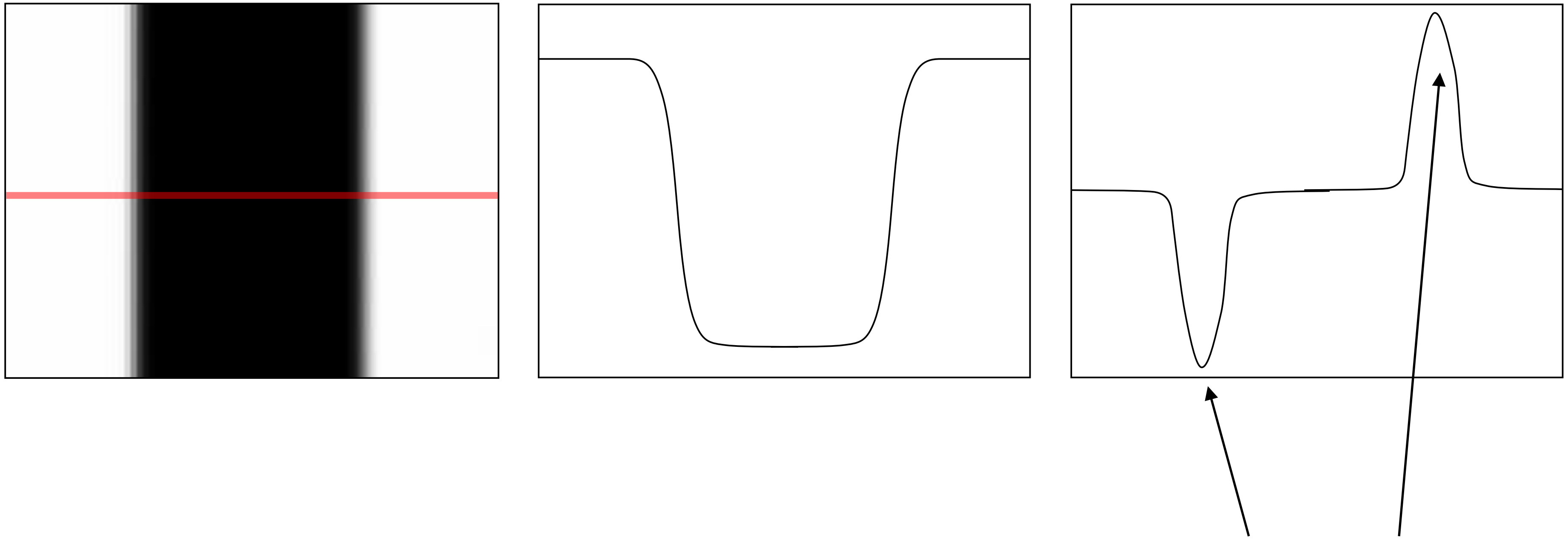
$$\frac{\partial F}{\partial x} = \lim_{h \to 0} \frac{F(x+h, y) - F(x, y)}{h}$$

Sumber: Noah Snavely, CS6670: Computer Vision

image

intensity function
(along horizontal scanline)

first derivative



edges correspond to
extrema of derivative

Source: L. Lazebnik

- Karena citra *f*(*x*,*y*) adalah fungsi dwimatra dalam bentuk diskrit, maka turunan pertamanya adalah secara parsial:

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

$$G_x = \frac{\partial f(x,y)}{\partial x} = \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$$

$$G_y = \frac{\partial f(x,y)}{\partial y} = \frac{f(x, y+\Delta y) - f(x,y)}{\Delta y}$$

- Biasanya $\Delta x = \Delta y = 1$, sehingga

$$G_x = \frac{\partial f(x,y)}{\partial x} = f(x+1, y) - f(x,y)$$ dan $$G_y = \frac{\partial f(x,y)}{\partial y} = f(x, y+1) - f(x,y)$$

Misalkan susunan *pixel-pixel* di dalam citra:

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

Diferensial maju (*forward differential*):

$$G_x = \frac{\partial f(x,y)}{\partial x} = f(x+1, y) - f(x,y)$$

$$G_y = \frac{\partial f(x,y)}{\partial y} = f(x, y+1) - f(x,y)$$

Kedua turunan parsial di atas dapat dipandang sebagai dua buah *mask* konvolusi:

$$G_x = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & 1 \end{bmatrix}$$

Bisa juga menggunakan *mask* gradien 2 x 2 sebagai berikut:

$$G_x = \begin{array}{|c|c|} \hline -1 & 1 \\ \hline -1 & 1 \\ \hline \end{array} \qquad G_y = \begin{array}{|c|c|} \hline 1 & 1 \\ \hline -1 & -1 \\ \hline \end{array}$$

- Kekuatan tepi (*edge strengthness*): $G[f(x,y)] = \sqrt{G_x^{\,2} + G_y^{\,2}}$

- Arah tepi: $\alpha(x,y) = \tan^{-1} \dfrac{G_y}{G_x}$

- Hasil pendeteksian tepi adalah **citra tepi** (*edges image*): $g(x,y) = G[f(x,y)]$

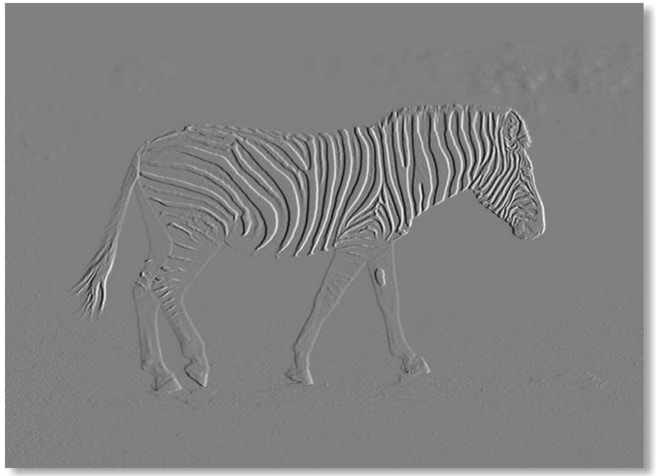
- Keputusan apakah suatu *pixel* merupakan tepi atau bukan tepi dinyatakan dengan operasi pengambangan:

$$g(x,y) = \begin{cases} 1, & \text{jika } G[f(x,y)] \geq T \\ 0, & \text{lainnya} \end{cases}$$

- *T* adalah nilai ambang, *pixel* tepi dinyatakan putih (1) sedangkan *pixel* bukan tepi dinyatakan hitam (0).

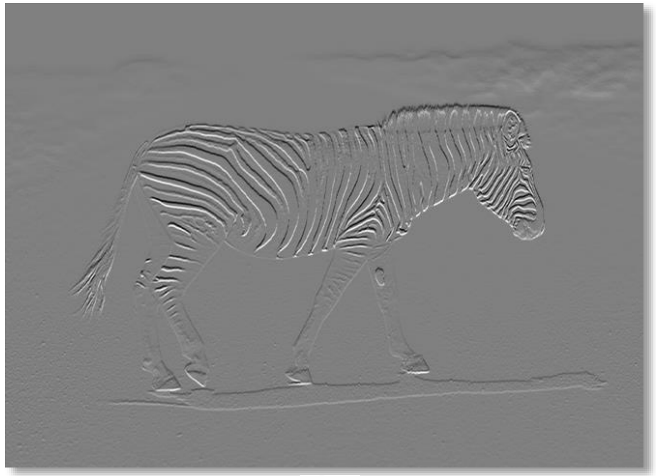
Sumber: Noah Snavely, CS6670: Computer Vision

$$f$$

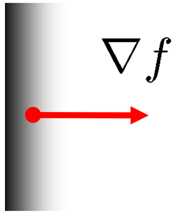


$$\frac{\partial f}{\partial x}$$

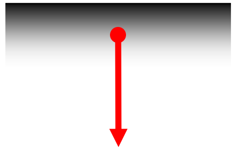
$$\|\nabla f\| = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2}$$



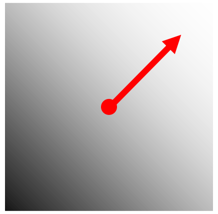
$$\frac{\partial f}{\partial y}$$



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

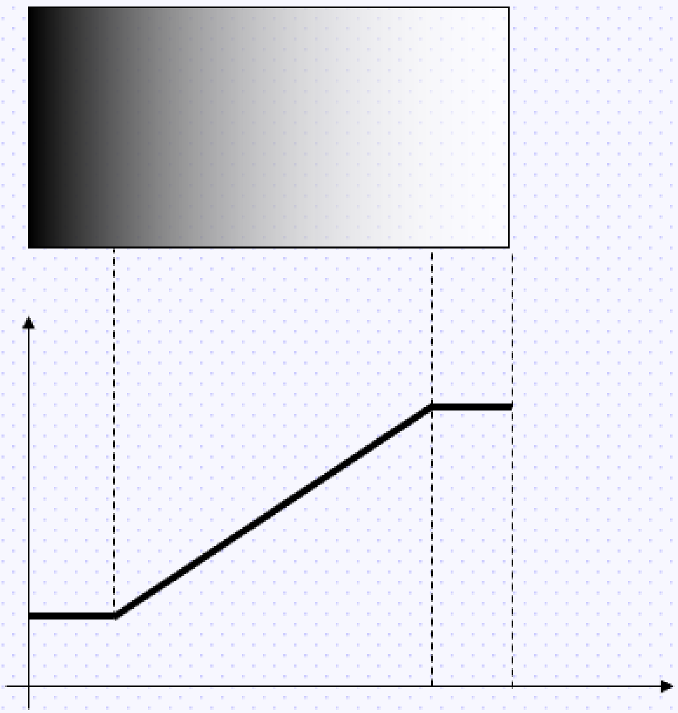


$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

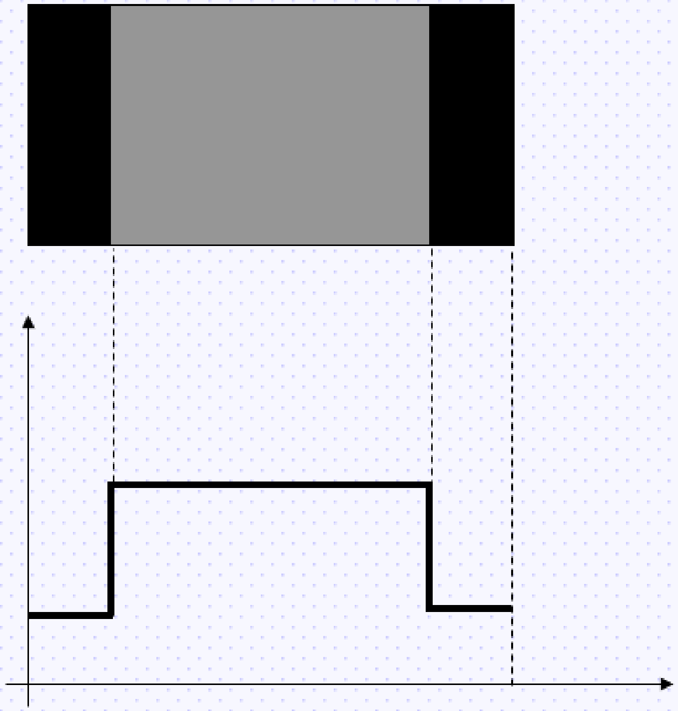


$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Citra



Gradien (turunan pertama)



Contoh. Misalkan terdapat sebuah 5×5 citra dengan dua derajat keabuan sebagai berikut:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$G_x = \frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$

$$G_y = \frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$

$$\alpha(x,y) = \tan^{-1} \frac{G_y}{G_x}$$

Hasil perhitungan gradien setiap *pixel* di dalam citra adalah sebagai berikut:

Citra	Gradien-*x*	Gradien-y	Arah gradien
1 1 1 1 1	0 0 0 0 0	0 0 0 0	* * * * *
1 1 1 1 1	0 0 0 −1 −	0 0 0 0	* * * ↔ ←
1 1 1 0 0	0 0 −1 0 0	0 0 −1 0	* * [illegible] * *
1 1 0 0 0	0 0 0 0 0	0 −1 0 0	* ↑ * * *
1 1 0 0 0	* * * * *	0 −1 0 0	* ↑ * * *

■

- Kembali ke rumus kekuatan tepi: $G[f(x,y)] = \sqrt{G_x^2 + G_y^2}$

- Karena menghitung akar lebih kompleks dan menghasilkan nilai riil, maka dalam praktek kekuatan tepi disederhanakan perhitungannya dengan menggunakan salah satu dari alternative:

(i) $G[f(x,y)] = |G_x^2| + |G_y^2|$, atau

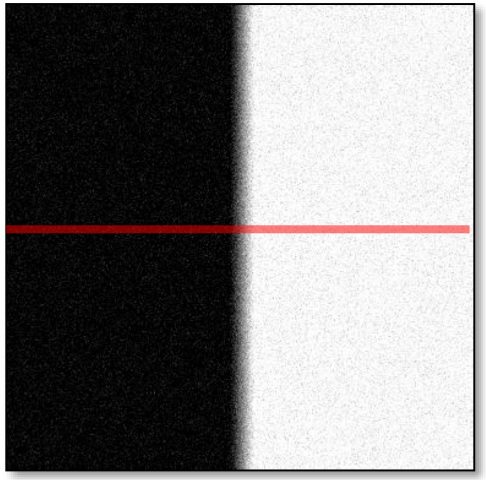
(ii) $G[f(x,y)] = |G_x| + |G_y|$, atau

(iii) $G[f(x,y)] = \max\{|G_x^2|, |G_y^2|\}$, atau

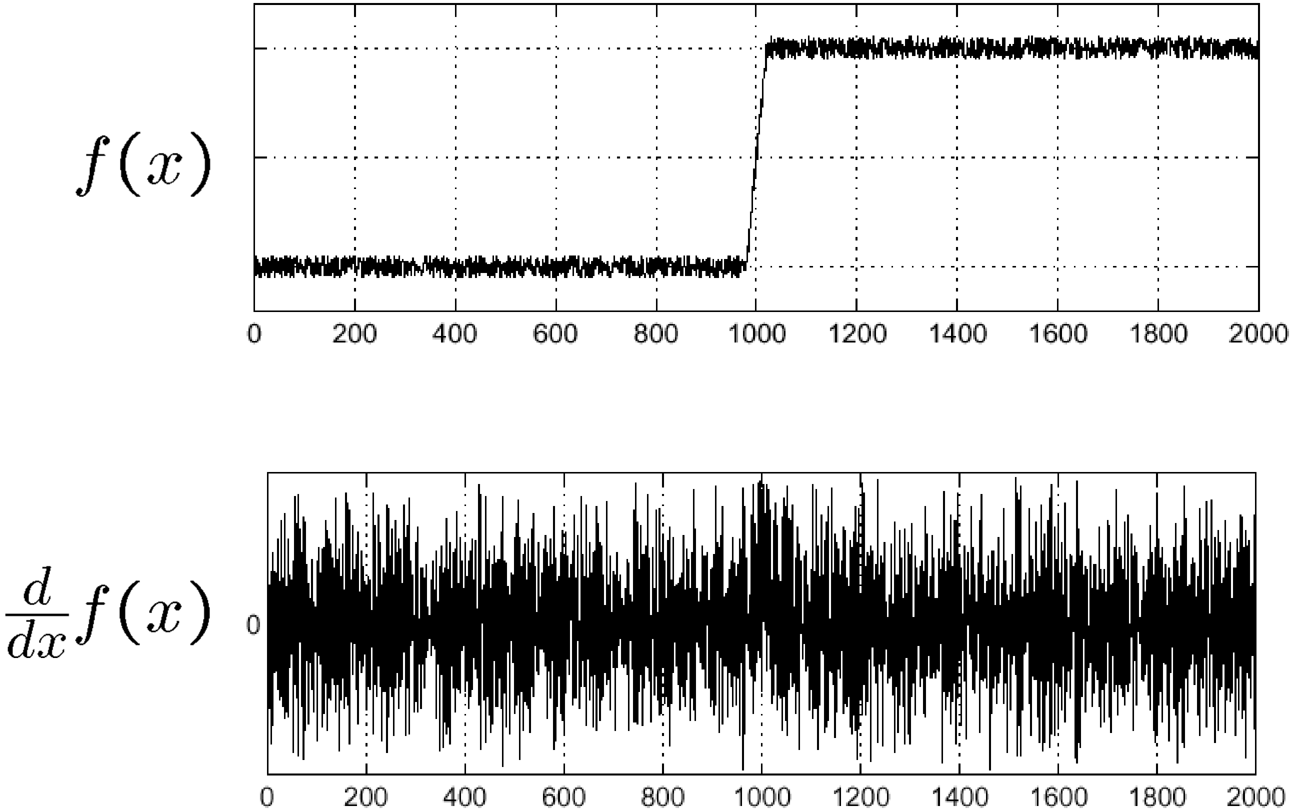
(iv) $G[f(x,y)] = \max\{|G_x|, |G_y|\}$.

Persamaan (ii) dan (iv) biasanya lebih disukai karena lebih mudah perhitungannya.

Efek derau di dalam citra



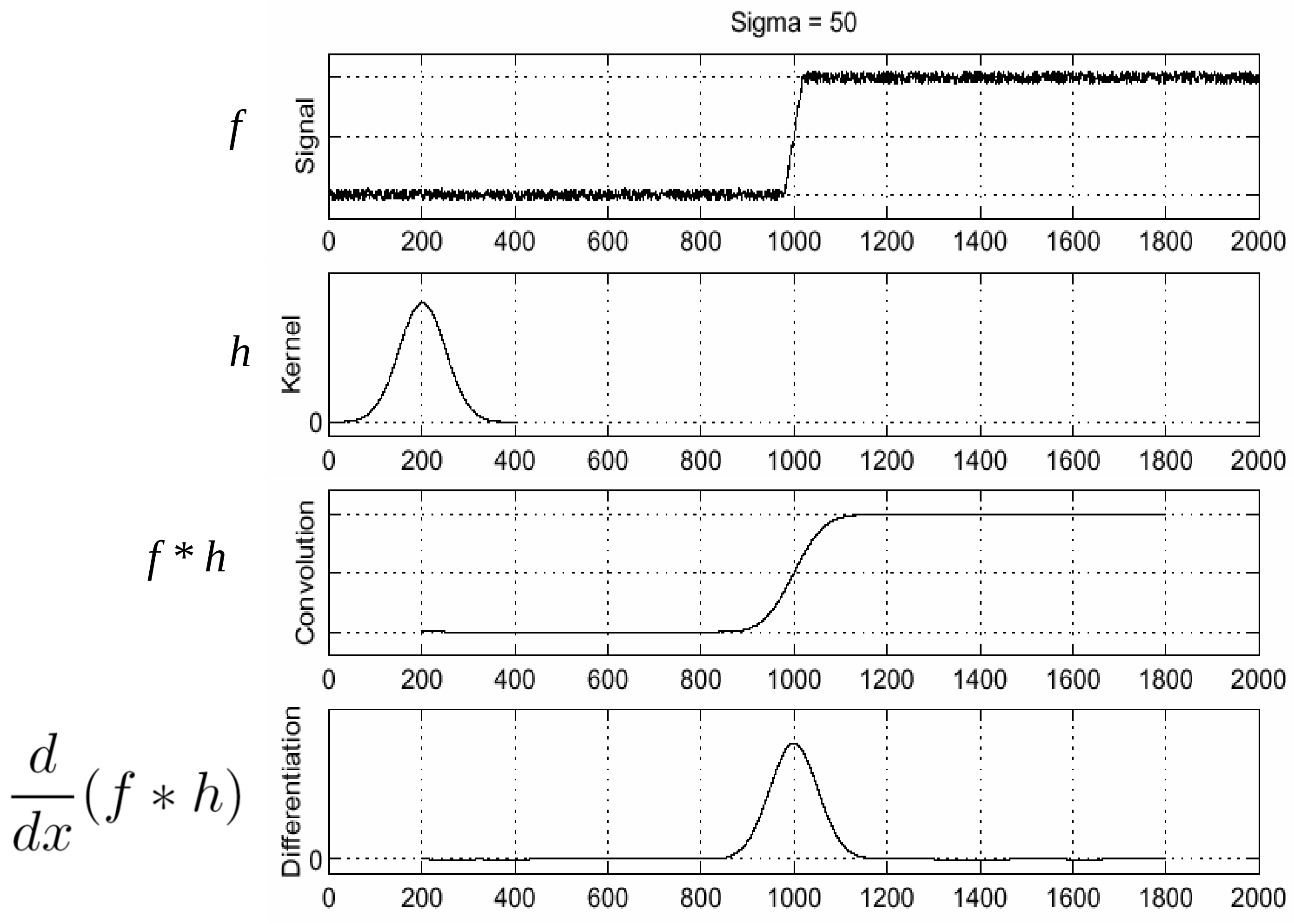
Noisy input image



Dimanakah tepi?

Source: S. Seitz

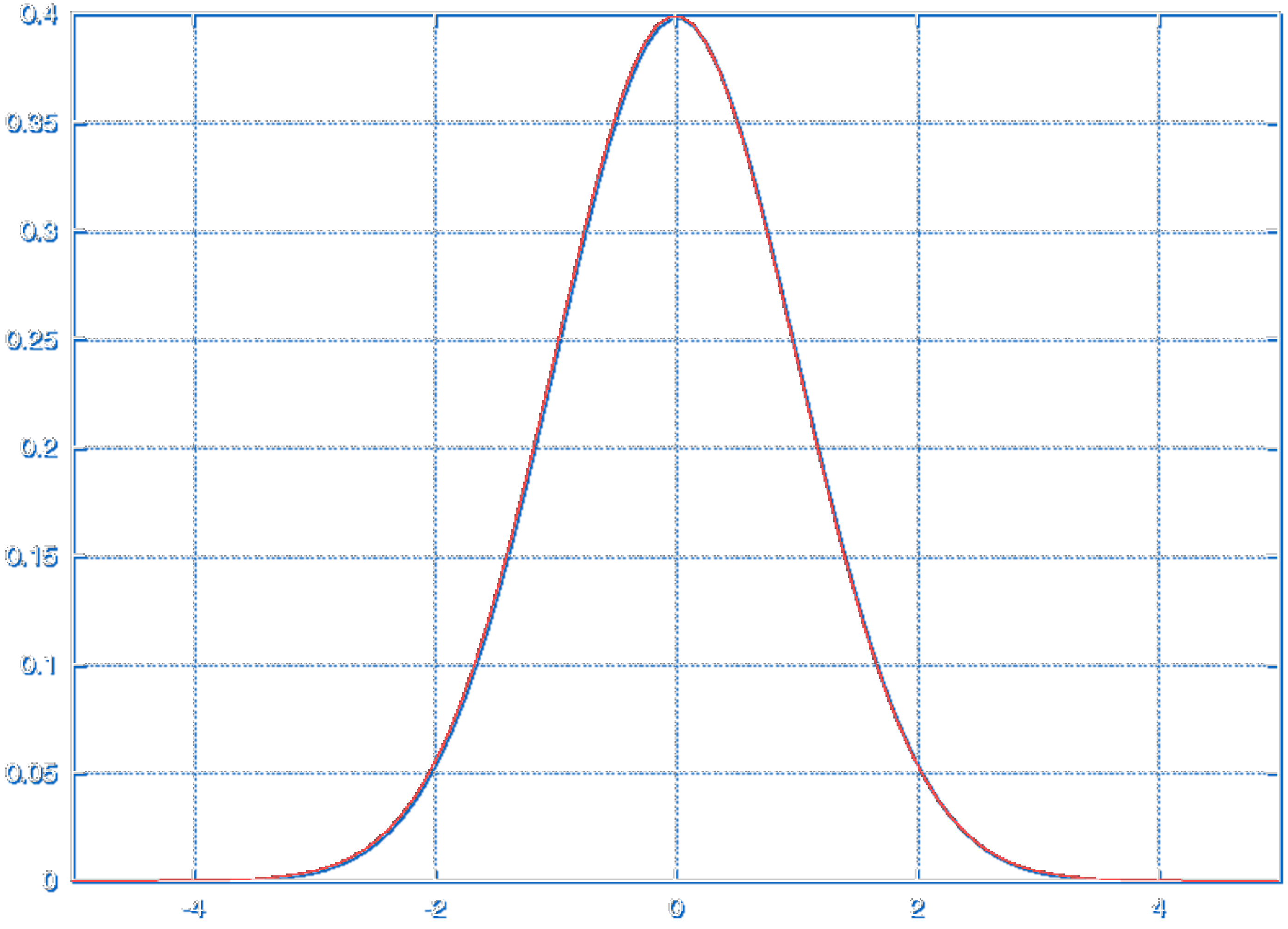
Solusi: lakukan pelembutan (*image smoothing*) terlebih dahulu, misalnya dengan penapis Gaussian



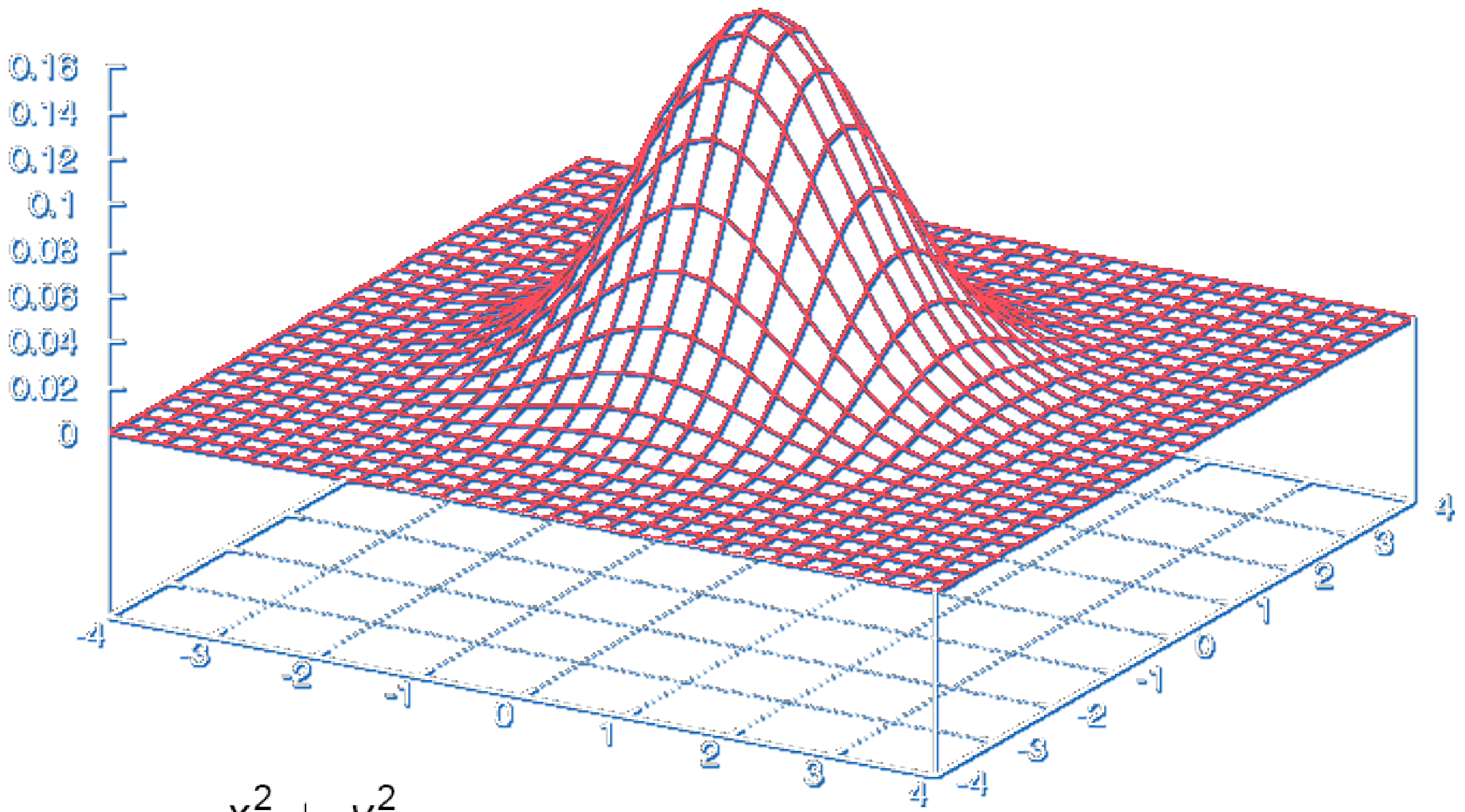
Sumber: Noah Snavely, CS6670: Computer Vision

Source: S. Seitz

Fungsi Gaussian (1-D)



Fungsi Gaussian (2-D)



$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

3 x 3 Gaussian mask

$$\frac{1}{16} \times$$

1	2	1
2	4	2
1	2	1

7×7 Gaussian mask

1	1	2	2	2	1	1
1	2	2	4	2	2	1
2	2	4	8	4	2	2
2	4	8	16	8	4	2
2	2	4	8	4	2	2
1	2	2	4	2	2	1
1	1	2	2	2	1	1

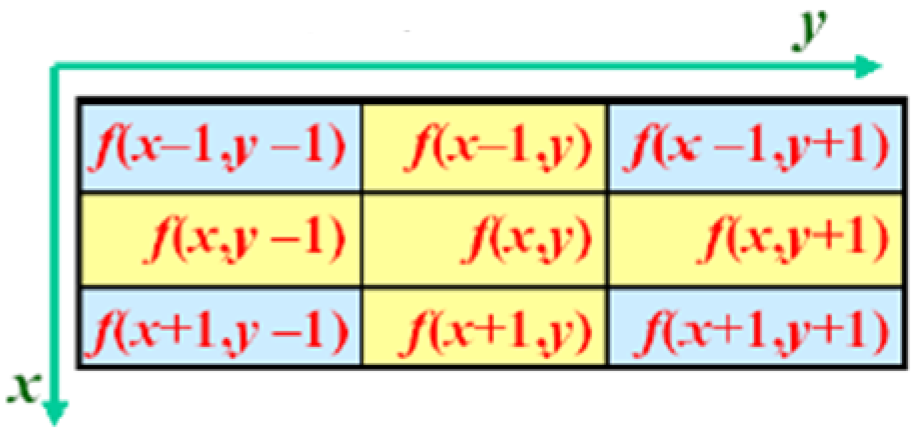
15×15 Gaussian mask

2	2	3	4	5	5	6	6	6	5	5	4	3	2	2
2	3	4	5	7	7	8	8	8	7	7	5	4	3	2
3	4	6	7	9	10	10	11	10	10	9	7	6	4	3
4	5	7	9	10	12	13	13	13	12	10	9	7	5	4
5	7	9	11	13	14	15	16	15	14	13	11	9	7	5
5	7	10	12	14	16	17	18	17	16	14	12	10	7	5
6	8	10	13	15	17	19	19	19	17	15	13	10	8	6
6	8	11	13	16	18	19	20	19	18	16	13	11	8	6
6	8	10	13	15	17	19	19	19	17	15	13	10	8	6
5	7	10	12	14	16	17	18	17	16	14	12	10	7	5
5	7	9	11	13	14	15	16	15	14	13	11	9	7	5
4	5	7	9	10	12	13	13	13	12	10	9	7	5	4
3	4	6	7	9	10	10	11	10	10	9	7	6	4	3
2	3	4	5	7	7	8	8	8	7	7	5	4	3	2
2	2	3	4	5	5	6	6	6	5	5	4	3	2	2

- Operator gradien lainnya adalah operator gradien selisih-terpusat (*center-difference*):

$$D_x(x,y) = \frac{\partial f(x,y)}{\partial x} = \frac{f(x+1,y) - f(x-1,y)}{2}$$

$$D_y(x,y) = \frac{\partial f(x,y)}{\partial y} = \frac{f(x,y+1) - f(x,y-1)}{2}$$

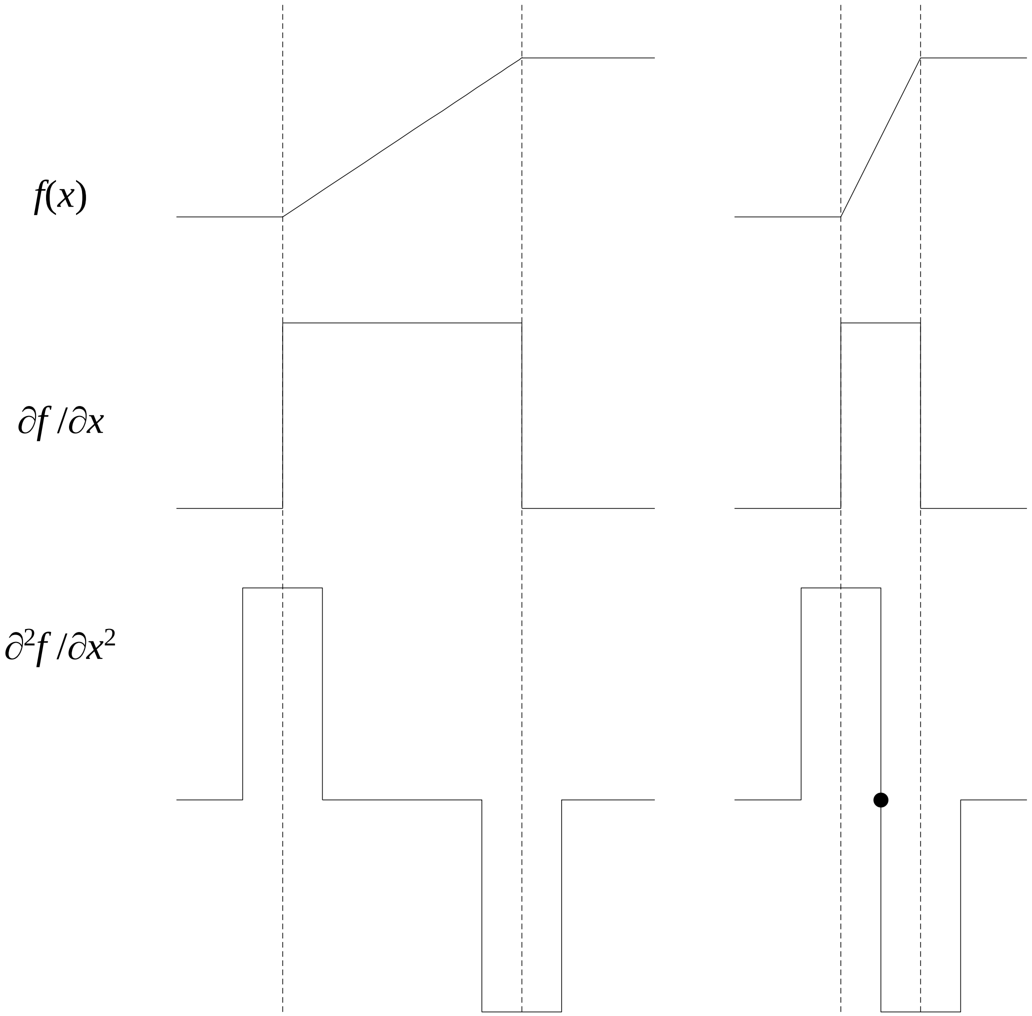


- Ekivalen dengan *mask*:

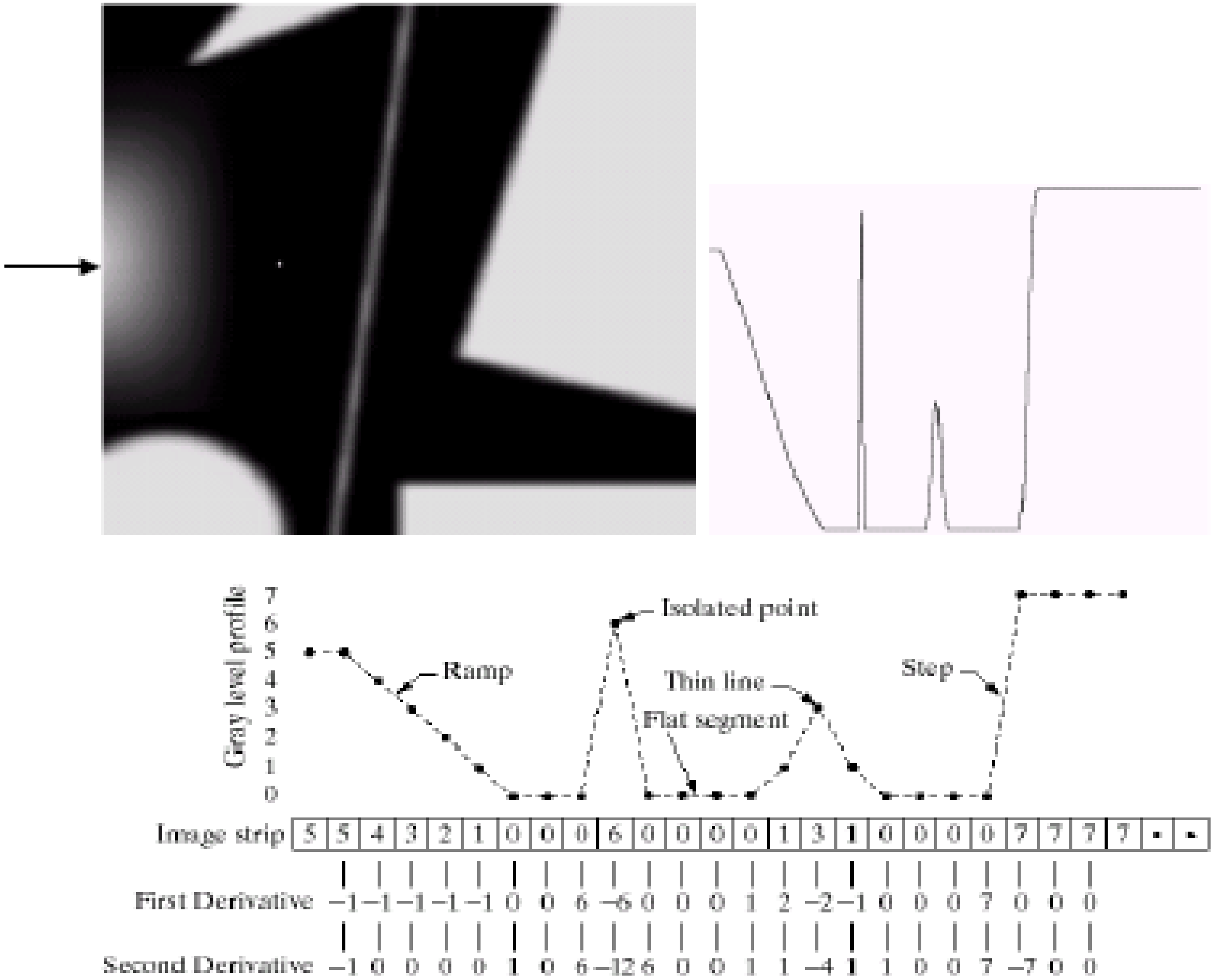
$$D_x = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \qquad D_y = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

Operator Turunan Kedua (Laplace)

- Turunan kedua: $\nabla^2 f = \dfrac{\partial^2 f}{\partial x^2} + \dfrac{\partial^2 f}{\partial y^2}$
- Operator turunan kedua disebut juga **operator Laplace**.
- Operator Laplace mendeteksi lokasi tepi lebih akurat khususnya pada tepi yang curam.
- Pada tepi yang curam, turunan keduanya mempunyai persilangan nol *(zero-crossing*), yaitu titik di mana terdapat pergantian tanda nilai turunan kedua dari positif ke negative atau sebaliknya.



(a) Tepi landai (b) Tepi curam

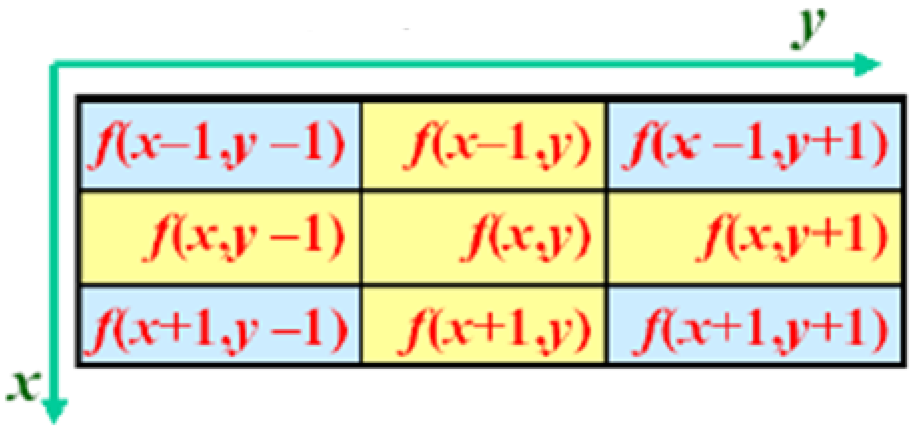

Gray level profile
7
6
5
4
3
2
1
0
Isolated point
Ramp
Thin line
Flat segment
Step
Image strip 5 5 4 3 2 1 0 0 0 6 0 0 0 0 1 3 1 0 0 0 0 7 7 7 7
First Derivative −1 −1 −1 −1 −1 0 0 6 −6 0 0 0 1 2 −2 −1 0 0 0 7 0 0 0
Second Derivative −1 0 0 0 0 1 0 6 −12 6 0 0 1 1 −4 1 1 0 0 7 −7 0 0

Penurunan rumus

- Hampiri turunan pertama dengan diferensial mundur (*backward differential*):

$$G_3(x) = \frac{\partial f(x,y)}{\partial x} = \frac{f(x,y) - f(x-\Delta x, y)}{\Delta x}$$

$$G_3(y) = \frac{\partial f(x,y)}{\partial y} = \frac{f(x,y) - f(x, y-\Delta y)}{\Delta y}$$



$\Delta x = \Delta y = 1$

- Maka, turunan kedua = turunan pertama diturunkan lagi:

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

$$= G_1(G_3(x)) + G_1(G_3(y))$$

$$= \frac{1}{\Delta x} G_1(f(x,y)) - G_1(f(x-\Delta x, y)) + \frac{1}{\Delta y} G_1(f(x,y)) - G_1(f(x, y-\Delta y))$$

$$= \frac{1}{\Delta x}\{\frac{f(x+\Delta x, y) - f(x,y) - f(x,y) + f(x-\Delta x, y)}{\Delta x}\}$$
$$+ \frac{1}{\Delta y}\{\frac{f(x, y+\Delta y) - f(x,y) - f(x,y) + f(x, y-\Delta y)}{\Delta y}\}$$

$$= \frac{f(x+\Delta x, y) - 2f(x,y) + f(x-\Delta x, y)}{(\Delta x)^2} + \frac{f(x, y+\Delta y) - 2f(x,y) + f(x, y-\Delta y)}{(\Delta y)^2}$$

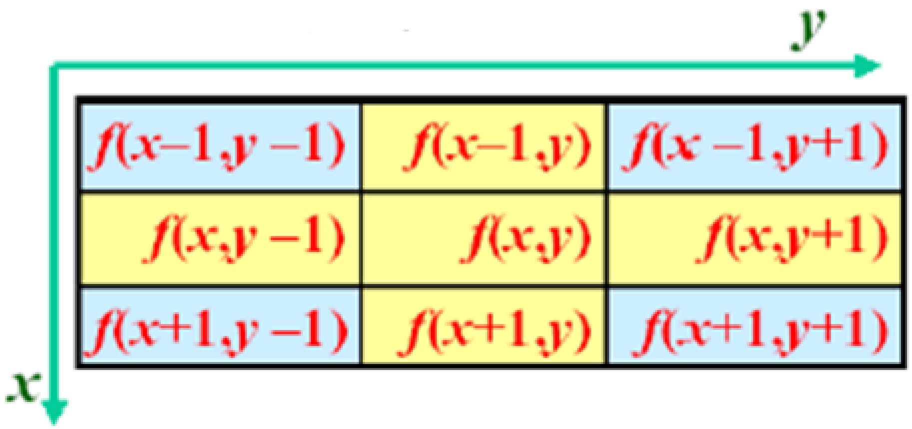
Dengan mengasumsikan $\Delta x = \Delta y = 1$, maka diperoleh:

$$\nabla^2 f(x,y) = f(x+1, y) - 2f(x,y) + f(x-1, y) + f(x, y+1) - 2f(x,y) + f(x, y-1)$$
$$= f(x, y-1) + f(x-1, y) - 4f(x,y) + f(x+1, y) + f(x, y+1)$$

$$
\begin{aligned}
\nabla^2 f(x,y) &= f(x+1,y) - 2f(x,y) + f(x-1,y) + f(x,y+1) - 2f(x,y) + f(x,y-1) \\
&= f(x,y-1) + f(x-1,y) - 4f(x,y) + f(x+1,y) + f(x,y+1) \\
&= 0 \quad + \quad f(x-1,y) + \quad 0 \quad + \\
&\quad f(x,y-1) - 4f(x,y) \quad + \quad f(x,y+1) + \\
&\quad 0 \quad + f(x+1,y) \quad + \quad 0
\end{aligned}
$$

Atau dalam bentuk *mask* konvolusi:

$$
\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}
$$



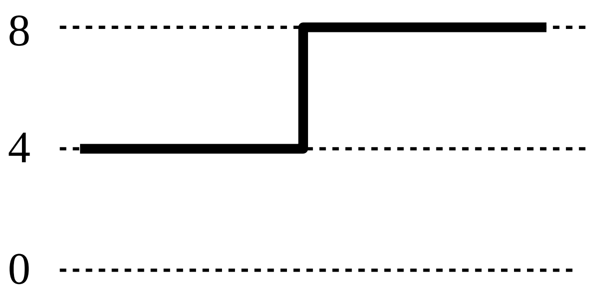
- Contoh berikut memperlihatkan memperlihatkan pendeteksian tepi vertikal dengan operator Laplace:

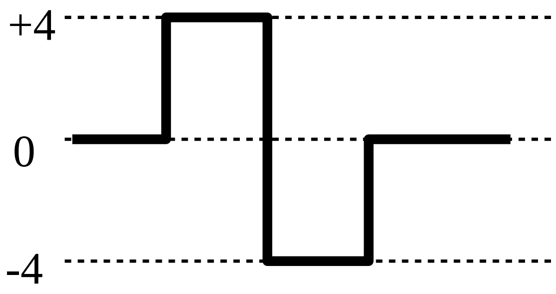
$$\left[\begin{array}{ccc|cccc} 4 & 4 & 4 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 \end{array}\right]$$

(i) Citra semula

$$\left[\begin{array}{cc:c|c:ccc} * & * & * & * & * & * & * \\ * & 0 & +4 & -4 & 0 & 0 & * \\ * & 0 & +4 & -4 & 0 & 0 & * \\ * & 0 & +4 & -4 & 0 & 0 & * \\ * & * & * & * & * & * & * \end{array}\right]$$

(ii) Hasil konvolusi





Satu baris dari hasil pendeteksian tepi: $0 \quad +4 \;|\; -4 \quad 0 \quad 0$

persilangan nol

- Contoh pendeteksian tepi diagonal (miring) dengan operator Laplace:

$$\begin{bmatrix} 4 & 4 & 8 & 8 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 8 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 4 & 8 & 8 & 8 & 8 \\ 4 & 4 & 4 & 4 & 4 & 8 & 8 & 8 \\ 4 & 4 & 4 & 4 & 4 & 4 & 8 & 8 \end{bmatrix}$$

(i) Citra semula

$$\begin{bmatrix} * & * & * & * & * & * & * & * \\ * & 0 & +8 & -4 & 0 & 0 & 0 & * \\ * & 0 & 0 & +8 & -4 & 0 & 0 & * \\ * & 0 & 0 & 0 & +8 & -4 & 0 & * \\ * & * & * & * & * & * & * & * \end{bmatrix}$$

(ii) Hasil konvolusi

- Contoh pendeteksian tepi landai dengan operator Laplace:

$$\begin{bmatrix} 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \end{bmatrix} \qquad \begin{bmatrix} * & * & * & * & * & * & * & * \\ * & 0 & +3 & 0 & -3 & 0 & 0 & * \\ * & 0 & +3 & 0 & -3 & 0 & 0 & * \\ * & 0 & +3 & 0 & -3 & 0 & 0 & * \\ * & * & * & * & * & * & * & * \end{bmatrix}$$

(i) Citra semula (ii) Hasil konvolusi

Satu baris dari hasil pendeteksian tepi: 0 +3 0 –3 0

Pada contoh di atas tidak terdapat persilangan nol; lokasi tepi yang sesungguhnya ditentukan secara interpolasi.

Operator Laplace lainnya:

(a)

0	1	0
1	–4	1
0	1	0

(b)

1	1	1
1	–8	1
1	1	1

(c)

0	–1	0
–1	4	–1
0	–1	0

(d)

–1	–1	–1
–1	8	–1
–1	–1	–1

a	b
c	d

FIGURE 3.39
(a) Filter mask used to implement the digital Laplacian, as defined in Eq. (3.7-4). (b) Mask used to implement an extension of this equation that includes the diagonal neighbors. (c) and (d) Two other implementations of the Laplacian.

- Kadang-kadang diinginkan memberi bobot yang lebih pada *pixel* tengah di antara *pixel* tetangganya:

$$\begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

- Operator Laplace termasuk ke dalam penapis lolos-tinggi sebab jumlah seluruh koefisiennya nol dan koefisiennya mengandung nilai negatif maupun positif.

```
I = imread('camera.bmp');
figure, imshow(I);
H = [0 1 0; 1 -4  1; 0  1  0];
J = uint8(convn(double(I), double(H)));
figure, imshow(J)
```

*

0	1	0
1	–4	1
0	1	0

=

```
I = imread('lada-gray.bmp');
figure, imshow(I);
H = [1 1 1; 1 -8  1; 1  1  1];
J = uint8(convn(double(I), double(H)));
figure, imshow(J)
```

*

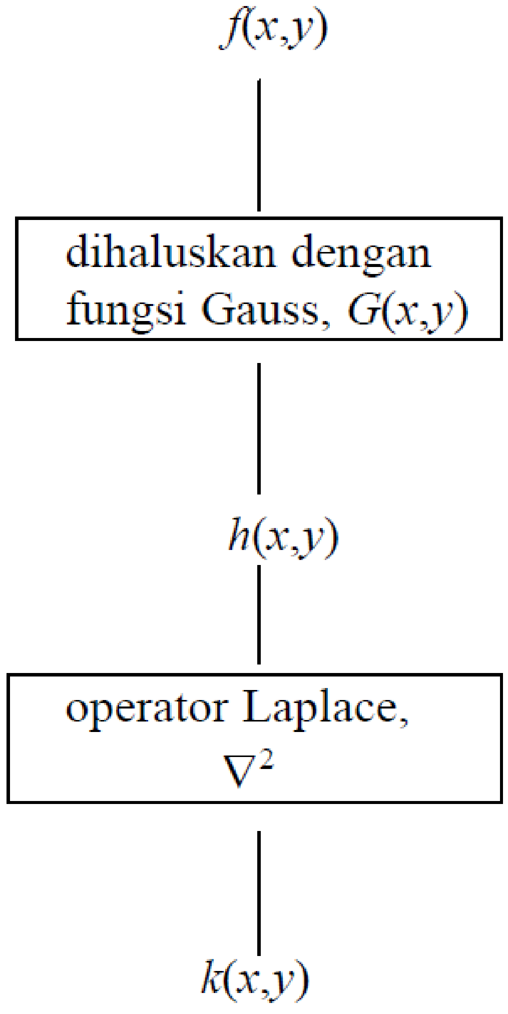
1	1	1
1	−8	1
1	1	1

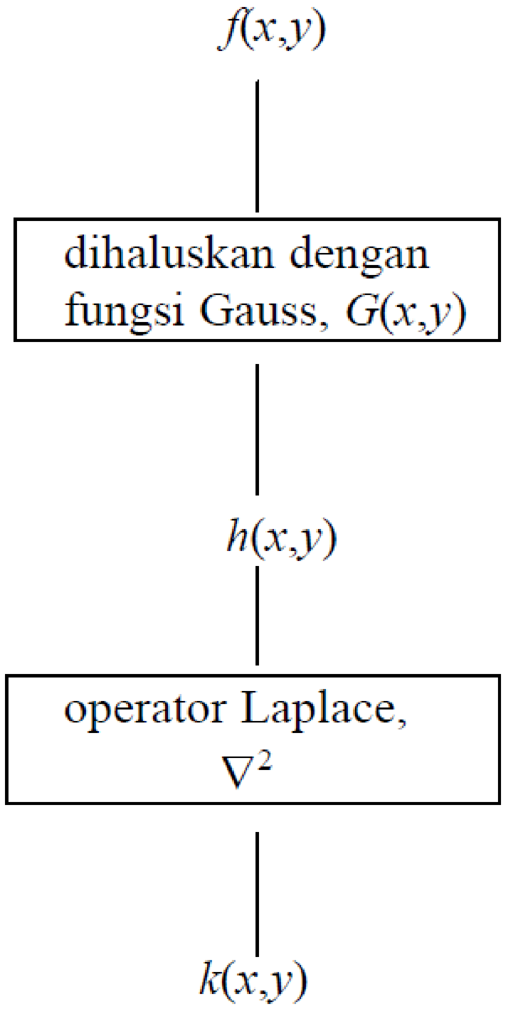
=

Operator Laplace of Gaussian (LoG)

- Kadangkala pendeteksian tepi dengan operator Laplace menghasilkan tepi-tepi palsu yang disebabkan oleh gangguan pada gambar.

- Untuk mengurangi kemunculan tepi palsu, citra ditapis dulu dengan fungsi Gaussian





Berdasarkan gambar di samping:

$$h(x,y) = f(x,y) * G(x,y)$$

$$k(x,y) = \nabla^2 h(x,y)$$

Dapat dibuktikan bahwa

$$\nabla^2 [f(x,y) * G(x,y)] = f(x,y) * \nabla^2 G(x,y)$$

Jadi,

$$k(x,y) = f(x,y) * \nabla^2 G(x,y)$$

yang dalam hal ini,

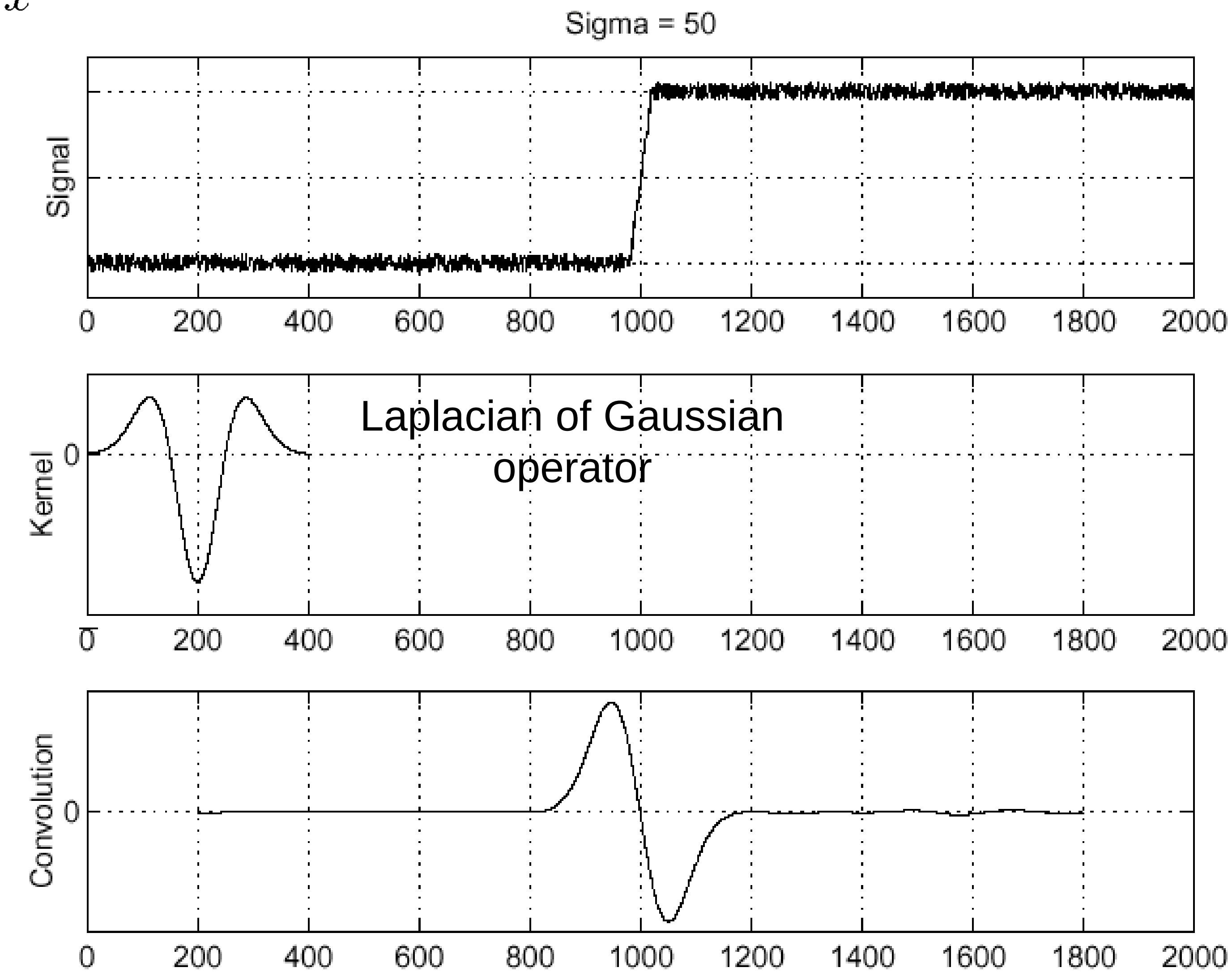
$$\nabla^2 G(x,y) = \left(\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right) e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

- Tinjau $\frac{\partial^2}{\partial x^2}(h \star f)$

f

$\frac{\partial^2}{\partial x^2}h$

$(\frac{\partial^2}{\partial x^2}h) \star f$

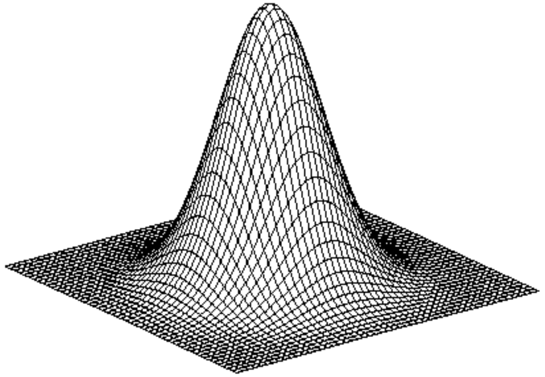


Di mana tepi?

Di tempat *zero-crossing*

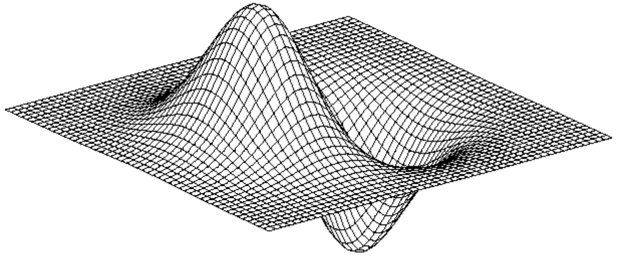
- Fungsi $\nabla^2 G(x,y)$ merupakan turunan kedua dari fungsi Gauss

- Kadang-kadang disebut juga fungsi *Laplacian of Gaussian* (*LoG*) atau fungsi topi orang Mexico (*Mexican Hat*), karena bentuk kurvanya seperti topi Meksiko.



Gaussian

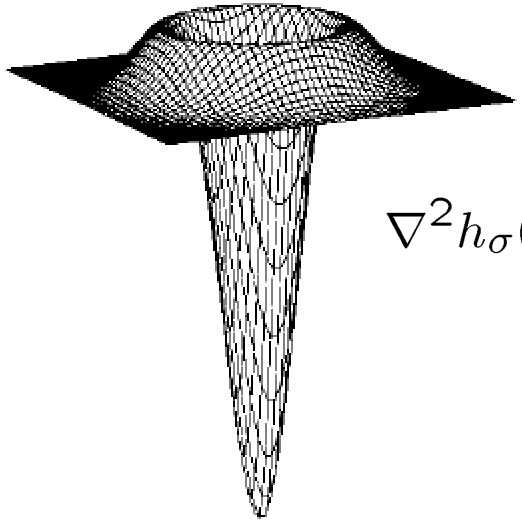
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



$$\nabla^2 h_\sigma(u,v)$$

∇^2 adalah operator **Laplace**: $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

Jadi, untuk mendeteksi tepi dari citra yang mengalami gangguan, kita dapat melakukan salah satu dari dua operasi ekivalen di bawah ini:

1. konvolusi citra dengan fungsi Gauss $G(x,y)$, kemudian lakukan operasi Laplace terhadap hasilnya, atau
2. konvolusi citra langsung dengan penapis *LoG*.

Contoh penapis *LoG* yang berukuran 5×5:

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

Mask konvolusi LoG dengan $\sigma = 1.4$

Citra masukan

Hasil operator Laplace

Hasil operator LoG

Operator gradien lainnya

- Oparator Sobel
- Operator Roberts
- Operator Prewitt
- Opreator 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

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

$$s_y = (a_0 + 2a_1 + a_{22}) - (a_6 + 2a_5 + a_4)$$

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

- 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

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

- Contoh:

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

(i) citra semula

$$\begin{bmatrix} * & * & * & * & * \\ * & 18 & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

(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 dengan T = 12.

Citra:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 4 & 3 \\ 0 & 0 & 2 & 0 & 2 & 4 & 3 & 3 & 2 & 3 \\ 0 & 0 & 1 & 3 & 3 & 4 & 3 & 3 & 3 & 3 \\ 0 & 1 & 0 & 4 & 3 & 3 & 2 & 4 & 3 & 2 \\ 0 & 0 & 1 & 2 & 3 & 3 & 4 & 4 & 4 & 3 \end{bmatrix}$$

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

$$\begin{bmatrix} * & * & * & * & * & * & * & * & * & * \\ * & 4 & 6 & 4 & 10 & 14 & 12 & 14 & 4 & * \\ * & 6 & 8 & 10 & 20 & 16 & 12 & 6 & 0 & * \\ * & 4 & 10 & 14 & 10 & 2 & 4 & 2 & 4 & * \\ * & 2 & 12 & 12 & 2 & 2 & 4 & 6 & 8 & * \\ * & * & * & * & * & * & * & * & * & * \end{bmatrix}$$

Hasil pengambangan dengan T = 12:

$$\begin{bmatrix} * & * & * & * & * & * & * & * & * & * \\ * & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & * \\ * & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & * \\ * & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & * \\ * & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & * \\ * & * & * & * & * & * & * & * & * & * \end{bmatrix}$$

```
%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 dihitung dengan rumus:

$$G(f(x,y)) = \sqrt{P_x^2 + P_y^2}$$

```
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

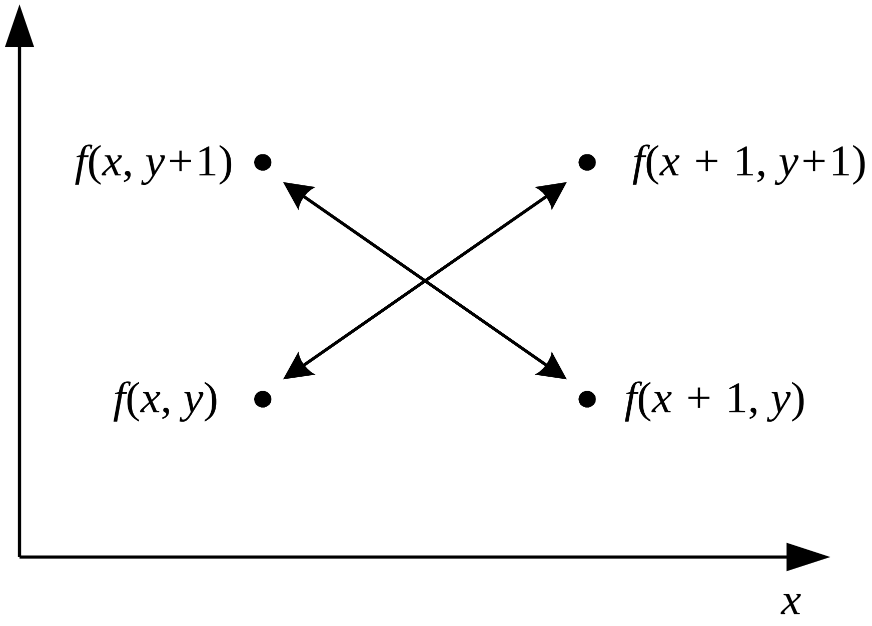
Hasil operator Prewitt

Hasil operator Sobel

Hasil operator Prewitt

Operator Roberts

- Operator Roberts sering disebut juga operator silang



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:

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

Arah tepi dihitung dengan rumus:

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

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

- Contoh berikut ini memeperlihatkan pendeteksian tepi dengan operator Roberts:

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

(i) citra semula (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

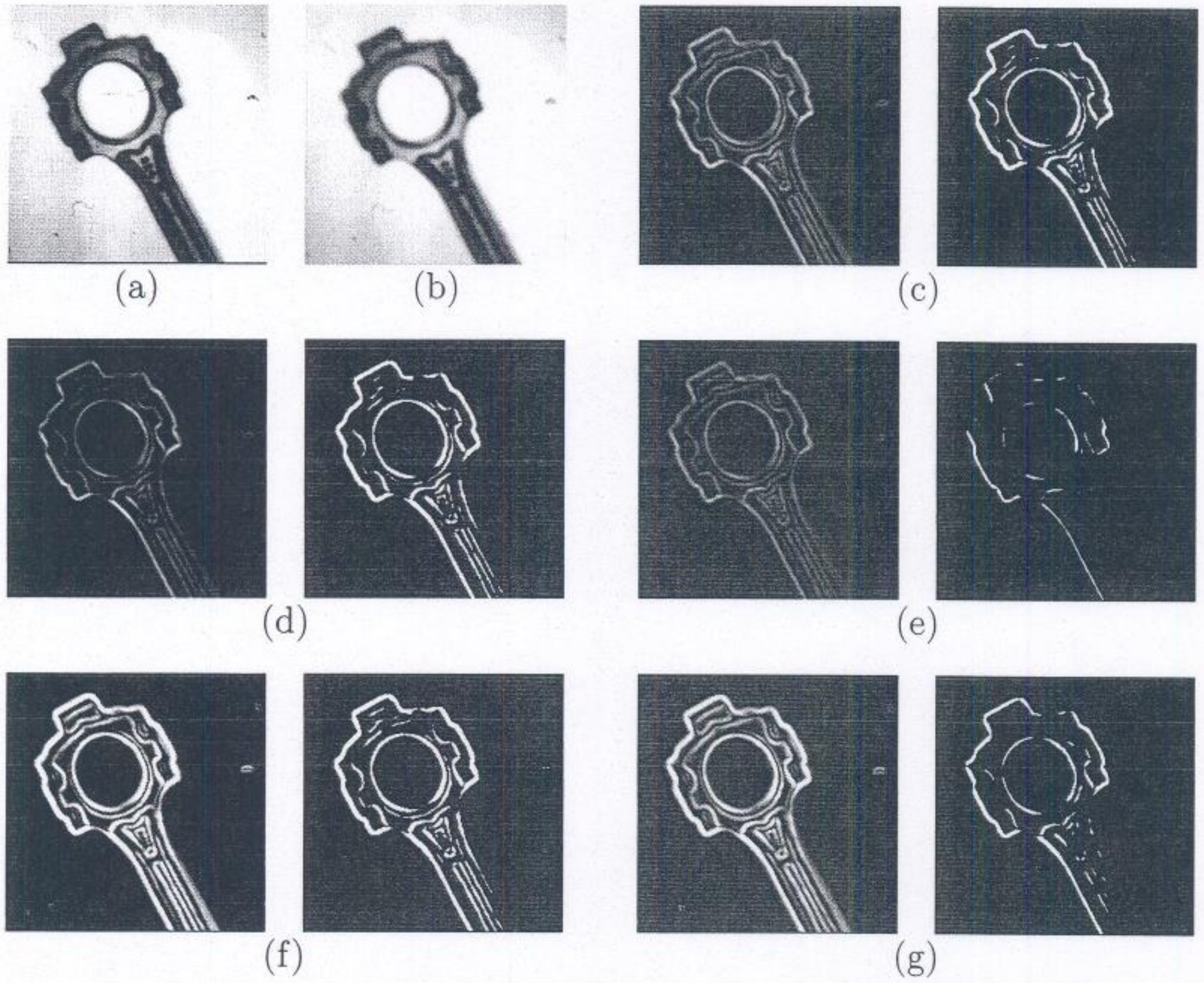


Figure 5.4: A comparison of various edge detectors. (a) Original image. (b) Filtered image. (c) Simple gradient using 1×2 and 2×1 masks, $T = 32$. (d) Gradient using 2×2 masks, $T = 64$. (e) Roberts cross operator, $T = 64$. (f) Sobel operator, $T = 225$. (g) Prewitt operator, $T = 225$.

Operator Canny

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

$G * I$

Langkah-langkah operator Canny:

- Haluskan citra *I* dengan penapis Gaussian:
- Hitung gradien setiap pixel dengan salah satu dari 4 operator sebelumnya (misalnya operator Sobel)
- Jika nilai mutlak gradien suatu pixel melebihi nilai ambang T, maka pixel termasuk pixel tepi.

original image (Lena)

magnitude of the gradient


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

Tugas

Buatlah program pendeteksian tepi dengan menggunakan:

- Operator gradien
- Operator turunan kedua
- Operator Laplace
- Operator LoG
- Operator Sobel
- Operator Prewitt
- Operator Roberts
- Operator Canny