# INF2310 – Digital bildebehandling

## FORELESNING 7

## FILTRERING I BILDEDOMENET – II

Andreas Kleppe

Høypassfiltrering: Bildeforbedring og kantdeteksjon
Gradient-operatorer
Laplace-operatoren og LoG-operatoren
Cannys kantdetektor

G&W: 3.6 og 10.2-10.2.6

# Høypassfiltre

- Slipper gjennom høye frekvenser,
  og demper eller fjerner lave frekvenser.
  - Typisk fjernes den aller laveste frekvensen helt,
    d.v.s. at homogene områder får ut-verdi 0.

- Effekt:
  - Demper langsomme variasjoner, f.eks. bakgrunn.
  - Fremhever skarpe kanter, linjer og detaljer.

- Typiske mål: «Forbedre» skarpheten, detektere kanter.

- Q: Hva skjer med støy?

# Høypassfiltrering med konvolusjon

- Summen av vektene i konvolusjonsfilteret er typisk 0.
  - Q: Hvorfor er dette lurt når vi skal høypassfiltrere?
- Da blir også summen av ut-bildets pikselverdier bli 0.
  - Antar nullutvidelse og bruker alle posisjoner med overlapp.
- => Positive og negative pikselverdier til ut-bildet.
- Ikke alltid en god ide å bruke $|g(x,y)|$.
- For framvisning: Gjør $g(x,y)$ positiv ved å addere med en konstant og skaler resultatet til ønsket intervall.

# Høypassfiltrering med konvolusjon (kursorisk)

- Når summen av vektene i konvolusjonsfilteret er 0, så blir også summen av ut-bildets pikselverdier 0.
  - Antar nullutvidelse og bruker alle posisjoner med overlapp.

$$\sum_{x=-a}^{M-1+a}\sum_{y=-b}^{N-1+b}(h*f)(x,y) = \sum_{s=-a}^{a}\sum_{t=-b}^{b}\sum_{x=-a}^{M-1+a}\sum_{y=-b}^{N-1+b}h(s,t)\,f(x-s,y-t)$$

$$\overset{\text{antar nullutvidelse}}{=} \sum_{s=-a}^{a}\sum_{t=-b}^{b}\sum_{x=s}^{M-1+s}\sum_{y=t}^{N-1+t}h(s,t)\,f(x-s,y-t)$$

$$= \sum_{s=-a}^{a}\sum_{t=-b}^{b}\sum_{m=0}^{M-1}\sum_{y=0}^{N-1}h(s,t)\,f(m,n)$$

$$= \sum_{m=0}^{M-1}\sum_{y=0}^{N-1}\left( f(m,n)\sum_{s=-a}^{a}\sum_{t=-b}^{b}h(s,t)\right) = 0$$

# Punkt-deteksjon

- Eksempel på et høypassfilter; Konvolusjonsfilteret:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Dette filteret kan bl.a. brukes til deteksjon av isolerte punkter:
  - Beregn konvolusjonen av filteret, betegnet *h*, og inn-bildet *f* :

$$g(x,y) = \sum_{s=x-1}^{x+1} \sum_{t=y-1}^{y+1} h(x-s, y-t)\, f(s,t)$$

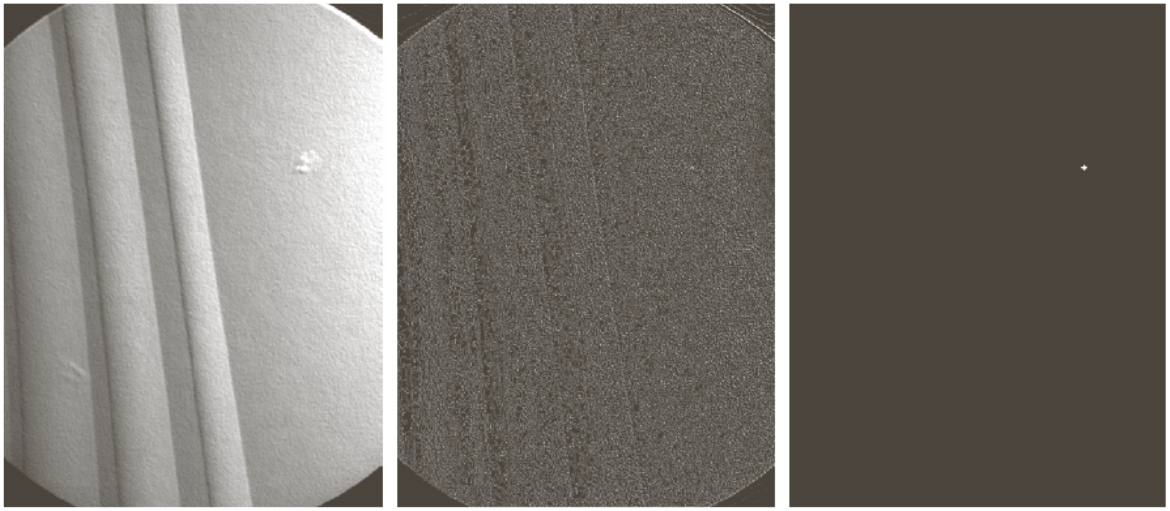
  - Isolerte punkter vil skille seg ut med høy respons (i absoluttverdi).
  - For passende terskel $T>0$ er de detektert punktene: $|g(x,y)| \geq T$

- Hva er responsen i homogene områder?
- Hva med en hellende gråtone-flate?

# Eksempel: Punkt-deteksjon

- **Oppgave**: Deteksjon av pore i turbinblad.

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | −8 | 1 |
| 1 | 1 | 1 |

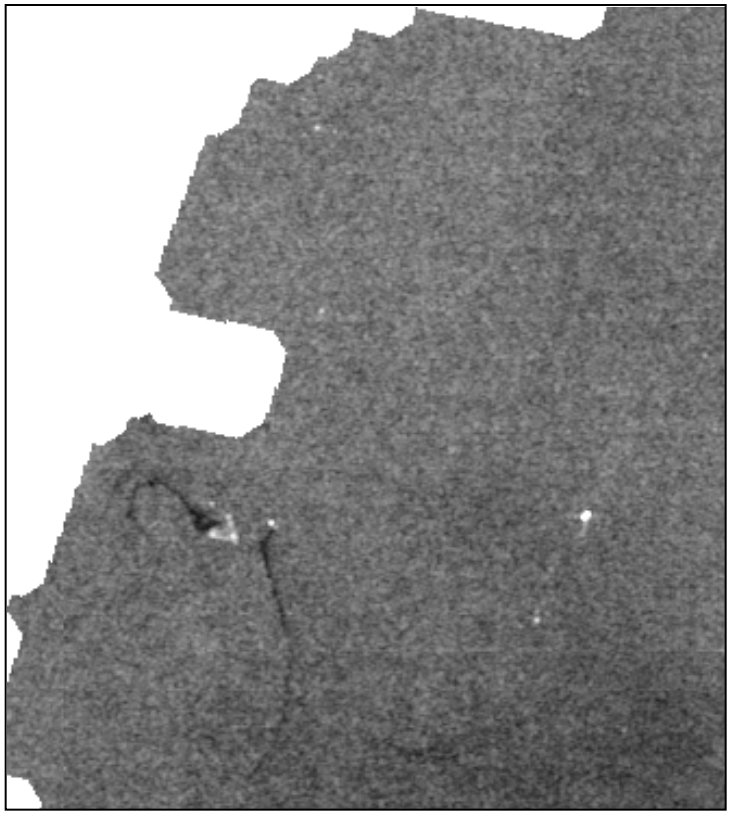


a
b c d

**FIGURE 10.4**
(a) Point detection (Laplacian) mask. (b) X-ray image of turbine blade with a porosity. The porosity contains a single black pixel. (c) Result of convolving the mask with the image. (d) Result of using Eq. (10.2-8) showing a single point (the point was enlarged to make it easier to see). (Original image courtesy of X-TEK Systems, Ltd.)

# Eksempel: Punkt-deteksjon

- **Oppgave**: Deteksjon av skip i radar-bilde over sjø.



- De små lyse punktene er skipene.
- Filteret vil gi flere høye responser for hvert skip. $\longrightarrow \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
  - Og nesten like høy respons i kanter og spesielt hjørner.
- Bedre å bruke et større filter av samme «type»:

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

# Bildeforbedring ved høypassfiltrering

- Konvolusjonsfilteret → $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ kan også brukes til bildeforbedring.

- Grunntanke:
  - Filtreringen detekterer starten og slutten av kanter.
  - Andre områder blir omtrent 0.
  - Derfor: Ved å **addere filtreringen til originalen** får vi sterkere kanter → bildet virker skarpere.

# Eksempel: Bildeforbedring ved høypassfiltrering

- **Oppgave**:
  Øk skarpheten
  i følgende bilde
  av Nordpolen:

- **Løsning**:
  1. Filtrer med → $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
  2. Summer filtreringen og originalen.

G&W fig. 3.38:
← (a) Original
(e) Resultat →

# Bildeforbedring ved høypassfiltrering

- Konvolusjonsfilteret vi har sett på kan uttrykkes slik:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

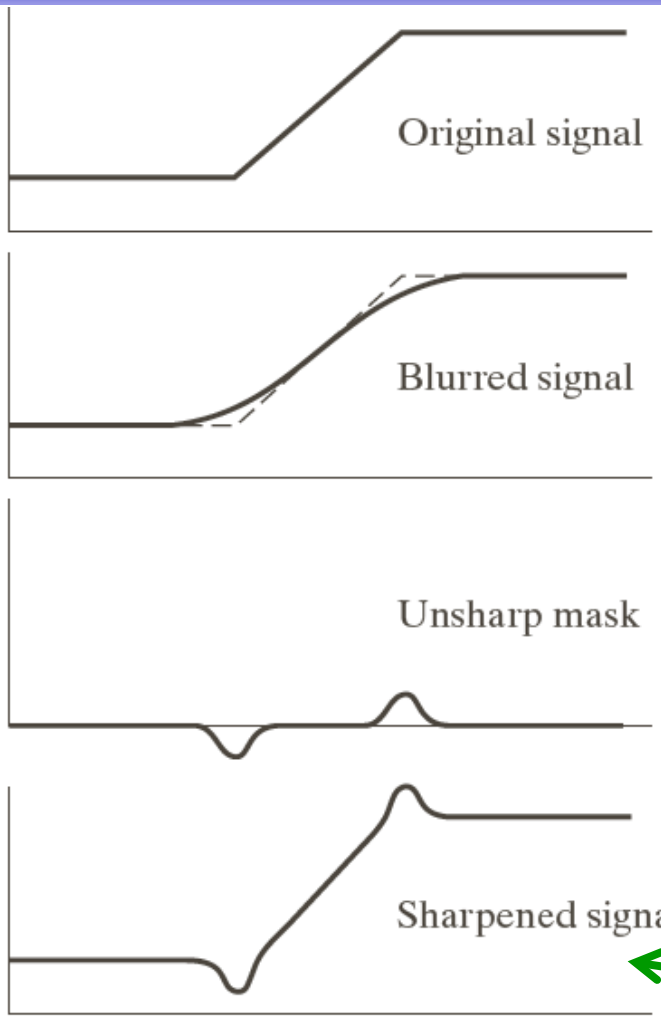
høypass = original – lavpass

Husk:
Konvolusjon er distributiv:
f*(g+h)
= (f*g) + (f*h)

- Bildeforbedringen vil har sett på tilsvarer altså:
  1. Lavpassfiltrer med 3x3-middelverdifilter.
  2. Subtraher resultatet fra originalen.
  3. Adder 9*differansen til originalen.
- Dette er én form for **highboost-filtrering**.

Husk: Konvolusjon er assosiativ ved skalar multiplikasjon:
a(f*g) = f*(ag)

# Unsharp masking og highboost-filtrering




G&W fig. 3.39

- Gitt et bilde (original):
  (til venstre: et 1D-bilde av en rampe)

1. Lavpassfiltrer.
   (til høyre er originalen stiplet)

2. Beregn differansen:
   *original – filtrering*

3. Resultatet er:
   *original + k • differansen*
   - *k* er en positiv konstant.
   - **Unsharp masking**: *k* = 1 (brukt til høyre)
   - **Highboost-filtrering**: *k* > 1

# Eksempel: Unsharp masking

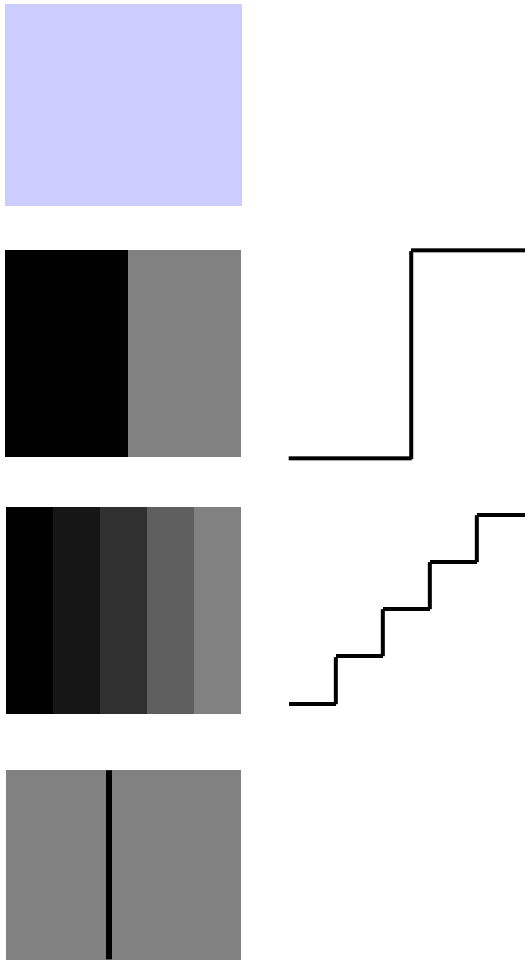
1. Lavpassfiltrering => uskarpt bilde.
   - Bruk f.eks. et middelverdifilter.
2. Subtraher uskarpt bilde fra originalen.
   - Original – Lavpass = Høypass
3. Adder differansen til originalen.
   - Forsterker kanter, resultatet er skarpere enn originalen.

# Motivasjon for kant-deteksjon

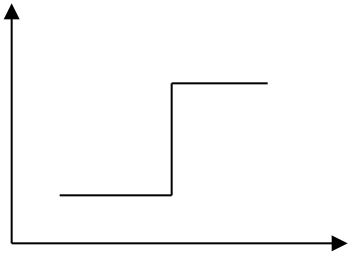
- Det meste av **informasjonen** i et bilde finnes ved **kantene** til objektene/regionene i bildet.
  - Med «kanter» menes her intensitets-kanter, farge-kanter, tekstur-kanter o.s.v.
- Biologiske visuelle systemer er basert på kant-deteksjon.
- Slike systemer arbeider ofte både parallelt og sekvensielt:
  - Alle lokale omgivelser behandles uavhengig av hverandre.
  - Lokale resultat kan være avhengig av tidligere resultater.

# Intensitets-flater, -kanter og -linjer

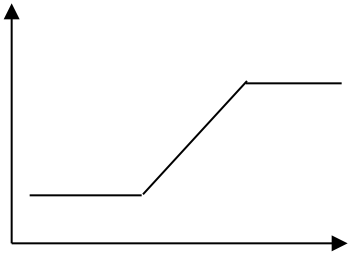
- Homogen flate: Et område der alle pikselverdiene er like.
- Kant: Overgangen mellom to områder med forskjellig middelverdi.
  - Steg-kant: En-piksels overgang.
  - Rampe: Fler-piksels overgang med konstant intensitetsendring (d.v.s. konstant gradient).
- «Kant» brukes også om skillepunktet mellom de to områdene.
  - Forskjellige måter å modellere hvor skiller er.
  - For steg-kanter:
    - Et alternativ er midt mellom nabo-piksler som tilhører forskjellig områder.
    - I segmentering ønsker man typisk å finne første piksel på siden som tilhører objektet.
- Merk at en linje består av to kanter.
  - Hver kan f.eks. være steg-kant eller rampe.
- Idealstrukturer er nyttig for modellering, men i praksis finner vi oftest strukturer som bare ligner.



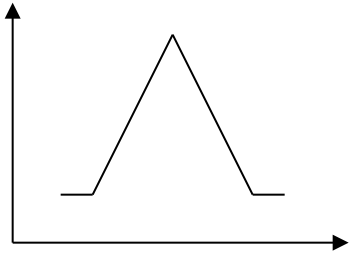
# Kant-typer



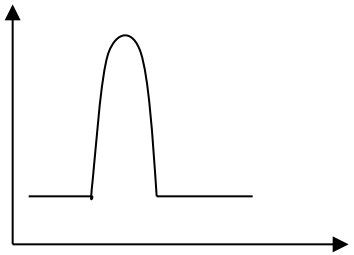
Steg-kant



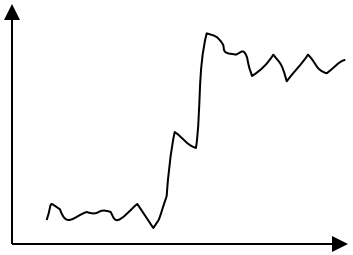
Rampe



Tak-kant



Linje
(litt glattet)



Kant med støy

# Digital derivasjon

- En kant kjennetegnes ved **endring i intensitetsverdi**.
  - Siden en intensitetskant er overgangen mellom to områder med forskjellig middelverdi, så må intensiteten endres i kanten.

- Den **deriverte** av en funksjon *f(x)* er definert som:

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

  og angir stigningstallet til *f* i punktet *x*,
  så */f(x)/* angir hvor mye *f* endrer seg i punktet *x*.

- Den deriverte er **ikke definert** for diskrete funksjoner, men vi kan **tilnærme** den ved å la $h \geq 1$ i definisjonen.
  - Tilnærme v.b.a. **differanser mellom nærliggende piksler**.

# Derivasjon av bilder

- Et digitalt bilde er en to-variabel, diskret funksjon.

- En kontinuerlig funksjon *f(x,y)* kan deriveres m.h.p. *x* og *y*.
  - Kalles å partiell-derivere m.h.p. *x* og *y*.
  - Betegnes henholdsvis $\partial f(x,y)/\partial x$ og $\partial f(x,y)/\partial y$

- Vektoren av de to partiell-deriverte kalles gradienten og betegnes $\nabla f$:

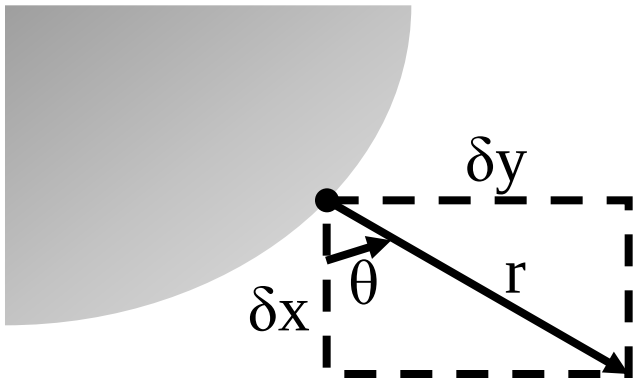
$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

# Gradient i et kontinuerlig bilde

- Gradienten peker i retningen der funksjonen øker mest:
  - Den retningsderiverte til $f$ i retning $\theta$ (d.v.s. langs $r$) er:

$$\frac{\partial f}{\partial r} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial r}$$

$$= \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$




  - Når den retningsderiverte er størst er:

$$\frac{\partial}{\partial\theta}\left(\frac{\partial f}{\partial r}\right) = 0$$

  - D.v.s. vinkelen $\theta_g$ der den retningsderiverte er størst oppfyller:

$$-\frac{\partial f}{\partial x}\sin\theta_g + \frac{\partial f}{\partial y}\cos\theta_g = 0 \quad \Leftrightarrow \quad \frac{\partial f}{\partial y}\cos\theta_g = \frac{\partial f}{\partial x}\sin\theta_g$$

# Gradient i et kontinuerlig bilde

- Gjentar: Når den retningsderiverte er størst er vinkelen $\theta_g$ :

$$\frac{\partial f}{\partial y}\cos\theta_g = \frac{\partial f}{\partial x}\sin\theta_g \quad \Rightarrow \quad \frac{\sin\theta_g}{\cos\theta_g} = \frac{\partial f / \partial y}{\partial f / \partial x} \quad \Rightarrow \quad \tan\theta_g = \frac{g_y}{g_x}$$

- Derfor: Gradienten peker i **retningen** der funksjonen **øker mest**:

$$\theta_g = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

$\theta_g$ angir bare en linje som er parallell med gradienten, men ved å dobbeltderivere *f* kan man vise at funksjonen øker mest med gradientretningen og avtar mest mot gradientretning.

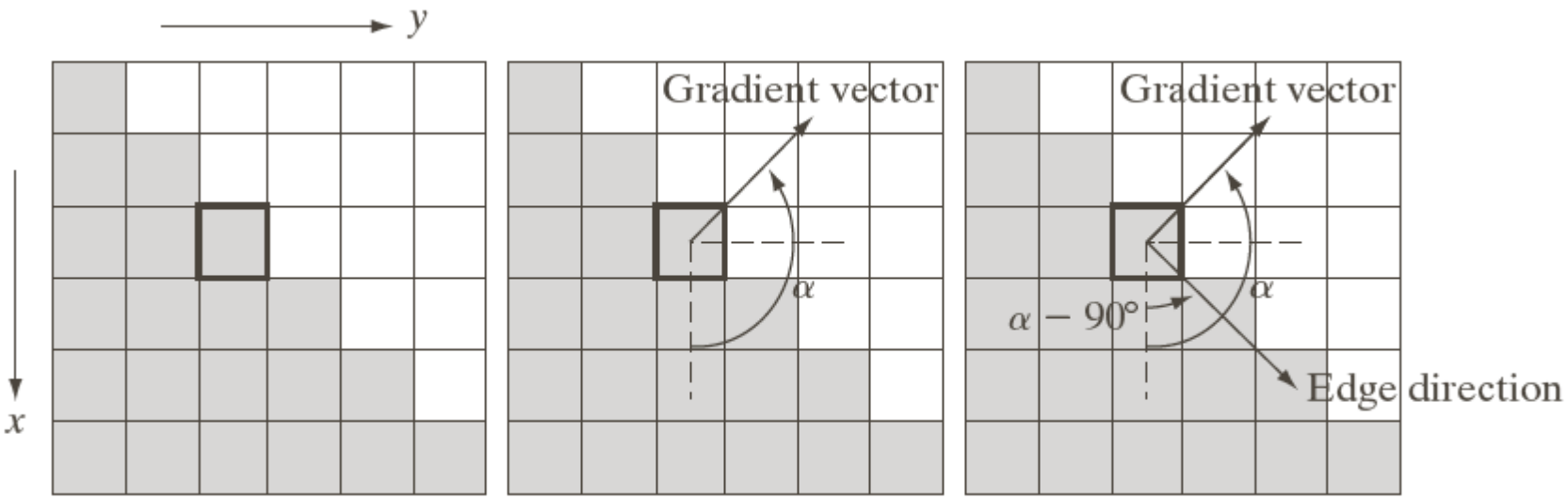
og økningen her, som kalles **gradient-magnituden**, er:

$$\frac{\partial f}{\partial r_g} = \sqrt{g_x^2 + g_y^2}$$

$\partial f/\partial r_g$ er den retningsderiverte i gradientretningen.

# Gradient $\perp$ Kant

- **Gradienten** peker i retningen der funksjonen **øker mest** og **kanten** går **vinkelrett på gradienten**.




a b c

**FIGURE 10.12** Using the gradient to determine edge strength and direction at a point. Note that the edge is perpendicular to the direction of the gradient vector at the point where the gradient is computed. Each square in the figure represents one pixel.

# Digitale gradient-tilnærminger

- Vi ønsker å tilnærme gradienten med **differanser mellom nærliggende piksler**.

- Til dette kan vi bruke to konvolusjonsfiltre som tilnærmer hver sin gradient-komponent.
  - To slike konvolusjonsfiltre kalles en **gradient-operator**.
  - Konvolusjonsfiltrene betegnes ofte som $h_x$ og $h_y$
  - F.eks. $h_x$ tilnærmer den partiell-derivert i $x$-retning ved å beregne differansen i vertikal retning av nærliggende piksler.

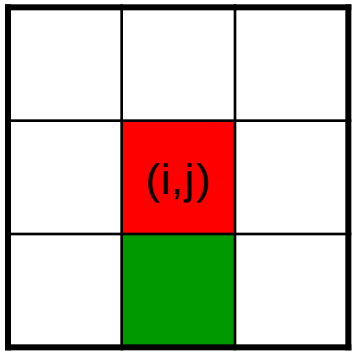
- Mange muligheter!

# Digitale gradient-tilnærminger

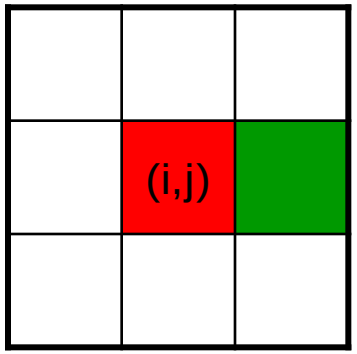
- Asymmetrisk 1D-operator:

  $g_x(i,j) = f(i+1,j) - f(i,j)$

  $g_y(i,j) = f(i, j+1) - f(i,j)$




- Definisjonene er gitt slik at gradient-komponentene er positive for en kant der intensiteten øker nedover og fra venstre mot høyre i bildet.



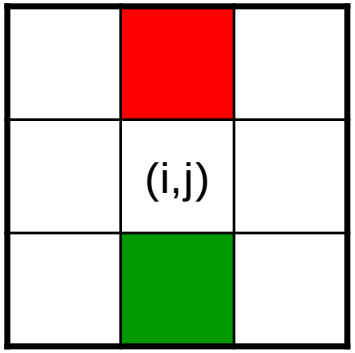

- Problemer med denne operatoren:
  - Hver av gradient-estimatene refererer til et punkt midt mellom to piksler.
  - *x*- og *y*-estimatet refererer ikke til samme punkt i bildet.

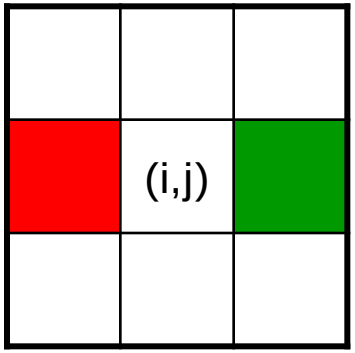
# Digitale gradient-tilnærminger

- Symmetrisk 1D-operator:

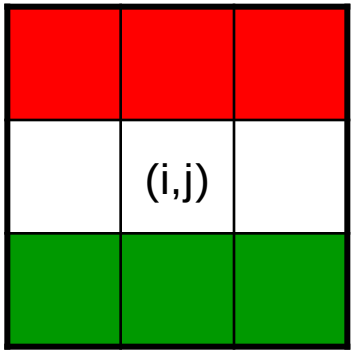
  $g_x(i,j) = f(i+1,j) - f(i-1,j)$

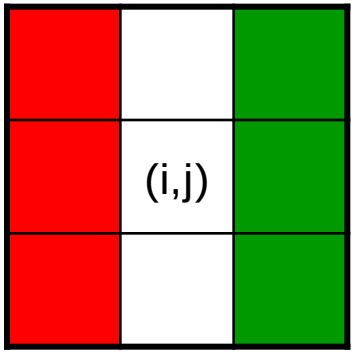
  $g_y(i,j) = f(i,j+1) - f(i,j-1)$




  - **Gradient-estimatene refererer nå til (i,j).**
- Problem: Operatoren er veldig følsom for støy.
  - Støy kan bli detektert som kanter.
- Løsning: Gjør det samme for tre symmetriske par:
  - **Gradient-estimatene blir mer robuste mot støy i bildet.**



# Digitale gradient-tilnærminger

- Vi gjorde 1D-operatoren symmetrisk for at gradient-estimatene skal referere til samme piksel.

- En positiv bieffekt er at vi innfører en mild glatting i retningen for gradient-tilnærmingen:

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

- For å gjøre operatoren mer støy-robust innfører vi deretter en glatting vinkelrett på denne retningen.

# Gradient-operatorer

- Asymmetrisk 1D-operator:

$$h_x(i,j) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

  - Også kalt «pixel difference»-operatoren.

- Symmetrisk 1D-operator:

$$h_x(i,j) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

  - Også kalt «separated pixel difference»-operatoren.

- Roberts-operatoren (også kalt Roberts kryssgradient-operator):

$$h_x(i,j) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

NB: Vi angir konvolusjonsfiltre i den tanke at de skal brukes til konvolusjon.

G&W angir filtermasker som skal brukes til korrelasjon.

Filtrene vil derfor avvike med en 180 graders rotasjon.

# Gradient-operatorer

- Prewitt-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

- Sobel-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

- Frei-Chen-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$$

NB: Vi angir konvolusjonsfiltre i den tanke at de skal brukes til konvolusjon.

G&W angir filtermasker som skal brukes til korrelasjon.

Filtrene vil derfor avvike med en 180 graders rotasjon.

# Separasjon av symmetriske gradient-operatorer

- Separasjon av Prewitt-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

- Separasjon av Sobel-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

- Separasjon av Frei-Chen-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 & \sqrt{2} & 1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 \\ \sqrt{2} \\ 1 \end{bmatrix}$$

# Symmetriske gradient-operatorer

- Operatoren gjøres mindre følsom for støy ved å lavpassfiltrere i en retning og derivere i den ortogonale retningen.
  - Sees tydelig fra separasjonene.
- Eksempler: Prewitt, Sobel, Frei-Chen.
- Prewitt er mer følsom for horisontale og vertikale enn for diagonale kanter.
- Det motsatte er tilfelle for Sobel.
- Frei-Chen gir samme gradient-magnitude om kanten ligger langs aksene eller diagonalt.

Prewitt

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

Sobel

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

Frei-Chen

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

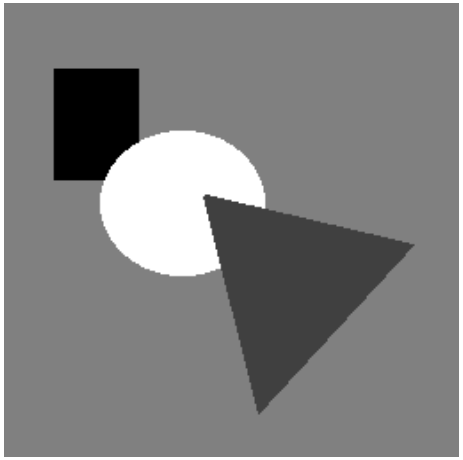
# Gradient-beregning

- Vi finner de vertikale kantene:
  - Beregn: $g_x(i,j) = h_x * f(i,j)$
- Vi finner de horisontale kantene:
  - Beregn: $g_y(i,j) = h_y * f(i,j)$
- Beregn gradient-magnitude og -retning:

$$M(i,j) = \sqrt{g_x^2(i,j) + g_y^2(i,j)}$$ **Gradient-magnitude**

$$\theta(i,j) = \tan^{-1}\left(\frac{g_y(i,j)}{g_x(i,j)}\right)$$ **Gradient-retning**

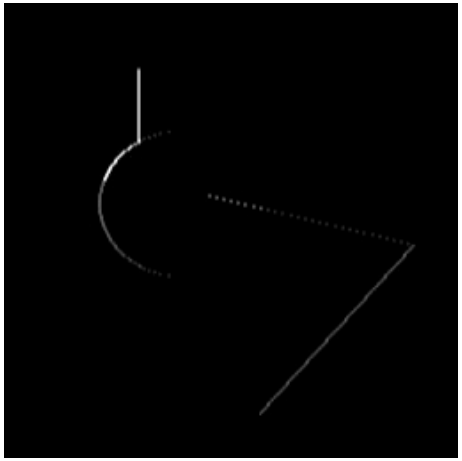
# Eksempel: Gradient-beregning med Sobel-operatoren



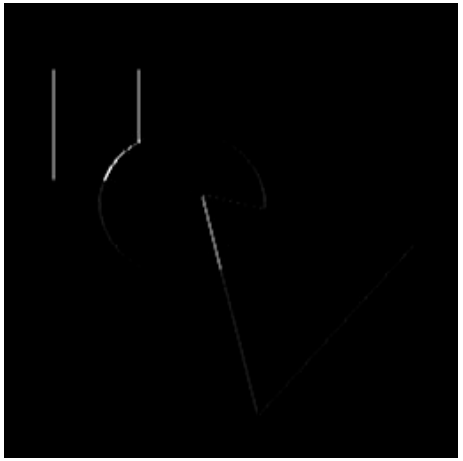
Inn-bilde f

$g_x = f*h_x$

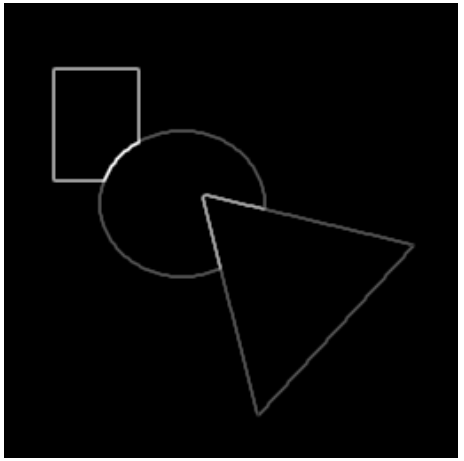
$g_x^2$



$g_y = f*h_y$



$g_y^2$



$(g_x^2+g_y^2)^{1/2}$

Merk:
Hvert bilde er skalert ved å dele på sitt maksimum.
De negative verdiene i $g_x$ og $g_y$ er satt til 0.

# Større gradient-operatorer

- De symmetriske gradient-operatorene kan gjøres mer støy-robuste ved å bygge inn mer lavpassfiltrering.

- Eksempel: Følgende 5×5-Sobel-operator:

$$h_x(i,j) = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 2 & 8 & 12 & 8 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & -8 & -12 & -8 & -2 \\ -1 & -4 & -6 & -4 & -1 \end{bmatrix} \quad h_y(i,j) = \begin{bmatrix} 1 & 2 & 0 & -2 & -1 \\ 4 & 8 & 0 & -8 & -4 \\ 6 & 12 & 0 & -12 & -6 \\ 4 & 8 & 0 & -8 & -4 \\ 1 & 2 & 0 & -2 & -1 \end{bmatrix}$$

er resultatet av konvolusjonene:

$$h_x(i,j) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

# Implementasjoner av gradient-operatorer

- Som vanlig lurt å utnytte separabilitet.

- For 5x5-Sobel-operatoren på forrige foil:
  - Med 5x5-filtrene kreves 50 multiplikasjoner.
  - Ved bruk av de fire 3x3-filtrene kreves 36 multiplikasjoner.
  - Finnes mange måter man dele opp på, men det raskeste blir å separere 5x5-filtrene direkte:

$$h_x(i,j) = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 0 \\ -2 \\ -1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 2 & 0 & -2 & -1 \end{bmatrix} * \begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix}$$

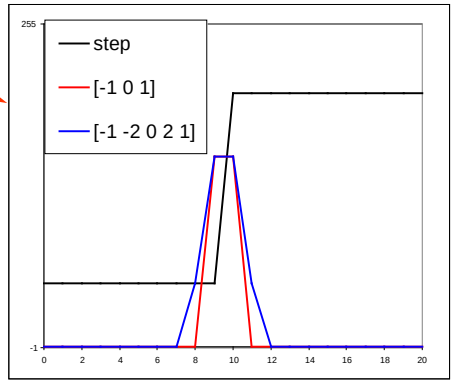
  Disse krever bare 20 multiplikasjoner.

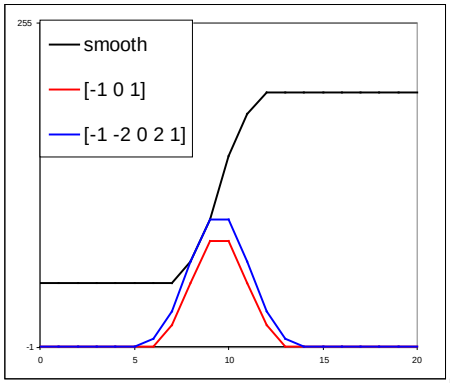
# Gradient til kant-deteksjon

- Gradient-magnituden indikerer **styrken av kanten** i et piksel.
- Kantene i gradient-magnituden blir typisk brede / tykke.
  - Kan være uønsket.
  - Bredden avhenger av størrelsen på filtrene og bredden på kanten i bildet.
- Mulige fremgangsmåter for å finne eksakt, tynn kant:
  - Terskle grad.mag. og tynne.
  - Finne maksimum i grad.mag. / bruke den andrederiverte.

# Gradient til kant-deteksjon

- Gradient-magnituden har «bred respons», men vi ønsker eksakt, tynn kant.
- For en steg-kant:
  - Bredden på responsen er avhengig av størrelsen på filteret.
- For en bred kant (glattet med [1 2 3 2 1]):
  - Bredden på responsen er avhengig av bredden på kanten.
- **Maksimumet er likt og fornuftig lokalisert!**
  - Bruke den andrederiverte til å finne maksimumene?



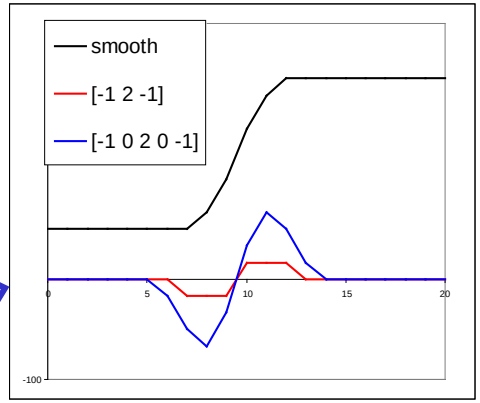



# Laplace-operatoren

- Laplace-operatoren er gitt ved:

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

  - Den endrer fortegn der $f$ et vendepunkt.
  - $\nabla^2 f = 0$ markerer kant-posisjon.
  - $|\nabla^2 f|$ har to ekstremverdier per kant; på starten og på slutten av kanten.
    - Derfor brukte vi den tidligere til å forbedre bildeskarpheten!




- Kantens eksakte posisjon er **nullgjennomgangen**.
- Dette gir tynne kanter.
- Vi finner bare kant-posisjoner, ikke kant-retninger.

# 1D-Laplace-operator

| Kontinuerlig | Digitalt |
|---|---|
| $f(x)$ | $f(i)$ |
| $f'(x)$ | $f(i+1)-f(i)$ |
| $f''(x)= \nabla^2 f$ | $f'(i+1)-f'(i)$<br>$= [f(i+2)-f(i+1)]$<br>$- [f(i+1)-f(i)]$<br>$= f(i+2)-2f(i+1)+f(i)$ |

- I 1D er $\nabla^2 f$ ekvivalent med den andrederiverte.
- Av symmetri-hensyn sentrerer vi om *i*.
- Dessuten bytter vi fortegn av konvensjon.
- Altså får vi:

  $\nabla^2 f = -f(i-1) + 2f(i) - f(i+1)$

# 2D-Laplace-operator

- Anvend 1D-Laplace-tilnærmingen i begge retninger:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$
$$\approx -f(i-1, j) + 2f(i, j) - f(i+1, j)$$
$$-f(i, j-1) + 2f(i, j) - f(i, j+1)$$

- Dette kan beregnes ved å konvolvere *f(i,j)* med

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

# Full 3x3-Laplace-operator

- Hvis vi i tillegg anvender 1D-Laplace-tilnærmingen langs begge diagonaler, får vi det som kan beregnes med:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Dette konvolusjonsfilteret kjenner vi igjen.
  - Punkt-deteksjon.
  - Øke bildeskarpheten.

# Laplace på andregradspolynom

- La de lokale intensitetene omkring *(x,y)* være modellert ved andregradspolynomet (*(m,n)* er koordinater relativt til *(x,y)*):

$$f(m,n) = k_1 + k_2 m + k_3 n + k_4 m^2 + k_5 mn + k_6 n^2$$

- I et 3x3-naboskap rundt *(x,y)* har vi da intensitetene:

| | | |
|---|---|---|
| $k_1-k_2-k_3+k_4+k_5+k_6$ | $k_1-k_3+k_6$ | $k_1+k_2-k_3+k_4-k_5+k_6$ |
| $k_1-k_2+k_4$ | $k_1$ | $k_1+k_2+k_4$ |
| $k_1-k_2+k_3+k_4-k_5+k_6$ | $k_1+k_3+k_6$ | $k_1+k_2+k_3+k_4+k_5+k_6$ |

- Den korrekte Laplace-verdien er gitt ved:

$$-\nabla^2 f(x,y) = -\left(\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}\right) = \boxed{-2(k_4 + k_6)}$$

- Både 4-nabo- og 8-nabo-Laplace-operatoren (høyre) gir korrekt estimat!

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad \frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

# Sobel vs Laplace

Sobel-filtrering
=> bred kant

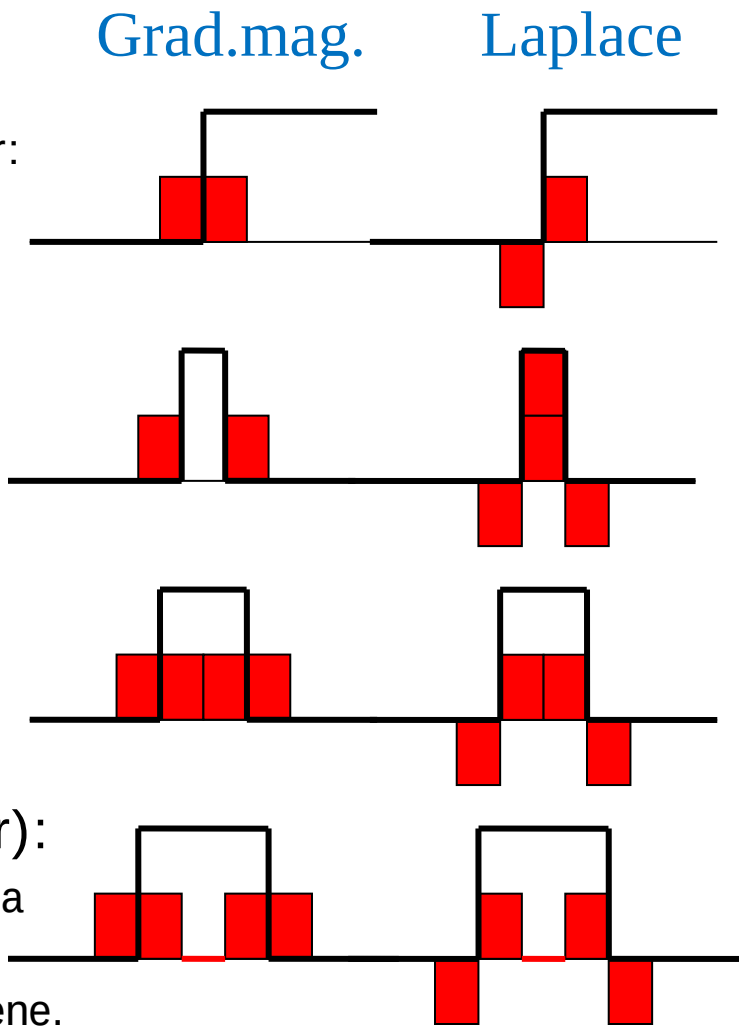
Laplace-filtrering
=> dobbelt-kant

# Steg-kanter og steg-kantede linjer

- Merk:
  - «Grad.mag.» bruker symmetrisk 1D-operator:
    $f(i+1) - f(i-1)$
  - «Laplace» er 1D-Laplace-operatoren:
    $-f(i-1) + 2f(i) - f(i+1)$

- For steg-kanter:
  - Gradientmagnituden gir samme respons i første piksel utenfor og første piksel innenfor kanten.
  - Laplace gir responser med motsatt fortegn, og riktig kant i null-gjennomgang.

- På tvers av en linje (med to steg-kanter):
  - 1-piksels linje gir respons på hver side av linja med gradientmagnituden – ingen respons i.
  - Laplace gir riktige kanter i nullgjennomgangene.



# Andre Laplace-operatorer

- Kan bl.a. finnes ved å bruke gradient-operatorer.

- Eksempel: Bruker 3x3-Sobel-operatoren:

$$-\nabla^2_{5\times5} = -\left(\begin{bmatrix}1&2&1\\0&0&0\\-1&-2&-1\end{bmatrix} * \begin{bmatrix}1&2&1\\0&0&0\\-1&-2&-1\end{bmatrix} + \begin{bmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{bmatrix} * \begin{bmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{bmatrix}\right)$$

$$= -\left(\begin{bmatrix}1&4&6&4&1\\0&0&0&0&0\\-2&-8&-12&-8&-2\\0&0&0&0&0\\1&4&6&4&1\end{bmatrix} + \begin{bmatrix}1&0&-2&0&1\\4&0&-8&0&4\\6&0&-12&0&6\\4&0&-8&0&4\\1&0&-2&0&1\end{bmatrix}\right) = \begin{bmatrix}-2&-4&-4&-4&-2\\-4&0&8&0&-4\\-4&8&24&8&-4\\-4&0&8&0&-4\\-2&-4&-4&-4&-2\end{bmatrix}$$

# Implementasjon av Laplace-operatorer

- Generelt ikke separable.
- Flere Laplace-operatorer kan likevel deles opp i 1D-operasjoner.
- Eksempel: Laplace-operatoren fra forrige foil:

$$-\nabla^2_{5\times5} = \begin{bmatrix} -2 & -4 & -4 & -4 & -2 \\ -4 & 0 & 8 & 0 & -4 \\ -4 & 8 & 24 & 8 & -4 \\ -4 & 0 & 8 & 0 & -4 \\ -2 & -4 & -4 & -4 & -2 \end{bmatrix}$$

er ikke separabel, men kan deles opp i 1D-operasjonene:

$$\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \end{bmatrix}^T + \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}^T$$

Krever da 37 operasjoner i stedet for 49 (v.b.a. 5x5-filteret).

# Fra Laplace til LoG

- Vi gjorde gradient-operatorene støy-robuste ved å bygge inn en lavpassfiltrering.
  - Eksempel: 3x3-Sobel-operatoren:

$$h_x(i,j) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \qquad h_y(i,j) = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

- Vi kan gjøre det samme med en Laplace-operator.
- Det er vanlig å bygge inn et Gauss-filter $G$ med gitt $\sigma$:

$$\nabla^2 * (G * f) = (\nabla^2 * G) * f = LoG * f$$

Husk:
Konvolusjon er assosiativ:
(f*g)*h = f*(g*h)

  - $\nabla^2$ er en Laplace-operator og $LoG = \nabla^2 * G$ er en **Laplacian-of-Gaussian-operator**.

# En 7×7-LoG-operator

- Konvolverer vi 3x3-Gauss-tilnærmingen:

$$G_{3\times 3} = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}^T$$

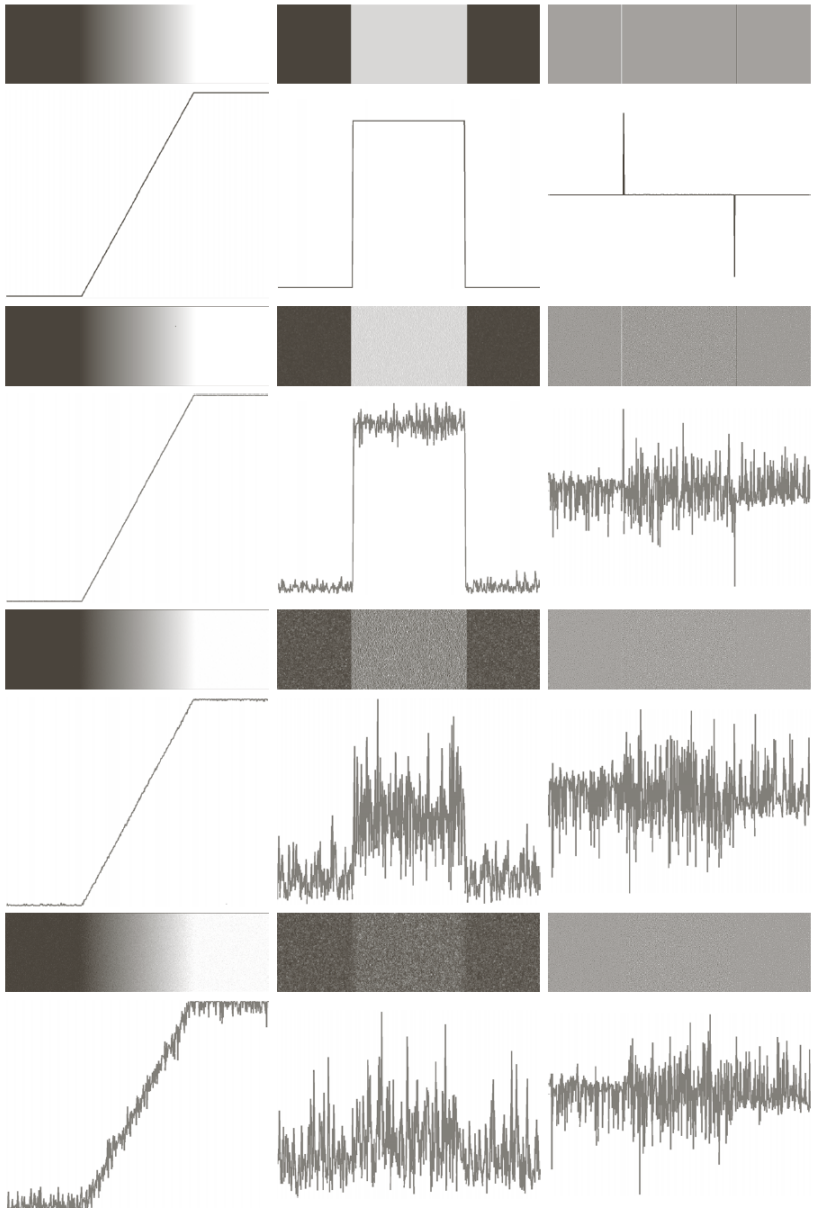
med Laplace-operatoren vil fikk ved å bruke 3x3-Sobel-operatoren:

$$-\nabla^2_{5\times 5} = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \end{bmatrix}^T + \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \end{bmatrix} * \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}^T$$

for å få følgende 7x7-LoG-operator:

$$LoG_{7\times 7} = -\nabla^2_{5\times 5} * G_{3\times 3} = \begin{bmatrix} -2 & -8 & -14 & -16 & -14 & -8 & -2 \\ -8 & -24 & -24 & -16 & -24 & -24 & -8 \\ -14 & -24 & 30 & 80 & 30 & -24 & -14 \\ -16 & -16 & 80 & 160 & 80 & -16 & -16 \\ -14 & -24 & 30 & 80 & 30 & -24 & -14 \\ -8 & -24 & -24 & -16 & -24 & -24 & -8 \\ -2 & -8 & -14 & -16 & -14 & -8 & -2 \end{bmatrix}$$

# Også lavpassfiltrere?



Signal Derivert Laplace

**FIGURE 10.11** First column: Images and intensity profiles of a ramp edge corrupted by random Gaussian noise of zero mean and standard deviations of 0.0, 0.1, 1.0, and 10.0 intensity levels, respectively. Second column: First-derivative images and intensity profiles. Third column: Second-derivative images and intensity profiles.

# To måter å lage LoG-operatorer

- Ofte lages og implementeres en LoG-operator som konvolusjonen av en Laplace-operator og et Gauss-filter.

- Ofte defineres en **LoG-operator** som en **sampling av LoG-funksjonen**, som er resultatet av å anvende Laplace-operatoren på Gauss-funksjonen i det **kontinuerlige domenet**.

- Disse fremgangsmåtene gir generelt ikke *helt* like filtre, men begge resulterer i filtre vi kaller LoG-operatorer.

# Utledning av LoG-funksjonen

2D - Gauss - funksjon :

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$$

Deriverer m.h.p. $x$ :

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

Deriverer m.h.p. $y$ :

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

Andrederivert m.h.t. $x$ :

$$\frac{\partial^2 G}{\partial x^2}=-\frac{1}{2\pi\sigma^4}\left(1-\frac{x^2}{\sigma^2}\right)e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$$

Andrederivert m.h.t. $y$ :

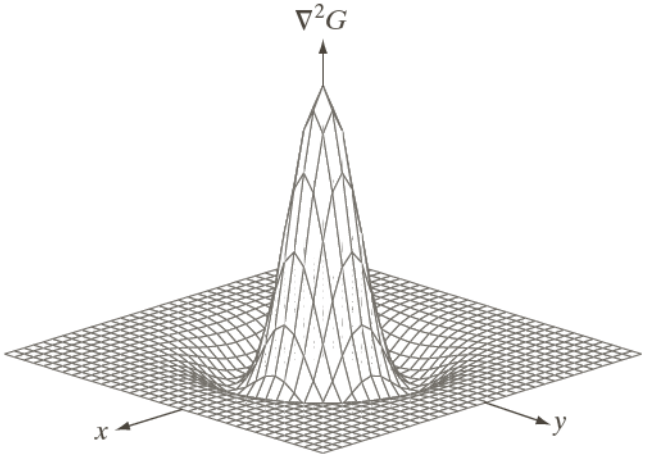
$$\frac{\partial^2 G}{\partial y^2}=-\frac{1}{2\pi\sigma^4}\left(1-\frac{y^2}{\sigma^2}\right)e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$$

Laplace er summen av disse :

$$-\nabla^2 G=\frac{1}{2\pi\sigma^4}\left(2-\frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$$

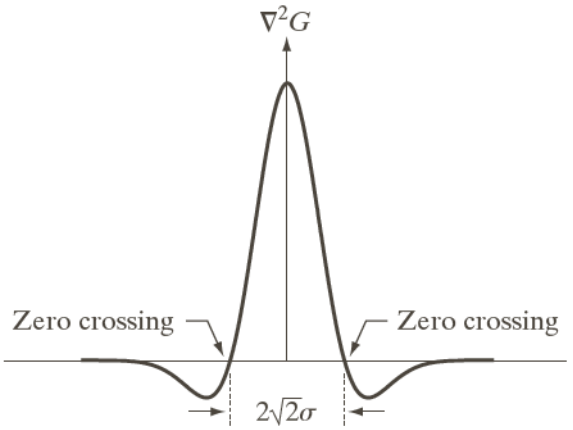
# LoG-funksjonen

- Kalles noen ganger «Mexican hat»-operatoren.




a b
c d

**FIGURE 10.21**
(a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d) $5 \times 5$ mask approximation to the shape in (a). The negative of this mask would be used in practice.

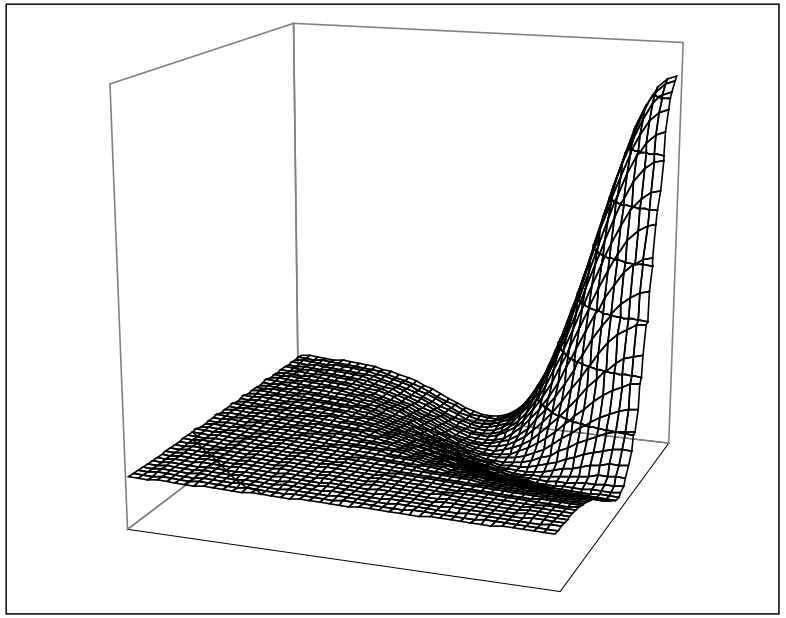



| | | | | |
|---|---|---|---|---|
| 0 | 0 | −1 | 0 | 0 |
| 0 | −1 | −2 | −1 | 0 |
| −1 | −2 | 16 | −2 | −1 |
| 0 | −1 | −2 | −1 | 0 |
| 0 | 0 | −1 | 0 | 0 |

G & W definerer $\nabla^2 G$ som LoG - funksjonen, men vi definerer den som $-\nabla^2 G$

# LoG-operator fra LoG-funksjonen

- Får en LoG-operator ved å sample (en variant av) LoG-funksjonen $\longrightarrow$ $-\nabla^2 G = \frac{1}{2\pi\sigma^4}\left(2 - \frac{x^2+y^2}{\sigma^2}\right)e^{-\frac{\left(x^2+y^2\right)}{2\sigma^2}}$ for heltallige *x* og *y*.
  - Vi bryr oss denne gangen ikke om implementasjonsdetaljene; justering slik at vektene summerer seg til 0 og eventuell heltallstilnærming av vektene.
- $\sigma$ er standard-avviket til Gauss-en og er en parameter.
- I de fleste tilfeller er størrelsen av operatoren $\approx 3w \approx 8.5\sigma$
  - LoG-funksjonen omtrent 0 utenfor dette området.
  - Den positive toppen til minus LoG-funksjonen kalles *kjernen* og $w = 2\sqrt{2}\,\sigma$ er bredden av denne.

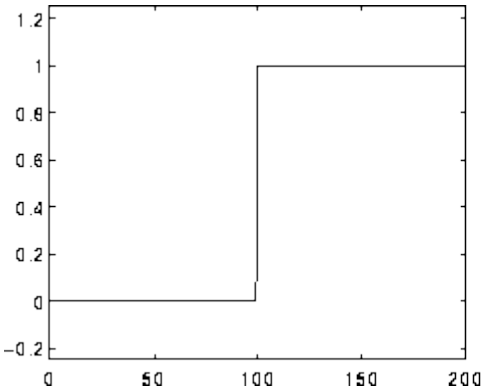


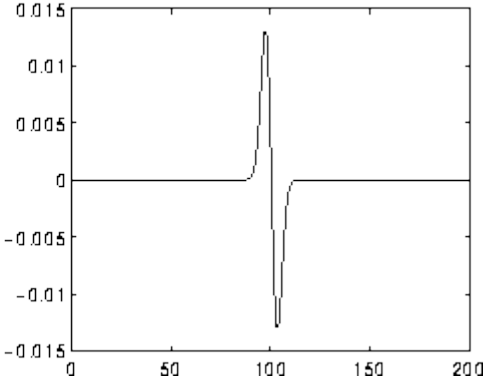
Et kvadrant av LoG-funksjonen

# Bruk av LoG-operatorer

- Laplace-operator detekterer kanter, men er følsomme for støy.
- Ofte må man lavpassfiltrere før Laplace-filtrering.
- En LoG-operator gjør begge disse operasjonene i ett.
- Fungerer ellers som en Laplace-operator:
  - I homogene områder vil en LoG-operator gi respons 0.
  - Den vil ha positiv respons på den ene siden av kanten, er ideelt sett 0 i kantskillet, og har negativ respons på den andre siden.
  - **Nullgjennomganger angir kanter**.

Intensitetsprofil av en steg-kant



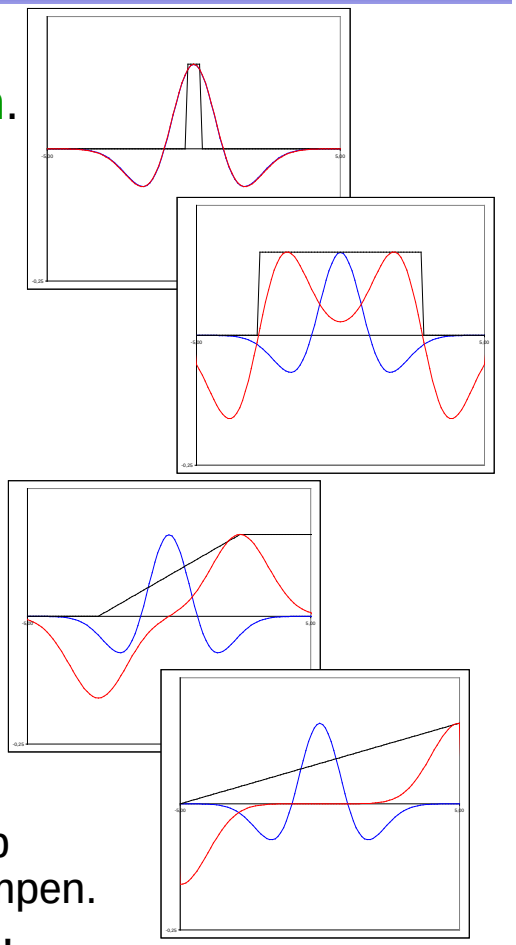

LoG-filtreringen av profilen

NB: Den anvendte LoG-operatoren har negativ verdi i senterposisjonen.

- PS: LoG-operatorer med varierende $\sigma$ fungerer også godt som «blob»-detektorer.

# Kantdeteksjon ved LoG-nullgjennomganger

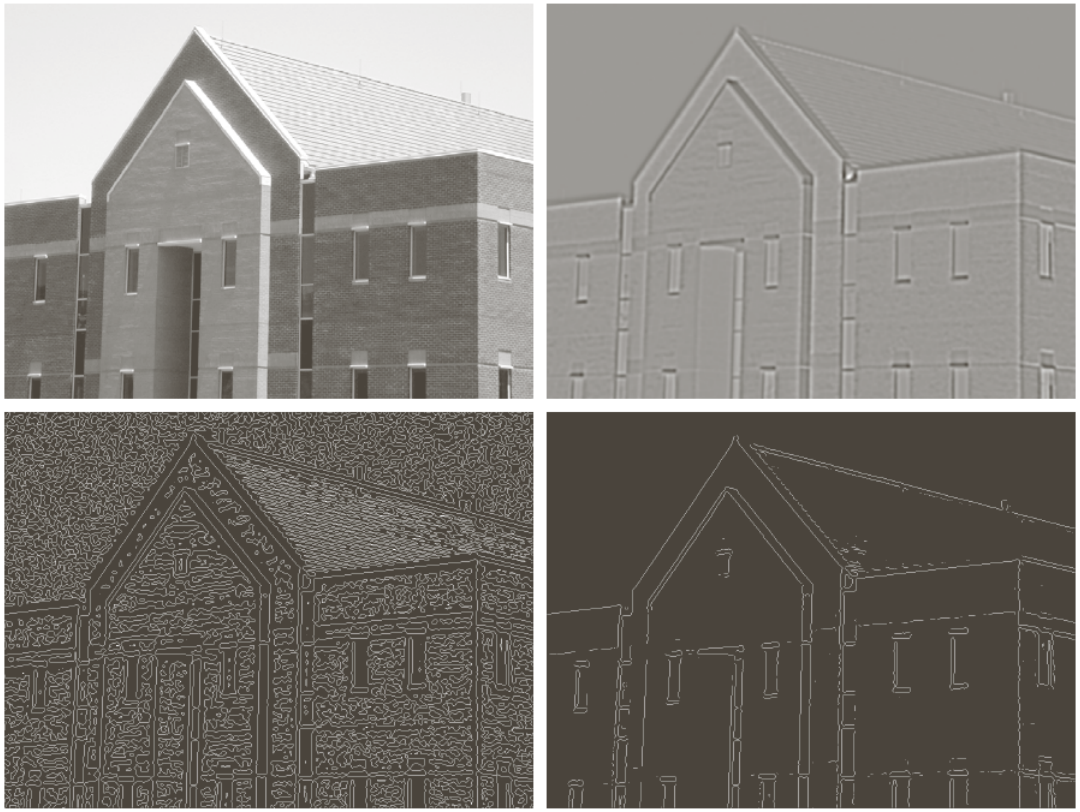
- Tommelfingerregel for strukturer:
  **LoG-kjernen må være smalere enn strukturen**.
  - Strukturen er mindre enn halvparten av LoG-kjernen
    => Nullgjennomgangene er utenfor kantskillene
  - Strukturen er større enn halvparten av LoG-filteret
    => Nullgjennomgangene er nøyaktig kantskillene
  - Et sted i mellom: Avhenger av
    diskretiseringen og tilnærmingen av LoG-filteret.
- Tommelfingerregel for ramper:
  **LoG-filteret må være større enn rampen**.
  - Rampen er bredere enn LoG-filteret,
    => Ingen nullgjennomgang, bare et null-platå.
  - Ellers: Nullgjennomgang midt på rampen
    (kan få én 0-respons akkurat på midten),
    altså en fornuftig definisjon av kantskillet til rampen.
    - P.g.a. støy krever ofte at nullpasseringen er skarp
      => LoG-filteret må være betydelig større enn rampen.
- => **Velg kjerne- og filterstørrelsen med omhu**!
  - Angis først og fremst av standardavviket til Gauss-funksjonen,
    som gir bredden av LoG-kjernen og antyder størrelsen av LoG-filteret.



# Eksempel: LoG-kantdeteksjon

- **Oppgave**: Finn fremtredende kanter.

Inn-bilde LoG-filtrering



Alle nullgjennomganger De med større diff. enn $T$

a b
c d

**FIGURE 10.22**
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found using a threshold equal to 4% of the maximum value of the image in (b). Note the thin edges.

# Robust kantdeteksjon

Vanligvis tre steg i robust kantdetektor:

1. **Støy-reduksjon**: Forsøker å fjerne så mye støy som mulig uten å glatte ut kantene for mye.
   - Lavpassfiltrering.
2. **Kant-filtrering**: Finner kantene.
   - Høypassfiltrering; anvender f.eks. gradient-operatorer.
3. **Kant-lokalisering**: Etterbehandler resultatet fra kant-filtreringen for å finne eksakte kant-posisjoner.
   - Kantresponsen skal helst være ett piksel tykk og være lokalisert der kanten faktisk er i inn-bildet.

# Hva kjennetegner en god kantdetektor?

- Finner **alle** og **bare** de relevante kantene.
- Posisjonen til detektert kant samsvarer med der kanten faktisk finnes i inn-bildet.
- En kant gir én enkelt respons.
- Robust for støy.
  - Trade-off / kompromiss mellom støy-robusthet og kant-lokalisering.

# Ideen til Canny

- Lag en kantdetektor som er optimal i forhold til følgende tre kriterier:
  - Best mulig deteksjon (alle kanter og bare kanter)
  - God kant-lokalisering
  - Én enkelt respons

- Optimer ved bruk av et bilde med støy.

- Resultat: Følgende enkle algoritme oppnår nesten optimumet:

# Cannys algoritme

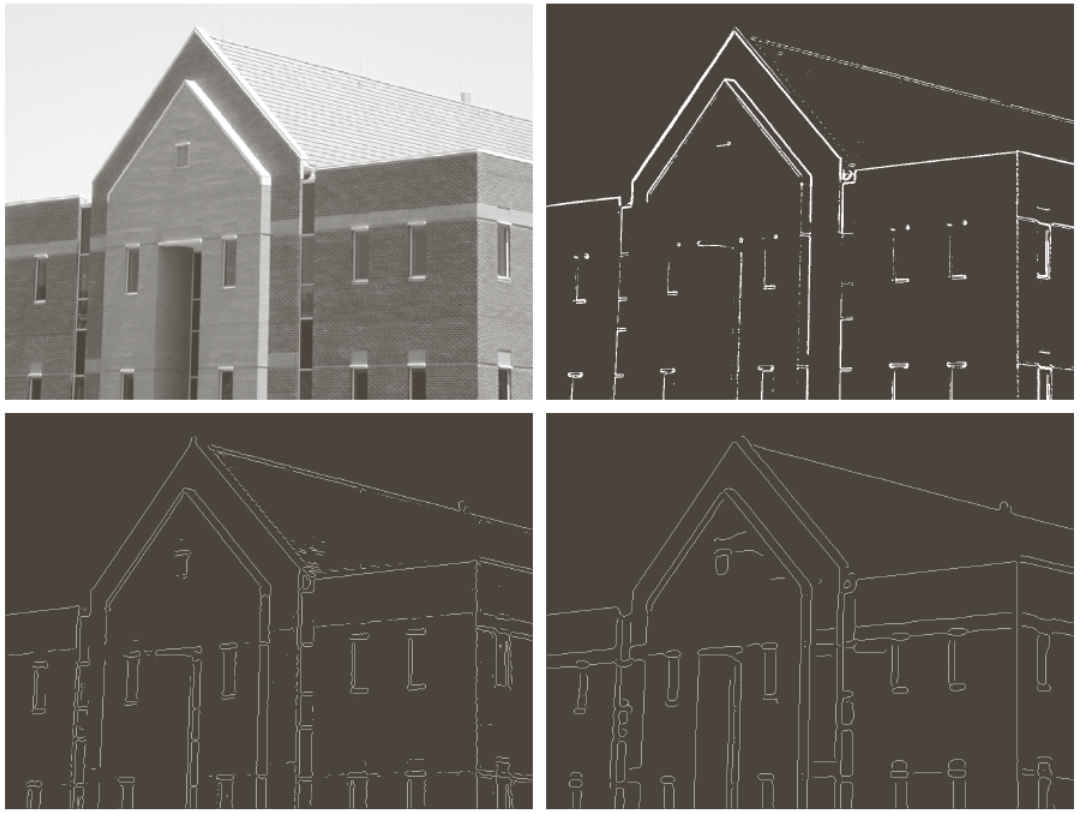
1. Lavpassfiltrer med Gauss-filter (med gitt $\sigma$).
2. Finn gradient-magnituden og gradient-retningen.
3. Tynning av gradient-magnitude ortogonalt på kant.
   - F.eks.: Hvis et piksel i gradient-magnitude-bildet har en 8-nabo i eller mot gradient-retningen med høyere verdi, så settes pikselverdien til 0.
4. Hysterese-terskling (to terskler, $T_h$ og $T_l$) :
   a. Merk alle piksler der $g(x,y) \geq T_h$
   b. For alle piksler der $g(x,y) \in [T_l, T_h)$ :
      - Hvis (4 eller 8)-nabo til et merket piksel, så merkes dette pikselet også.
   c. Gjenta fra trinn b til konvergens.

# Eksempel: Kantdeteksjon

- **Oppgave**: Finn fremtredende kanter.

Inn-bilde

Tersklet glattet grad.mag.



LoG-kantdetektor

Cannys kantdetektor

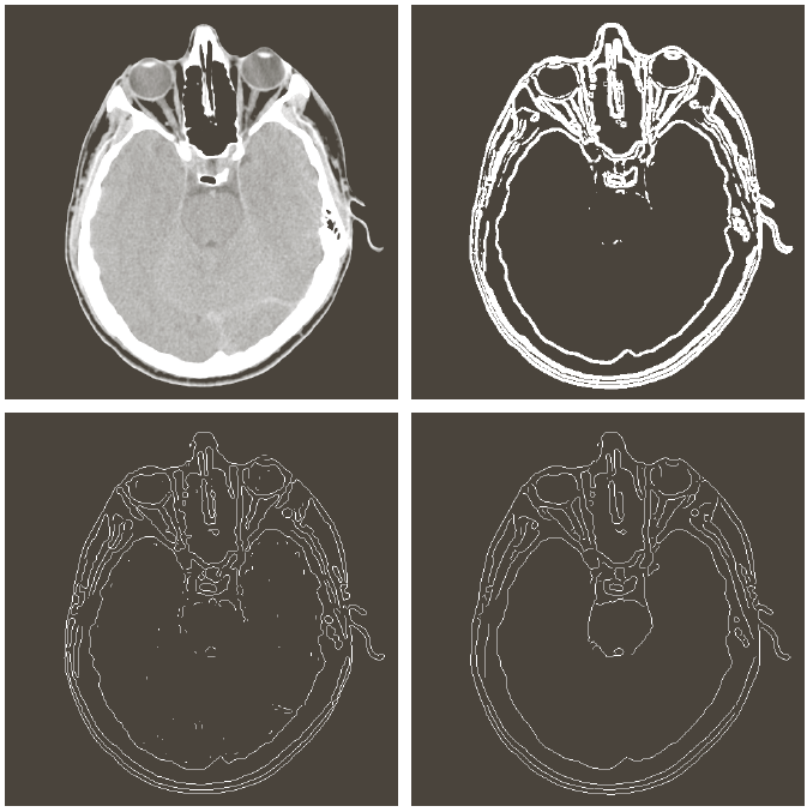
a b
c d

**FIGURE 10.25**
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1].
(b) Thresholded gradient of smoothed image.
(c) Image obtained using the Marr-Hildreth algorithm.
(d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.

# Eksempel: Kantdeteksjon

- **Oppgave**: Finn fremtredende kanter.

Inn-bilde Tersklet glattet grad.mag.



LoG-kantdetektor Cannys kantdetektor

# Oppsummering

- Vi har utledet enkle kant-deteksjonsoperatorer.
- Gradient-operatorene glatter i den ene retningen og gjør kantdeteksjon i den andre retningen.
- Gradient-operatorer gir både kant-styrke og retning.
- Laplace-operatorer gir presis lokalisering av kanten, men forsterker støy.
- LoG-operatoren er en mer robust versjon av Laplace som inkluderer Gauss-glatting.
  - Kjernens og filterets størrelse må passe til oppgaven!
- Cannys kantdetektor gir et kompromiss mellom støyreduksjon og kantlokalisering.