EE 584
MACHINE VISION

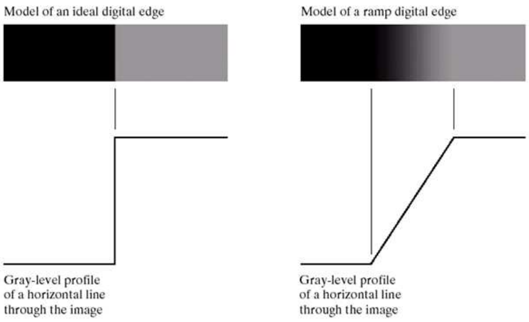
Edge Detection
- Differential Operators
- Discrete Approximations
  - Roberts, Prewit & Sobel
- Laplacian of Gaussian (LoG) Detector
- Canny Edge Detector

Corner Detection

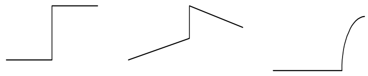
Edge Finding

- A curve in image where rapid changes occur
- Edges usually contain important info
  - Surface orientation changes
  - Shadows due to non-uniform illumination
  - Occlusions of objects
  - Discontinuity in the surface reflectance
- → discontinuity in image brightness is expected
- → derivatives can be used to detect edges
- Edge detection is complementary to segmentation, since edges divide image into regions

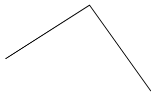
Types of Edges



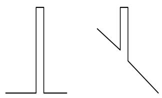
© Gonzalez & Woods, Digital Image Processing,2nd ed., Prentice-Hall, 2002



Step edges



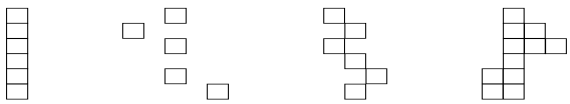
Roof-top edge



Line edges

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2012

Major Steps in Edge Detection



True edge | Noise susceptibility | Poor localization | Too many responses

- Steps for most edge detection algorithms:
  - Filtering : Noise is a critical factor; filtering noise is possible while losing edge strength
  - Enhancement : Using gradient information, significant changes in intensity is located
  - Detection : Finding edge pixels among all the pixels with non-zero gradient information
  - Localization : (Optional) non max. suppression and subpixel resolution

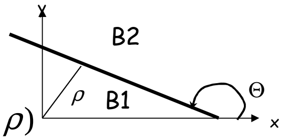
Differential Operators (1/3)

- A simple edge model : $u(z)=\begin{cases}1 & \text{for } z>0\\ \frac{1}{2} & \text{for } z=0\\ 0 & \text{for } z<0\end{cases}$ where $u(z)=\int_{-\infty}^{z}\delta(t)dt$

- Assume edge is a line :

$$x\sin\Theta - y\cos\Theta + \rho = 0$$

$$E(x,y)=B_1+(B_2-B_1)u(x\sin\Theta - y\cos\Theta+\rho)$$



- Partial derivatives of the intensity field E(x,y)

$$\frac{\partial E(x,y)}{\partial x}=\sin\Theta(B_2-B_1)\,\delta(x\sin\Theta - y\cos\Theta+\rho)$$

$$\frac{\partial E(x,y)}{\partial y}=-\cos\Theta(B_2-B_1)\,\delta(x\sin\Theta - y\cos\Theta+\rho)$$

Differential Operators (2/3)

- Magnitude for brightness gradient $\begin{bmatrix}\frac{\partial E(x,y)}{\partial x}\\ \frac{\partial E(x,y)}{\partial y}\end{bmatrix}$

➔ $$\left(\frac{\partial E(x,y)}{\partial x}\right)^2+\left(\frac{\partial E(x,y)}{\partial y}\right)^2=(B_2-B_1)^2\,\delta^2\,(x\sin\Theta - y\cos\Theta+\rho)$$

- Similarly, *Laplacian* can be found as

$$\frac{\partial^2 E(x,y)}{\partial x^2}=\sin^2\Theta(B_2-B_1)\,\delta'(x\sin\Theta - y\cos\Theta+\rho)$$

$$\frac{\partial^2 E(x,y)}{\partial y^2}=\cos^2\Theta(B_2-B_1)\,\delta'(x\sin\Theta - y\cos\Theta+\rho)$$

➔ $$\frac{\partial^2 E(x,y)}{\partial x^2}+\frac{\partial^2 E(x,y)}{\partial y^2}=(B_2-B_1)\,\delta'(x\sin\Theta - y\cos\Theta+\rho)$$

Differential Operators (3/3)

$$\left(\frac{\partial E(x,y)}{\partial x}\right)^2+\left(\frac{\partial E(x,y)}{\partial y}\right)^2=(B_2-B_1)^2\,\delta^2\,(x\sin\Theta-y\cos\Theta+\rho)$$

$$\frac{\partial^2 E(x,y)}{\partial x^2}+\frac{\partial^2 E(x,y)}{\partial y^2}=(B_2-B_1)\,\delta'(x\sin\Theta-y\cos\Theta+\rho)$$

- Note that the magnitude of brightness gradient and *Laplacian* do <u>not depend on orientation</u> (rotation or translation) of the edge
  ➔ Isotropic operators
- *Laplacian* retains the sign of the brightness difference across the edge, which allows to determine the brighter side of the image

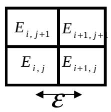
Discrete Approximations (1/4)

- Using finite-difference approximation of a derivative :

$$\frac{\partial E(x,y)}{\partial x}\approx\frac{1}{2\varepsilon}\left\{(E_{i+1,j+1}-E_{i,j+1})+(E_{i+1,j}-E_{i,j})\right\}$$

$$\frac{\partial E(x,y)}{\partial y}\approx\frac{1}{2\varepsilon}\left\{(E_{i+1,j+1}-E_{i+1,j})+(E_{i,j+1}-E_{i,j})\right\}$$

- Discrete approximation to the magnitude of the brightness gradient can be obtained as :



$$\left(\frac{\partial E(x,y)}{\partial x}\right)^2+\left(\frac{\partial E(x,y)}{\partial y}\right)^2\approx\left\{(E_{i+1,j+1}-E_{i,j})^2+(E_{i,j+1}-E_{i+1,j})^2\right\}$$

- Discrete approximation to the angle of the brightness gradient is not accurate since edge pixels may have intermediate values

Discrete Approximations (2/4)

- Roberts operator

$$(\frac{\partial E(x,y)}{\partial x})^2 + (\frac{\partial E(x,y)}{\partial y})^2 \approx \left\{(E_{i+1,j+1} - E_{i,j})^2 + (E_{i,j+1} - E_{i+1,j})^2\right\}$$

1	0
0	-1

0	-1
1	0

- Prewitt operator
  - Averaging to suppress noise

−1	0	1
−1	0	1
−1	0	1

1	1	1
0	0	0
−1	−1	−1

- Sobel operator
  - Averaging with emphasis to center pixels

−1	0	1
−2	0	2
−1	0	1

1	2	1
0	0	0
−1	−2	−1

Discrete Approximations (3/4)

- Compass Gradients for different orientations

- Sobel Compass

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

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

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

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

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

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

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

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

- Nevatia-Babu Compass

-100	-100	0	100	100
-100	-100	0	100	100
-100	-100	0	100	100
-100	-100	0	100	100
-100	-100	0	100	100

0^0

-100	-32	100	100	100
-100	-78	92	100	100
-100	-100	0	100	100
-100	-100	-92	78	100
-100	-100	-100	-32	100

30^0

100	-100	100	100	100
-32	78	100	100	100
-100	-92	0	92	100
-100	-100	-100	-78	32
-100	-100	-100	-100	-100

60^0

100	100	100	100	100
100	100	100	100	100
0	0	0	0	0
-100	-100	-100	-100	-100
-100	-100	-100	-100	-100

90^0

100	100	100	100	100
100	100	100	78	-32
100	92	0	-92	-100
32	-78	-100	-100	-100
-100	-100	-100	-100	-100

120^0

100	100	100	32	-100
100	100	92	-78	-100
100	100	0	-100	-100
-100	78	-92	-100	-100
-100	-32	-100	-100	-100

150^0

Discrete Approximations (4/4)

- Finite-difference approximation of a *Laplacian* in 3x3 picture cells :

$E_{i-1,j+1}$	$E_{i,j+1}$	$E_{i+1,j+1}$
$E_{i-1,j}$	$E_{i,j}$	$E_{i+1,j}$
$E_{i-1,j-1}$	$E_{i,j-1}$	$E_{i+1,j-1}$

$$\frac{\partial^2 E(x,y)}{\partial x^2} \approx \frac{1}{\varepsilon^2}\left\{E_{i+1,j} - 2E_{i,j} + E_{i-1,j}\right\}$$

$$\frac{\partial^2 E(x,y)}{\partial y^2} \approx \frac{1}{\varepsilon^2}\left\{E_{i,j-1} - 2E_{i,j} + E_{i,j+1}\right\}$$

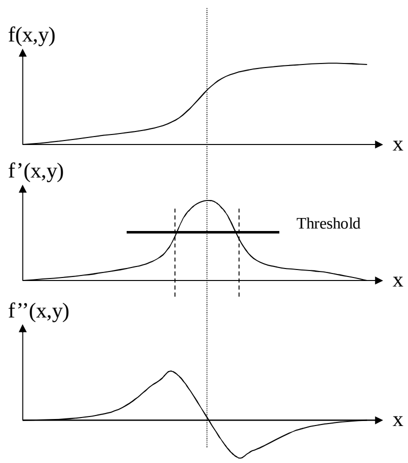
$$\frac{\partial^2 E(x,y)}{\partial x^2} + \frac{\partial^2 E(x,y)}{\partial y^2} \approx \frac{4}{\varepsilon^2}\left\{\underbrace{\frac{1}{4}(E_{i+1,j} + E_{i-1,j} + E_{i,j-1} + E_{i,j+1})}_{\text{Average of neighbors}} - E_{i,j}\right\}$$

- *Laplacian* will return 0 values around constant and linearly changing regions
- Corresponding stencil (kernel) →

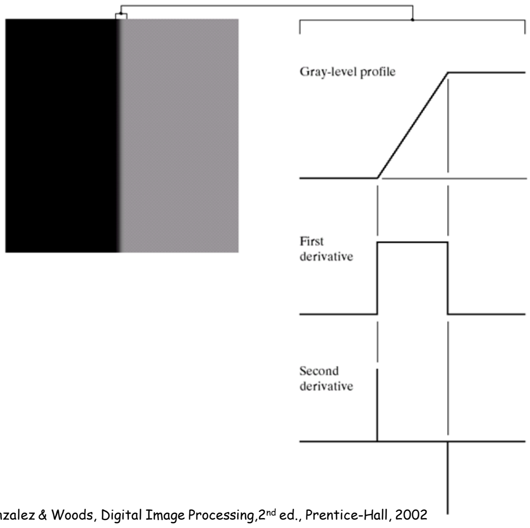
	1	
1	−4	1
	1	

Detection of Edges (1/2)

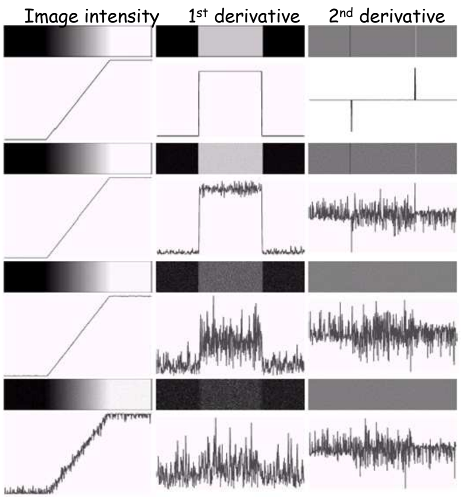
- Detection of an edge differs for 1st and 2nd derivatives
- Maxima of 1st derivative gives edge location
- Zero-crossing of 2nd derivative shows the location of edge



Detection of Edges (2/2)



Effect of Noise on Edges



No noise

Gaussian noise with σ=0.1

Gaussian noise with σ=1.0

Gaussian noise with σ=10.0

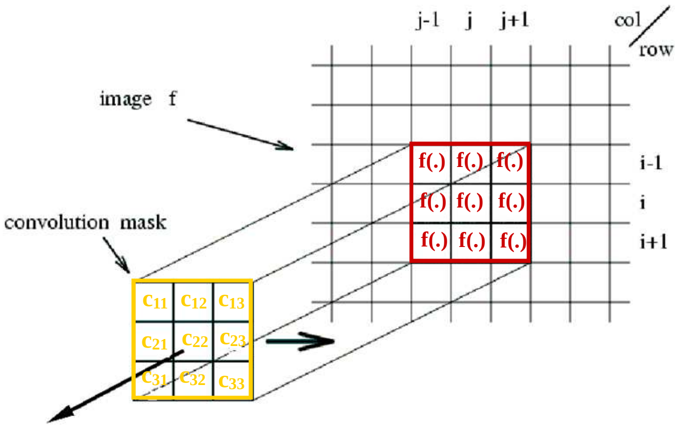
Detection vs. Localization

- Noise generates spurious edges
- In order to suppress noise, filtering is one option but the support of such filters are quite large
- Filtering makes the edges thicker
- Hence, edge localization becomes weaker.

This dilemma can not be avoided

Slide from Marc Pollefeys, Lecture Notes on Computer Vision

Convolution



$$o(i,j) = c_{11}\, f(i-1,j-1) + c_{12}\, f(i-1,j) + c_{13}\, f(i-1,j+1) + c_{21}\, f(i,j-1) + c_{22}\, f(i,j) + c_{23}\, f(i,j+1) + c_{31}\, f(i+1,j-1) + c_{32}\, f(i+1,j) + c_{33}\, f(i+1,j+1)$$

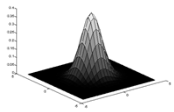
Slide from Marc Pollefeys, Lecture Notes on Computer Vision

Convolution

Slide from Marc Pollefeys, Lecture Notes on Computer Vision

Convolution

$$\exp\left(-\left(\frac{x^2+y^2}{2\sigma^2}\right)\right)$$



Laplacian of Gaussian (LoG)

- Smoothing filter is a Gaussian
- Enhancement step is a Laplacian
- Detection is based on zero-crossings
- Edge location can be obtained by linear int.

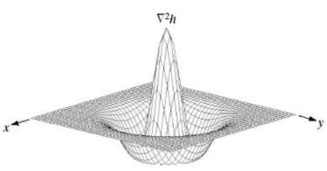
$$h(x,y) = \nabla^2\big(g(x,y) * f(x,y)\big) \quad \text{where } g(x,y) \text{ is a Gaussian filter}$$
$$= \big(\nabla^2 g(x,y)\big) * f(x,y) \quad \text{where}$$
$$\nabla^2 g(x,y) = \left(\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

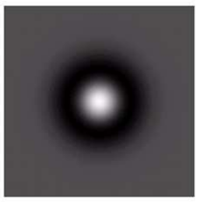
METU EE 584 Lecture Notes by A.Aydin ALATAN © 2012

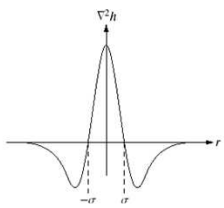
Laplacian of Gaussian (LoG)

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

Inverted LoG function (Mexican hat)







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

© Gonzalez & Woods, Digital Image Processing, 2nd ed., Prentice-Hall, 2002

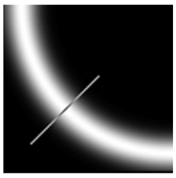
Canny Edge Detector

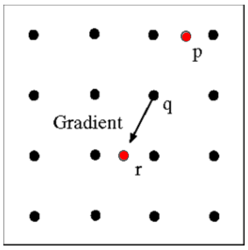
- Smooth the image with a Gaussian
- Compute the gradient magnitude and angle
- Apply non-maxima suppression to magnitude
  - remove all pixels except the maximum along the gradient direction
- Use double thresholding (*hysteresis*) the detect/link edges
  - obtain two edges maps with two thresholds
    1. High threshold → thick edges → start tracing
    2. From staring points, low threshold → trace on thin edges
  - link the thin edges using the other to obtain the final

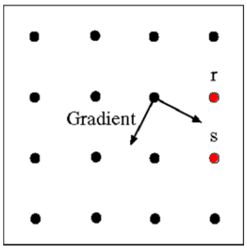
Slide from Marc Pollefeys, Lecture Notes on Computer Vision

Canny Edge Detector

- Non-maxima suppression
  - which point is the maximum?
    - along the gradient direction, q should be larger than p & r (both p & r are interpolated)
  - where is the next maximum?
    - next one should be around the line perpendicular to the gradient vector, i.e. r or s







Surface Fitting for Edge Detection

- Discrete approximations of derivatives limits the performance
- Let $z=f(x,y)$ be a continuous image intensity function to be found, after fitted to the discrete pixel values at each neighborhood
- $f(x,y)$ function can be approximated locally at every pixel of the image, so that derivatives can be found
- Let $f(x,y)=k_1+k_2x+ k_3y+ k_4x^2+ k_5xy+ k_6y^2+ k_7x^3+k_8x^2y+ k_9xy^2+ k_9y^3$
- First solve for k_i using the discrete pixel values, then analytically find the partial derivatives to determine the location of the edges

Comparison between Edge Detectors

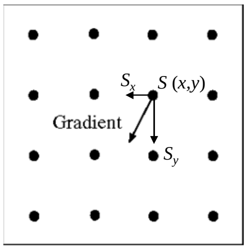
- There are 2 main approaches in edge detection
  - Extracting local maxima of the magnitude gradient in the direction of gradient
  - Finding zero crossings of the Laplacian
- The idea behind these two approaches is quite similar
- Note that following two approaches are equivalent
  - Extracting local maxima of the magnitude gradient in the direction of gradient
  - Finding points, where the 2nd directional derivative in the direction of the gradient is zero.
- Hence, how are
  - Zero crossings of the Laplacian
  - Zero crossings of the 2nd directional derivative in the direction of the gradient

  related?

Comparison between Edge Detectors

- Parameterize the directional derivative by *t in the direction of gradient*

$$\frac{\partial^2 \bar{S}(x,y)}{\partial n^2} = \frac{\partial^2}{\partial t^2}\bar{S}\left(x+t\frac{S_x}{\sqrt{S_x^2+S_y^2}},\, y+t\frac{S_y}{\sqrt{S_x^2+S_y^2}}\right)\Bigg|_{t=0}$$



$$= \frac{\partial}{\partial t}\left[\begin{array}{l}\frac{S_x}{\sqrt{S_x^2+S_y^2}}\bar{S}_x\left(x+t\frac{S_x}{\sqrt{S_x^2+S_y^2}},\, y+t\frac{S_y}{\sqrt{S_x^2+S_y^2}}\right)\\ +\frac{S_y}{\sqrt{S_x^2+S_y^2}}\bar{S}_y\left(x+t\frac{S_x}{\sqrt{S_x^2+S_y^2}},\, y+t\frac{S_y}{\sqrt{S_x^2+S_y^2}}\right)\end{array}\right]\Bigg|_{t=0}$$

$$= \frac{S_{xx}S_x^2 + 2S_{xy}S_xS_y + S_{yy}S_y^2}{S_x^2+S_y^2}$$

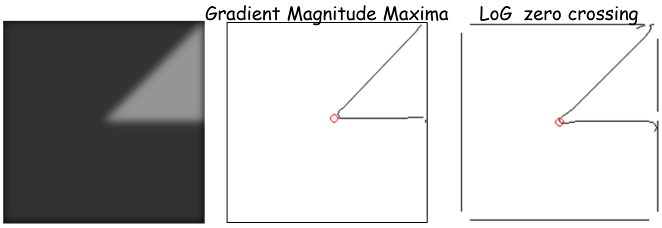
Comparison between Edge Detectors

$$\frac{\partial^2 \bar{S}(x,y)}{\partial n^2} = \frac{S_{xx}S_x^2 + 2S_{xy}S_xS_y + S_{yy}S_y^2}{S_x^2+S_y^2}$$

- Finding edges via LoG $\nabla^2 S(x,y) = S_{xx}(x,y) + S_{yy}(x,y)$

$$\nabla^2 S(x,y) = \begin{bmatrix} a & b \end{bmatrix}\begin{bmatrix} S_{xx} & S_{xy} \\ S_{yx} & S_{yy}\end{bmatrix}\begin{bmatrix} a \\ b\end{bmatrix} + \begin{bmatrix} -b & a \end{bmatrix}\begin{bmatrix} S_{xx} & S_{xy} \\ S_{yx} & S_{yy}\end{bmatrix}\begin{bmatrix} -b \\ a\end{bmatrix} \qquad a^2+b^2=1$$

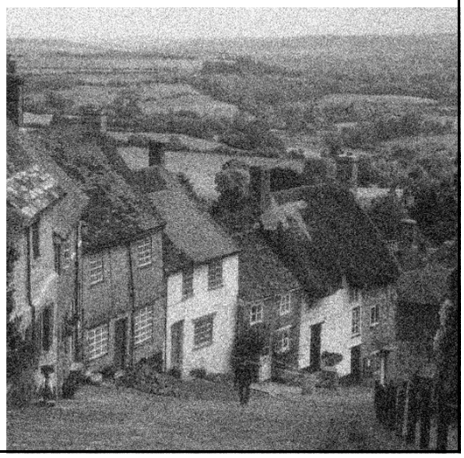
$$\text{let } a = \frac{S_x}{\sqrt{S_x^2+S_y^2}},\, b = \frac{S_y}{\sqrt{S_x^2+S_y^2}} \Rightarrow \nabla^2 S(x,y) = \frac{\partial^2 S}{\partial n^2} + \frac{\partial^2 S}{\partial n_\perp^2}$$



Simulations : Edge Detection

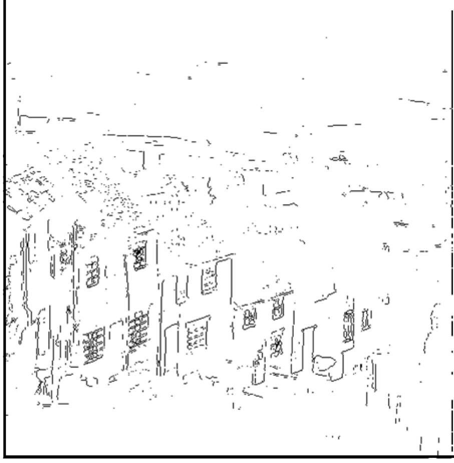
Original gray-scale

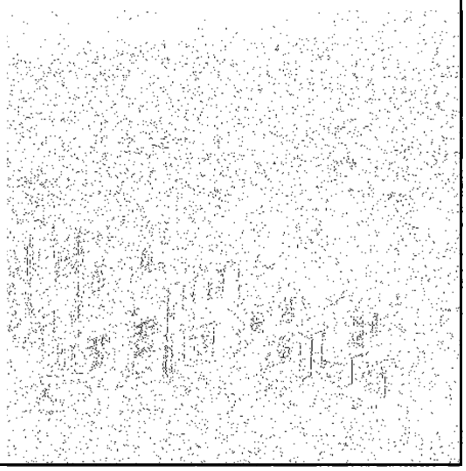
Additive Gaussian Noise



Roberts Operator

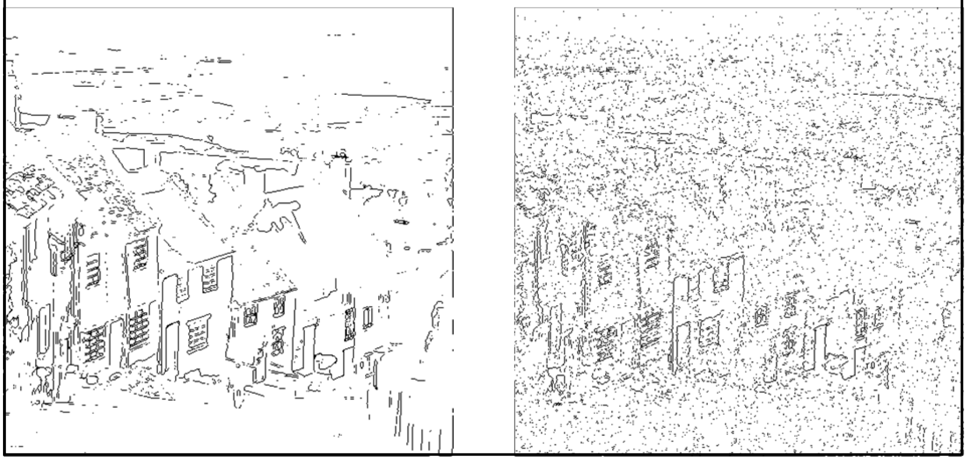
Poor robustness to noise, low detection





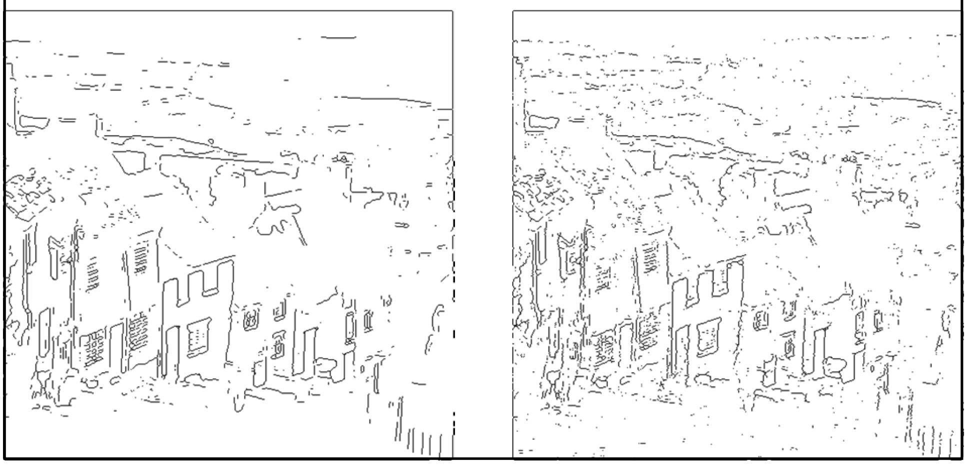
Sobel & Prewitt Operators

Better robustness to noise, better detection



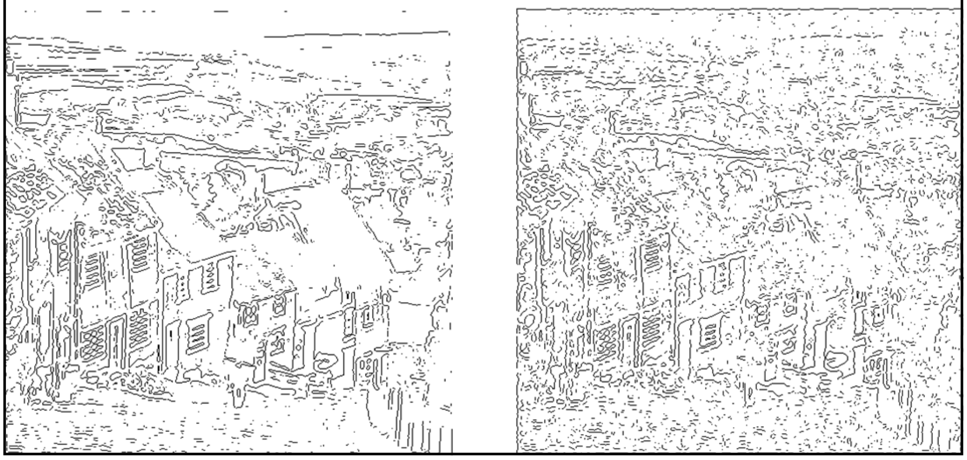
Compass Gradients (Nevatia-Babu)

Good robustness to noise, noise/localization trade-off



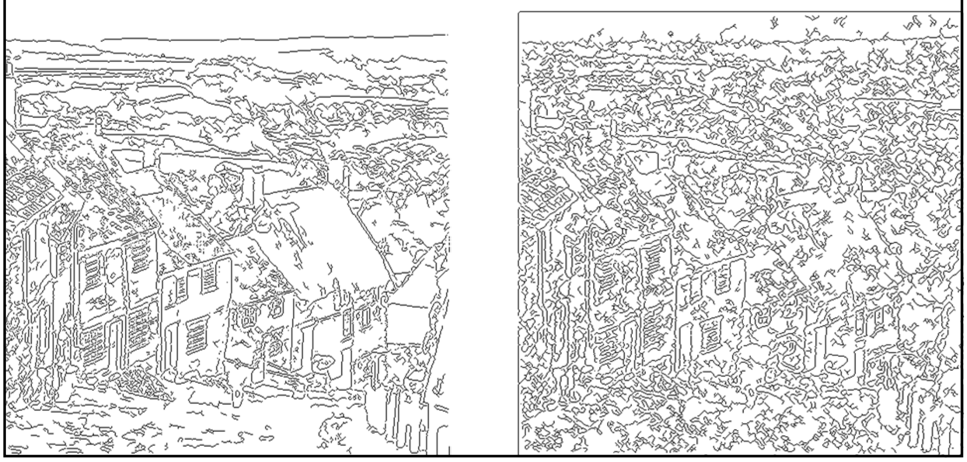
LoG Operator

Better robustness to noise, good detection, better localization
(May fail at very nonlinear intensity gradients)



Canny Detector

Better robustness to noise, very good detection, good localization



Performance Comparison

- Canny (5)
- LoG (4)
- Compass Gradients (3)
- Sobel & Prewitt (2)
- Roberts' Cross (1)

- Detection vs. Localization problem still exists

Final Words on Edge Finding

- A simple model is used : unit step + noise
- Better models are emerging for more realistic situations
- Fundamental problem of detection vs. localization still exists

Although edge detection is assumed to be a simple problem, it is not possible to obtain perfect results in real applications

Corner Detection : Intro

- Extraction of salient features from an image is necessary in many applications
- Requirements for such feature detectors
  - Accurate localization
  - Repeatability (detectable under different views)
  - Invariance under geometric and photometric transformations
- Corners are typical salient features

A corner model

- An ideal corner with an edge along the x-axis and an angle Θ

$$I_{\Theta}(x,y) = u(mx - y) \cdot u(y)$$

$$m = \tan\Theta \qquad u(x) = \begin{cases} 1 & if \quad x > 0 \\ 0 & otherwise \end{cases}$$

- Convolving with a 2-D Gaussian yields (more realistic data)

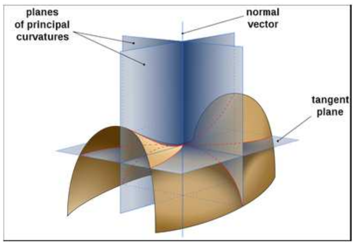
$$S(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x-a)\cdot g(y-b)\cdot u(ma-b)\cdot u(b) \;\; dadb$$

$$g(x) = \frac{1}{\sigma\sqrt{2\Pi}} e^{-\frac{x^2}{2\sigma^2}}$$

Gaussian Curvature (1/4)

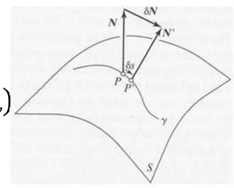
- *Curvature* of a point, *p*, on a <u>curve</u> is defined as rate of change of the tangent vector

- For a <u>surface</u>, there exist too many (infinitely many) curves passing through a given point, *p*
- Consider the intersection of this surface with the planes which pass thru the normal vector at point *p*
  - Intersection of these planes with the surface results with curves of various curvatures
  - <u>Principal curvatures</u> : min & max of such curvatures, K_{min} & K_{max} (planes always perpendicular)



Slide from Marc Pollefeys, Lecture Notes on Computer Vision

Gaussian Curvature (2/4)



- Parametric surface: $\boldsymbol{x}(u,v)$, $\boldsymbol{x_u}=\partial x(u,v)/\partial u$,
- Unit surface normal: $\boldsymbol{N} = \dfrac{1}{|\boldsymbol{x}_u \times \boldsymbol{x}_v|}(\boldsymbol{x}_u \times \boldsymbol{x}_v)$
- First fundamental form: I($\boldsymbol{t}, \boldsymbol{t}$)

  $\boldsymbol{t} = u' \boldsymbol{x}_u + v' \boldsymbol{x}_v$: a vector in the tangent plane at $\boldsymbol{x}$

  $I(\boldsymbol{t},\boldsymbol{t}) \equiv \boldsymbol{t}\,.\,\boldsymbol{t} = Eu'^2 + 2Fu'v' + Gv'^2$

  $$\begin{cases} E = \boldsymbol{x}_u \,.\, \boldsymbol{x}_u \\ F = \boldsymbol{x}_u \,.\, \boldsymbol{x}_v \\ G = \boldsymbol{x}_v \,.\, \boldsymbol{x}_v \end{cases}$$

- Second fundamental form: II($\boldsymbol{t}, \boldsymbol{t}$)

  $II(\boldsymbol{t},\boldsymbol{t}) \equiv \boldsymbol{t}\,.\,dN(\boldsymbol{t}) = eu'^2 + 2fu'v' + gv'^2$

  $$\begin{cases} e = -\boldsymbol{N}\,.\,\boldsymbol{x}_{uu} = \boldsymbol{N}_u\,.\,\boldsymbol{x}_u \\ f = -\boldsymbol{N}\,.\,\boldsymbol{x}_{uv} = \boldsymbol{N}_v\,.\,\boldsymbol{x}_u \\ g = -\boldsymbol{N}\,.\,\boldsymbol{x}_{vv} = \boldsymbol{N}_v\,.\,\boldsymbol{x}_v \end{cases}$$

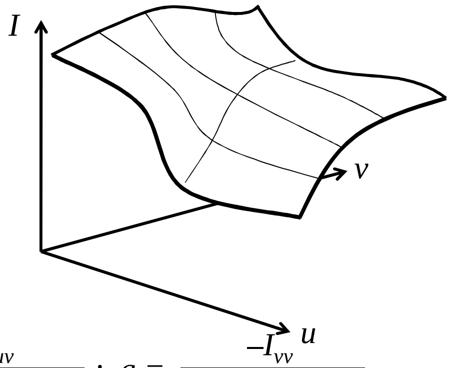
- Normal and Gaussian curvatures:

$$\kappa_t \equiv \frac{II(\boldsymbol{t},\boldsymbol{t})}{I(\boldsymbol{t},\boldsymbol{t})} \qquad K \equiv \frac{eg - f^2}{EG - F^2}$$

Gaussian Curvature (3/4)

Monge Patches

$$\mathbf{x}(u, v) = (u, v, I(u, v))$$



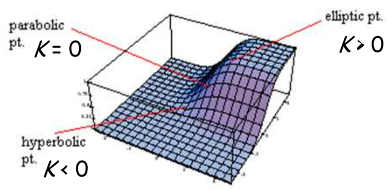
In this case

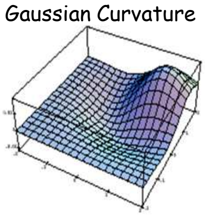
- $\mathbf{N} = \dfrac{1}{(1+I_u^2+I_v^2)^{1/2}} (-I_u, -I_v, 1)^T$
- $E = 1+I_u^2;\ F = I_u I_v;\ G = 1+I_v^2$
- $e = \dfrac{-I_{uu}}{(1+I_u^2+I_v^2)^{1/2}}\ ;\ f = \dfrac{-I_{uv}}{(1+I_u^2+I_v^2)^{1/2}}\ ;\ g = \dfrac{-I_{vv}}{(1+I_u^2+I_v^2)^{1/2}}$

Gaussian curvature is equal to: $K = \dfrac{I_{uu}I_{vv}-I_{uv}^2}{(1+I_u^2+I_v^2)^2}$.

Gaussian Curvature (4/4)

Gaussian curvature: $K = K_{\min} \cdot K_{\max}$ $\qquad K = \dfrac{I_{xx}I_{yy} - I_{xy}^2}{\left(1 + I_x^2 + I_y^2\right)^2}$





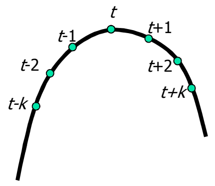
Gaussian curvature is 0 for plane&cylindir, +1 for spheres, (-) for saddle points.

It determines whether a surface is locally convex or locally saddle ➔ corner at 0.

1. Compute the Gaussian curvature.
2. Select locations of Gaussian curvature extrema.
3. Match each *elliptic* maxima with a *hyperbolic* minima by principal curvatures
4. For a particular match, consider the segment joining the elliptic maximum with the hyperbolic minimum.
5. The point at which the Gaussian curvature is equal to zero ➔ corner

K-Curvature

- Extract the edges in the image using one of the edge detection methods
- Represent the edges as chain codes



- Calculate *k-curvature* on the curve
  1. Let the point denoted by *t*.
  2. Subtract the difference of the directions of the vectors defined by [*t*,*t+k*] and [*t-k*,*t*].
  3. Define this difference to be k-curvature at *t*
  4. Average k-curvatures with possibly different weights(emphasizing small *k*'s) to obtain the curvature.

- The local maximum of a curvature is taken as a corner, if its curvature is above some threshold.

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2012

Kitchen-Rosenfeld Cornerness

- A *cornerness measure* may be proposed, as the change of gradient direction along an edge contour multiplied by the local gradient magnitude.

- Calculating the derivative of $\tan^{-1}\left(\frac{I_y}{I_x}\right)$, along

$$\frac{d\tan^{-1}(x)}{dx} = \frac{1}{1+x^2} \quad \Longrightarrow \quad \frac{\partial \tan^{-1}\left(\frac{I_y}{I_x}\right)}{\partial x} = \frac{I_x I_{yx} - I_y I_{xx}}{I_x^2 + I_y^2}$$

$$K = \left(\frac{\partial \tan^{-1}\left(\frac{I_y}{I_x}\right)}{\partial x} \quad \frac{\partial \tan^{-1}\left(\frac{I_y}{I_x}\right)}{\partial y} \right) \bullet \frac{(-I_y \quad I_x)}{\sqrt{I_x^2 + I_y^2}} \cdot \sqrt{I_x^2 + I_y^2}$$

$$K = \frac{1}{\sqrt{I_x^2 + I_y^2}} \begin{bmatrix} -I_y & I_x \end{bmatrix} \cdot \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix} \cdot \begin{bmatrix} -I_y \\ I_x \end{bmatrix} \frac{1}{\sqrt{I_x^2 + I_y^2}}$$

Zuniga-Haralick Facet Model:

- Image patch (7x7 or 5x5), can be modeled by a bi-cubic polynomial

$$f(x,y) \approx k_1 + k_2 x + k_3 y + k_4 x^2 + k_5 xy + k_6 y^2 + k_7 x^3 + k_8 x^2 y + k_9 xy^2 + k_{10} y^3$$

- Cornerness is again calculated as the rate of change of the direction of the gradient along the edge contour via

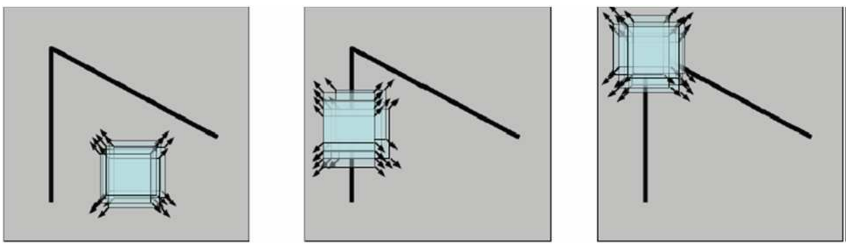
$$K = \frac{I_{xx} I_x^2 - 2 I_{xy} I_x I_y + I_{yy} I_y^2}{\left(I_x^2 + I_y^2\right)^{\frac{3}{2}}} \Longrightarrow K = 2\frac{k_4 k_2^2 - k_5 k_2 k_3 + k_6 k_3^2}{\left(k_2^2 + k_3^2\right)^{\frac{3}{2}}}$$

- If this measure has a local maximum and is above a threshold, an edge point, detected after the non-maxima gradient magnitude suppression ➔ corner

Harris-Stephens Corner Detector:

- If the following function, at some point , assumes high values for any direction, $(\Delta x, \Delta y)$, the point can be considered to be significantly distinct.

$$m(x,y)_{(\Delta x, \Delta y)} = \left(\int_R \left(I(p,q) - I(p+\Delta x, q+\Delta y) \right)^2 dR \right)$$



Harris-Stephens Corner Detector:

$$m(x,y)_{(\Delta x,\Delta y)} = \left(\int_R \left(I(p,q) - I(p+\Delta x, q+\Delta y) \right)^2 dR \right)$$

- Using Taylor series expansion, as

$$I(x+\Delta x, y+\Delta y) \approx I(x,y) + \Delta x I_x(x,y) + \Delta y I_y(x,y)$$

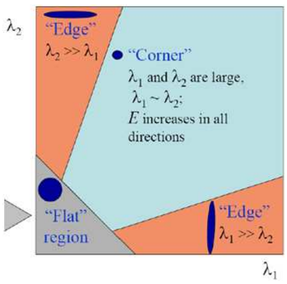
- The distinctness measure can be written as

$$m(x,y)_{(\Delta x,\Delta y)} \approx \sum_R \left(\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2 = \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \underbrace{\begin{bmatrix} \sum_R I_x I_x & \sum_R I_x I_y \\ \sum_R I_x I_y & \sum_R I_y I_y \end{bmatrix}}_{C'} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

Harris-Stephens Corner Detector:

$$m(x,y)_{(\Delta x,\Delta y)} \approx \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \underbrace{\begin{bmatrix} \sum_R I_x I_x & \sum_R I_x I_y \\ \sum_R I_x I_y & \sum_R I_y I_y \end{bmatrix}}_{C'} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

- If C' has 2 significant eigenvalues, λ_1, λ_2,measure should yield high values signaling an interest point



- Instead of calculating the eigenvalues, the product of the eigenvalues is compared via $Det(C) - kTrace^2(C)$

$$= \lambda_1 \cdot \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

Harris-Stephens Corner Detector:

- Concentrating the measure around center of patch, the summations are done using a Gaussian window,

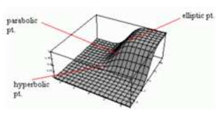
$$C(x,y)=\begin{bmatrix} \sum_R w_R I_x^2 & \sum_R w_R I_x I_y \\ \sum_R w_R I_x I_y & \sum_R w_R I_y^2 \end{bmatrix}$$

- There is also Hessian-Laplace detector

$$Hessian=\begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix} \qquad Det = I_{xx} I_{yy} - I_{xy}^2$$

- Note its relation to Gaussian Curvature

$$Gaussian\_Curvature = (K_{\min}.K_{\max}) = \frac{I_{xx} I_{yy} - I_{xy}^2}{(1+I_x^2+I_y^2)^2}$$

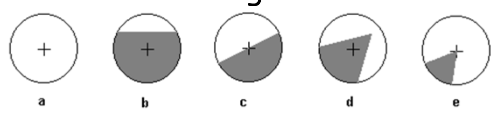


- Search for elliptic maximas at corners or blobs

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2012

SUSAN & Fast Corner Detection

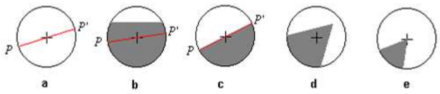
- Radically different interest point detectors
- An area is defined by the pixels having brightness similar to that of the nucleus (center point)
  - USAN: Univalue Segment Assimilating Nucleus



- Near a corner, USAN significantly decreases and attains a local minimum at the corner point
  - *brightness difference threshold* is utilized for deciding whether a pixel in the circular mask belongs to USAN
  - *geometrical threshold* is deciding whether a local minimum is a corner point.

SUSAN & Fast Corner Detection

- In another method, an arbitrary line k containing the nucleus and intersecting the boundary of the circular window at two opposite points is assumed

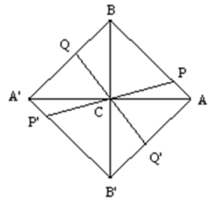


- For various lines k, corner response function (CRF) is calculated by minimizing :

$$R_C = \min_k \left[(I_P - I_N)^2 + (I_{P'} - I_N)^2 \right]$$

- For discrete case, interpolation is used

$$R = \min_{x \in (0,1)} \left(r_1(x), r_2(x) \right)$$
$$r_1(x) = (I_P - I_C)^2 + (I_{P'} - I_C)^2$$
$$r_2(x) = (I_Q - I_C)^2 + (I_{Q'} - I_C)^2$$



Blob Detectors : Scale Invariant Feature Transform

- The scale-normalized LoG function $\sigma^2 \nabla^2 G$
  - σ^2 is required for scale invariance.
  - The amplitude of the scale space representation in general decreases with scale, and the factor σ^2 compensates for this decrease.
- The maxima and minima of $\sigma^2 \nabla^2 G$ produce the most stable image features compared to other image functions such as Harris corner detectors.
- How to implement scale normalized LoG in an efficient manner?

Courtesy Elif Vural

Scale Invariant Feature Transform

- Scale Invariant Feature Transform (SIFT), is a method for extracting distinctive features from images.
- The features are invariant to image scale and rotation.
- In order to assure scale invariance, the image must be searched for stable features across all possible scales.
- This requires the usage of a continuous function of scale known as scale space.

Courtesy Elif Vural

Scale Space Representation

- Given a continuous signal f, the scale space representation L of f is defined as the solution to the diffusion equation:

$$\partial_t L = \frac{1}{2}\nabla^2 L = \frac{1}{2}\sum_{i=1}^{D} \partial_{x_i x_i} L \qquad f : \mathbb{R}^D \to \mathbb{R} \qquad L : \mathbb{R}^D \times \mathbb{R}_+ \to \mathbb{R}$$

with initial condition $L(\cdot;\ 0) = f(\cdot)$

- Equivalently, this family can be defined by convolution with Gaussian kernels of variable width t.

$$L(\cdot;\ t) = g(\cdot;\ t) * f(\cdot) \qquad g : \mathbb{R}^D \times \mathbb{R}_+ \to \mathbb{R}$$

$$g(x;\ t) = \frac{1}{(2\pi t)^{N/2}} e^{-(x_1^2+\ldots+x_D^2)/(2t)}$$

Courtesy Elif Vural

Scale Invariant Feature Transform

- Only possible scale space kernel is the Gaussian function.
  - Scale space L(x,y,σ) of an image I(x,y) as

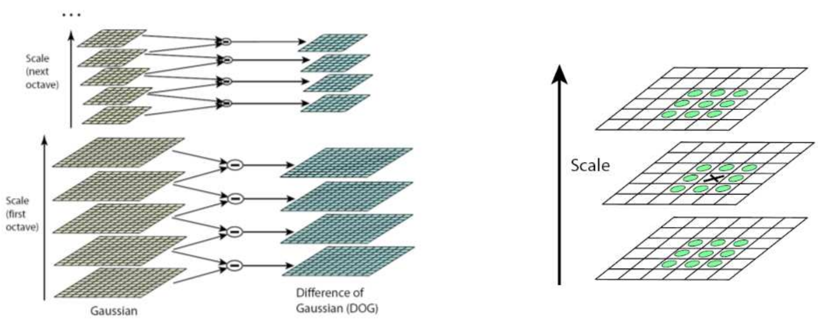
$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

- The method SIFT suggests that the scale-space extrema of difference of Gaussian functions with two nearby scales (σ and kσ) convolved with the image turn out to be stable features.

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

Courtesy Elif Vural

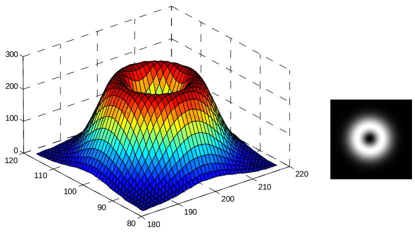
Scale Invariant Feature Transform



- The maxima and minima of the images obtained by convolution with DoG's are detected by comparing a pixel to its 26 neighbours at the current and adjacent scales.

Courtesy Elif Vural

Scale Invariant Feature Transform



Difference of Gaussian

- The reason for using the DoG function is that it is a close approximation to the scale-normalized Laplacian-of-Gaussian function $\sigma^2\nabla^2 G$.

$$\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G \qquad \sigma \nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x,y,k\sigma) - G(x,y,\sigma)}{k\sigma - \sigma}$$

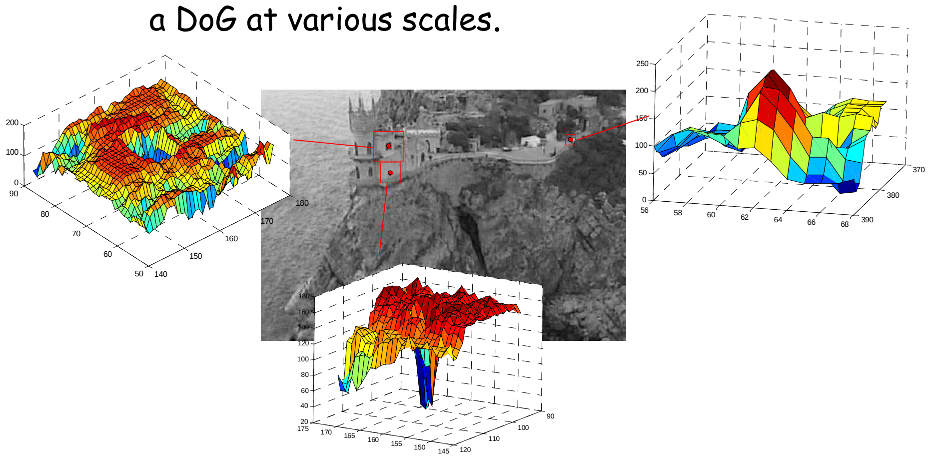
$$G(x,y,k\sigma) - G(x,y,\sigma) \approx (k-1)\sigma^2 \nabla^2 G$$

Courtesy Elif Vural

METU EE 584 Lecture Notes by A.Aydin ALATAN © 2012

Scale Invariant Feature Transform

- SIFT is capable of detecting features resembling a DoG at various scales.



Courtesy Elif Vural

Final Words

- Different salient point detectors exist
  - Edge, corner, blob
- In different applications, these detectors have performances
  - Edges are more suitable to define segments of objects
  - Corners perform better to match two scenes that are observed from similar distance and angles
  - Blobs are more salient in case of scale and view changes