syllabus update

spatial image processing — edges.
(regions)

geometric transformations
- warping
- image registration

- color

- cameras, image formation, camera calibration

image domain processing

Jain, Kasturi, Schunck - Ch. 5, Edge Detection

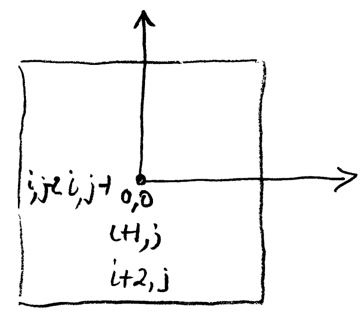
image is an array of samples of a continuous function

So, come up with a function that approximates image and computes image properties from estimated function

Let $z = f(x, y)$, a continuous intensity function

Fitting a polynomial to 240 or more points would be too high a degree so we do piecewise functions called facets.

For an image we might use a 5x5 facet



- For example we might model $f(x,y)$ for simple images as a constant or a bilinear function (a plane)
- For more complex images use biquadratic, bicubics, or higher.

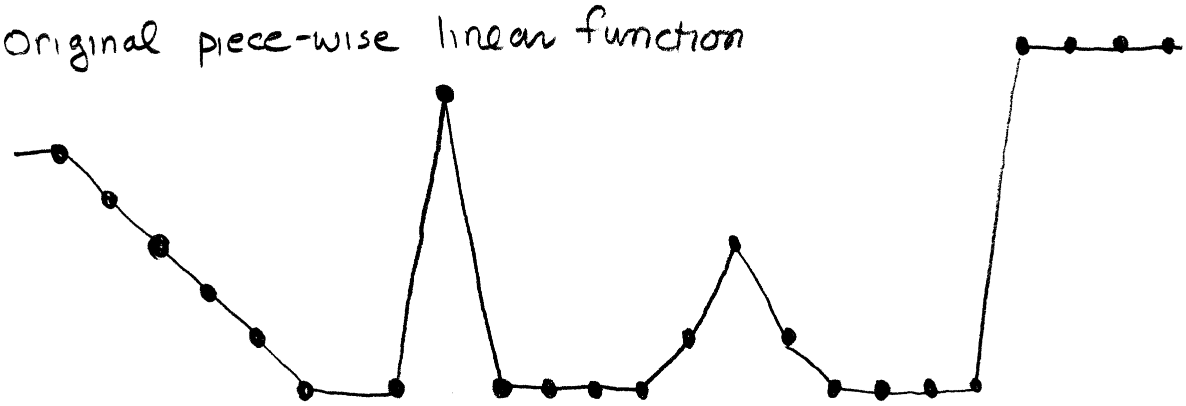
For a bicubic we might fit a bicubic polynomial to this facet, i.e.,

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

We can use least squares (or other methods such as masks or SVD) to compute the k's.

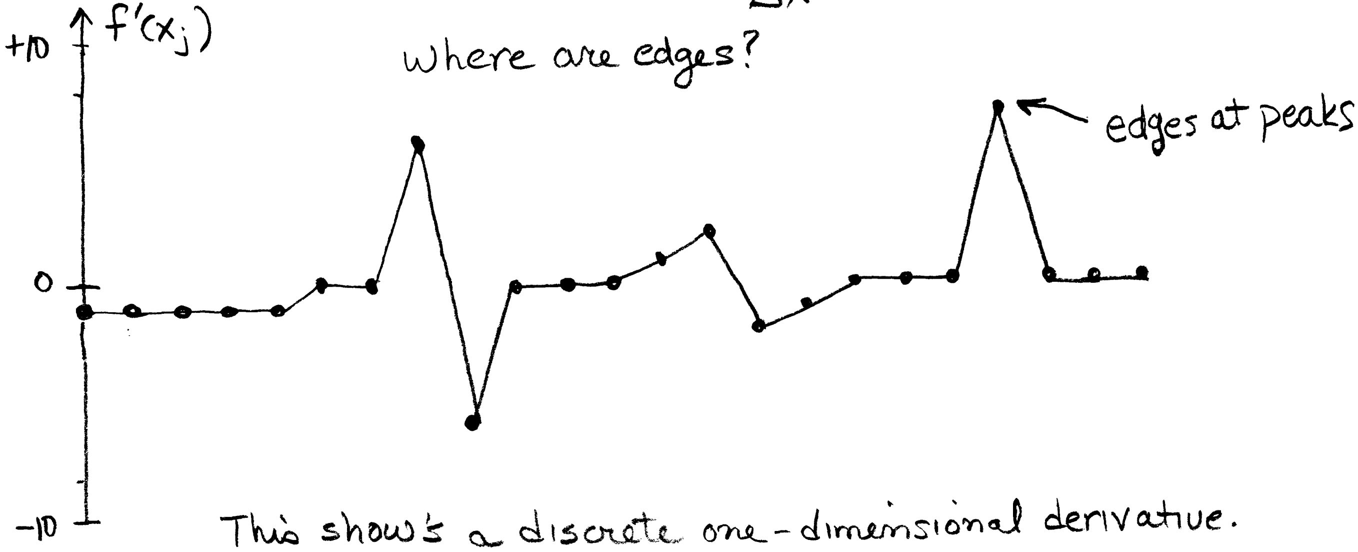
edges might be extreme maxima in the first derivative or zero crossings in the second derivative.

Original piece-wise linear function



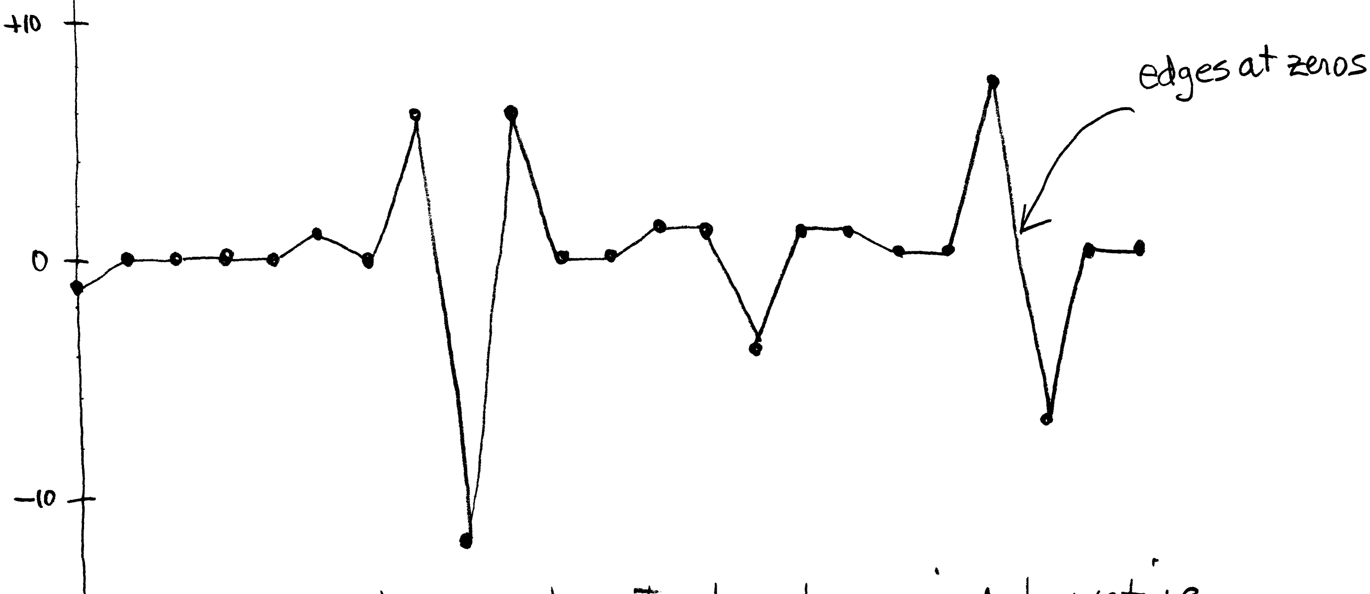
first derivative $f'(x_j) = \dfrac{f(x_{j+1}) - f(x_j)}{\Delta x}$ Assume spacing $\Delta x = 1$

Where are edges?



This shows a discrete one-dimensional derivative.

$f''(x_j) = \dfrac{f'(x_{j+1}) - f'(x_j)}{\Delta x}$



This shows a discrete two-dimensional derivative.

If we have fit the k's of our polynomial we can simply calculate directional derivatives to find an arbitrary edge.

$$f'_\theta(x,y) = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$ is the derivative in the θ-direction

and we look for local maxima.

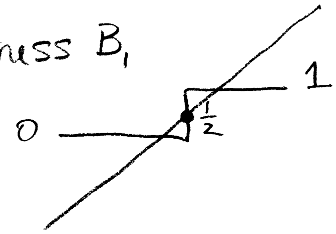
The zero's of the second derivative are usually better for locating edges accurately.

Mark as an edge if $f''_\theta(x_0, y_0; \rho) = 0$ and $f'_\theta(x_0, y_0 \rho) \neq 0$

Horn, differential brightness S8.2

line $x\sin\theta - y\cos\theta + \rho = 0$

brightness B_1



brightness B_2

define unitstep $u(t) = \begin{cases} 1 & z > 0 \\ \frac{1}{2} & z = 0 \\ 0 & z < 0 \end{cases}$

image brightness function

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

computing the partial derivatives

$$\frac{\partial B}{\partial x} = (B_2 - B_1)\sin\theta\ \delta(x\sin\theta - y\cos\theta + \rho)$$

$$\frac{\partial B}{\partial y} = -(B_2 - B_1)\cos\theta\ \delta(x\sin\theta - y\cos\theta + \rho)$$

These are dependent upon rotation and translation.

We can define a <u>brightness gradient</u> (a vector)

$$\begin{bmatrix} \frac{\partial B}{\partial x} \\ \frac{\partial B}{\partial y} \end{bmatrix}$$

magnitude and direction follow rotations and translations of image

<u>squared gradient</u>

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

This is non-linear <u>BUT</u> it is rotationally symmetric finding edges at all angles θ equally well.

<u>Laplacian</u>

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

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

δ' is known as a doublet

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

We can use this to compute the Laplacian

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

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

The Laplacian is <u>linear</u>

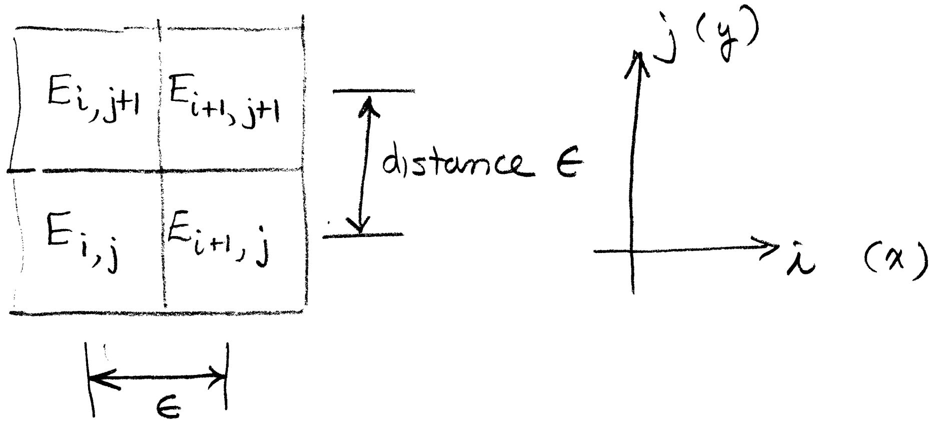
<u>maintains sign of brightness</u>

<u>rotationally symmetric</u>

We can compute a quadratic Laplacian

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

8.3 Horn   Approximate derivative operators discretely



slope in top row $= \dfrac{E_{i+1,j+1} - E_{i,j+1}}{\epsilon}$

slope in bottom row $= \dfrac{E_{i+1,j} - E_{i,j}}{\epsilon}$

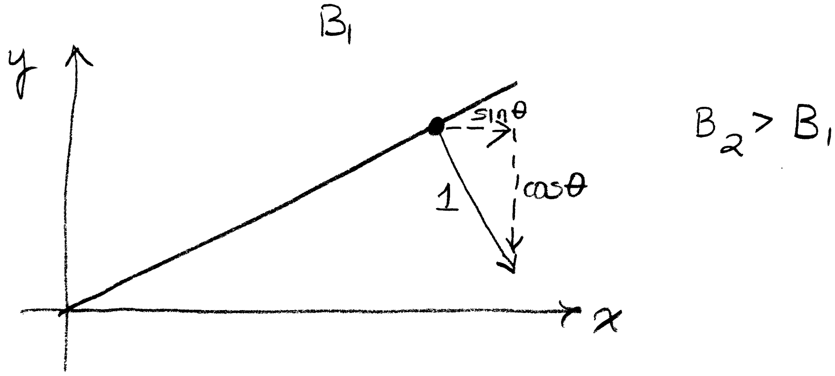
average derivative $\overline{\dfrac{\partial E}{\partial x}} \approx \dfrac{1}{2\epsilon}\left[\left(E_{i+1,j+1} - E_{i,j+1}\right) + \left(E_{i+1,j} - E_{i,j}\right)\right]$

you can do the same in the y-direction

$$\overline{\frac{\partial E}{\partial y}} \approx \frac{1}{2\epsilon}\left[\left(E_{i+1,j+1} - E_{i+1,j}\right) + \left(E_{i,j+1} - E_{i,j}\right)\right]$$

gradient

$$\begin{bmatrix} \dfrac{\partial E}{\partial x} \\ \dfrac{\partial E}{\partial y} \end{bmatrix} = (B_2 - B_1)\ \delta(x \sin\theta - y\cos\theta + \rho) \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix}$$



The gradient points in the direction of largest increase.

Approximating the second derivative requires at least a 3x3 mask.

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

$j \uparrow$

$i \longrightarrow$

$$\frac{\partial^2 E}{\partial x^2} = \frac{\frac{E_{i+1,j} - E_{i,j}}{\epsilon} - \frac{E_{i,j} - E_{i-1,j}}{\epsilon}}{\epsilon}$$

$$= \frac{E_{i+1,j} - 2E_{i,j} + E_{i-1,j}}{\epsilon^2}$$

$$\frac{\partial^2 E}{\partial x^2} \cong \frac{1}{\epsilon^2}\left[E_{i-1,j} - 2E_{i,j} + E_{i+1,j}\right]$$

$$\frac{\partial^2 E}{\partial y^2} \cong \frac{1}{\epsilon^2}\left[E_{i,j-1} - 2E_{i,j} + E_{i,j+1}\right]$$

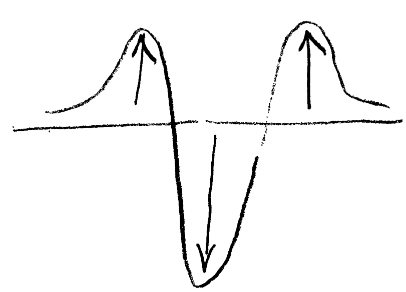
if we combine

$$\frac{\partial^2 E}{\partial x^2} + \frac{\partial^2 E}{\partial y^2} \cong \frac{1}{\epsilon^2}\left[\left(E_{i,j+1} + E_{i+1,j} + E_{i,j-1} + E_{i-1,j}\right) - 4E_{i,j}\right]$$

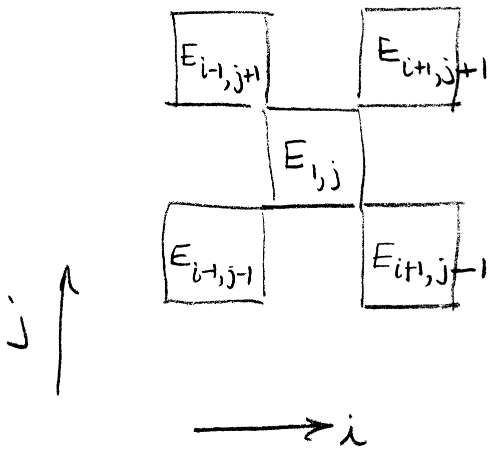
As a mask

	1	
1	−4	1
	1	

This is a reasonable approximation to a Laplacian on a square grid



You can approximate other functions such as a laplacian



$$\frac{\partial^2 E}{\partial x^2} = \frac{\left.\frac{\partial E}{\partial x}\right|_+ - \left.\frac{\partial E}{\partial x}\right|_-}{\sqrt{2}\,\epsilon} = \frac{\frac{E_{i+1,j+1}-E_{i,j}}{\sqrt{2}\,\epsilon} - \frac{E_{i,j}-E_{i-1,j-1}}{\sqrt{2}\,\epsilon}}{\sqrt{2}\,\epsilon}$$

$$\frac{\partial^2 E}{\partial x^2} = \frac{E_{i+1,j+1} - 2E_{i,j} + E_{i-1,j-1}}{2\epsilon^2}$$

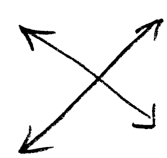
$$\frac{\partial^2 E}{\partial y^2} = \frac{E_{i-1,j+1} - 2E_{i,j} + E_{i+1,j-1}}{2\epsilon^2}$$

$$\frac{\partial^2 E}{\partial x^2} + \frac{\partial^2 E}{\partial y^2} \cong \frac{1}{2\epsilon^2}\left[\left(E_{i+1,j+1} + E_{i+1,j-1} + E_{i-1,j-1} + E_{i-1,j+1}\right) - 4E_{i,j}\right]$$

corresponding mask

$\frac{1}{\epsilon^2}$

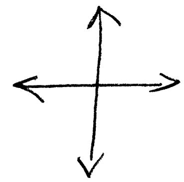
+1		+1
	−4	
+1		+1



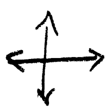
diagonal Laplacian

We can also do a rectangular Laplacian

$$\frac{1}{\epsilon^2}\begin{array}{|c|c|c|}\hline & 1 & \\ \hline 1 & -4 & 1 \\ \hline & 1 & \\ \hline\end{array}$$



A popular Laplacian estimate is

$\frac{2}{3}$  $+\ \frac{1}{3}$ 

$$\frac{1}{6\epsilon^2}\begin{array}{|c|c|c|}\hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline\end{array}$$

which is approximately rotationally symmetric