Pixels

- Neighborhood
- Adjacency
- Connectivity
- Paths
- Regions and boundaries

Neighbors of a Pixel

- Any pixel p(x, y) has two vertical and two horizontal neighbors, given by

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

- This set of pixels are called the 4-neighbors of P, and is denoted by $N_4(P)$.

- Each of them are at a unit distance from P.

Neighbors of a Pixel (Contd..)

- The four diagonal neighbors of p*(x,y)* are given by,

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

- This set is denoted by $N_D(P)$.

- Each of them are at Euclidean distance of 1.414 from P.

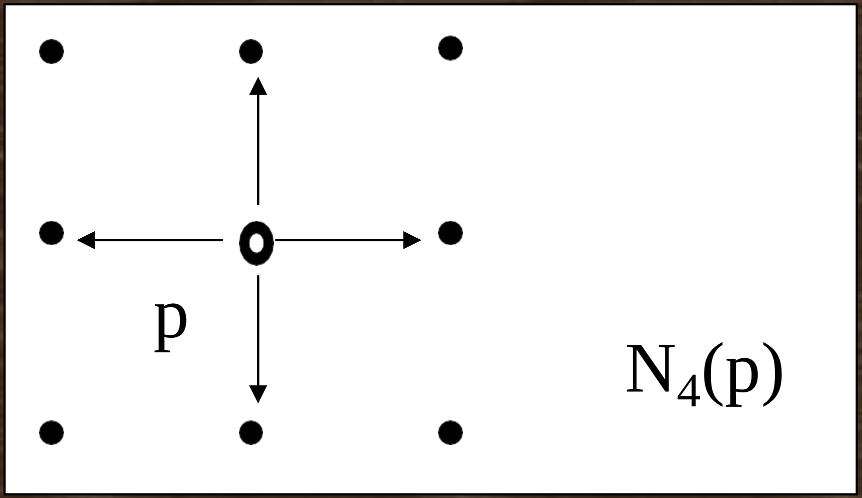
Neighbors of a Pixel (Contd..)

- The points $N_D(P)$ and $N_4(P)$ are together known as 8-neighbors of the point P, denoted by $N_8(P)$.
- Some of the points in the N_4, N_D and N_8 may fall outside image when P lies on the border of image.

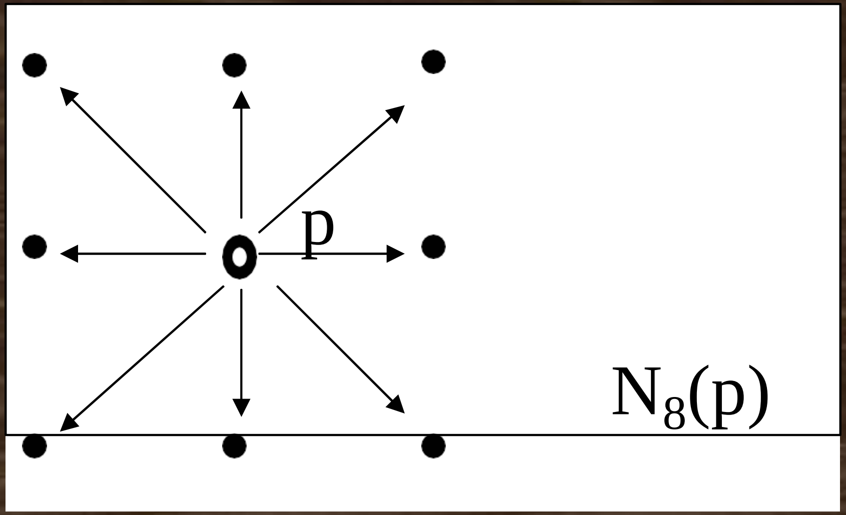
Neighbors of a Pixel (Contd..)

Neighbors of a pixel

a. 4-neighbors **of a pixel p are its vertical and horizontal neighbors denoted by $N_4(p)$**



b. 8-neighbors **of a pixel p are its vertical horizontal and 4 diagonal neighbors denoted by $N_8(p)$**



N_D	N_4	N_D
N_4	**P**	N_4
N_D	N_4	N_D

- N_4 - 4-neighbors
- N_D - diagonal neighbors
- N_8 - 8-neighbors (N_4 U N_D)

Connectivity :

To determine whether the pixels are adjacent in some sense.

Let V be the set of gray-level values define connectivity; then Two pixels p, q that have values from the set V are:

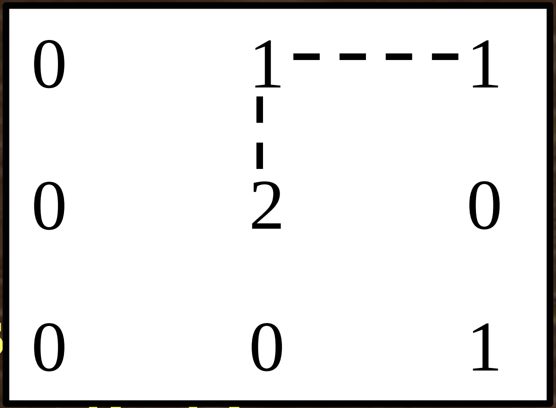
a. 4-connected, **if q is in the set N_4(p)**

b. 8-connected, **if q is in the set N_8(p)**

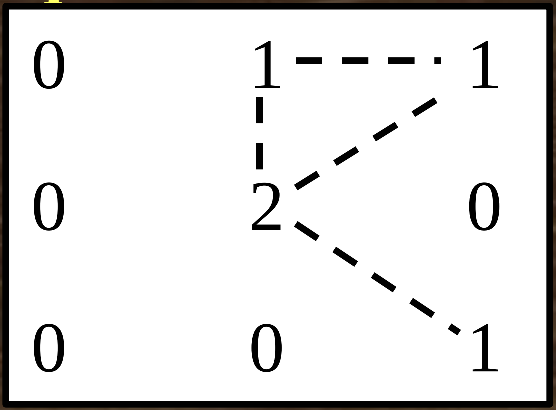
c. m-connected, **iff**

   i. q is in N_4(p) or

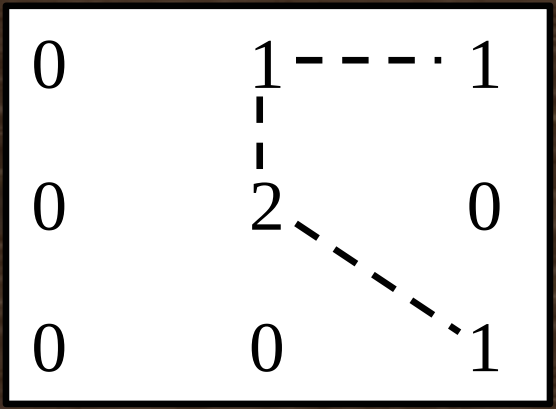
   ii. q is in N_D(p) and the set $N_4(p) \cap N_4(q)$ is empty



a.



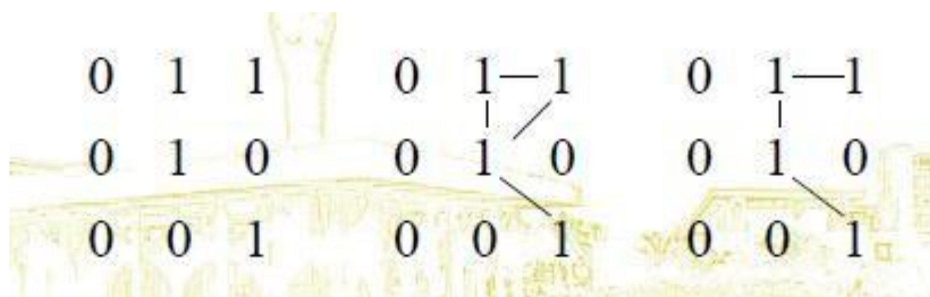
b.



c.

Types of Adjacency

- In this example, we can note that to connect between two pixels (finding a path between two pixels):
  - In 8-adjacency way, you can find multiple paths between two pixels
  - While, in m-adjacency, you can find only one path between two pixels
- So, m-adjacency has eliminated the multiple path connection that has been generated by the 8-adjacency.



- A *path* from pixel p with coordinates (x, y) to pixel q with coordinates (s, t) is a sequence of distinct pixels with coordinates:

  $(x_0, y_0), (x_1, y_1), (x_2, y_2) \ldots (x_n, y_n)$,

  where $(x_0, y_0)=(x, y)$ and $(x_n, y_n)=(s, t)$;

  (x_i, y_i) is **adjacent** to (x_{i-1}, y_{i-1}) $1 \le i \le n$

- Here n is the *length* of the path.

- We can define 4-, 8-, and m-paths based on type of adjacency used.

- **If *p* and *q* are pixels of an image subset S then *p* is *connected* to *q* in S if there is a path from *p* to *q* consisting entirely of pixels in S.**
- **For every pixel *p* in S, the set of pixels in S that are connected to *p* is called a *connected component* of S.**
- **If S has only one connected component then S is called *Connected Set.***

Regions and Boundaries

- A subset R of pixels in an image is called a *Region* of the image if R is a connected set.
- The *boundary* of the region R is the set of pixels in the region that have one or more neighbors that are not in R.
- If R happens to be entire Image?

Distance measures

Given pixels *p*, *q* and *z* with coordinates *(x, y)*, *(s, t)*, *(u, v)* respectively, the distance function D has following properties:

a. $D(p, q) \geq 0$ [$D(p, q) = 0$, iff $p = q$]

b. $D(p, q) = D(q, p)$

c. $D(p, z) \leq D(p, q) + D(q, z)$

The following are the different Distance measures:

- **Euclidean Distance :**
  $D_e(p, q) = [(x-s)^2 + (y-t)^2]$

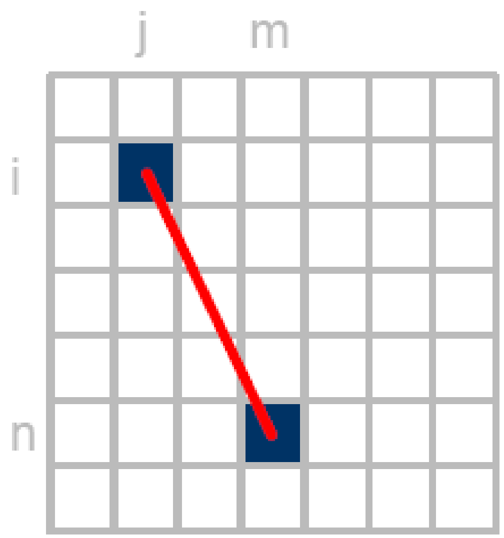
b. **City Block Distance:** →
   $D_4(p, q) = |x-s| + |y-t|$

		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

c. **Chess Board Distance:** →
   $D_8(p, q) = max(|x-s|, |y-t|)$

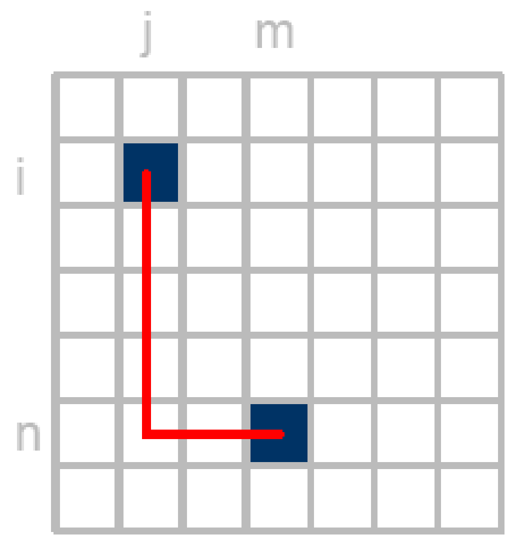
2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Distance measures



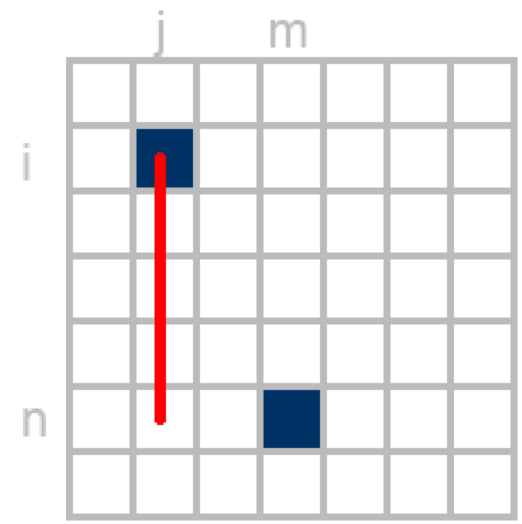
Euclidean Distance

$$=\sqrt{(i-n)^2 + (j-m)^2}$$



City Block Distance

$$= |i-n| + |j-m|$$



Chessboard Distance

$$= \max\left[\, |i-n|, |j-m| \,\right]$$

Example:

Compute the distance between the two pixels using the three distances :

q:(1,1)

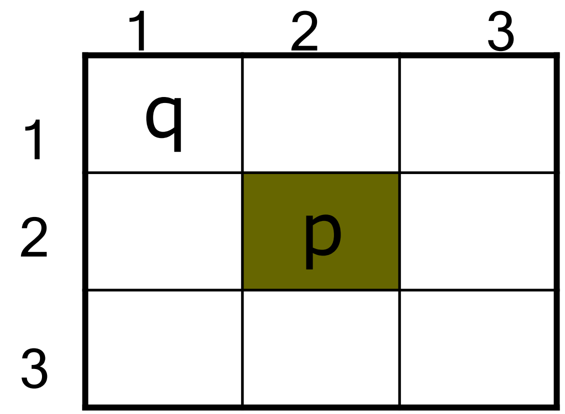
P: (2,2)

Euclidian distance : $((1-2)^2+(1-2)^2)^{1/2}$ = sqrt(2).

D4(City Block distance): |1-2| +|1-2| =2

D8(chessboard distance) : max(|1-2|,|1-2|)= 1

(because it is one of the 8-neighbors)



Distance measures

Example :

Use the city block distance to prove 4-neighbors ?

	1	2	3
1		d	
2	a	p	c
3		b	

Pixel A : | 2-2| + |1-2| = 1

Pixel B: | 3-2|+|2-2|= 1

Pixel C: |2-2|+|2-3| =1

Pixel D: |1-2| + |2-2| = 1

Now as a homework try the chessboard distance to proof the 8- neighbors!!!!

Arithmetic/Logic Operations:

- **Addition :** $p + q$
- **Subtraction:** $p - q$
- **Multiplication:** $p * q$
- **Division:** p / q
- **AND:** p **AND** q
- **OR :** p **OR** q

Neighborhood based arithmetic/Logic :

Value assigned to a pixel at position '*e*' is a function of its neighbors and a set of window functions.

		⋮		
	a	b	c	
⋯	d	e	f	⋯
	g	h	i	
		⋮		

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

$$p = (\mathbf{w}_1\mathbf{a} + w_2\mathbf{b} + w_3\mathbf{c} + w_4\mathbf{d} + w_5\mathbf{e} + w_6\mathbf{f} + w_7\mathbf{g} + w_8\mathbf{h} + w_9\mathbf{i})$$

$$= \sum w_i f_i$$

- **Tasks done using neighborhood processing:**
  - **Smoothing / averaging**
  - **Noise removal / filtering**
  - **Edge detection**
  - **Contrast enhancement**

What is convolution?

- Convolution is a general purpose filter effect for images.
- Is a matrix applied to an image and a mathematical operation comprised of integers
- It works by determining the value of a central pixel by adding the weighted values of all its neighbors together
- The output is a new modified filtered image

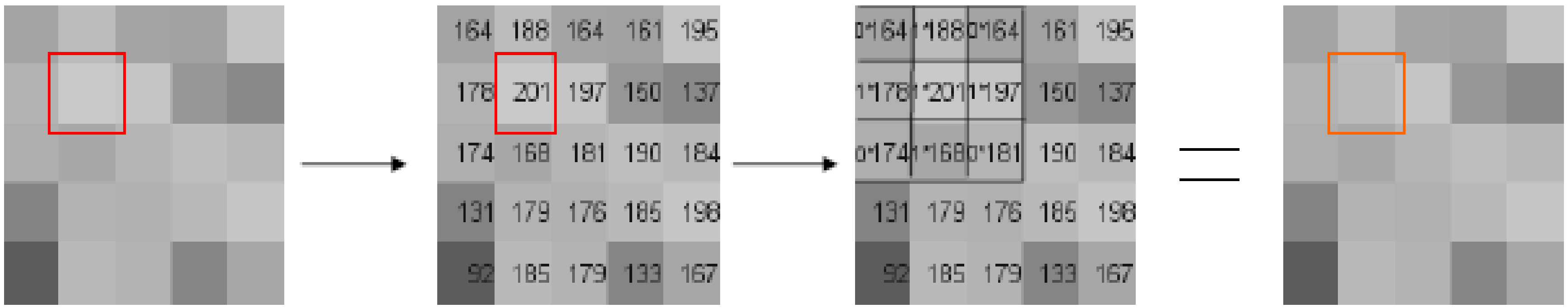
The process of image convolution

- A convolution is done by multiplying a pixel's and its neighboring pixels color value by a matrix
- **Kernel:** A kernel is a (usually) small matrix of numbers that is used in image convolutions.
  - Differently sized kernels containing different patterns of numbers produce different results under convolution.
  - The size of a kernel is arbitrary but 3x3 is often used

Example kernel:

0	1	0
1	1	1
0	1	0

Example



Original image

Image with color values placed over it

Image with 3x3 kernel placed over it

Output image

164	188	164
178	201	197
174	168	181

Color values

×

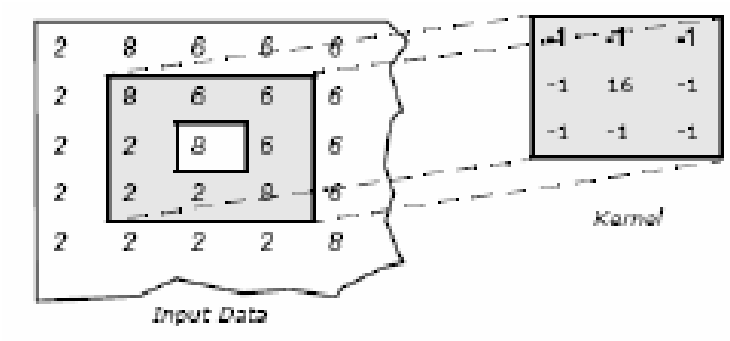
0	1	0
1	1	1
0	1	0

Kernel

Divided by the sum of the kernel

932\5 = new pixel color

More examples



integer [(-1 × 8) + (-1 × 6) + (-1 × 6) + (-1 × 2) + (16 × 8) + (-1 × 6) +
(-1 × 2) + (-1 × 2) + (-1 × 8) ÷ (-1 + -1 + -1 + -1 + 16 + -1 + -1 + -1 + -1)]

= int [(128-40) / (16-8)]

= int (88 / 8) = int (11) = 11

Some other kernel examples

1	1	1
1	1	1
1	1	1

Unweighted 3x3 smoothing kernel

0	1	0
1	4	1
0	1	0

Weighted 3x3 smoothing kernel with Gaussian blur

0	-1	0
-1	5	-1
0	-1	0

Kernel to make image sharper

-1	-1	-1
-1	9	-1
-1	-1	-1

Intensified sharper image

Gaussian Blur

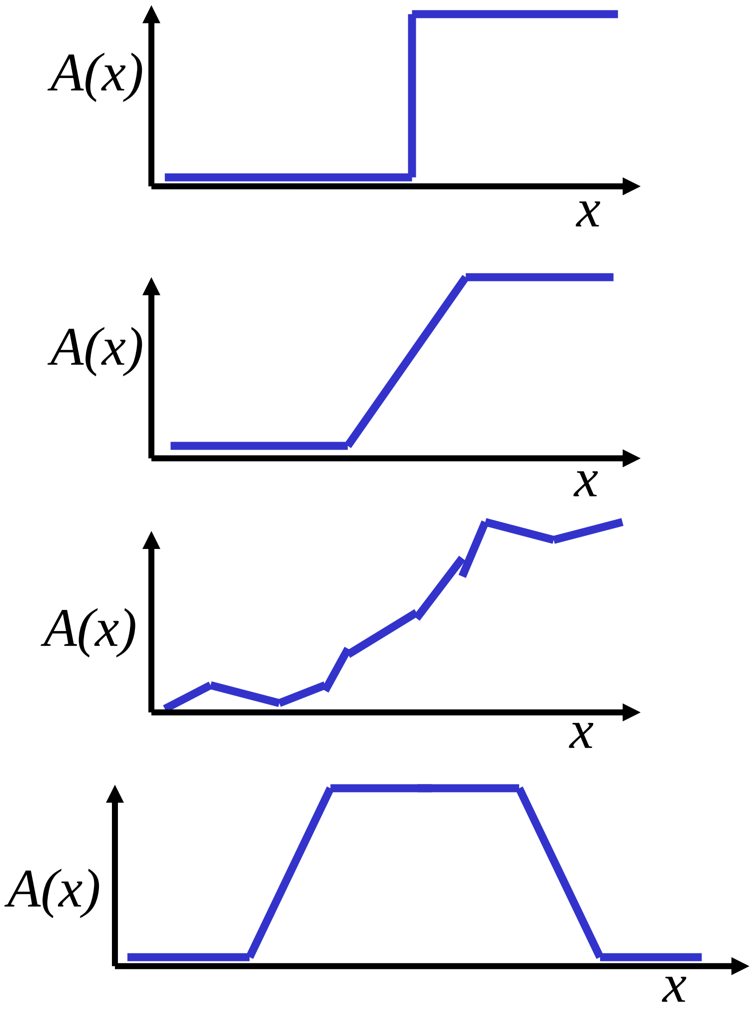
Sharpened image

What are edges ?

- Change, or discontinuity, in image brightness between two reasonably smooth regions.
- Fundamentally important primitive image characteristics.
- Only information in most black and white images.

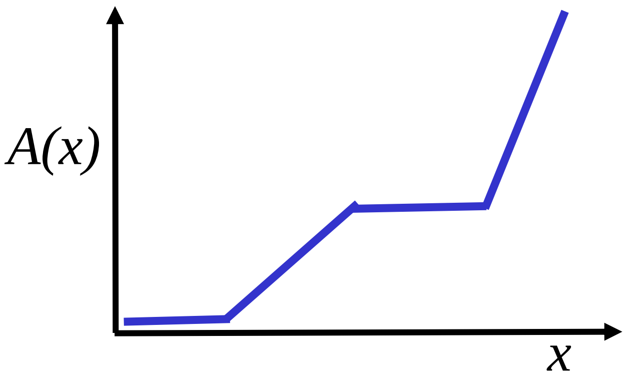
What are edges?

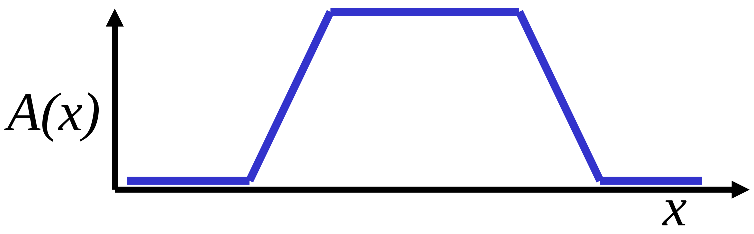
- Ideal edge
- It is usually *ramped* because of sensor processing during capture
- Noisy edge
- Line



Edge Properties

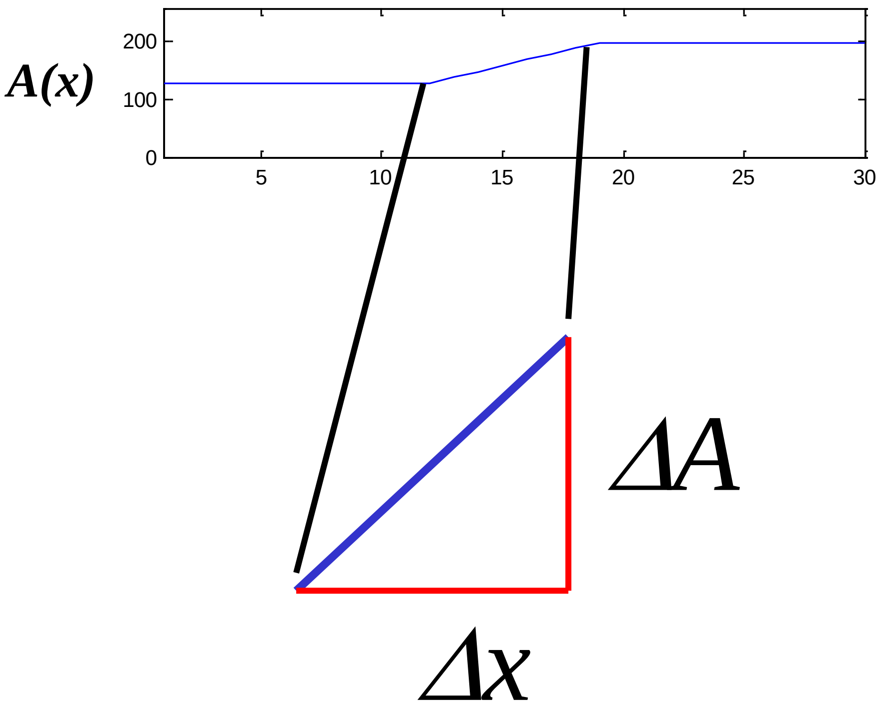
- Edge has two properties:
  - –how steep it is
  - –direction, ie, is it pointing towards the left or right?





Edge Properties-gradient

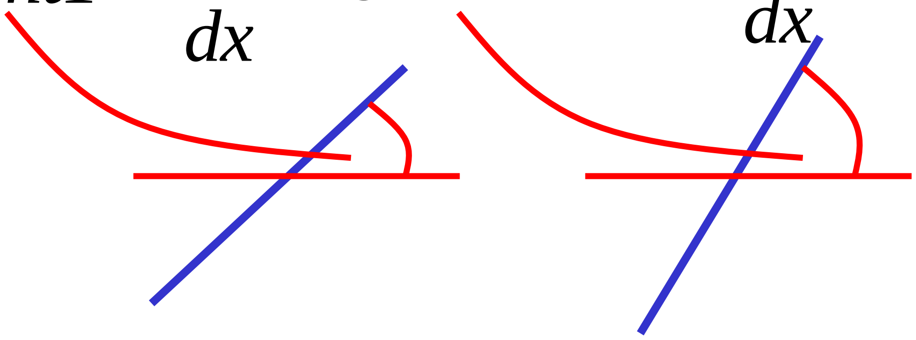
- Consider a 1-d continuous image of an edge, denoted by *A(x)*
- Edge properties can be obtained from the gradient = $\Delta A / \Delta x$
- gradient=dA/dx as $\Delta x \rightarrow 0$.



Edge Properties

- Gradient has two properties
  - magnitude
  - direction
- Magnitude, or steepness, given by $|dA/dx|$
- Direction, left or right, given by sign of $dA/d\,x$

$$gradient1 = \frac{dA_1}{dx} \qquad gradient2 = \frac{dA_2}{dx}$$

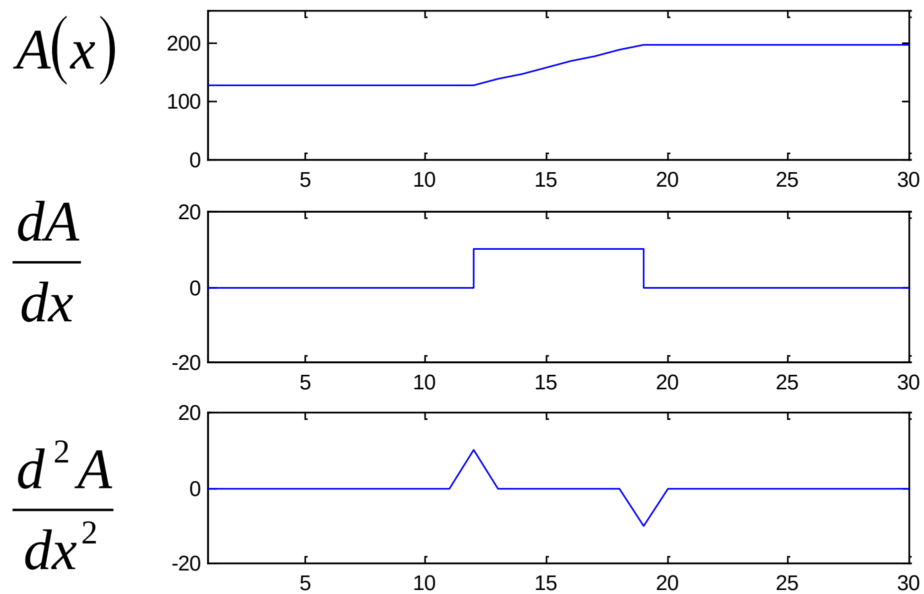


$$\left|\frac{dA_2}{dx}\right| > \left|\frac{dA_1}{dx}\right|$$

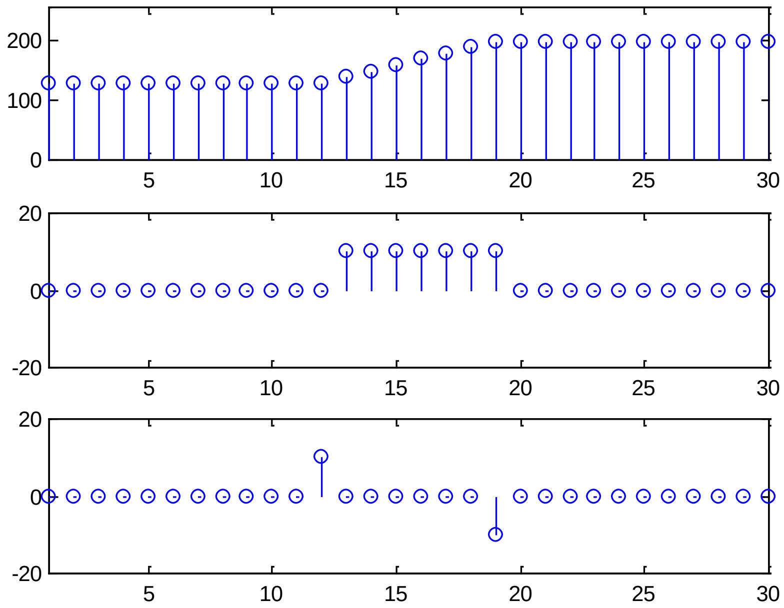
$$\operatorname{sgn}\left(\frac{dA_2}{dx}\right) = \operatorname{sgn}\left(\frac{dA_1}{dx}\right)$$

Edge Properties- gradient

- Gradient given by first derivative dA/dx.
- Second derivative, d^2A/dx^2, generates two peaks at beginning and end of edge.
- Called 'ringing'.



Edge Properties-discrete gradient



$$A(x) = A_i$$

B=[-1 1]

$$\frac{dA}{dx} \approx A_{i+1} - A_i = \Delta A_i$$

$$\frac{d^2A}{dx^2} \approx \Delta A_{i+1} - \Delta A_i$$

B=[1 -2 1]

$$= A_{i+2} - A_{i+1} - \left(A_{i+1} - A_i\right)$$

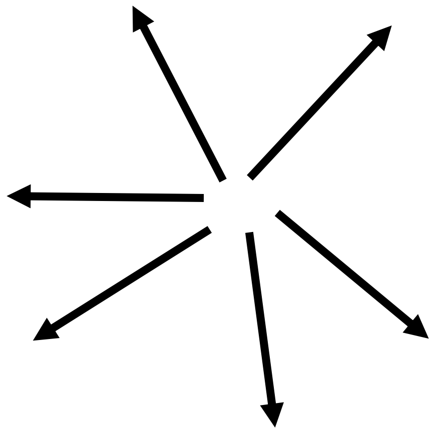
$$= A_{i+2} - 2A_{i+1} + A_i = \Delta^2 A_i$$

Neighbourhood Operators

- First derivative can be calculated by convolving with mask B=[-1 1].
- Second derivative can be calculated by convolving with mask B = [1 -2 1].

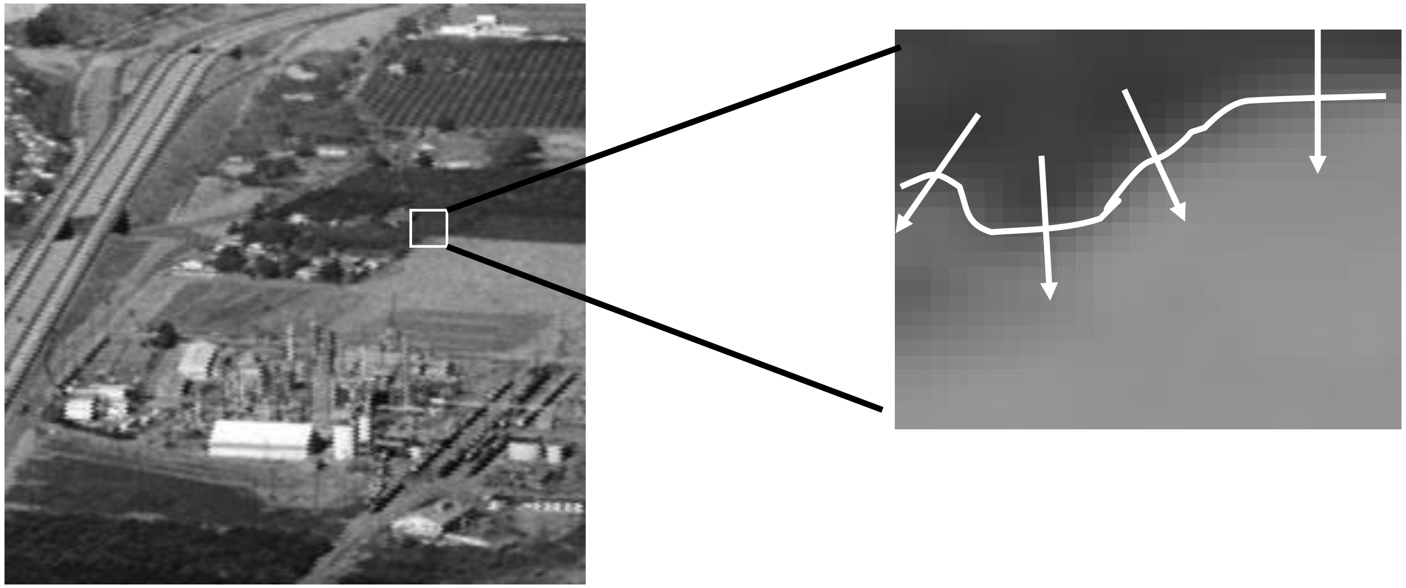
Edges in 2-D Images

- Edge properties are provided by gradient of image brightness *A(x,y)*
- 1-d case the gradient direction is either → or←
- 2-d gradient has a magnitude and orientation

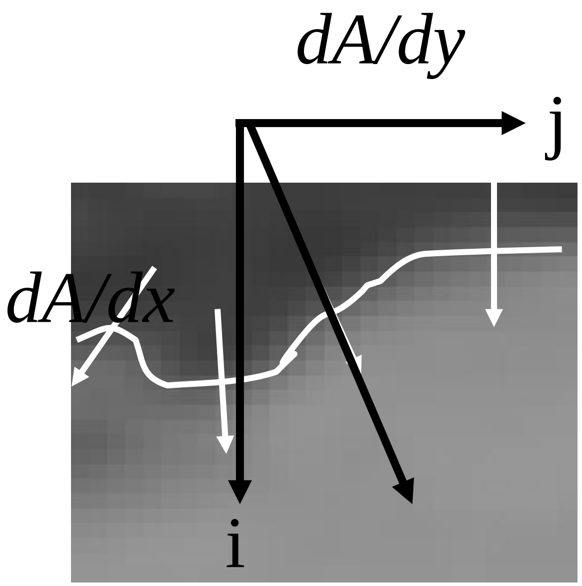


Edges in 2-D Images

- Direction of gradient at any point is the direction of maximum change.



2-d Gradient Operator



$$\Delta A = \frac{dA}{dx}\mathbf{i} + \frac{dA}{dy}\mathbf{j}$$

$$\text{Magnitude} = \sqrt{\left(\frac{dA}{dx}\right)^2 + \left(\frac{dA}{dy}\right)^2}$$

$$\text{Orientation} = \tan^{-1}\left(\frac{{dA}/{dy}}{{dA}/{dx}}\right)$$

Discrete 2-d gradient operator

$$A(x,y) = A_{i,j}$$

$$\frac{\partial A}{\partial x} \approx A_{i+1,j} - A_{i,j} = \Delta_i A_{i,j}$$

$$\frac{\partial A}{\partial y} \approx A_{i,j+1} - A_{i,j} = \Delta_j A_{i,j}$$

$$\Delta A_{i,j} = \Delta_i A_{i,j}\mathbf{i} + \Delta_j A_{i,j}\mathbf{j}$$

$\longrightarrow$

Neighbour hood operators

$$B_i = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$B_j = \begin{bmatrix} -1 & 1 \end{bmatrix}$$

Neighbourhood Operators

$$A(x,y) = A_{i,j}$$

$$\frac{\partial^2 A}{\partial x^2} \approx \Delta_i A_{i+1,j} - \Delta_i A_{i,j} = \Delta_i^2 A_{i,j}$$

$$= A_{i+2,j} - A_{i+1,j} - \left(A_{i+1,j} - A_{i,j}\right)$$

$$= A_{i+2,j} - 2A_{i+1,j} + A_{i,j}$$

$$\frac{\partial^2 A}{\partial y^2} \approx \Delta_j A_{i,j+1} - \Delta_j A_{i,j} = \Delta_j^2 A_{i,j}$$

$$= A_{i,j+2} - A_{i,j+1} - \left(A_{i,j+1} - A_{i,j}\right)$$

$$= A_{i,j+2} - 2A_{i,j+1} + A_{i,j}$$

$$\Longrightarrow$$

$$\Delta_i^2 \mathbf{A} = \mathbf{A} * \mathbf{B}_i$$

$$\mathbf{B}_i = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

$$\Delta_j^2 \mathbf{A} = \mathbf{A} * \mathbf{B}_j$$

$$\mathbf{B}_j = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

Neighbourhood Operators

$$\begin{aligned}
\Delta^2 \mathbf{A} &= \Delta_i^2 \mathbf{A} + \Delta_j^2 \mathbf{A} \\
&= \mathbf{A} * \mathbf{B}_i + \mathbf{A} * \mathbf{B}_j \\
&= \mathbf{A} * \left(\mathbf{B}_i + \mathbf{B}_j \right) \\
&= \mathbf{A} * \mathbf{B}
\end{aligned}$$

$$\mathbf{B} = \mathbf{B}_i + \mathbf{B}_j = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Laplacian Image

$$\mathbf{L} = -\Delta^2 \mathbf{A}$$

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