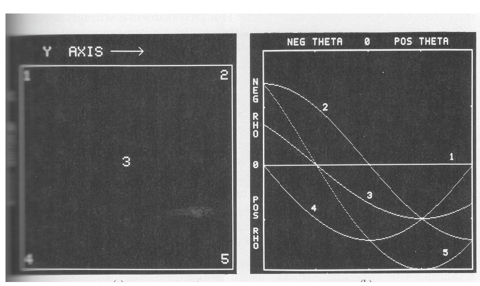
Lecture-12

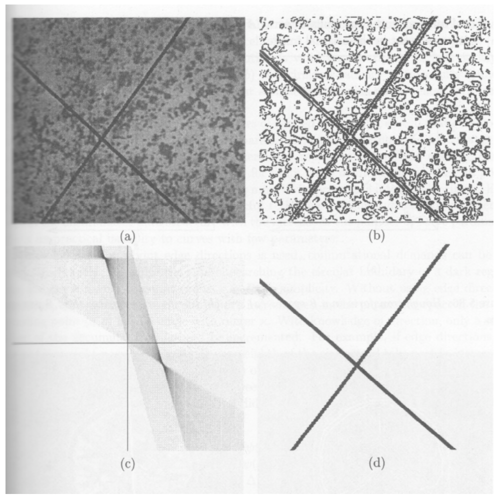
Hough Transform
Examples

Hough Space

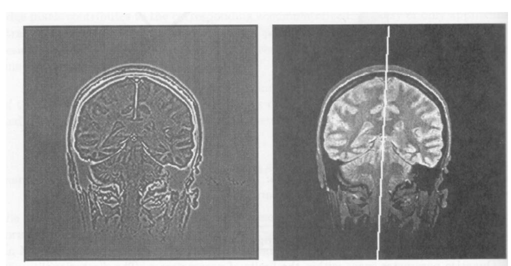


Theta is from -90 to +90

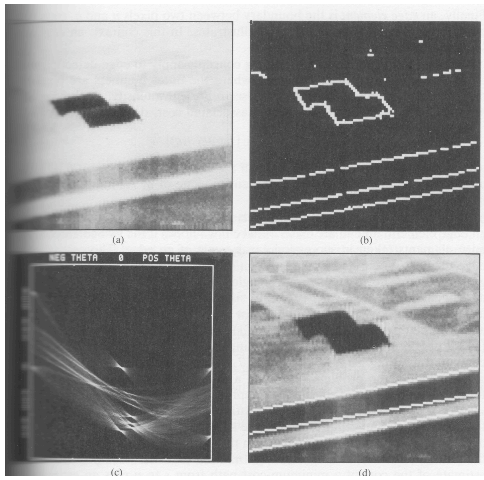
Fitting Lines In an Image



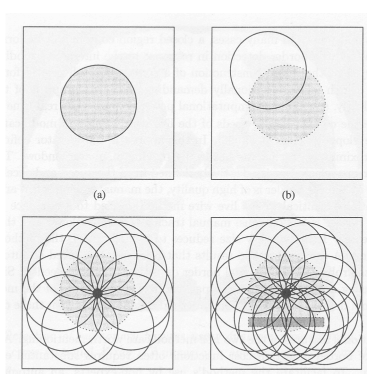
Fitting Lines In an Image



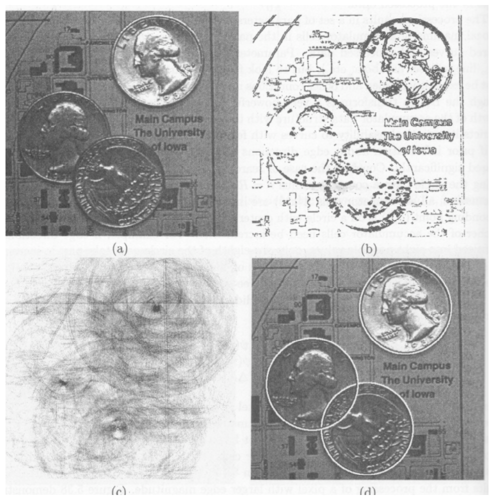
Fitting lines in an image



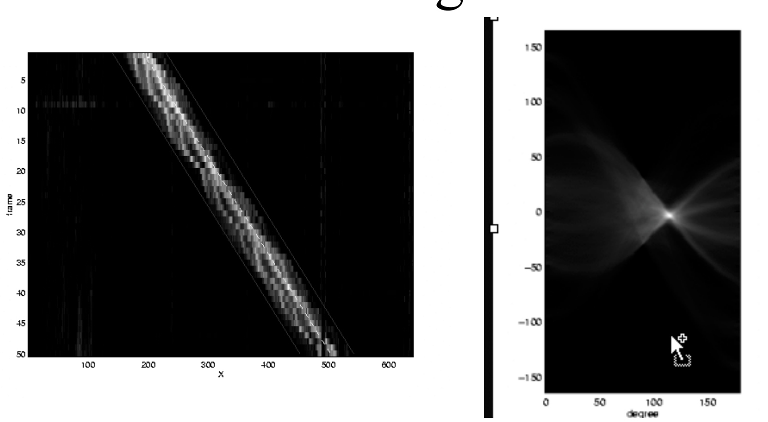
Fitting Circles



Fitting Circles



Detecting Lines in Gray Level Images

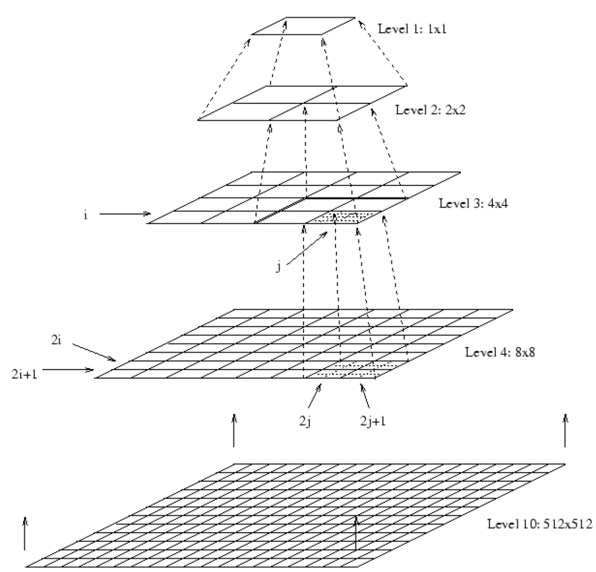


Detect yellow line in the middle
Use gray levels instead of edges
Increment the parameter space by gray level at a pixel instead of by 1.

Pyramids

- Very useful for representing images.
- Pyramid is built by using multiple copies of image.
- Each level in the pyramid is 1/4 of the size of previous level.
- The lowest level is of the highest resolution.
- The highest level is of the lowest resolution.

Pyramid

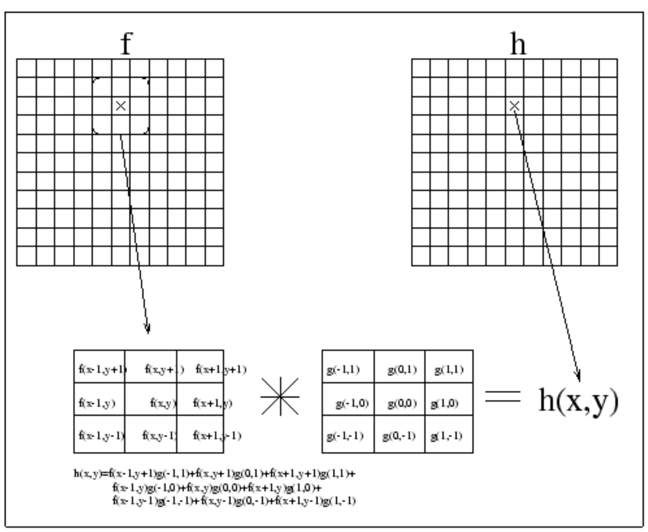


Gaussian Pyramids

$$g_l(i,j) = \sum_{m=-2}^{2}\sum_{n=-2}^{2} w(m,n)\, g_{l-1}(2i+m, 2j+n)$$

$$g_l = REDUCE[g_{l-1}]$$

Convolution



Gaussian Pyramids

$$g_{l,n}(i,j) = \sum_{p=-2}^{2}\sum_{q=-2}^{2} w(p,q) g_{l,n-1}\left(\frac{i-p}{2}, \frac{j-q}{2}\right)$$

$$g_{l,n} = EXPAND[g_{l,n-1}]$$

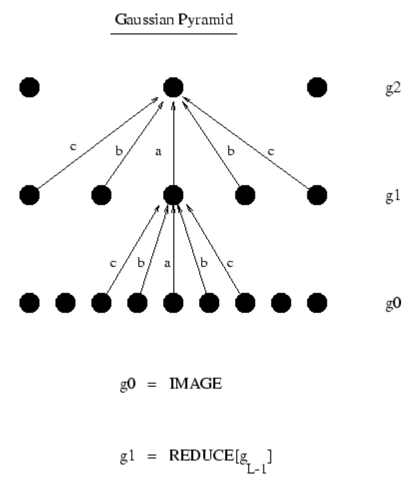
Reduce (1D)

$$g_l(i) = \sum_{m=-2}^{2} \hat{w}(m) g_{l-1}(2i+m)$$

$$g_l(2) = \hat{w}(-2)g_{l-1}(4-2) + \hat{w}(-1)g_{l-1}\hat{w}(4-1) + \hat{w}(0)g_{l-1}(4) + \hat{w}(1)g_{l-1}(4+1) + \hat{w}(2)g_{l-1}(4+2)$$

$$g_l(2) = \hat{w}(-2)g_{l-1}(2) + \hat{w}(-1)g_{l-1}\hat{w}(3) + \hat{w}(0)g_{l-1}(4) + \hat{w}(1)g_{l-1}(5) + \hat{w}(2)g_{l-1}(6)$$

Reduce



Expand (1D)

$$g_{l,n}(i) = \sum_{p=-2}^{2} \hat{w}(p)\, g_{l,n-1}\left(\frac{i-p}{2}\right)$$

$$g_{l,n}(4) = \hat{w}(-2)\, g_{l,n-1}\left(\frac{4-2}{2}\right) + \hat{w}(-1)\, g_{l,n-1}\left(\frac{4-1}{2}\right) +$$

$$\hat{w}(0)\, g_{l,n-1}\left(\frac{4}{2}\right) + \hat{w}(1)\, g_{l,n-1}\left(\frac{4+1}{2}\right) + \hat{w}(2)\, g_{l,n-1}\left(\frac{4+2}{2}\right)$$

$$g_{l,n}(4) = \hat{w}(-2)\, g_{l,n-1}(1) + \hat{w}(0)\, g_{l,n-1}(2) + \hat{w}(2)\, g_{l,n-1}(3)$$

Expand (1D)

$$g_{l,n}(i) = \sum_{p=-2}^{2} \hat{w}(p) g_{l,n-1}\left(\frac{i-p}{2}\right)$$

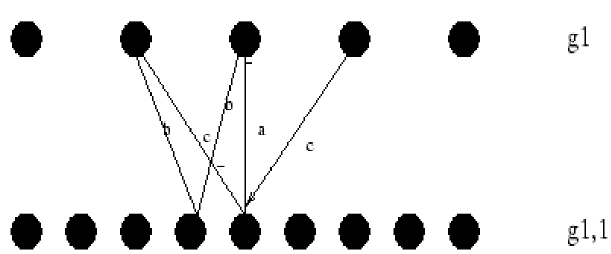
$$g_{l,n}(3) = \hat{w}(-2) g_{l,n-1}\left(\frac{3-2}{2}\right) + \hat{w}(-1) g_{l,n-1}\left(\frac{3-1}{2}\right) +$$

$$\hat{w}(0) g_{l,n-1}\left(\frac{3}{2}\right) + \hat{w}(1) g_{l,n-1}\left(\frac{3+1}{2}\right) + \hat{w}(2) g_{l,n-1}\left(\frac{3+2}{2}\right)$$

$$g_{l,n}(3) = \hat{w}(-1) g_{l,n-1}(1) + \hat{w}(1) g_{l,n-1}(2)$$

Expand

Gaussian Pyramid



g1,1 = EXPAND[g 1]

Convolution Mask

$$[w(-2), w(-1), w(0), w(1), w(2)]$$

Convolution Mask

- Separable

$$w(m,n) = \hat{w}(m)\hat{w}(n)$$

- Symmetric

$$\hat{w}(i) = \hat{w}(-i)$$

$$[c, b, a, b, c]$$

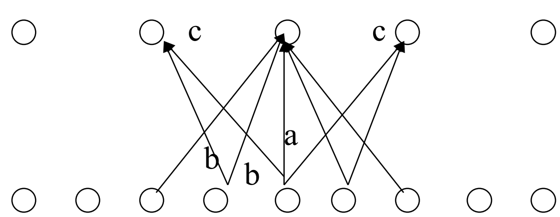
Convolution Mask

- The sum of mask should be 1.

$$a + 2b + 2c = 1$$

•All nodes at a given level must contribute the same total weight to the nodes at the next higher level.

$$a + 2c = 2b$$



Convolution Mask

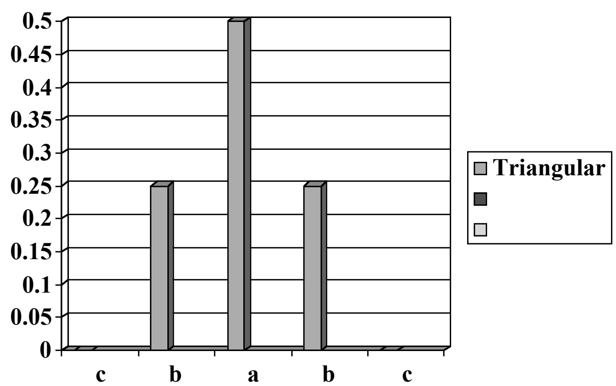
$$\hat{w}(0) = a$$

$$\hat{w}(-1) = \hat{w}(1) = \frac{1}{4}$$

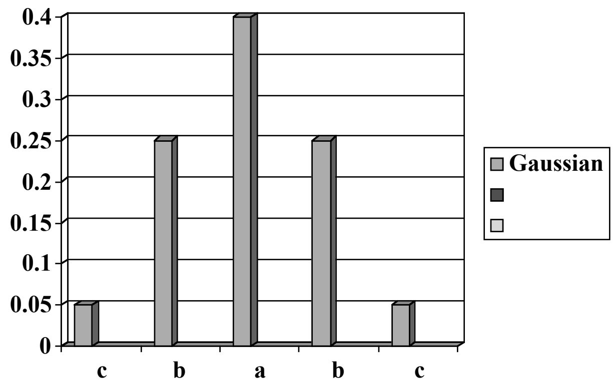
$$\hat{w}(-2) = \hat{w}(2) = \frac{1}{4} - \frac{a}{2}$$

a=.4 GAUSSIAN, a=.5 TRINGULAR

Triangular



Approximate Gaussian



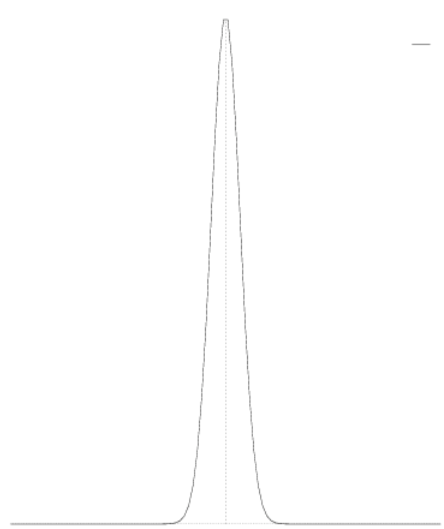
Gaussian

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

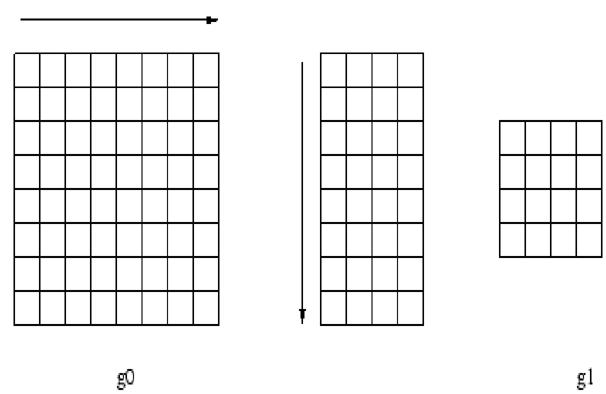
Gaussian

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

x	-3	-2	-1	0	1	2	3
g(x)	.011	.13	.6	1	.6	.13	.011



Separability



Algorithm

- Apply 1-D mask to alternate pixels along each row of image.
- Apply 1-D mask to alternate pixel along each column of resultant image from previous step.

Gaussian Pyramid

Laplacian Pyramids

- Similar to edge detected images.
- Most pixels are zero.
- Can be used for image compression.

$$L_1 = g_1 - EXPAND[g_2]$$

$$L_2 = g_2 - EXPAND[g_3]$$

$$L_3 = g_3 - EXPAND[g_4]$$

536

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983

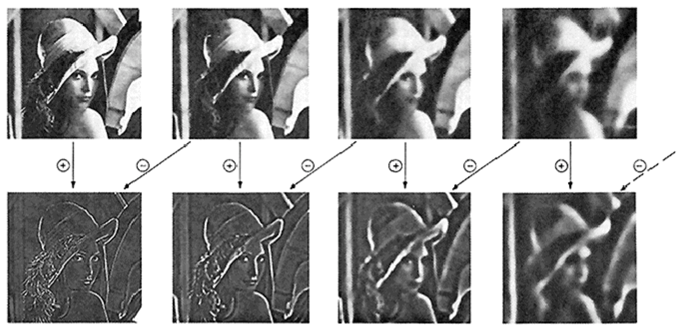


Fig 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

Coding using Laplacian Pyramid

- Compute Gaussian pyramid

$$g_1, g_2, g_3, g_4$$

- Compute Laplacian pyramid

$$L_1 = g_1 - EXPAND[g_2]$$
$$L_2 = g_2 - EXPAND[g_3]$$
$$L_3 = g_3 - EXPAND[g_4]$$
$$L_4 = g_4$$

- Code Laplacian pyramid

Decoding using Laplacian pyramid

- Decode Laplacian pyramid.
- Compute Gaussian pyramid from Laplacian pyramid.

$$g_4 = L_4$$

$$g_3 = EXPAND[g_4] + L_3$$

$$g_2 = EXPAND[g_3] + L_2$$

$$g_1 = EXPAND[g_2] + L_1$$

- is reconstructed image.

Laplacian Pyramid

536

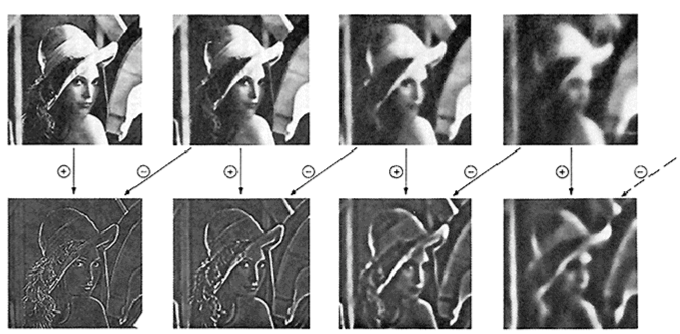


Fig 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-31, NO. 4, APRIL 1983

Image Compression (Entropy)

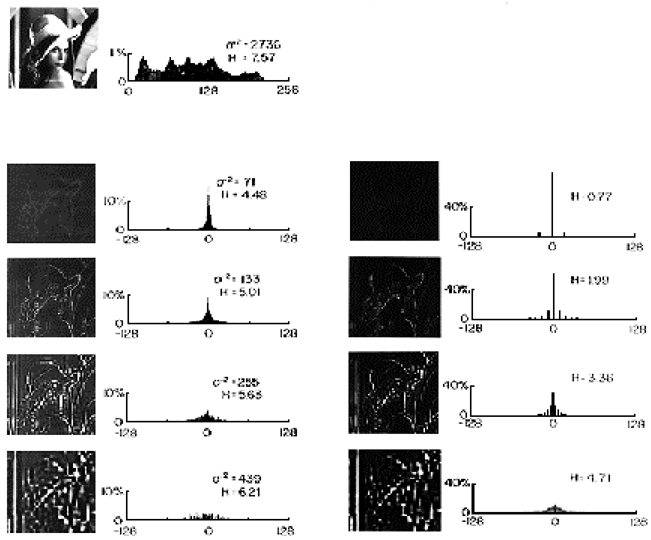
7.6

4.4 .77

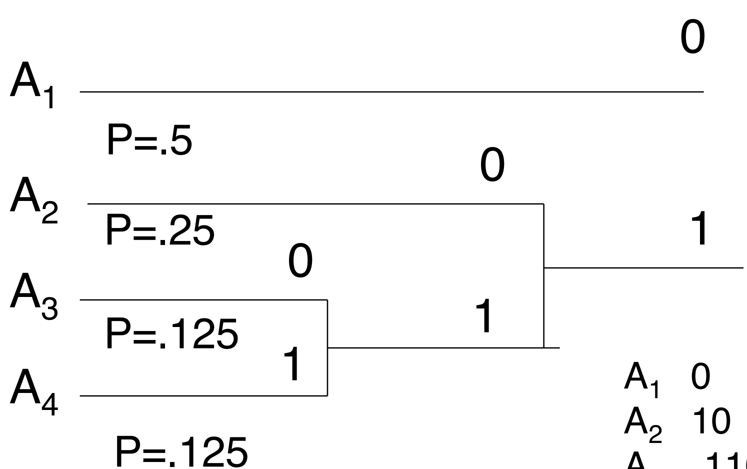
5.0 1.9

5.6 3.3

6.2 4.2



Huffman Coding (Example-1)



A_1	0
A_2	10
A_3	110
A_4	111

Huffman Coding

Entropy $$H = -\sum_{i=0}^{255} p(i)\log_2 p(i)$$

$$H = -.5\log .5 - .25\log .25 - .125\log .125 -$$
$$.125\log .125 = 1.75$$

Image Compression

1.58

1


(a)

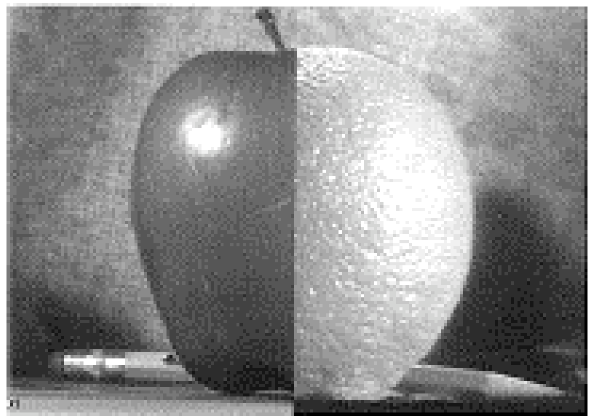

(b)

.73

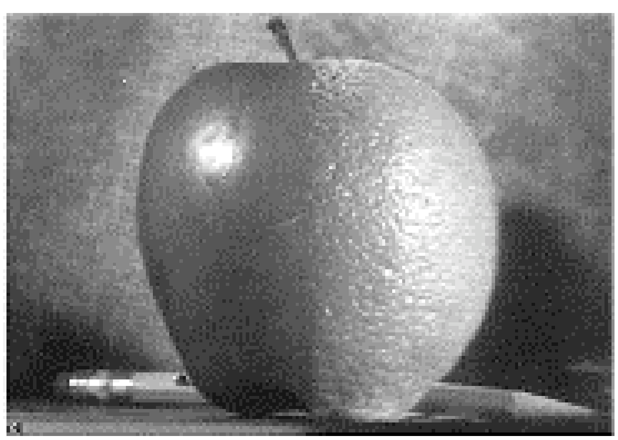

(c)


(d)

Combining Apple & Orange



Combining Apple & Orange

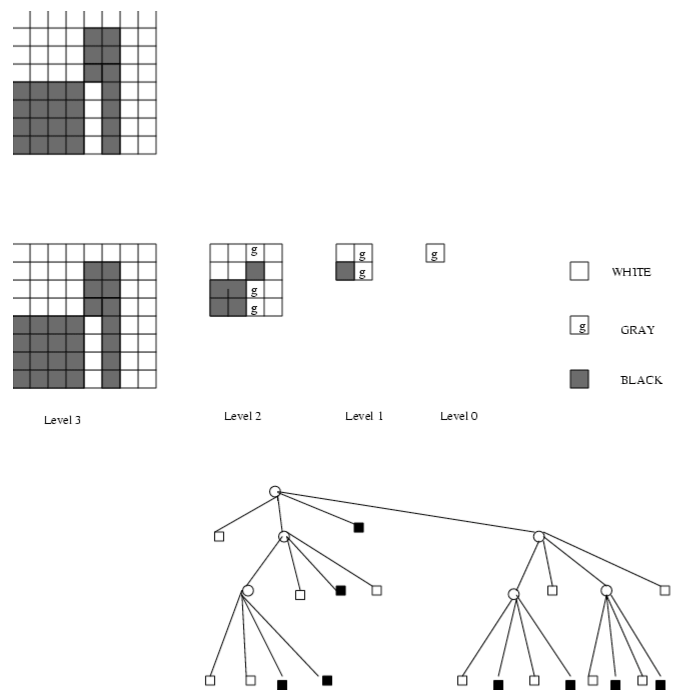


Algorithm

- Generate Laplacian pyramid Lo of orange image.
- Generate Laplacian pyramid La of apple image.
- Generate Laplacian pyramid Lc by copying left half of nodes at each level from apple and right half of nodes from orange pyramids.
- Reconstruct combined image from Lc.

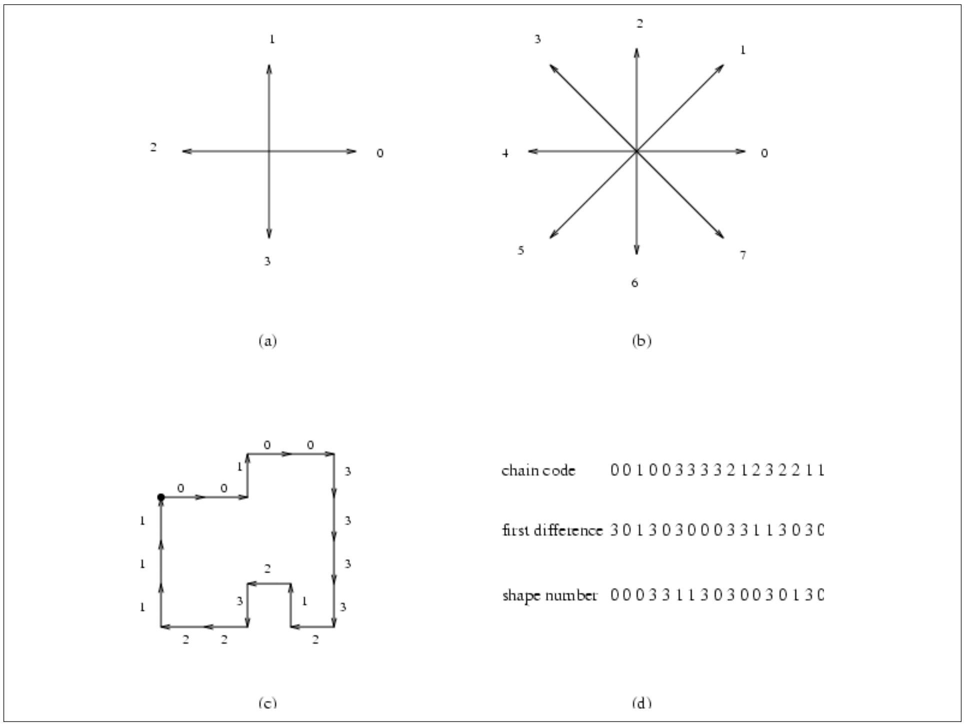
Quad Trees

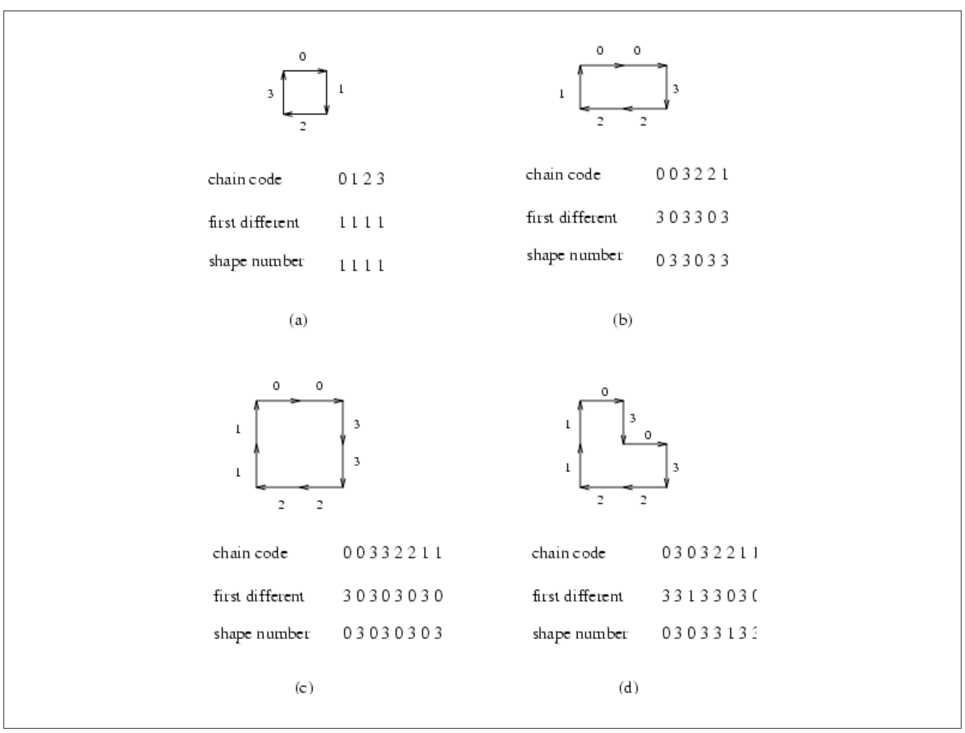
- Data structure to represent regions
- Three types of nodes: gray, black and white
- First generate the pyramid, then:
- If type of pyramid is black or white then return else
  - Recursively find quad tree of SE quadrant
  - Recursively find quad tree of SW quadrant
  - Recursively find quad tree of NE quadrant
  - Recursively find quad tree of NW quadrant
  - Return



Chain Code

- A simple technique to represent a shape of boundary.
- Each directed line segment is assigned a code.
- Chain code is integer obtained by putting together the codes of all consecutive line segments.
- Shape number is a normalized chain code, which is invariant to translation and rotation.


(a)
(b)
chain code 0 0 1 0 0 3 3 3 3 2 1 2 3 2 2 1 1
first difference 3 0 1 3 0 3 0 0 0 3 3 1 1 3 0 3 0
shape number 0 0 0 3 3 1 1 3 0 3 0 0 3 0 1 3 0
(c)
(d)


chain code 0 1 2 3
first different 1 1 1 1
shape number 1 1 1 1
(a)
chain code 0 0 3 2 2 1
first different 3 0 3 3 0 3
shape number 0 3 3 0 3 3
(b)
chain code 0 0 3 3 2 2 1 1
first different 3 0 3 0 3 0 3 0
shape number 0 3 0 3 0 3 0 3
(c)
chain code 0 3 0 3 2 2 1 1
first different 3 3 1 3 3 0 3 0
shape number 0 3 0 3 3 1 3 3
(d)