Pyramids

Lecture-7

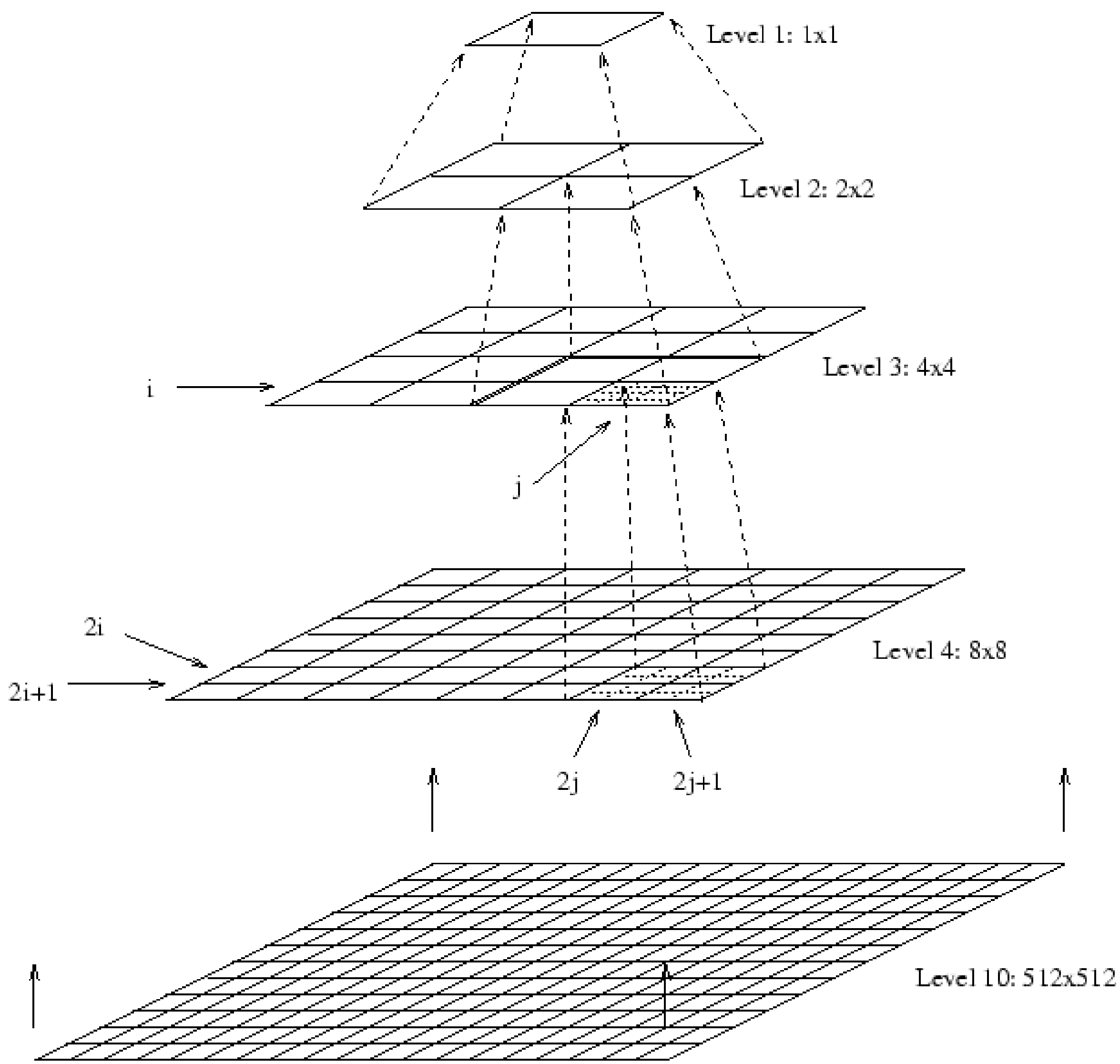
Contents

- Gaussian and Laplacian Pyramids
  - Reduce
  - Expand
- Applications of Laplacian pyramids
  - Image compression
  - Image composting
- Optical flow using Pyramids
  - interpolation

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



Level 1: 1x1
Level 2: 2x2
Level 3: 4x4
i
j
2i
2i+1
Level 4: 8x8
2j
2j+1
Level 10: 512x512

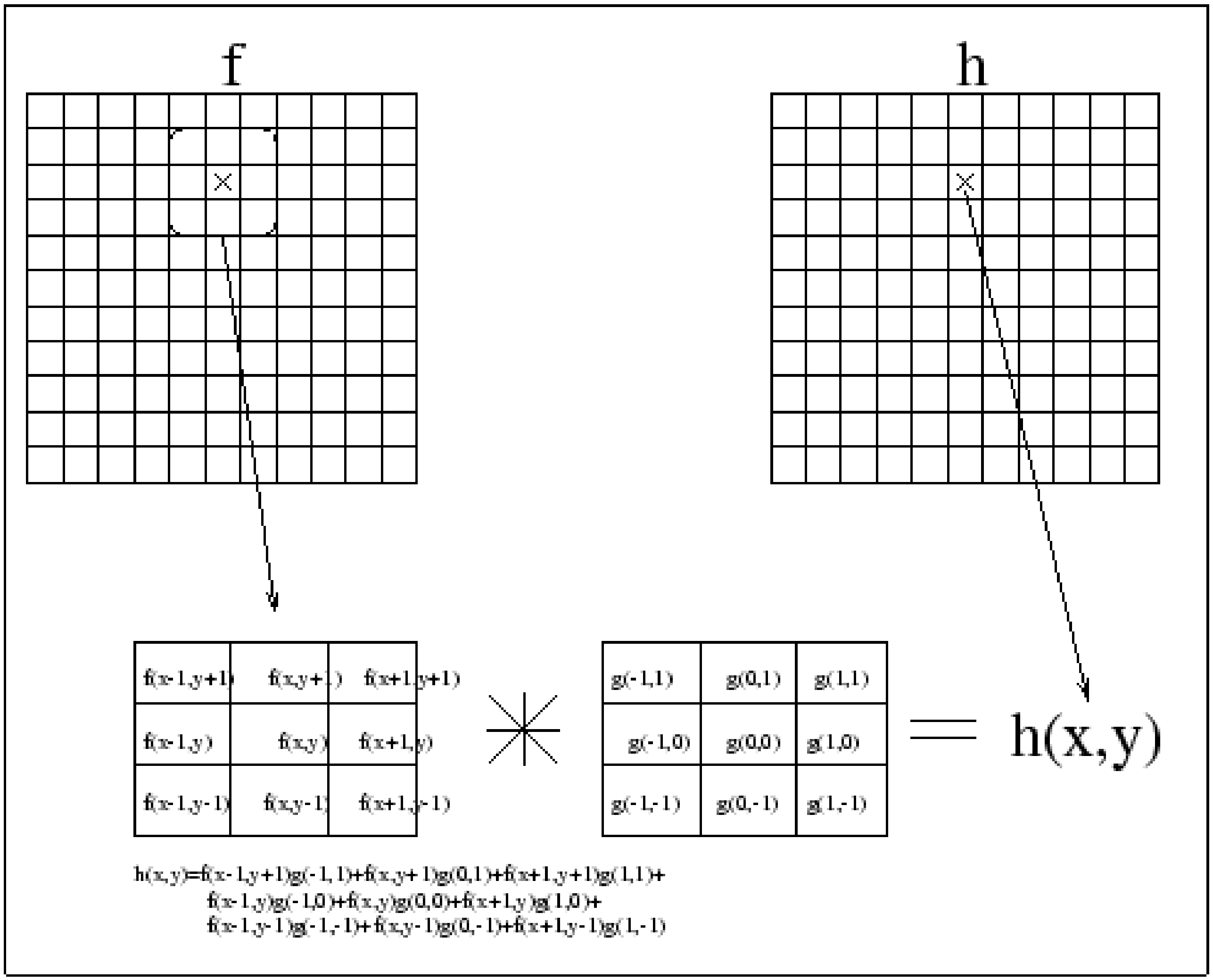
Gaussian Pyramids (reduce)

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

Level l

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

Convolution



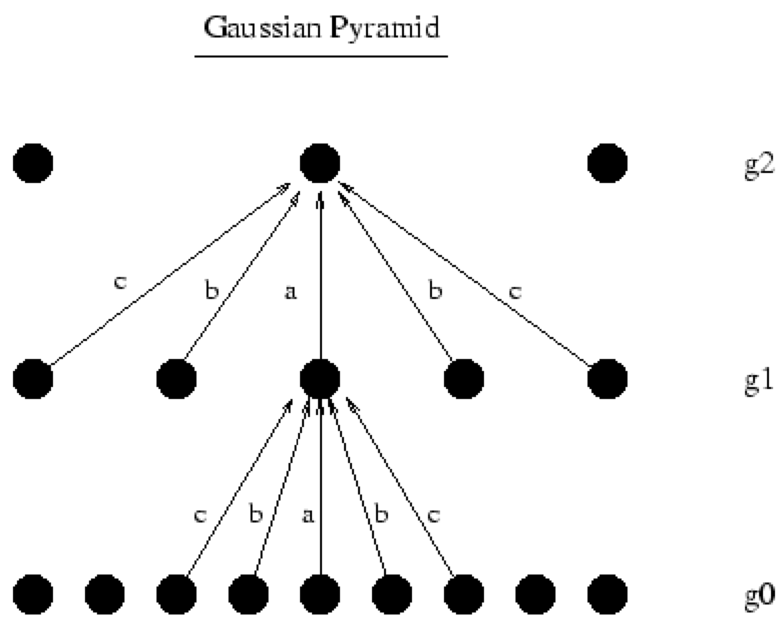
Reduce (1D)

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

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

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

Reduce



0

g0 = IMAGE

g1 = REDUCE[g_{L-1}]

Gaussian Pyramids (expand)

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

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

Expand (1D)

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

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

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

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

Expand (1D)

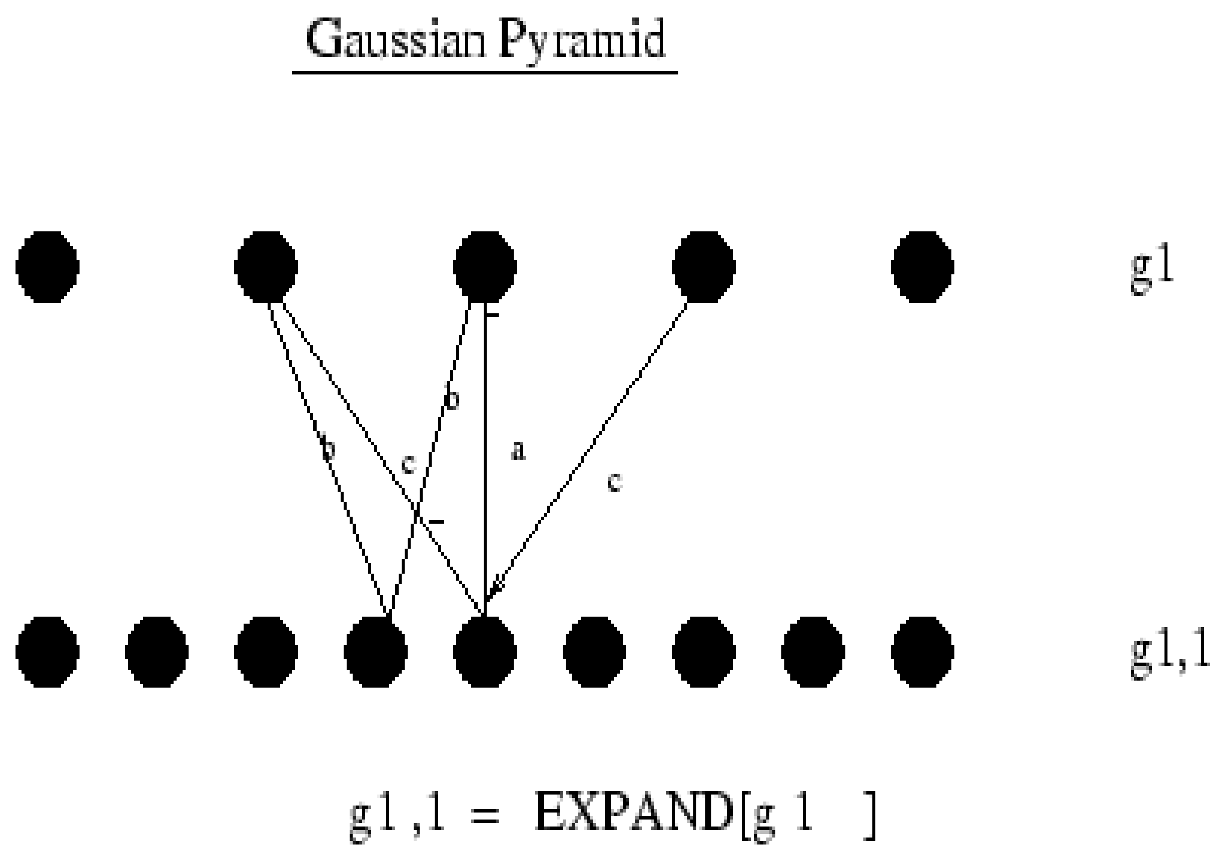
$$g_{l,n}(i) = \sum_{p=-2}^{2} \hat{w}(p) g_{l,n-1}\left(\frac{i-p}{2}\right)$$

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

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

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

Expand



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

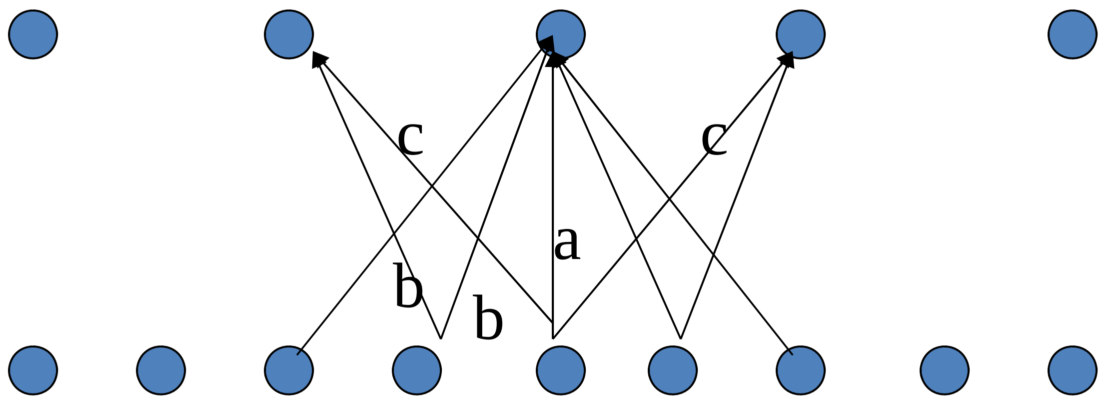
- The sum of mask should be 1.

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

•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



$$a + 2c = 2b$$

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

$$b = \frac{1}{4}$$

$$c = \frac{1}{2}(2b - a)$$

$$c = \frac{1}{4} - \frac{a}{2}$$

Convolution Mask

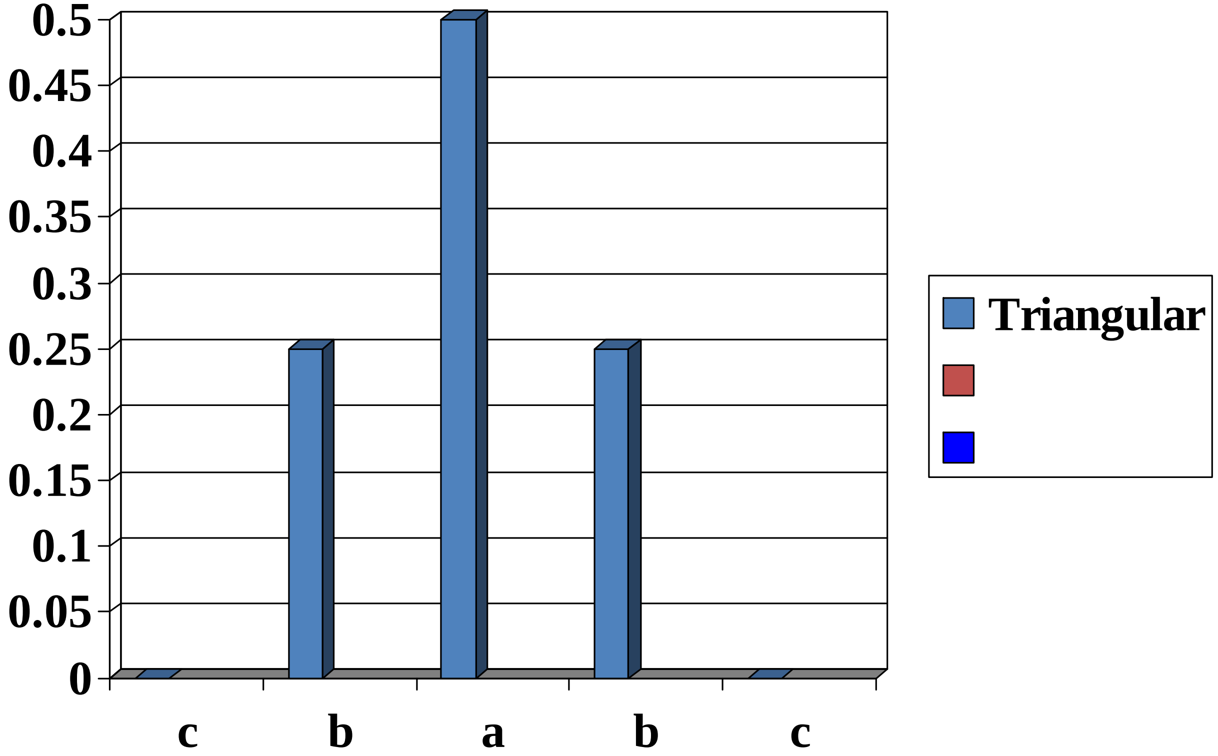
$$\hat{w}(0) = a$$

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

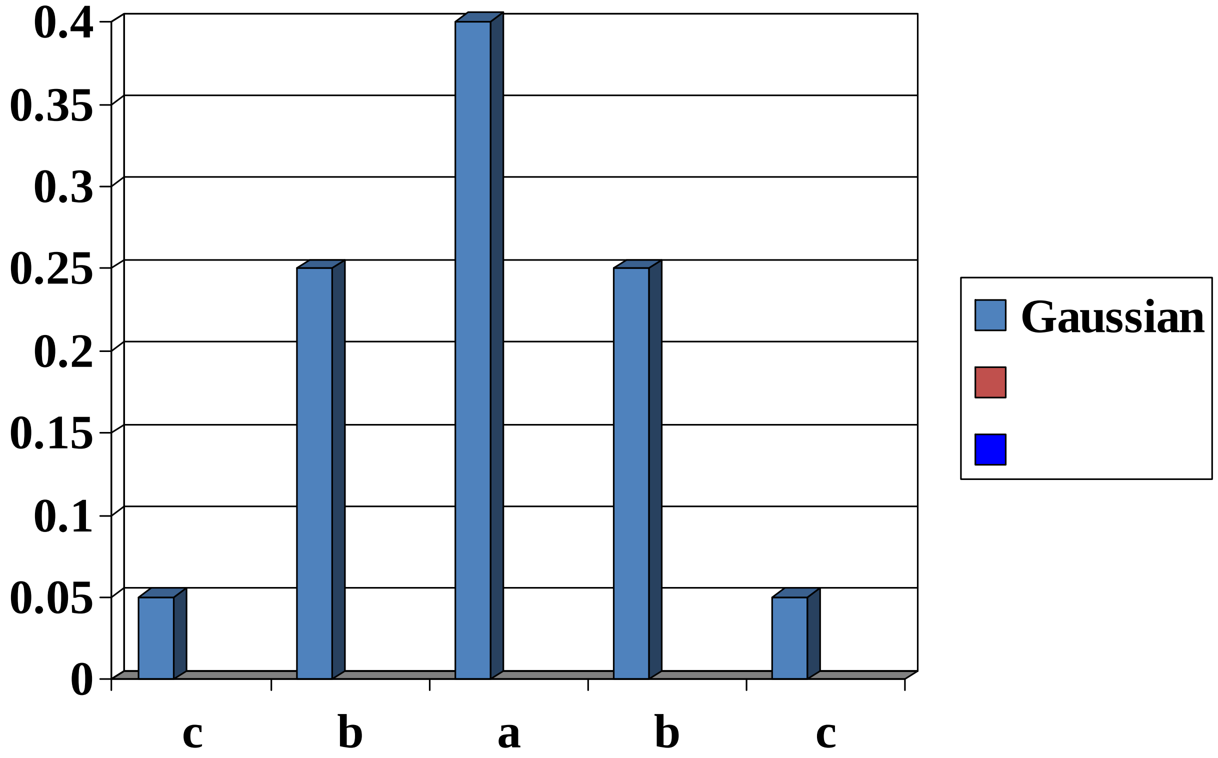
$$\hat{w}(-2) = \hat{w}(2) = \frac{1}{4} - \frac{a}{2}$$

a=.4 GAUSSIAN, a=.5 TRINGULAR

Triangular



Approximate Gaussian



Gaussian

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

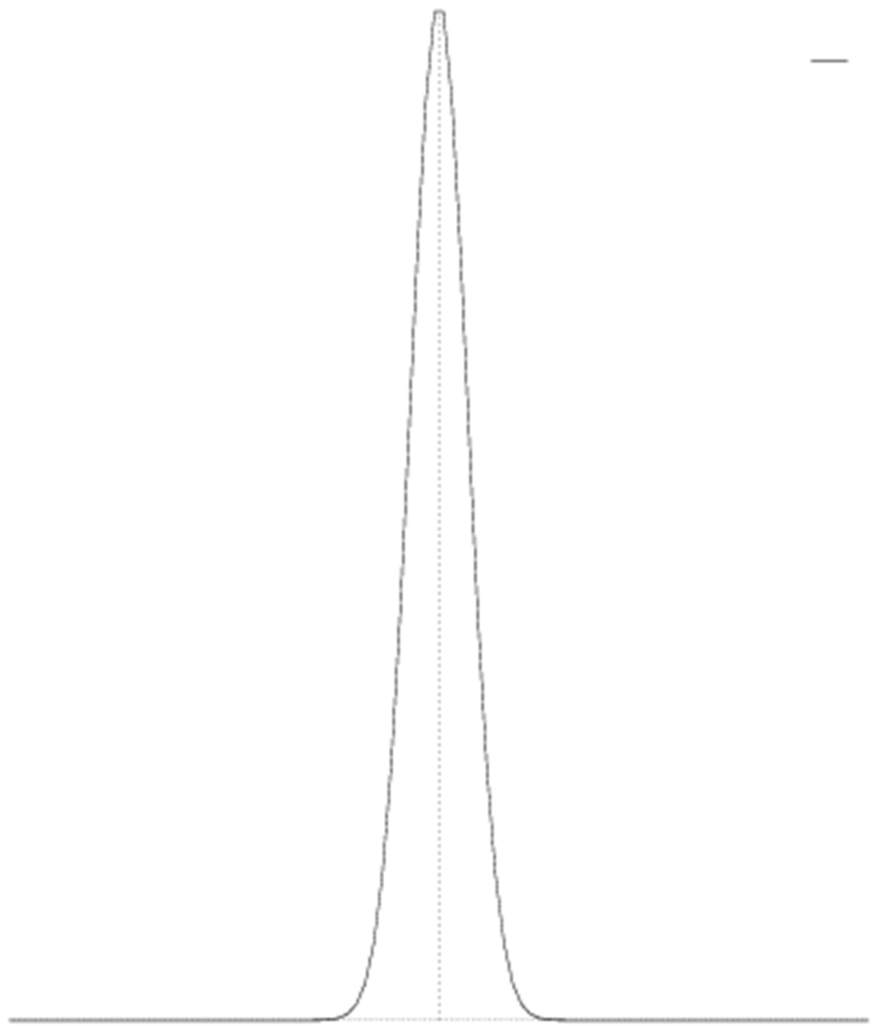
$$\hat{w}(-1) = \hat{w}(1) = \frac{1}{4}$$

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

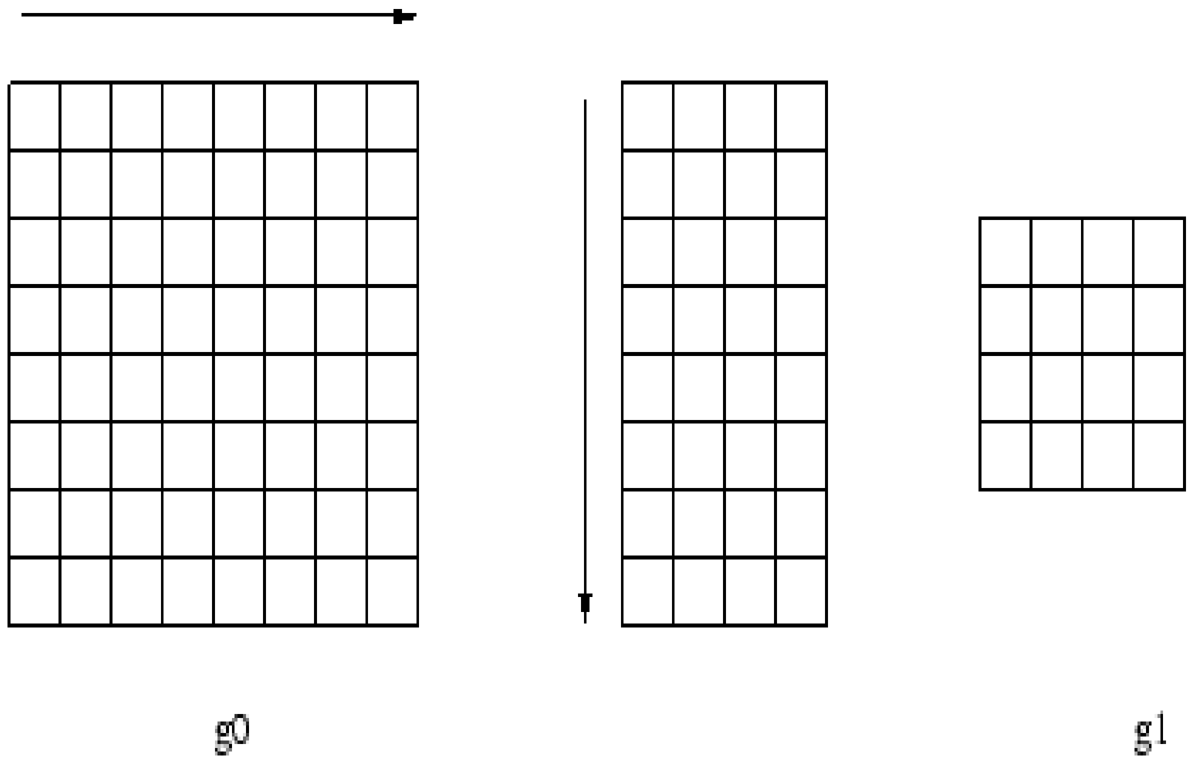
Gaussian

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

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 each pixel along alternate columns 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.

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

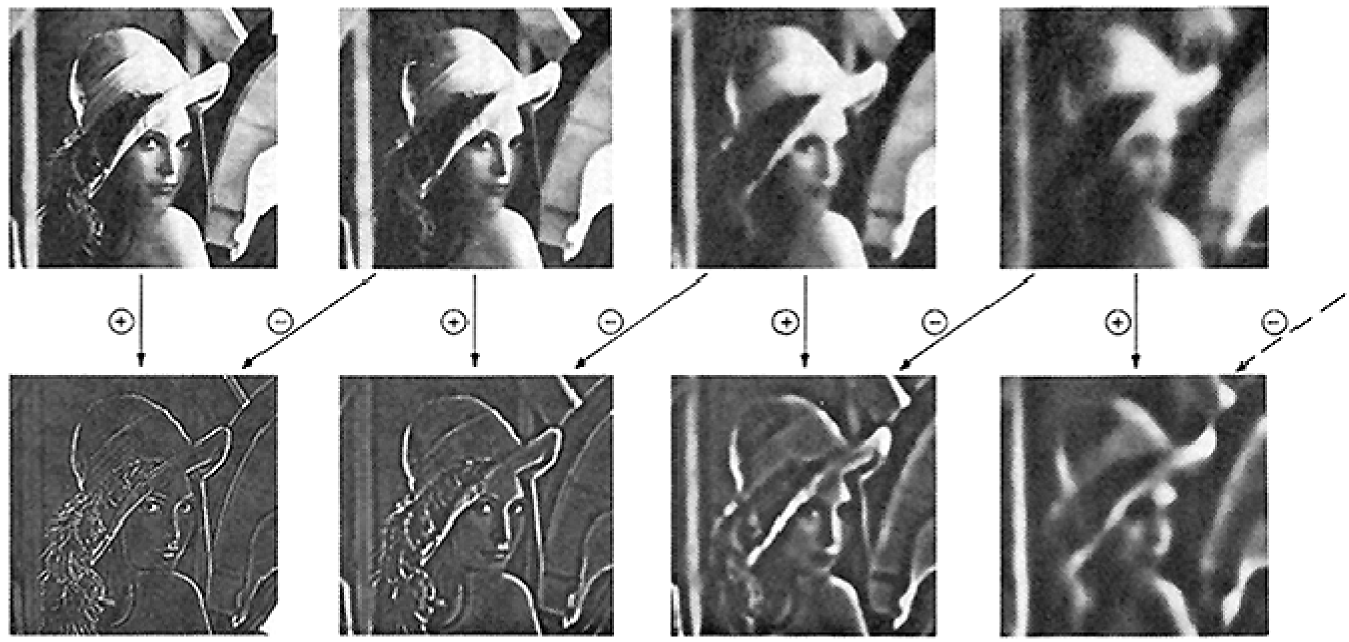


Fig. 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

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.

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

- g_1 is reconstructed image.

Laplacian Pyramid

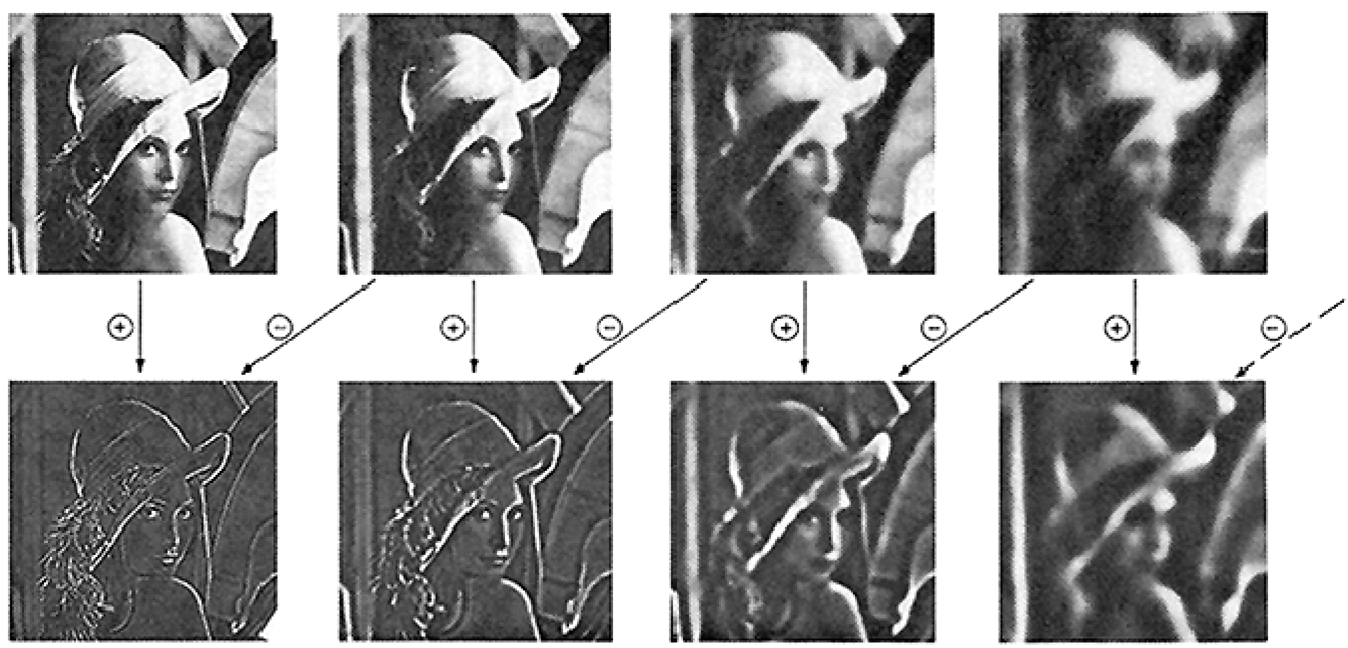


Fig 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

Image Compression (Entropy)

Bits per pixel

7.6

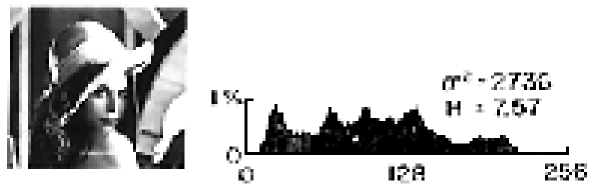


Image Compression

1.58

(a)

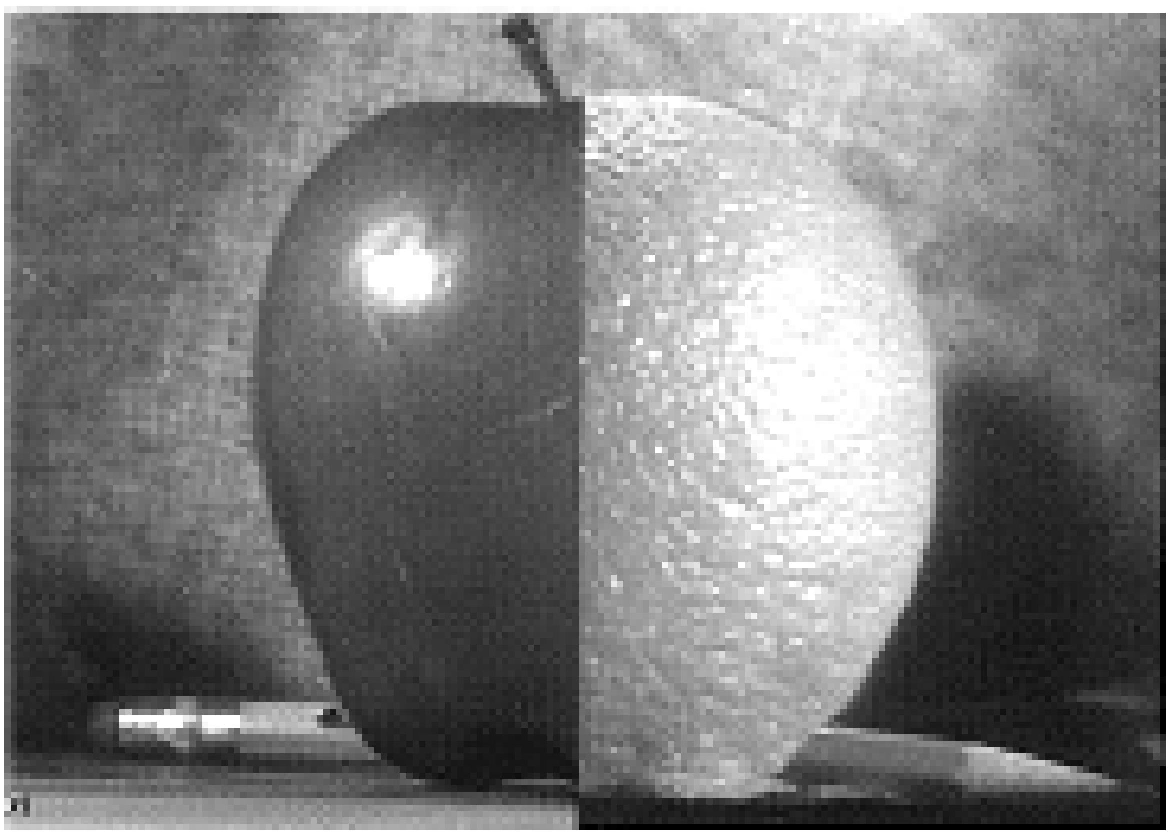
(b)

.73

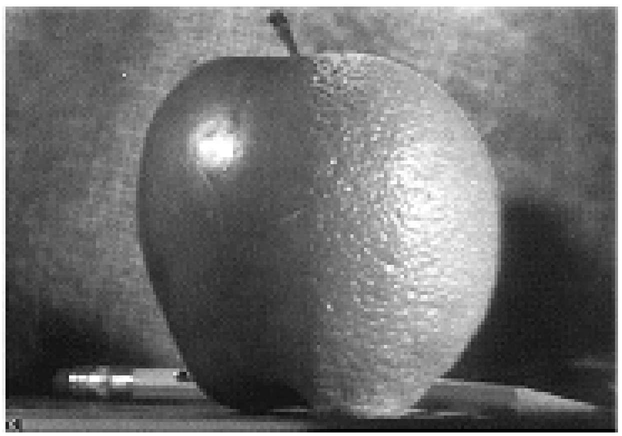
(c)

(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.

Reading Material

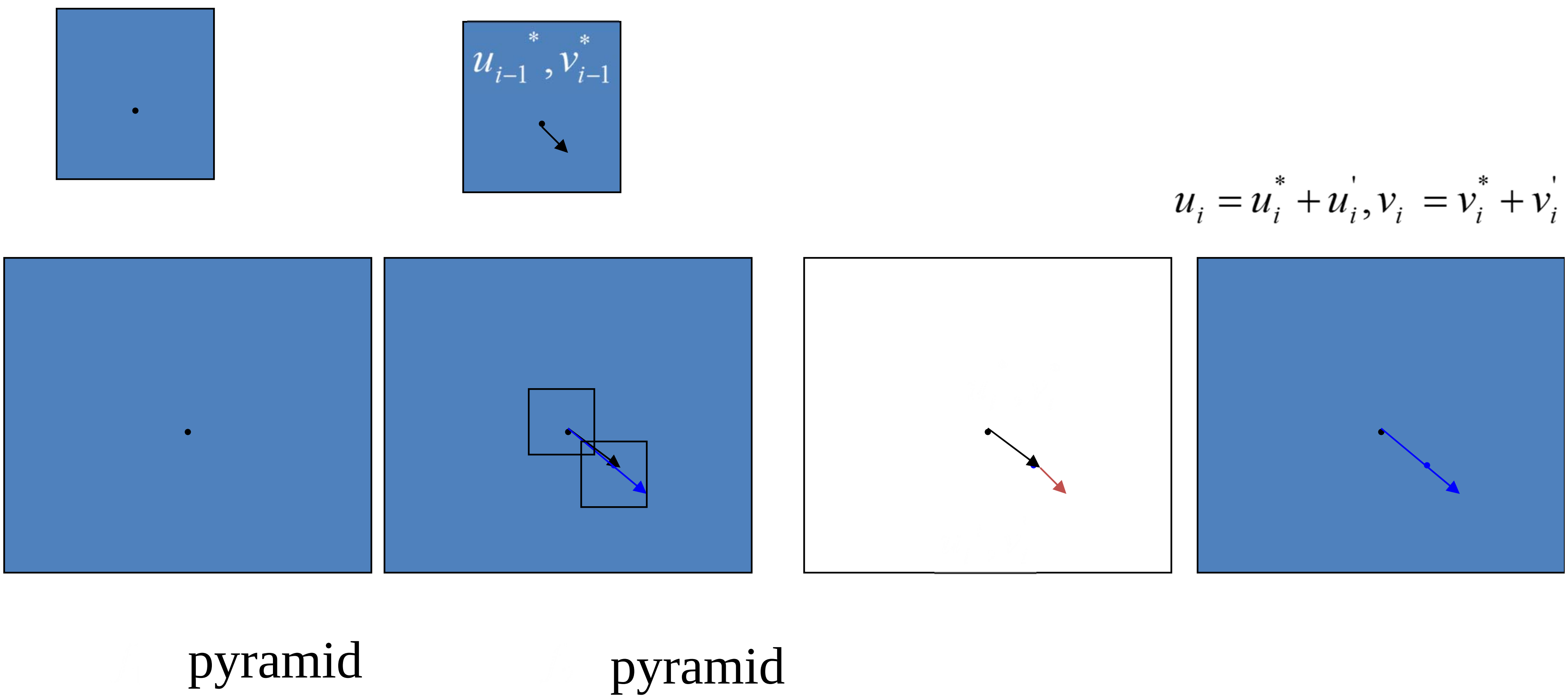
- http://ww-bcs.mit.edu/people/adelson/papers.html
  - The Laplacian Pyramid as a compact code, Burt and Adelson, IEEE Trans on Communication, 1983.
- Fundamental of Computer Vision, Section 4.5. http://www.cs.ucf.edu/courses/cap6411/book.pdf

Lucas Kanade with Pyramids

- Compute 'simple' LK optical flow at highest level
- At level *i*
  - Take flow $u_{i\text{-}1}$, $v_{i\text{-}1}$ from level *i*-1
  - bilinear interpolate it to create u_i^*, v_i^* matrices of twice resolution for level *i*
  - multiply u_i^*, v_i^* by 2
  - compute f_{t} from a block displaced by $u_i^*(x,y)$, $v_i^*(x,y)$
  - Apply LK to get $u_i'(x, y)$, $v_i'(x, y)$ (the correction in flow)
  - Add corrections u_i' v_i', *i.e.* $u_i = u_i^* + u_i'$,

  $v_i = v_i^* + v_i'$.

Pyramids



Interpolation

$$
\begin{array}{r|cccc}
 & 0 & 1 & 2 & 3 \\
0 & \bullet & \bullet & \bullet & \bullet \\
u = 1 & \bullet & \bullet & \bullet & \bullet \\
2 & \bullet & \bullet & \bullet & \bullet \\
3 & \bullet & \bullet & \bullet & \bullet
\end{array}
$$

$$
\begin{array}{r|cccc}
 & 0 & 1 & 2 & 3 \\
0 & \bullet & \bullet & \bullet & \bullet \\
v = 1 & \bullet & \bullet & \bullet & \bullet \\
2 & \bullet & \bullet & \bullet & \bullet \\
3 & \bullet & \bullet & \bullet & \bullet
\end{array}
$$

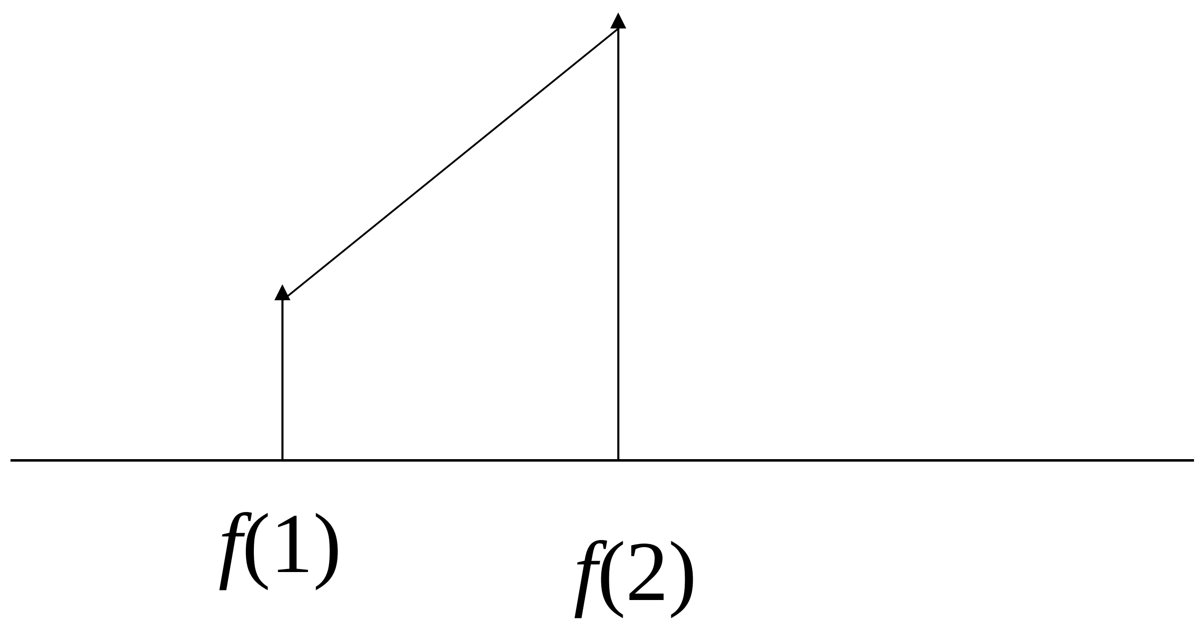
$$
\begin{array}{r|cccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
0 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
1 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
2 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
u^* = 3 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
4 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
5 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
6 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
7 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ
\end{array}
$$

$$
\begin{array}{r|cccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
0 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
1 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
2 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
v^* = 3 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
4 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
5 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ \\
6 & \bullet & \circ & \bullet & \circ & \bullet & \circ & \bullet & \circ \\
7 & \circ & \circ & \circ & \circ & \circ & \circ & \circ & \circ
\end{array}
$$

1-D Interpolation

$$y = mx + c$$

$$f(x) = mx + c$$



2-D Interpolation

$$f(x,y) = a_1 + a_2 x + a_3 y + a_4 xy$$

Bilinear

X X

O X

Bi-linear Interpolation

Four nearest points of (x,y) are:

$$(\underline{x}, \underline{y}), (\overline{x}, \underline{y}), (\underline{x}, \overline{y}), (\overline{x}, \overline{y})$$

$$(3,5), (4,5), (3,6), (4,6)$$

$$\underline{x} = \mathrm{int}(x)$$ 3 (3.2,5.6)

$$\underline{y} = \mathrm{int}(y)$$ 5

$$\overline{x} = \underline{x} + 1$$ 4

$$\overline{y} = \underline{y} + 1$$ 6

X (3,6) X (4,6)

X (3,5) X (4,5)

Bi-linear Interpolation

$$f(x,y) = \overline{\varepsilon_x}\,\overline{\varepsilon_y} f(\underline{x},\underline{y}) + \underline{\varepsilon_x}\,\overline{\varepsilon_y} f(\overline{x},\underline{y}) +$$

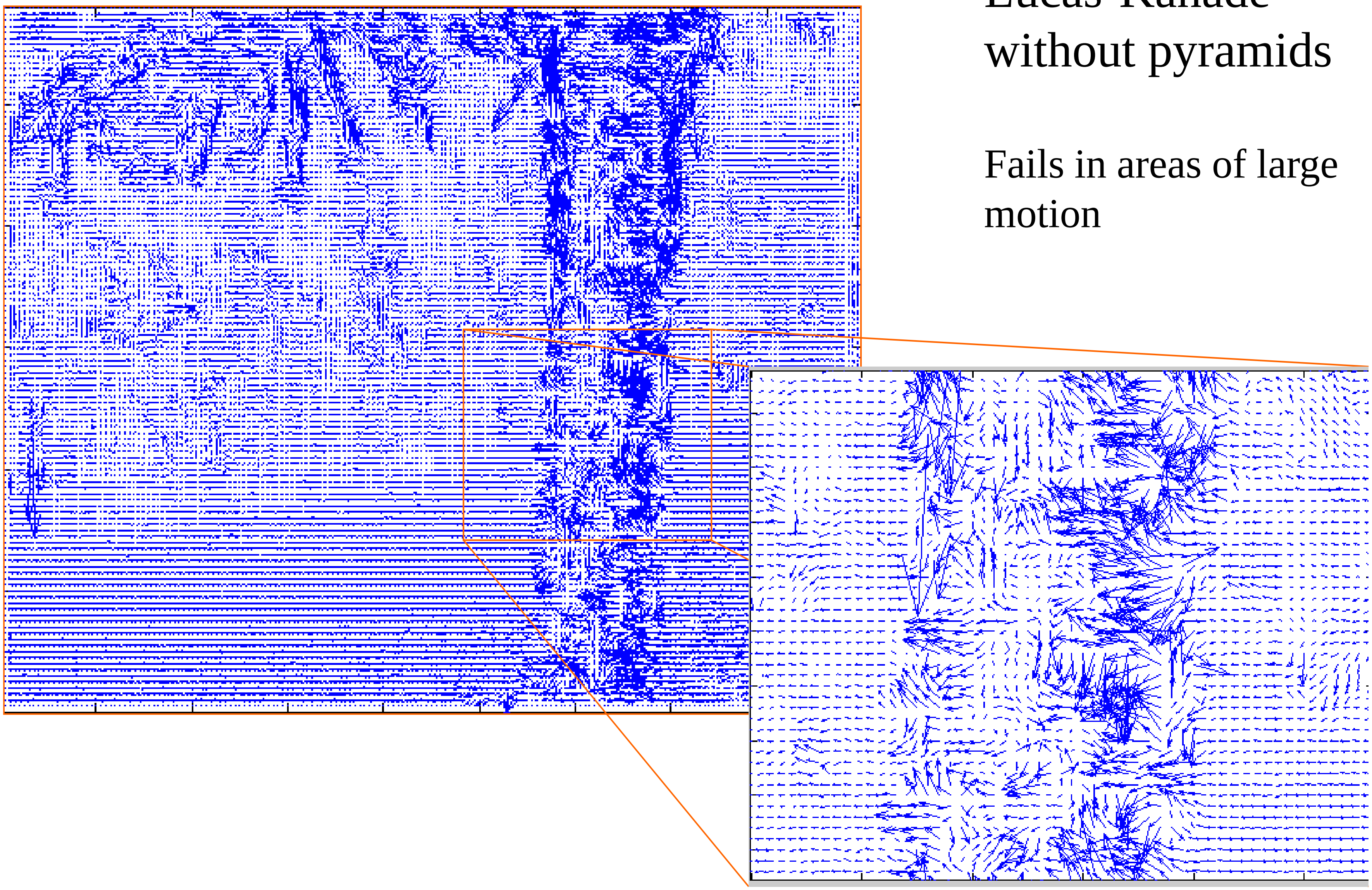
$$\overline{\varepsilon_x}\,\underline{\varepsilon_y} f(\underline{x},\overline{y}) + \underline{\varepsilon_x}\,\underline{\varepsilon_y} f(\overline{x},\overline{y})$$

$$\overline{\varepsilon_x} = \overline{x} - x \qquad \overline{\varepsilon_x} = \overline{x} - x = 4 - 3.2 = .8$$

$$\overline{\varepsilon_y} = \overline{y} - y \qquad \overline{\varepsilon_y} = \overline{y} - y = 6 - 5.6 = .4$$

$$\underline{\varepsilon_x} = x - \underline{x} \qquad \underline{\varepsilon_x} = x - \underline{x} = 3.2 - 2 = .2$$

$$\underline{\varepsilon_y} = y - \underline{y} \qquad \underline{\varepsilon_y} = y - \underline{y} = 5.6 - 5 = .6$$



Lucas-Kanade
without pyramids

Fails in areas of large motion

Lucas-Kanade with Pyramids