Pyramids and their Applications

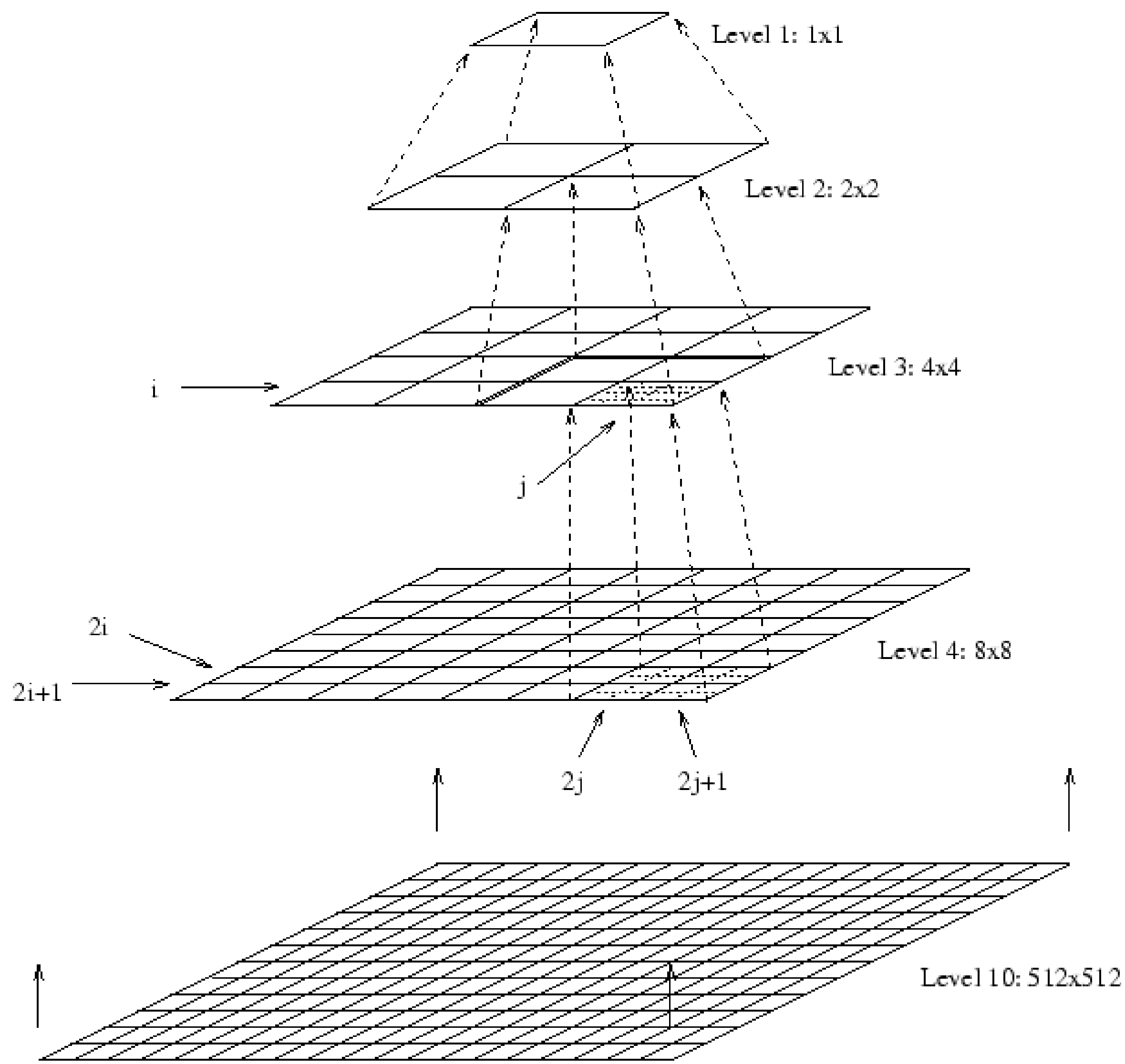
Contents

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

Pyramids

- Very useful for representing images at different scales.
- Pyramid is built by using multiple copies of image at different levels.
- 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

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

Reduce (1D)

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

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

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

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)

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

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

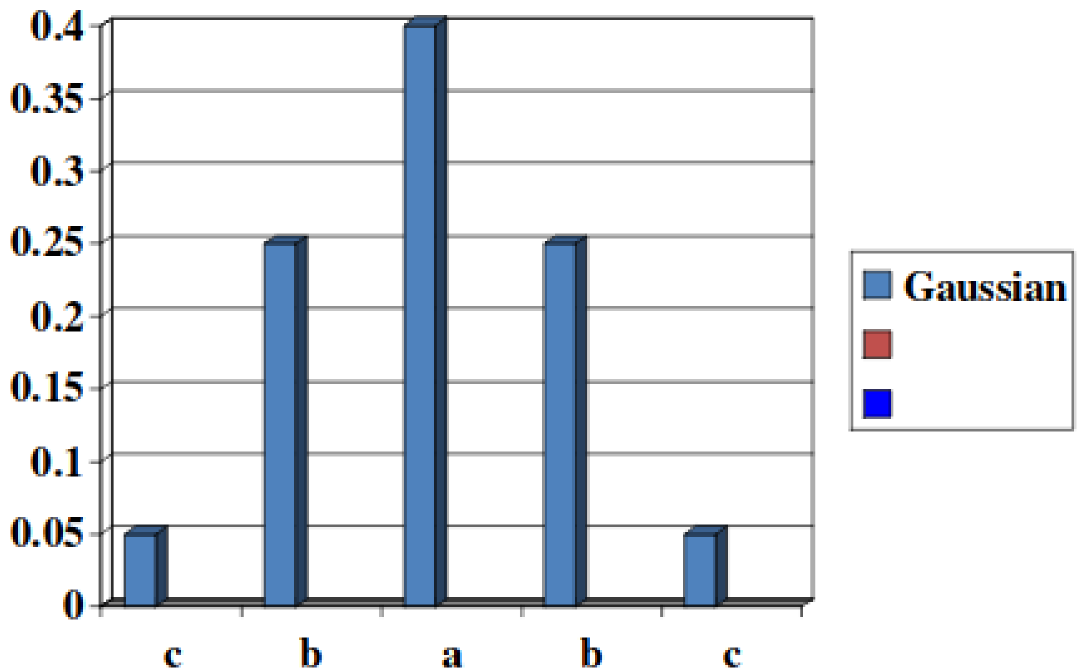
$$\hat{w}(0)\cancel{g_{l,n-1}(\frac{3}{2})} + \hat{w}(1) g_{l,n-1}(\frac{3-1}{1}) + \hat{w}(2)\cancel{g_{l,n-1}(\frac{3-2}{2})}$$

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

Convolution Mask

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

Approximate Gaussian

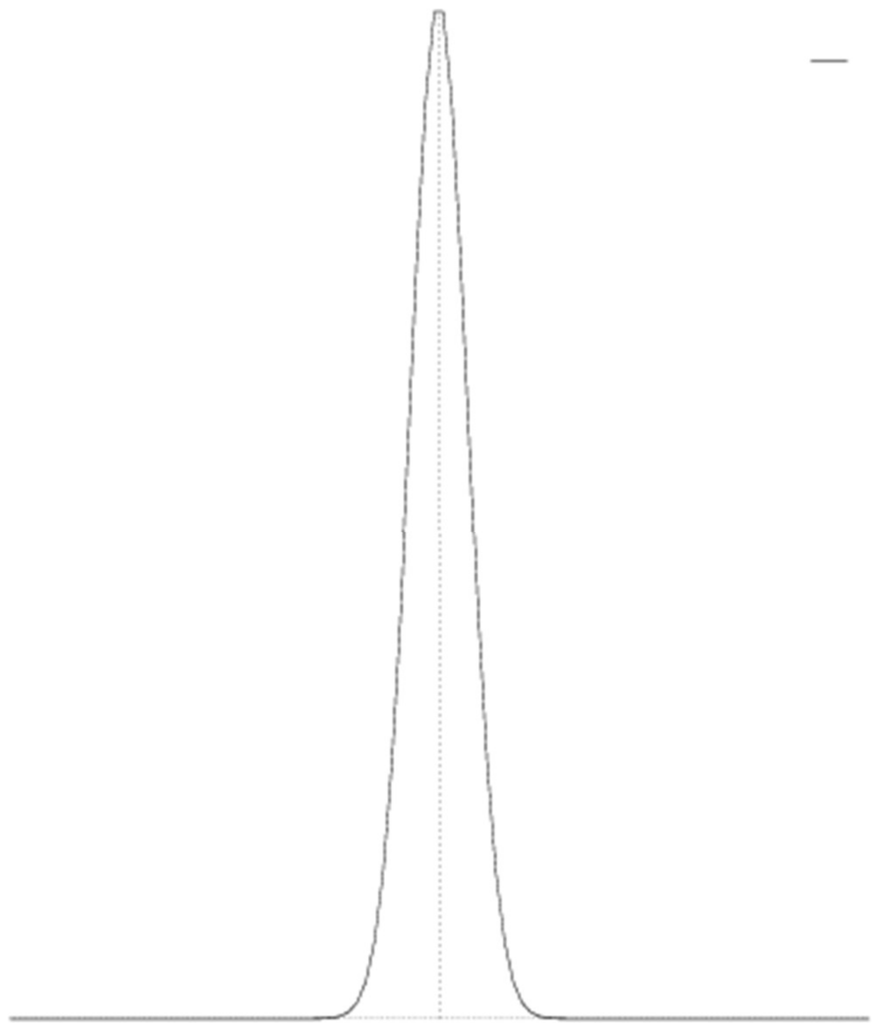


Gaussian

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

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

Normalize g(x)



Gaussian Pyramid

Laplacian Pyramids

WKT, **LOG=DOG**

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

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

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

- L_l is similar to edge image
- Most pixels are zero in L_i .
- Can be used for image compression

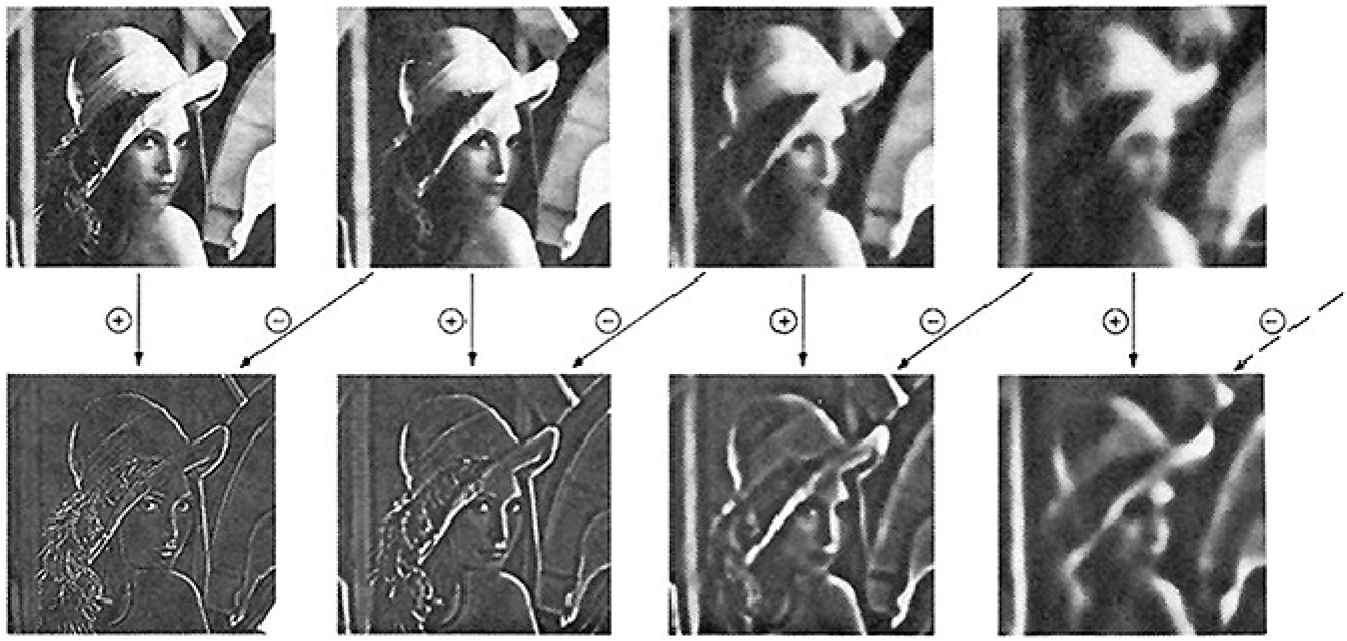


Fig. 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

Coding using Laplacian Pyramid

- Compute Gaussian pyramid

$$g_1, g_2, g_3, g_4$$

- Compute Laplacian pyramid

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

- Image Compression:

  Code of Laplacian pyramid: L_1 , L_2, L_3, L_4

Decoding using Laplacian pyramid

- Decode code of 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

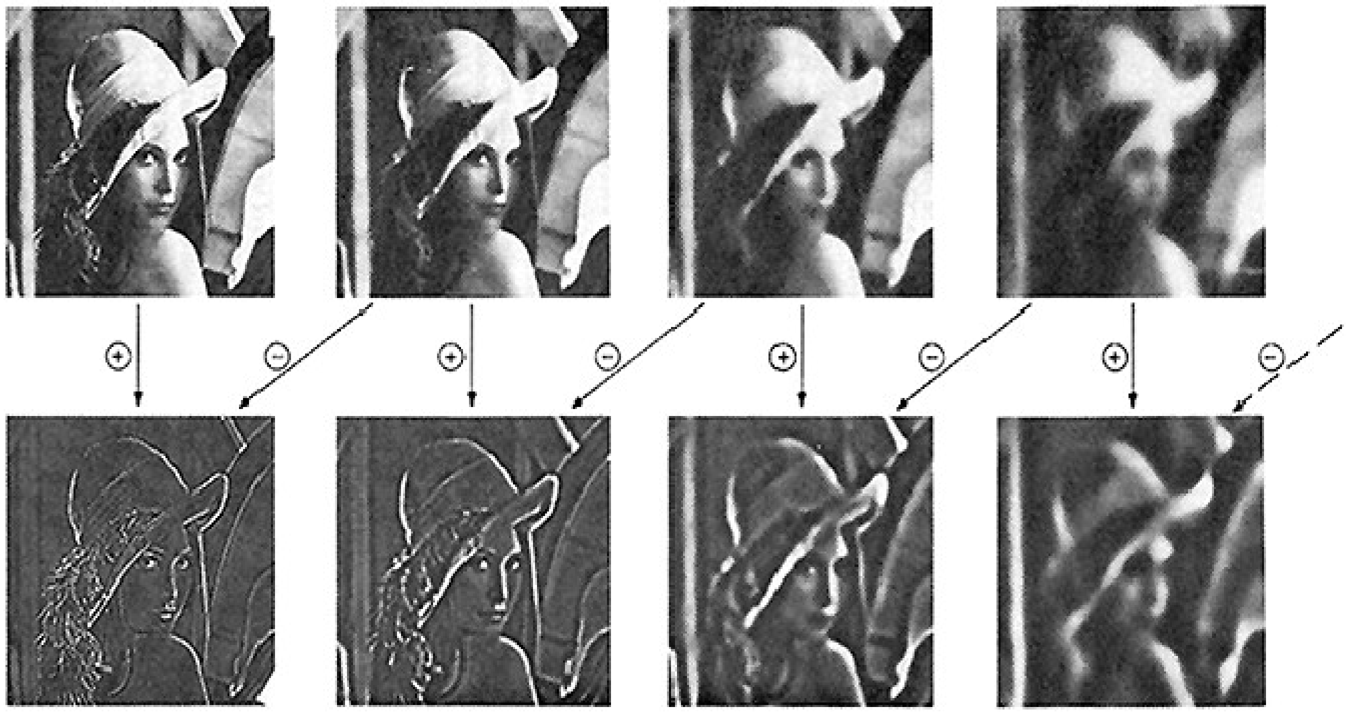


Fig 5. First four levels of the Gaussian and Laplacian pyramid. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

Image Compression (Entropy)

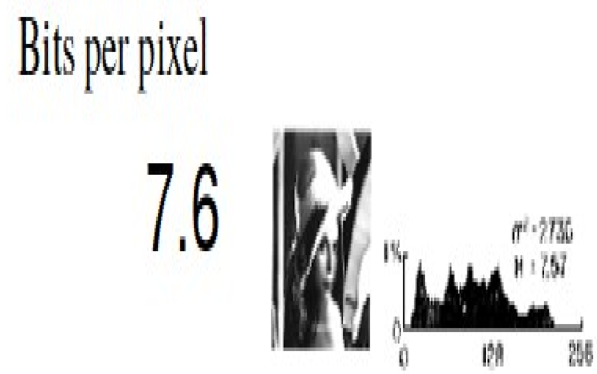
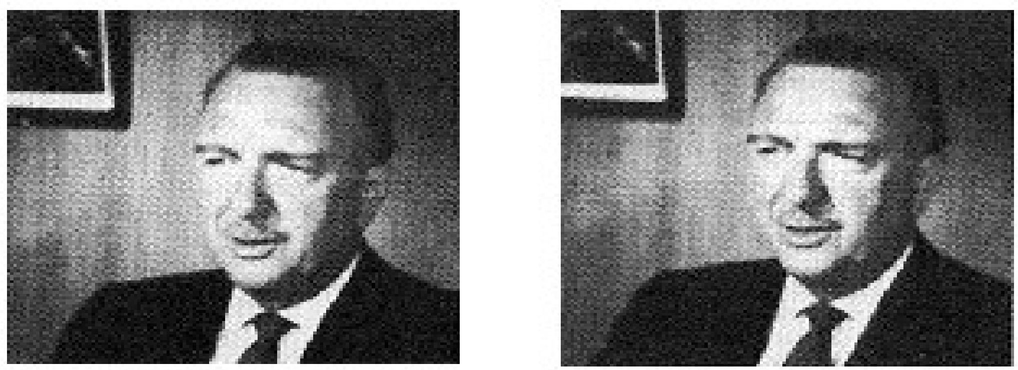


Image Compression

1.58

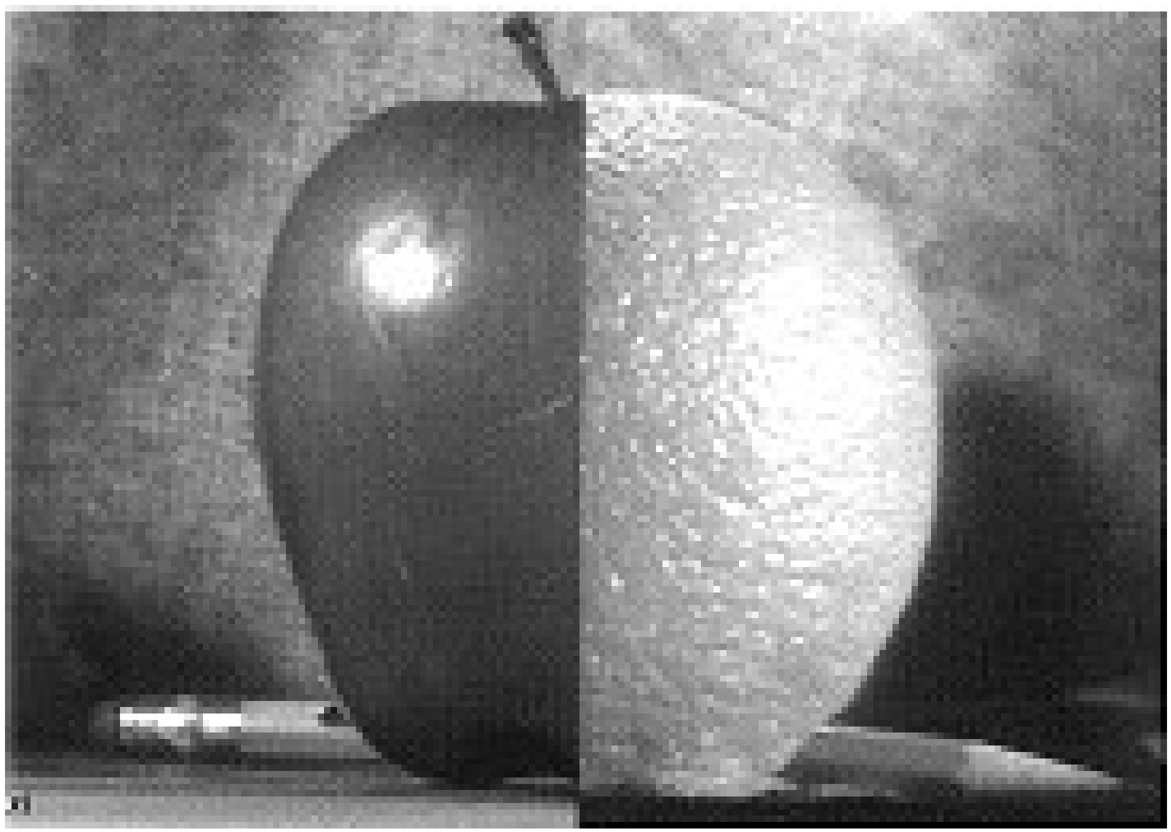
(a) (b)

.73

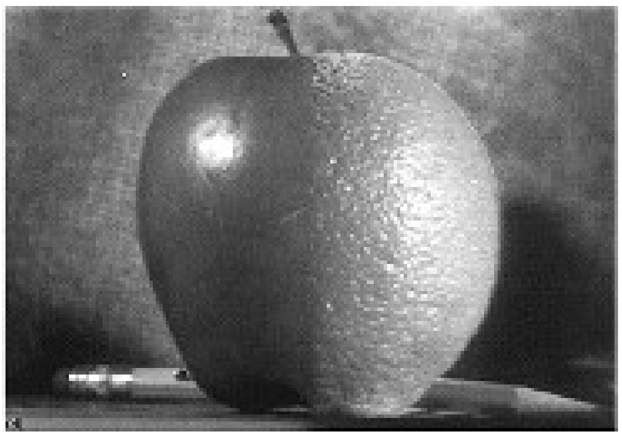


(c) (d)

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.

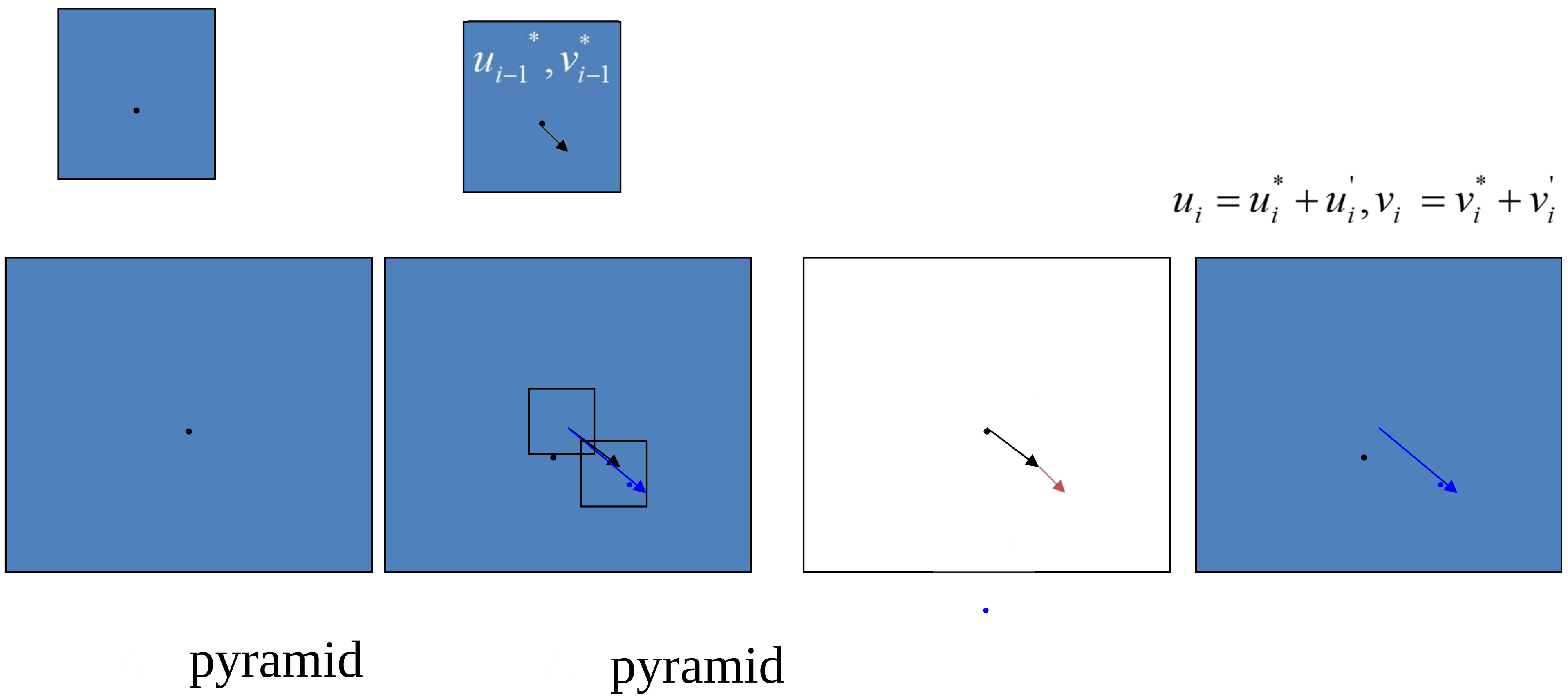
Reading Material

- 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-1}, v_{i-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

	0	1	2	3
0	●	●	●	●
$u = 1$	●	●	●	●
2	●	●	●	●
3	●	●	●	●

	0	1	2	3
0	●	●	●	●
$v = 1$	●	●	●	●
2	●	●	●	●
3	●	●	●	●

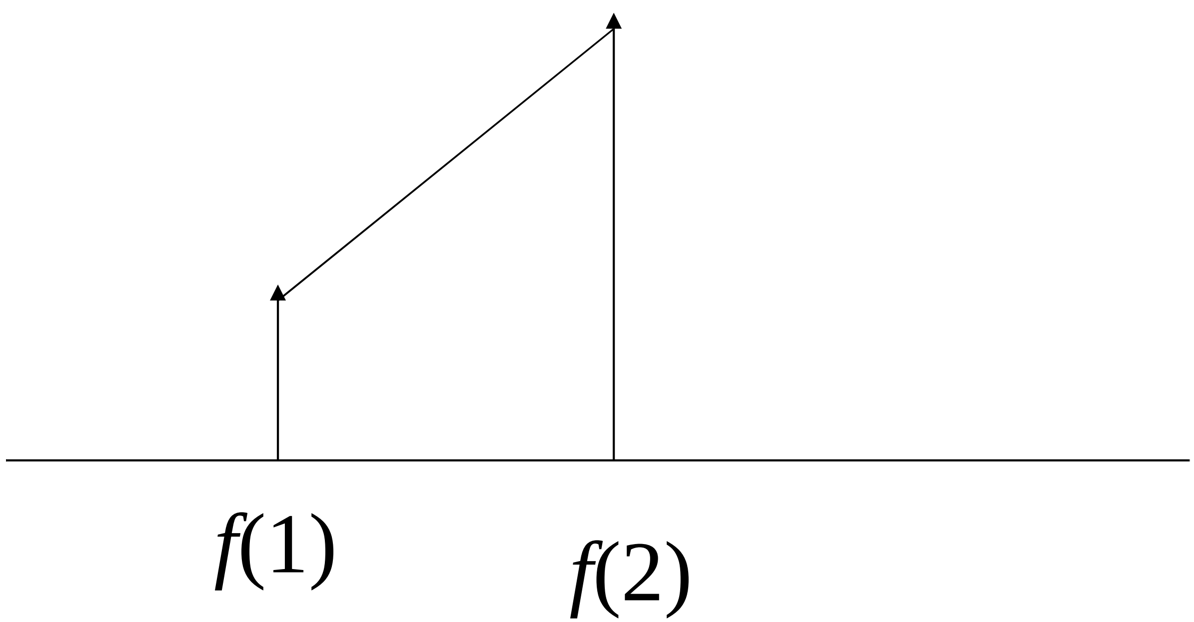
	0	1	2	3	4	5	6	7
0	●	○	●	○	●	○	●	○
1	○	○	○	○	○	○	○	○
2	●	○	●	○	●	○	●	○
$u^* = 3$	○	○	○	○	○	○	○	○
4	●	○	●	○	●	○	●	○
5	○	○	○	○	○	○	○	○
6	●	○	●	○	●	○	●	○
7	○	○	○	○	○	○	○	○

	0	1	2	3	4	5	6	7
0	●	○	●	○	●	○	●	○
1	○	○	○	○	○	○	○	○
2	●	○	●	○	●	○	●	○
$v^* = 3$	○	○	○	○	○	○	○	○
4	●	○	●	○	●	○	●	○
5	○	○	○	○	○	○	○	○
6	●	○	●	○	●	○	●	○
7	○	○	○	○	○	○	○	○

1-D Interpolation

$$y = mx + c$$

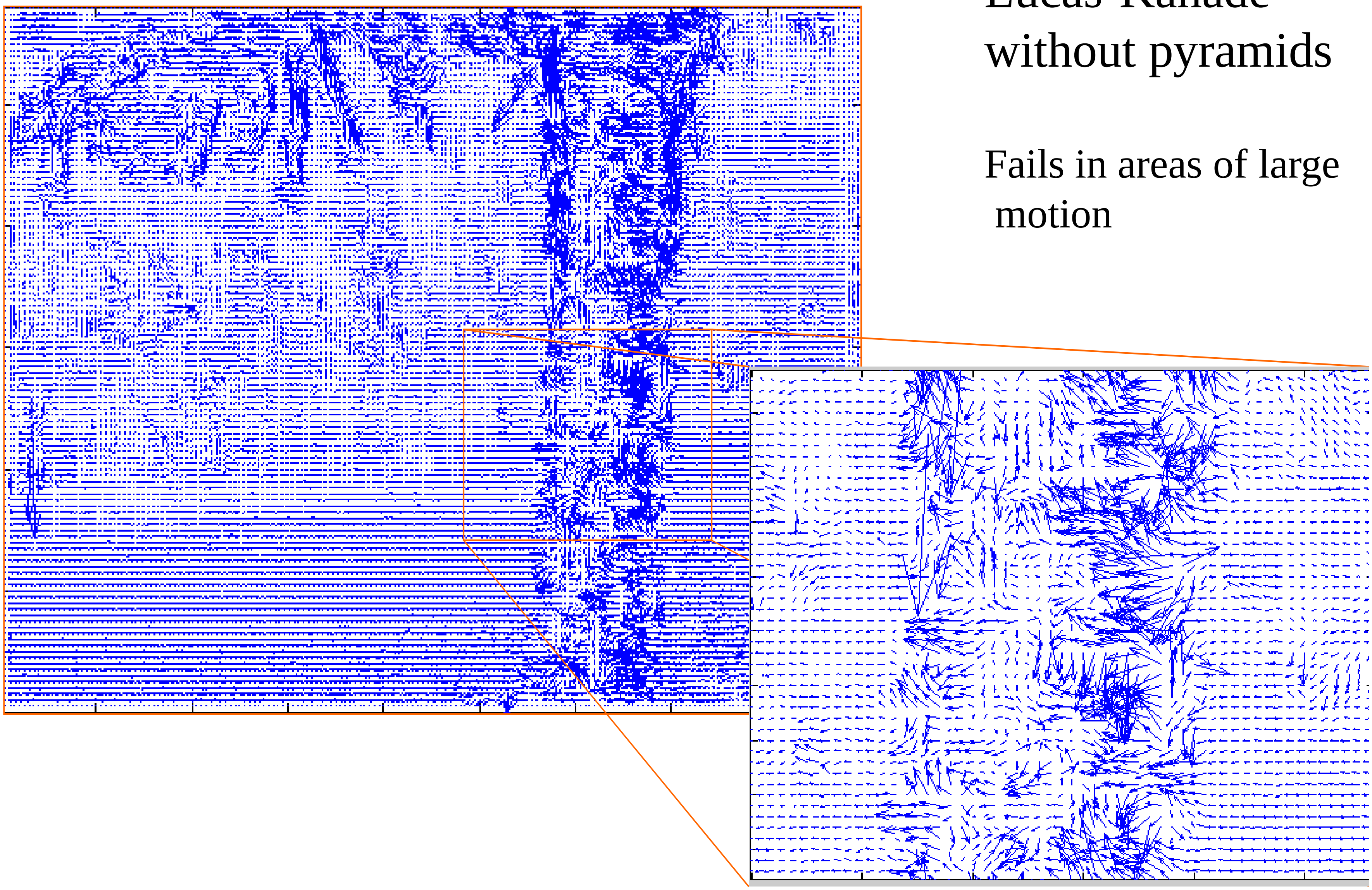
$$f(x) = mx + c$$



Bilinear Interpolation

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

Find the values of unknowns using the pixel values of neighbours of (x,y)



Lucas-Kanade without pyramids

Fails in areas of large motion

Lucas-Kanade with Pyramids