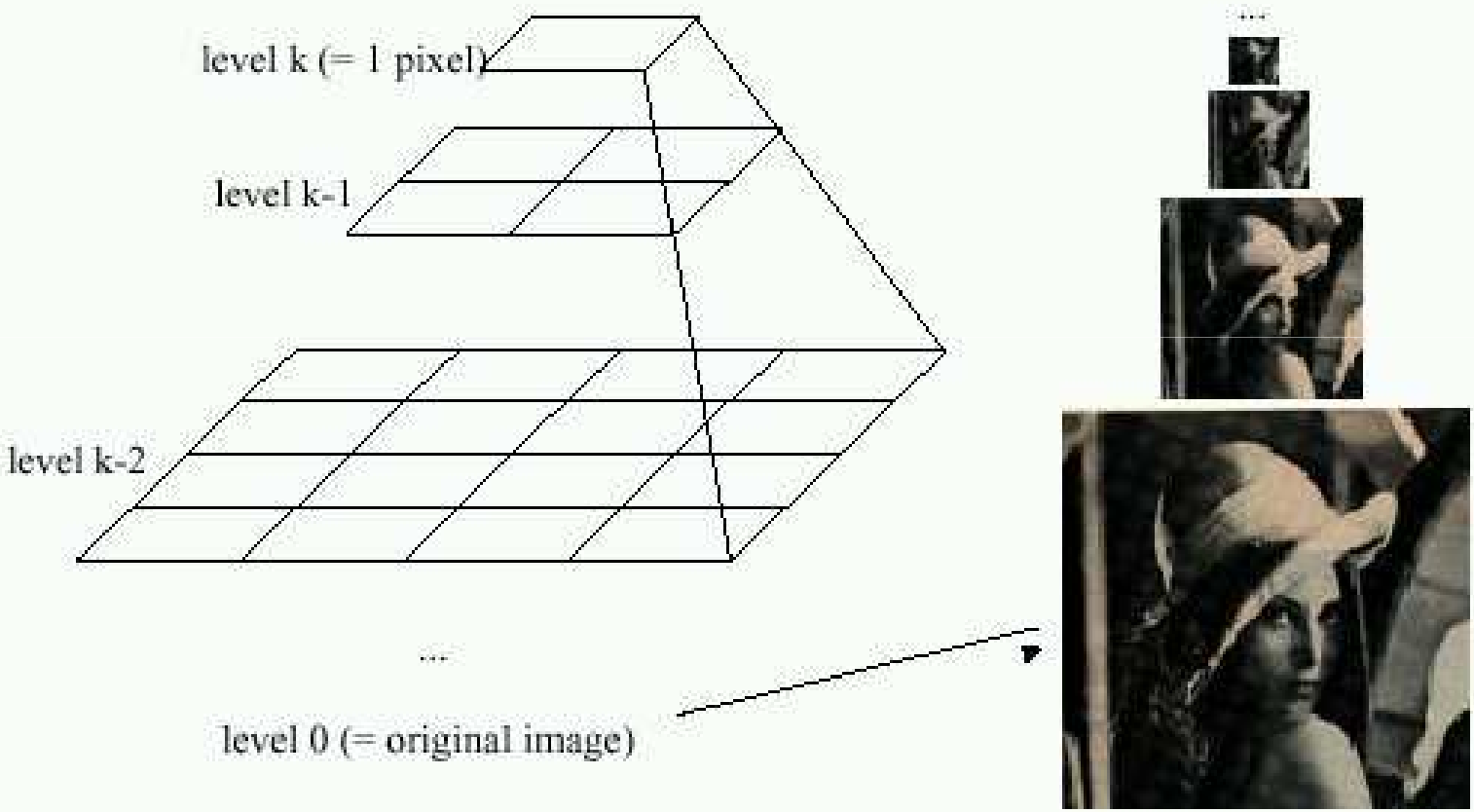
Gaussian Pyramid (Burt and Adelson 1983)

- Image pyramid is a collection of representations of an image
- Typically each layer of the pyramid is half the width and half the height of the previous layer
- If we stack the layers on top of each other, we get a pyramid:
  - lowest level, highest resolution
  - highest level, lowest resolution

Idea: Represent NxN image as a "pyramid" of
1x1, 2x2, 4x4,…, 2^kx2^k images (assuming N=2^k)

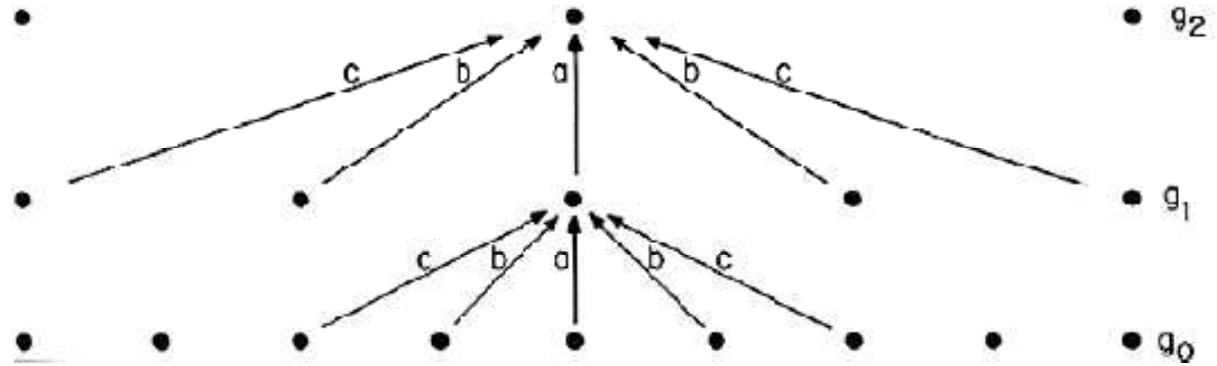


- In a Gaussian pyramid, each layer is smoothed by a symmetric Gaussian kernel, and resampled to get the next layer
- The smallest image is the most heavily smoothed

Reduction

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

GAUSSIAN PYRAMID



g_0 = IMAGE

g_L = REDUCE $[g_{L-1}]$

- Usually the mask is gaussian like function
- Set of weights:

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

$$\sum_{m=-2}^{2} \hat{w}(m) = 1$$ normalization

$$\hat{w}(i) = \hat{w}(-i) \quad \text{i=0,1,2}$$ symmetry

$$a + 2c = 2b$$ equal contribution

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

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

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

- Iterative pyramid generation is equivalent to convolving the image with a set of equivalent weighting functions:

$$g_l = h_l \oplus g_0$$

or

$$g_l(i,j) = \sum_{m=-M_l}^{Ml} \sum_{n=-Ml}^{Ml} h_l(m,n) g_0(i2^l + m, j2^l + n)$$

…where Ml, the size of the equivalent weighting function, doubles from one level to the next

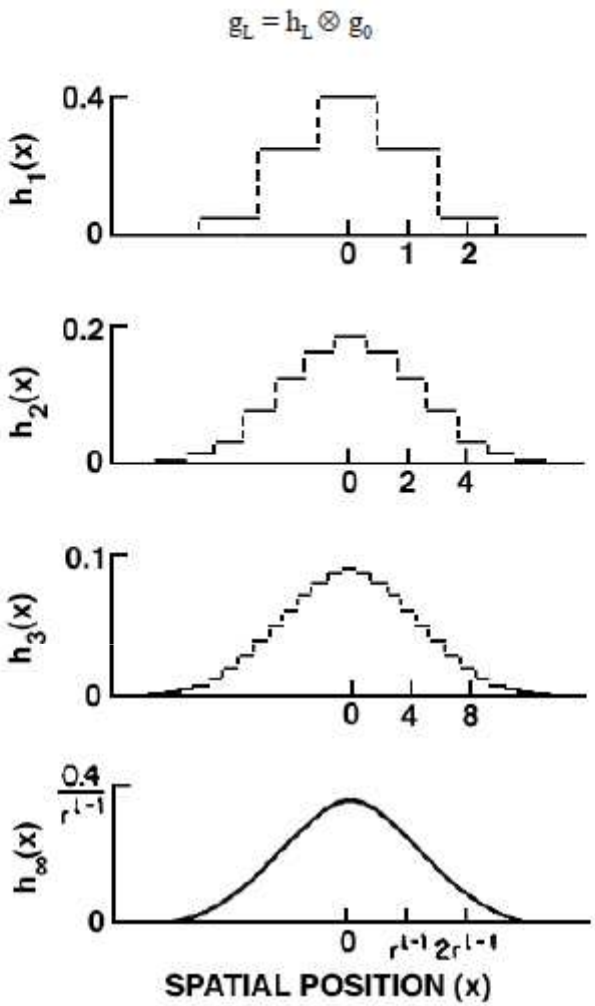


Fig. 2. The equivalent weighting functions $h_j(x)$ for nodes in levels 1, 2, 3, and infinity of the Gaussian pyramid. Note that axis scales have been adjusted by factors of 2 to aid comparison Here the parameter a of the generating kernel is 0.4, and the resulting equivalent weighting functions closely resemble the Gaussian probability density functions.

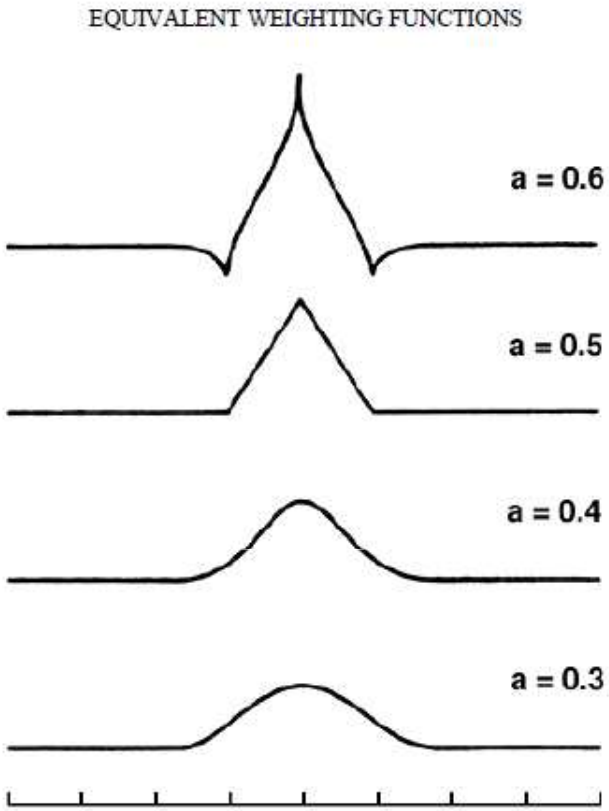


Fig. 3. The shape of the equivalent weighting function depends on the choice of parameter a. For $a = 0.5$, the function is triangular; for $a = 0.4$ it is Gaussian-like, and for $a = 0.3$ it is broader than Gaussian. For $a = 0.6$ the function is trimodal.

512 256 128 64 32 16 8

from: David Forsythhttp://www.cs.berkeley.edu/~daf/

Applications (1)

- Search over scale – objects can be represented as small image patterns; if we want to search across different *scales* we can search across the layers of the gaussian pyramid; bigger objects will be found in the coarser scale layers, smaller objects will be found in the finer scales

Applications (2)

- Spatial search: often we have a point in one image and want to find the same point in another image (example: stereo vision). This can be achieved more efficiently if we first start to search the object in the coarser layers, and then refine the match by searching in the finer layers (*coarse-to-fine matching)*

Applications (3)

- Feature tracking: features (e.g. edges) found at coarse levels are associated with high-contrast image events (low contrast patches are easily lost during consequent smoothing); at fine scales there are probably many more features with lower contrast. A common strategy for improving a set of features obtained at a fine scale is to track features to coarser scales and accept only the fine scale features that are identifiable at coarser scales (*feature tracking)*

Laplacian pyramid

- Produce layers of the pyramid
- Idea compute:

$$L_0(i,j) = g_0(i,j) - g_l(i,j)$$

- Instead of g0 encode gl and L0; since L0 is a “prediction” error and can be encoded with fewer bits than g0
- Repeat…

$$L_2(i,j) = g_1(i,j) - g_2(i,j)$$

$$\Rightarrow L_0, L_1, .., L_n$$

Encoding

- expand a MxN image into a 2Mx2N image

$$g_l = EXPAND(g_{l+1})$$

$$g_l(i,j) = 4\sum_{m=-2}^{2}\sum_{n=-2}^{2} w(m,n)\cdot g_{l+1}\left(\frac{i-m}{2},\frac{j-n}{2}\right)$$

build sequence $L_0, L_1, \ldots, L_N$

$$L_0 = g_0 - EXPAND(g_1)$$

$$L_1 = g_1 - EXPAND(g_2)$$

...

$$L_{N-1} = g_{N-1} - EXPAND(g_N)$$

$$L_N = g_N$$

Decoding

from $L_N, L_{N-1}, .., L_0$

$g_N = L_N$

$g_{N-1} = L_{N-1} + EXPAND(g_N)$

...

$g_0 = L_0 + EXPAND(g_1)$

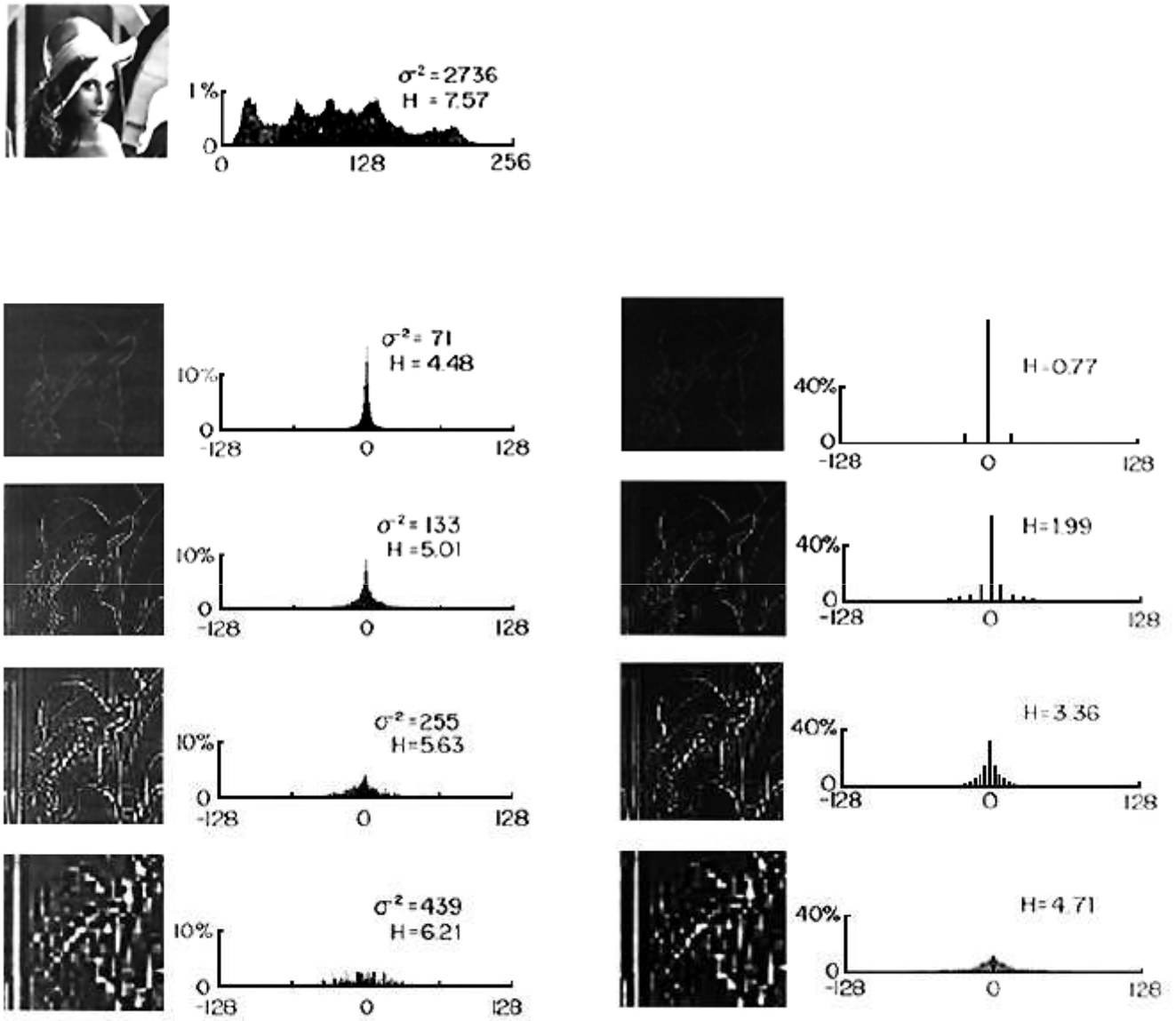


FIG 6. The distribution of pixel gray level values at various stages of the encoding process. The histogram of the original image is given in (a). (b)-(e) give histograms for levels 0-3 of the Laplacian pyramid with generating parameter a=0.6. Histograms following quantization at each level are shown in (f)-(i). Note that pixel values in the Laplacian pyramid are concentrateed near zero, permitting data compression through shortened and varable length code words. Substantial further reduction is realized through quantization (particularly at low pyramid levels) and reduced sample density (particularly at high pyramid levels).