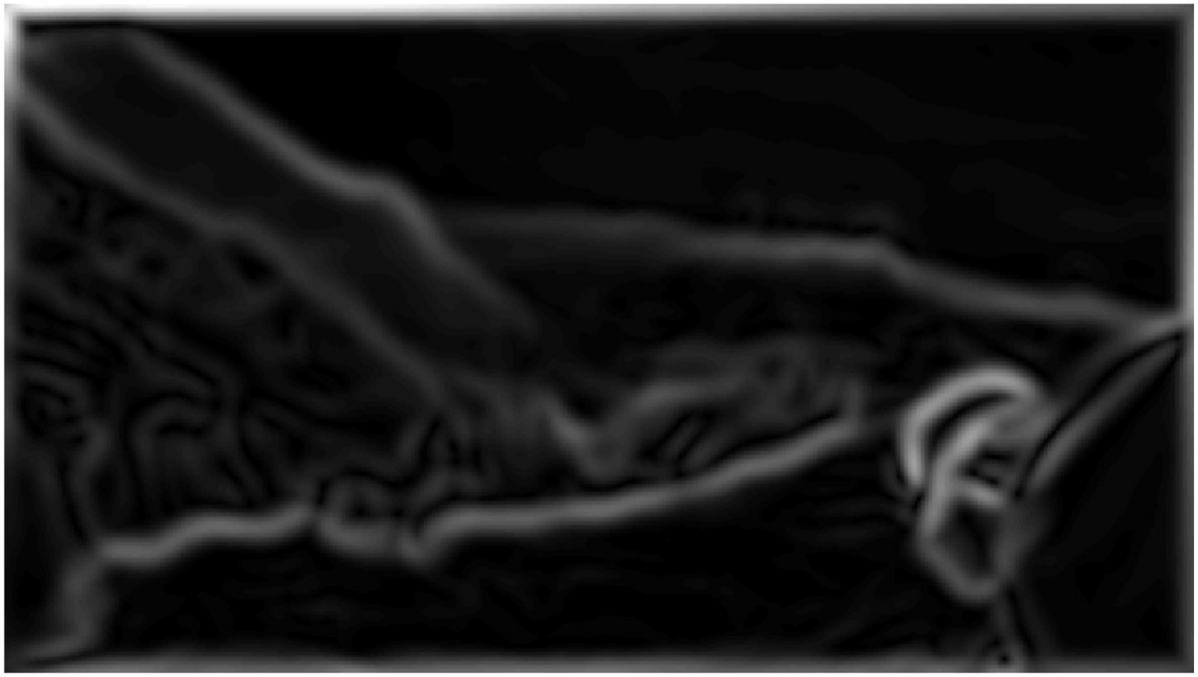
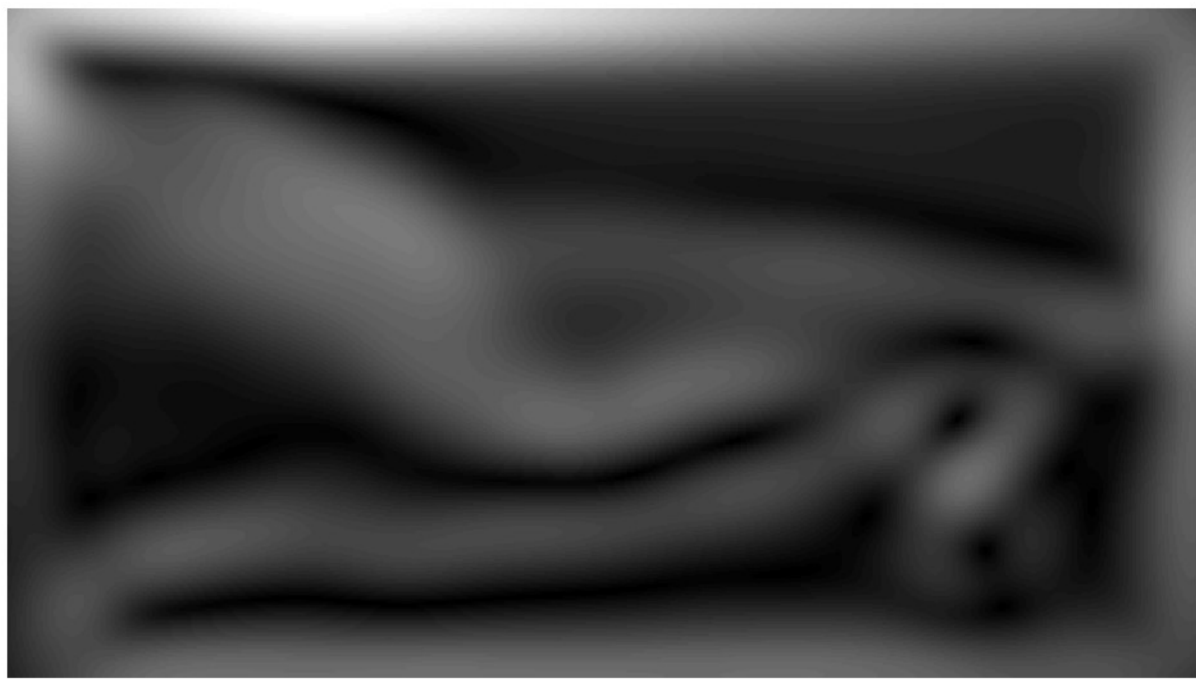
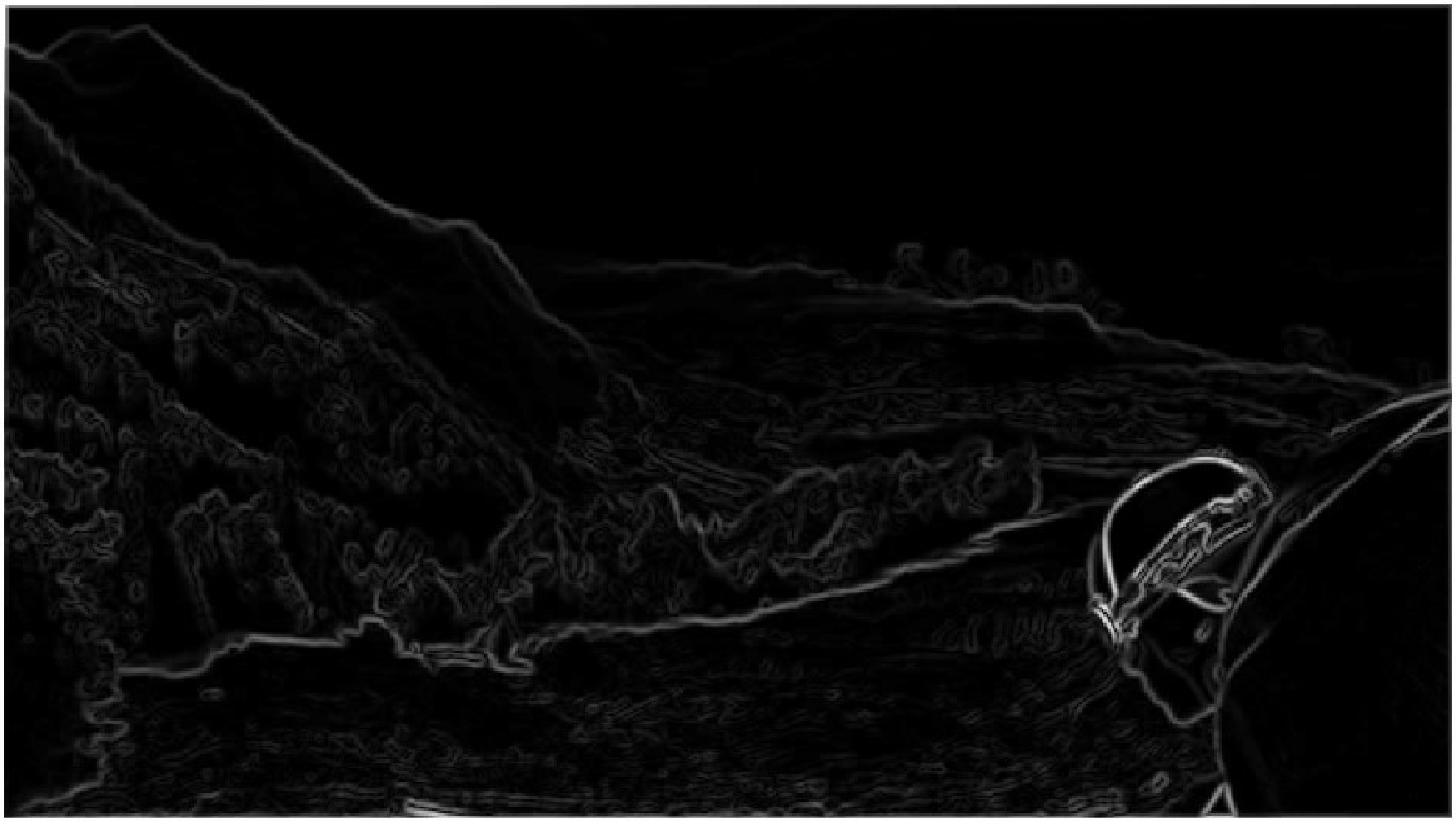
Image Pyramid, CIS581

Image Scale

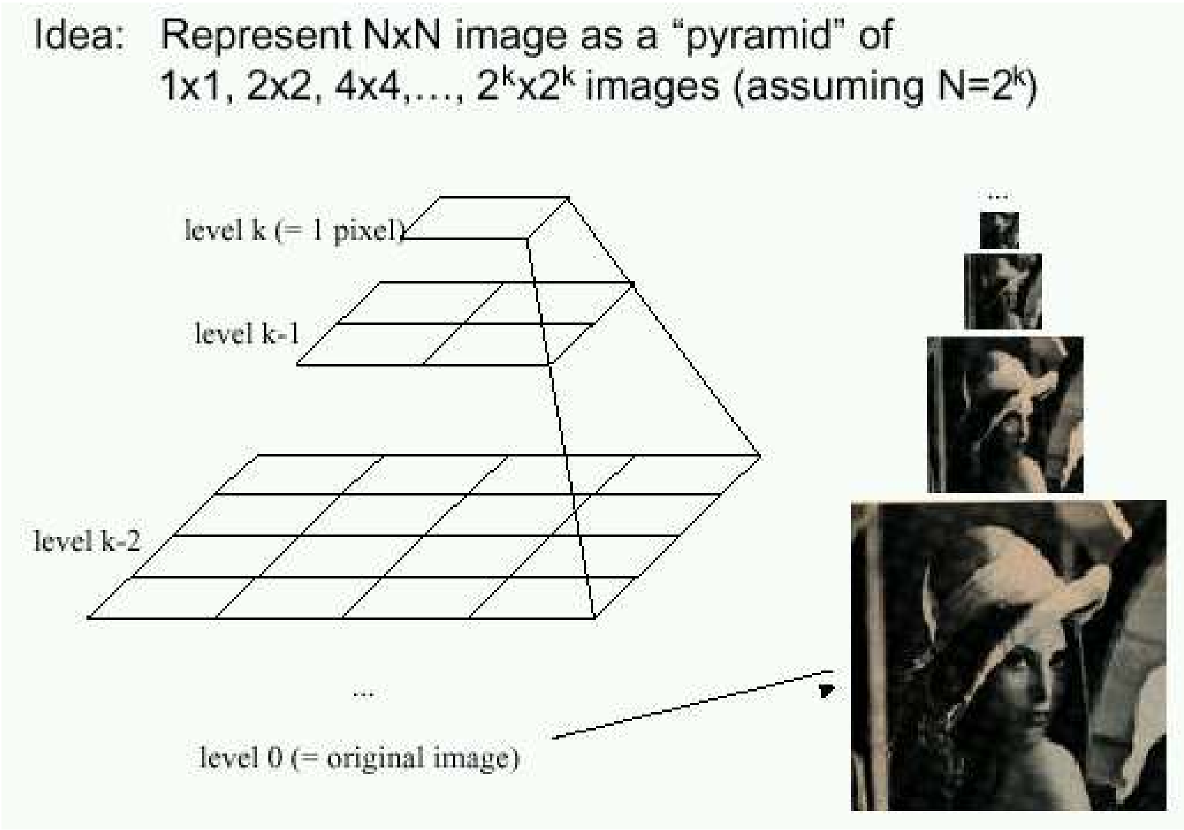






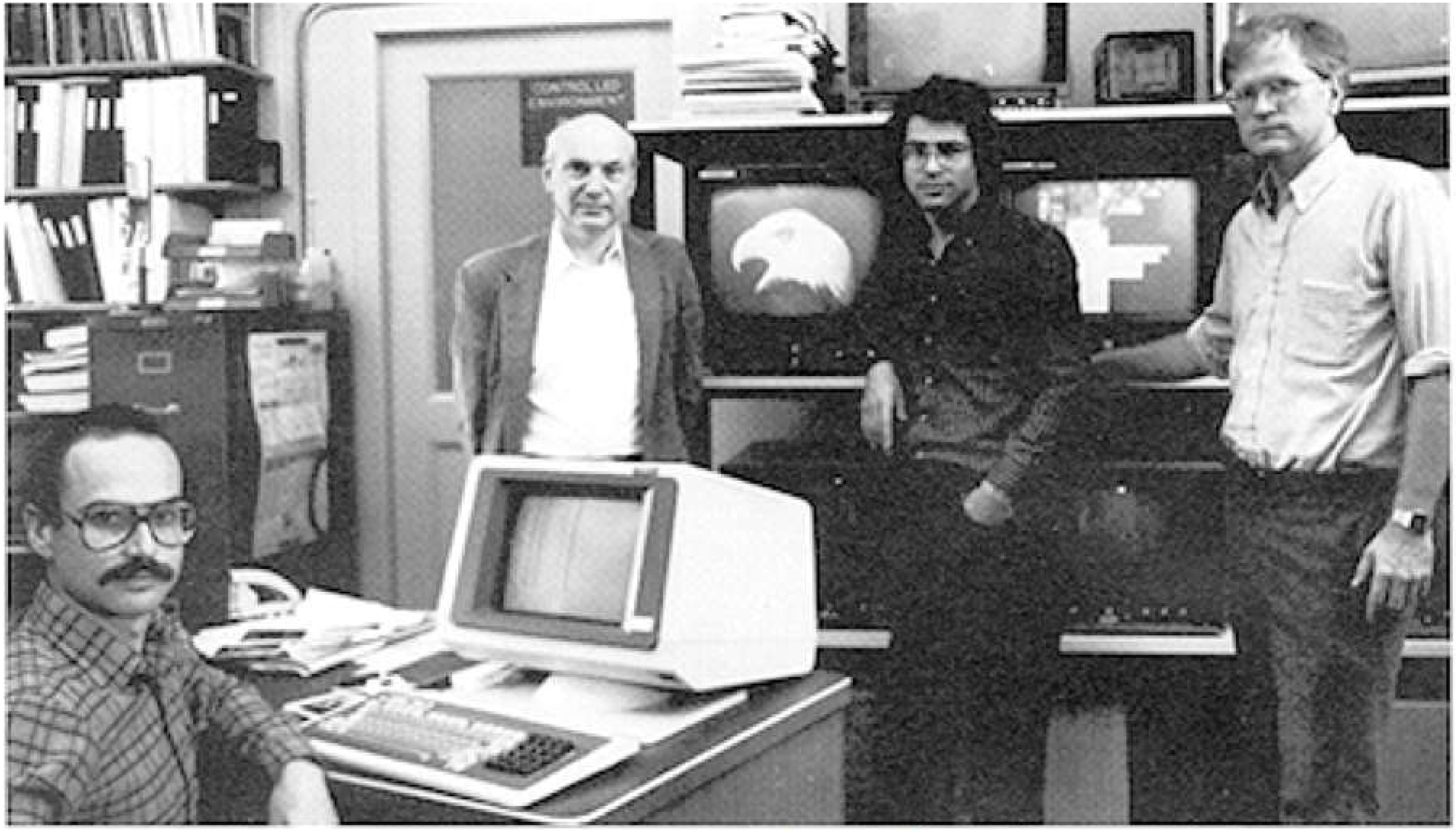
Different scale of image encodes different edge response.

Image Pyramids



Known as a Gaussian Pyramid [Burt and Adelson, 1983]

- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*



Authors, left to right: Bergen, Anderson, Adelson, Burt.

A detour through scale space

Image encoding-decoding

1) Image statistics: pixel in neighborhood are correlated, encode per pixel value is redundant

2) Predictive Coding:use raster scan, predict based on pass value, and store only the error in prediction. Simple and fast

signal	10	10	20	22	24	24
prediction	10	10	10	15	21	23
error-encoded	0	0	10	7	3	1

Image encoding-decoding (part1)

	S[1]	S[2]	S[3]	S[4]	S[5]	S[6]
signal	**10**	**10**	**20**	**22**	**24**	**24**

P[1]=S[1]

P[2]=S[1]

	P[1]	P[2]	P[3]	P[4]	P[5]	P[6]
prediction	**10**	**10**				

	E[1]	E[2]	E[3]	E[4]	E[5]	E[6]
error-encoded						

Image encoding-decoding (part2)

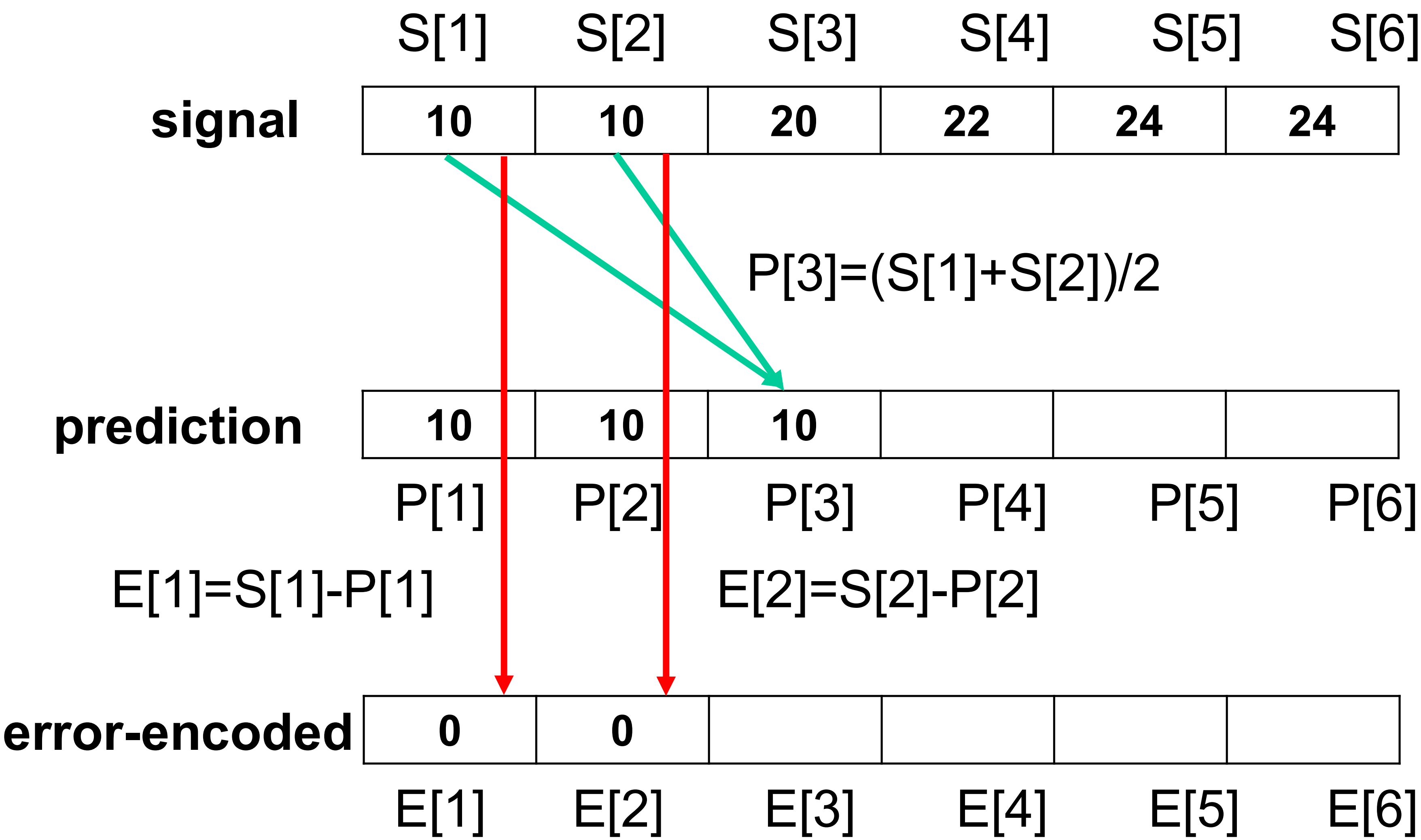


Image encoding-decoding (part2)

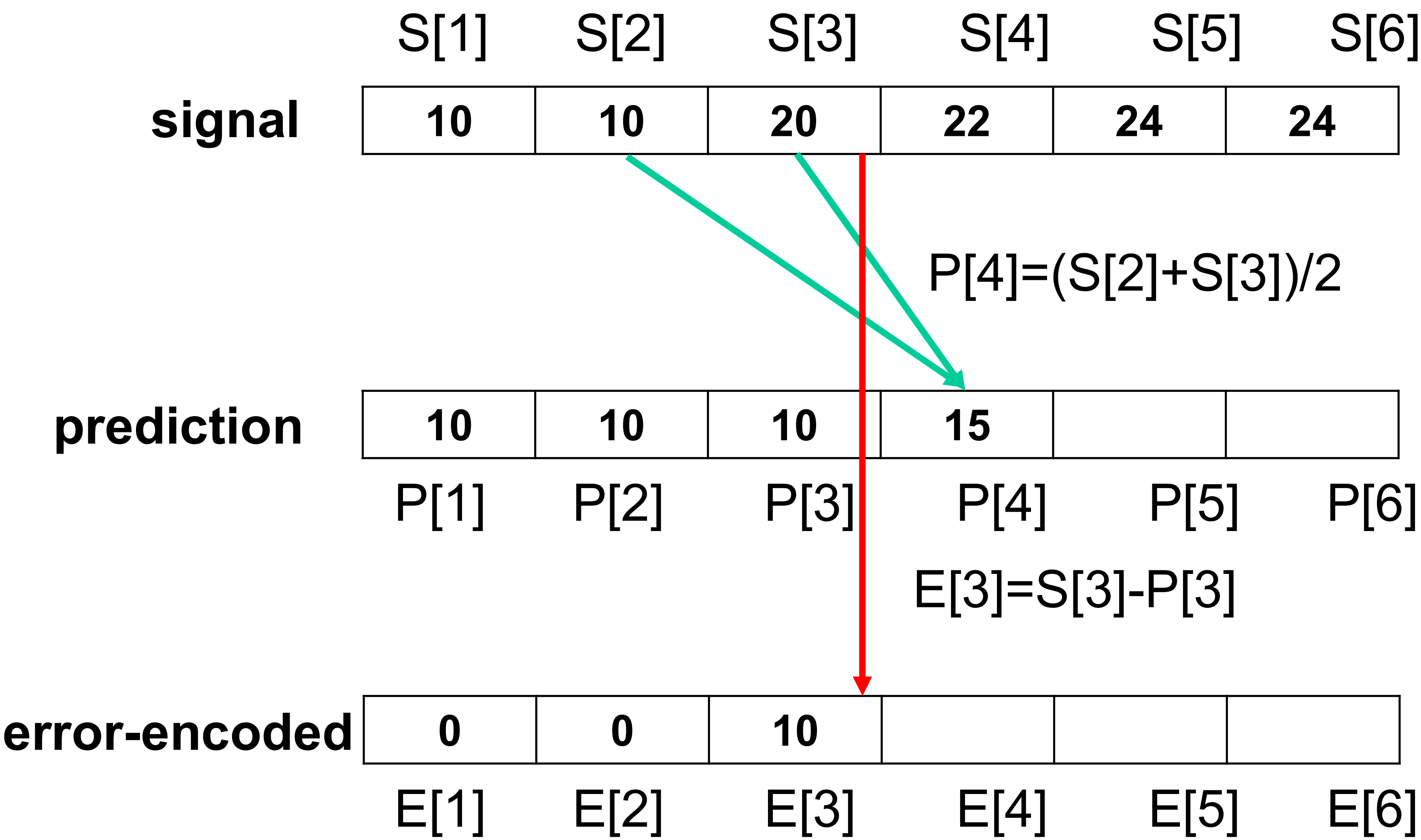


Image encoding-decoding (part3)

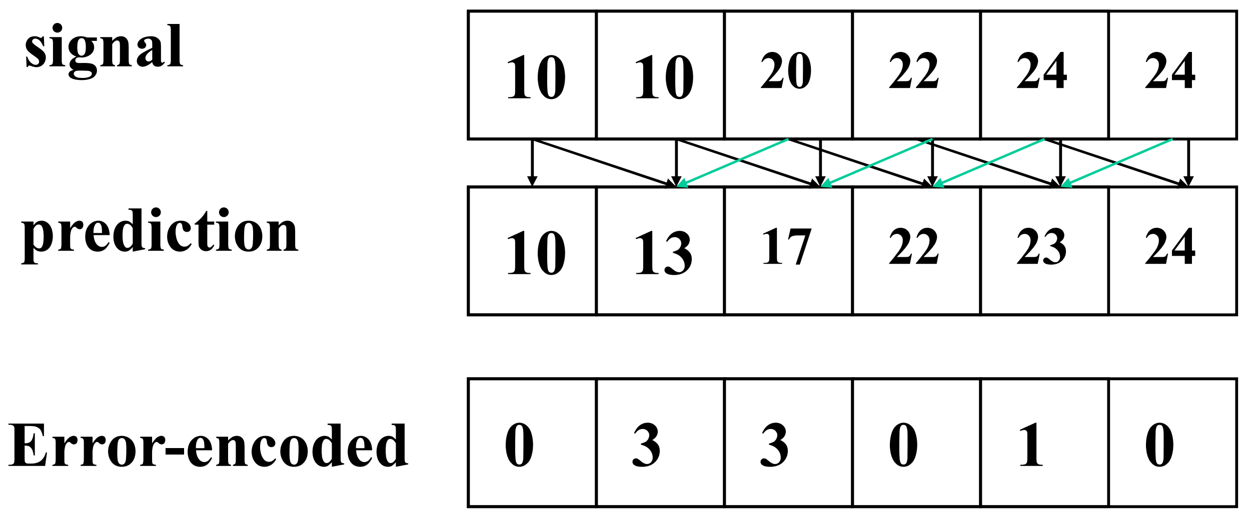
	S[1]	S[2]	S[3]	S[4]	S[5]	S[6]
signal	**10**	**10**	**20**	**22**	**24**	**24**
prediction	**10**	**10**	**10**	**15**	**21**	**23**
	P[1]	P[2]	P[3]	P[4]	P[5]	P[6]
error-encoded	**0**	**0**	**10**	**7**	**3**	**1**
	E[1]	E[2]	E[3]	E[4]	E[5]	E[6]

P[i]=(S[i-1]+S[i-2])/2

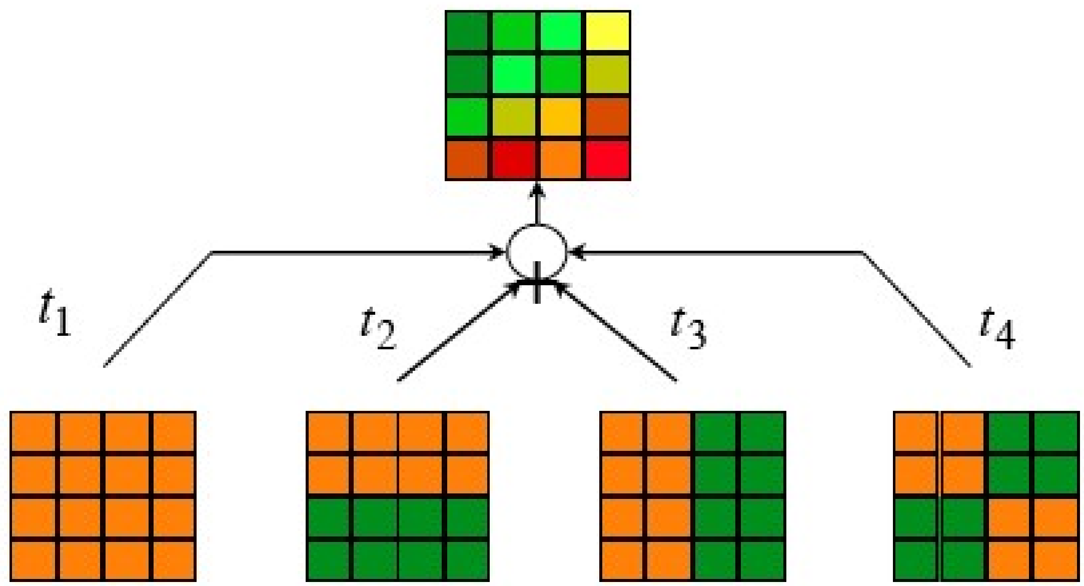
E[i]=S[i]-P[i]

Non-causal prediction

non-causal involves typically transform, or solution to a large sets of equations. Encode block by block. Bigger compression but slower.



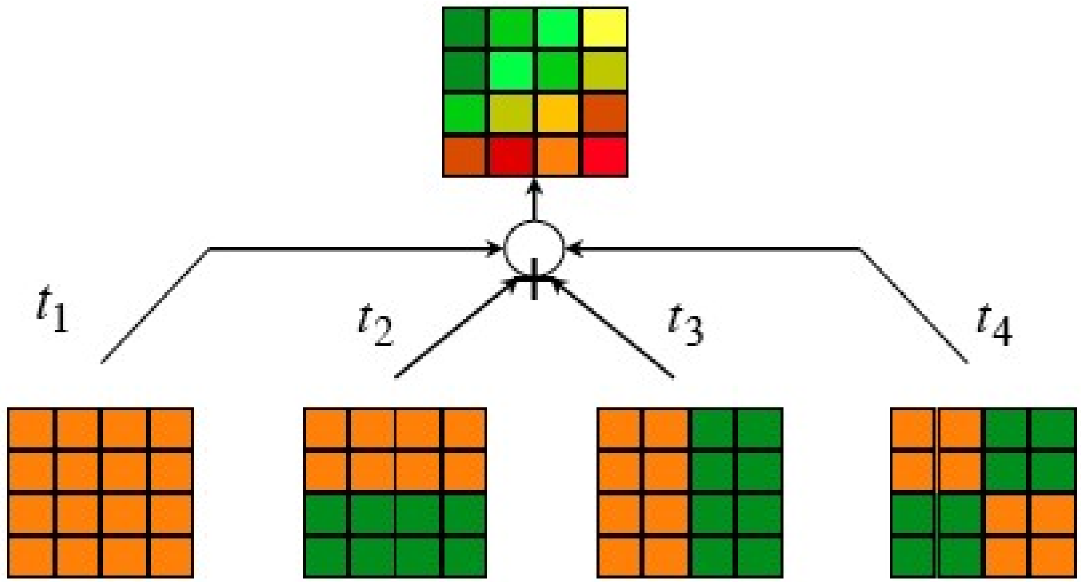
More general building blocks of image



We can have a block of pixels as a building block, “code”.

Think image as made of LEGO, how to take it apart, and put it back.

More general building blocks of image



$$I = T_1 * t_1 + T_2 * t_2 + T_3 * t_3 + T_4 * t_4;$$

How to estimate t_i

Coding book of DCT (discrete Cosine)

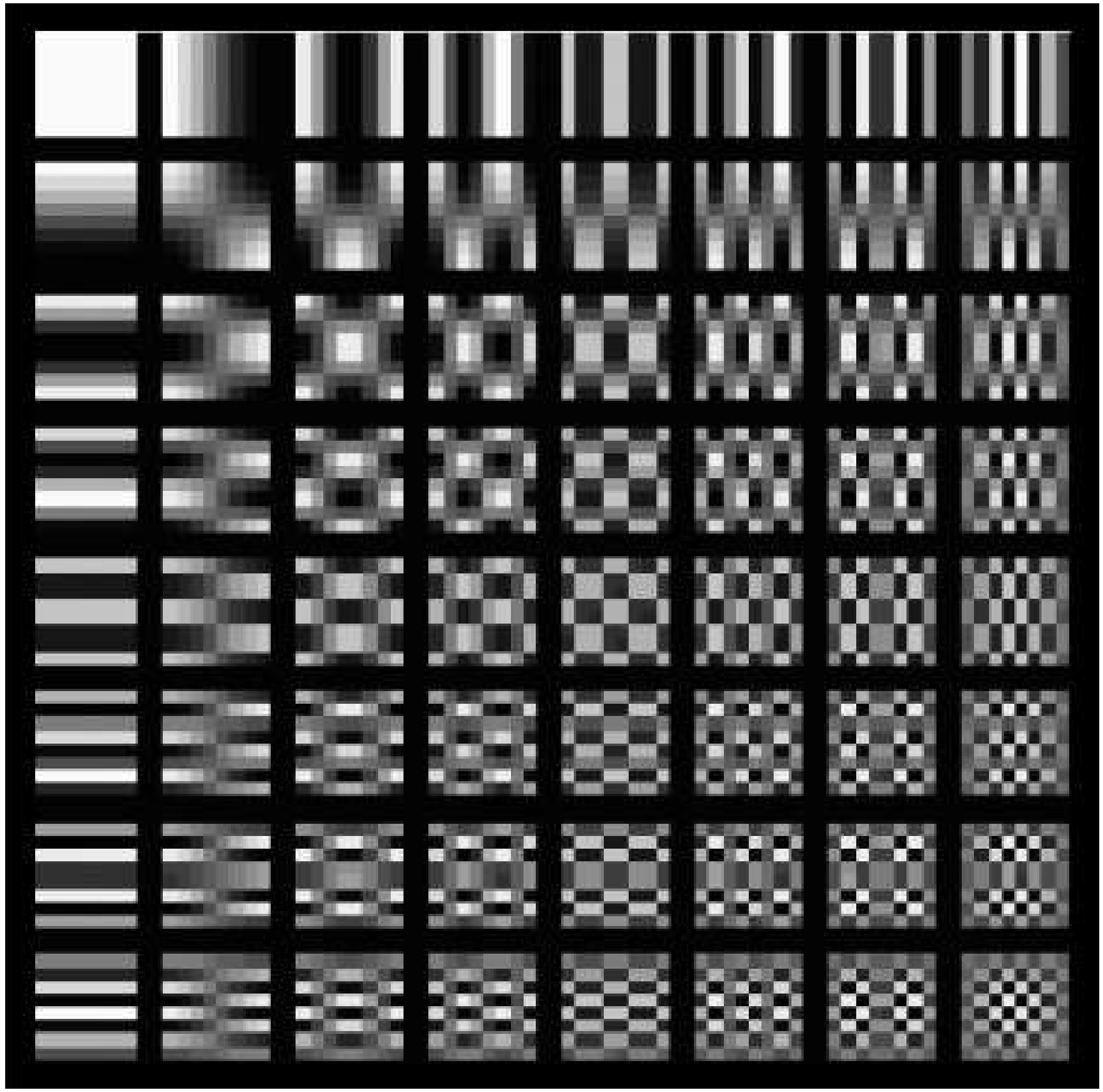
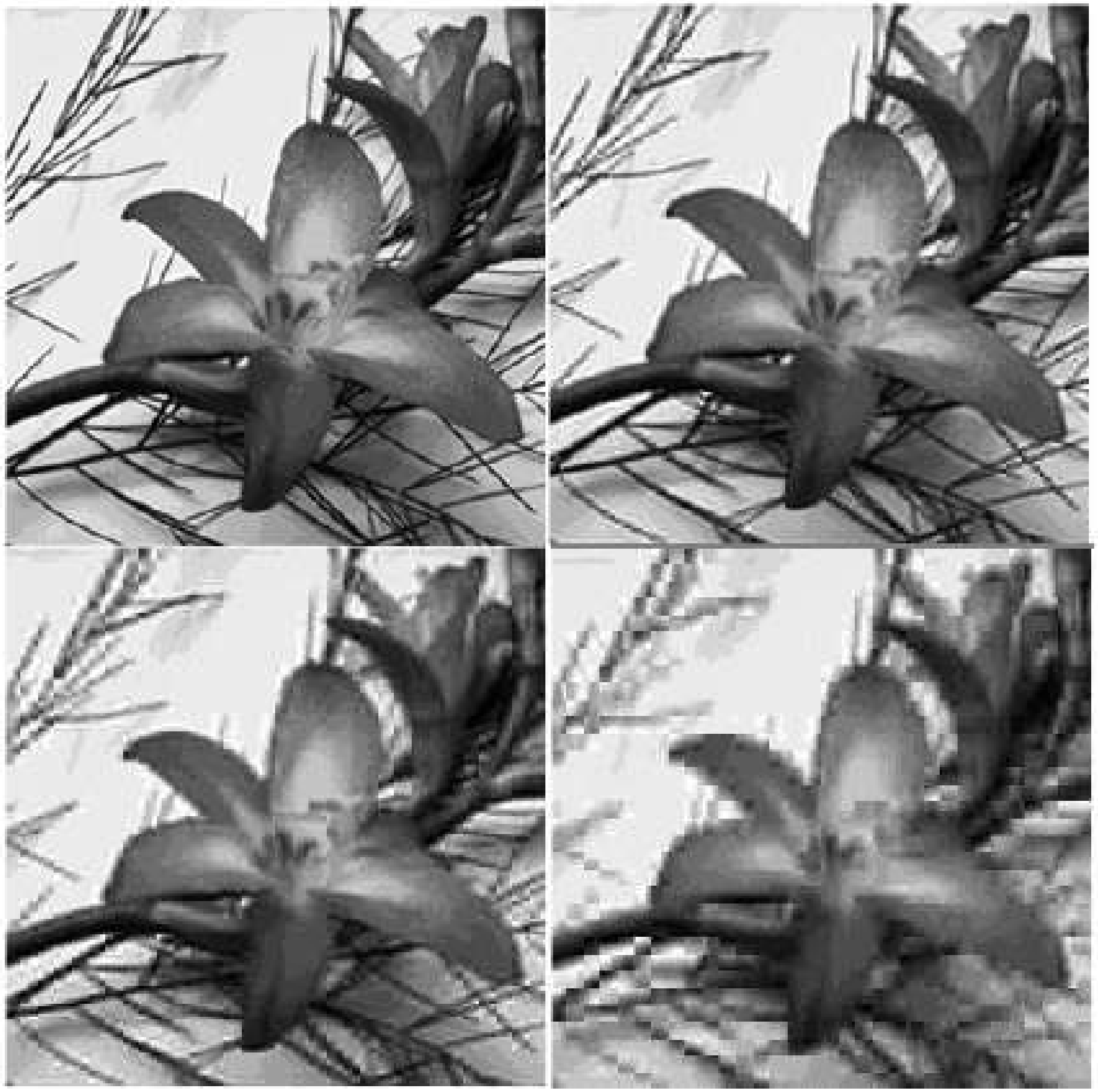


Image coding with DCT



Original

With 16/64 Coefficients

With 8/64 Coefficients

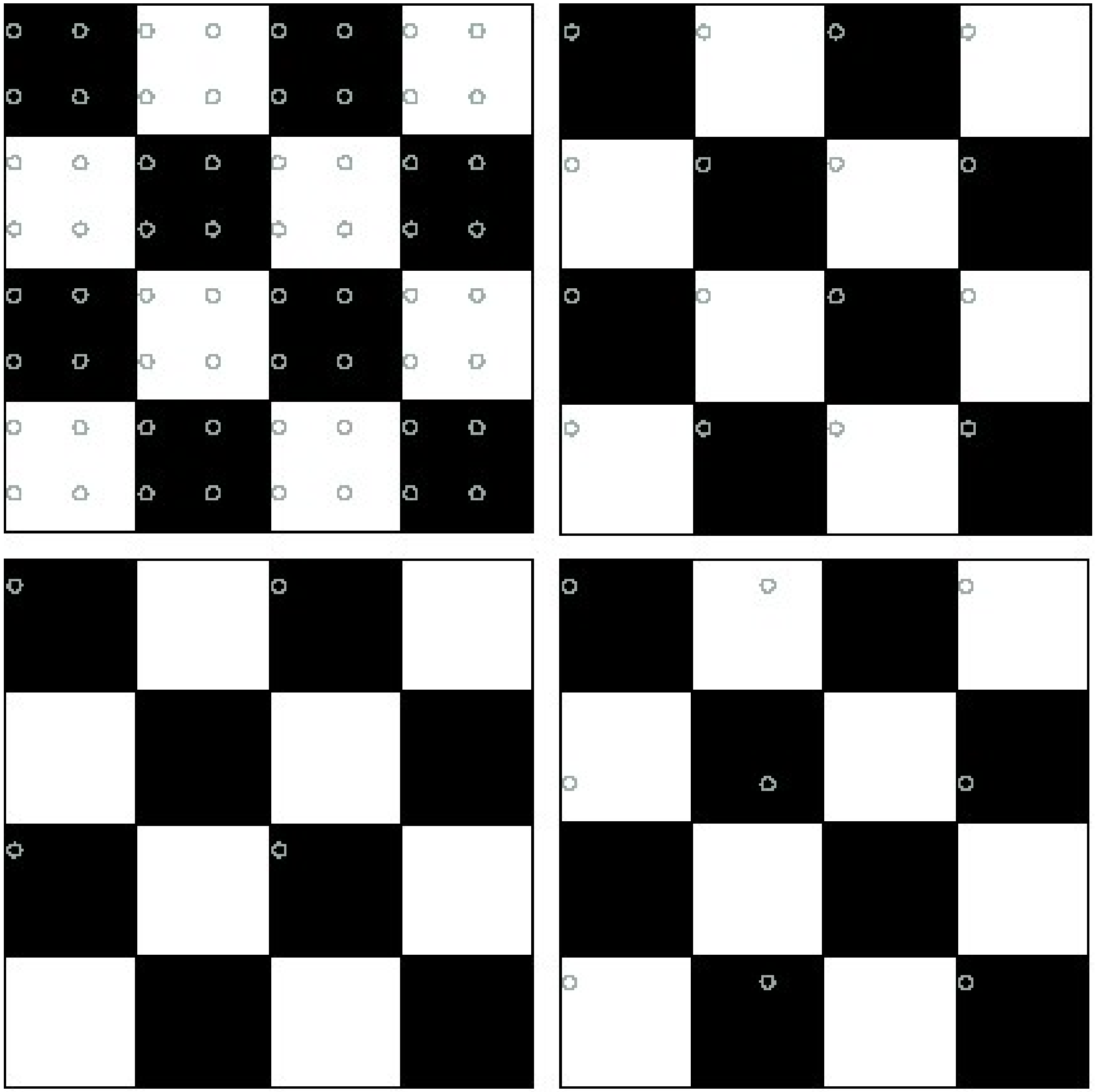
With 4/64 Coefficients

Image Pyramids

© Kenneth Kwan

Slides Modified from Alexei Efros, CMU,

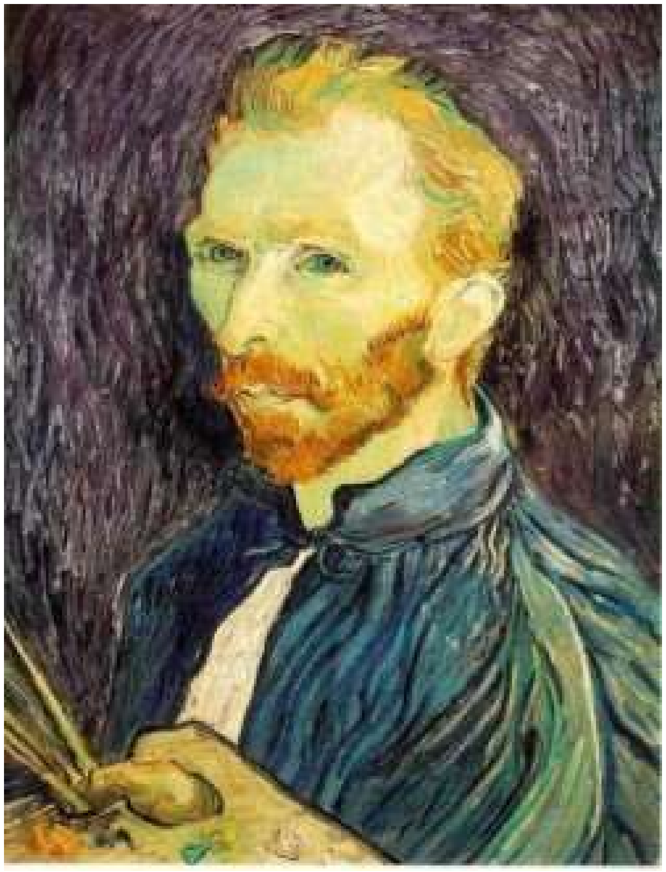
Sampling



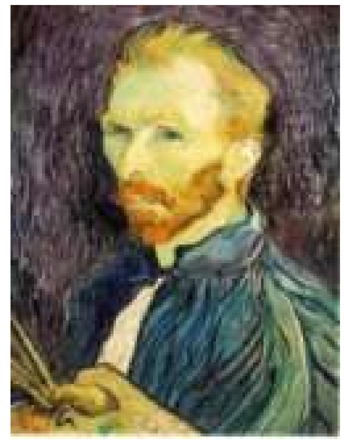
Good sampling:
- Sample often or,
- Sample wisely

Bad sampling:
- see aliasing in action!

Gaussian pre-filtering



Gaussian 1/2



G 1/4

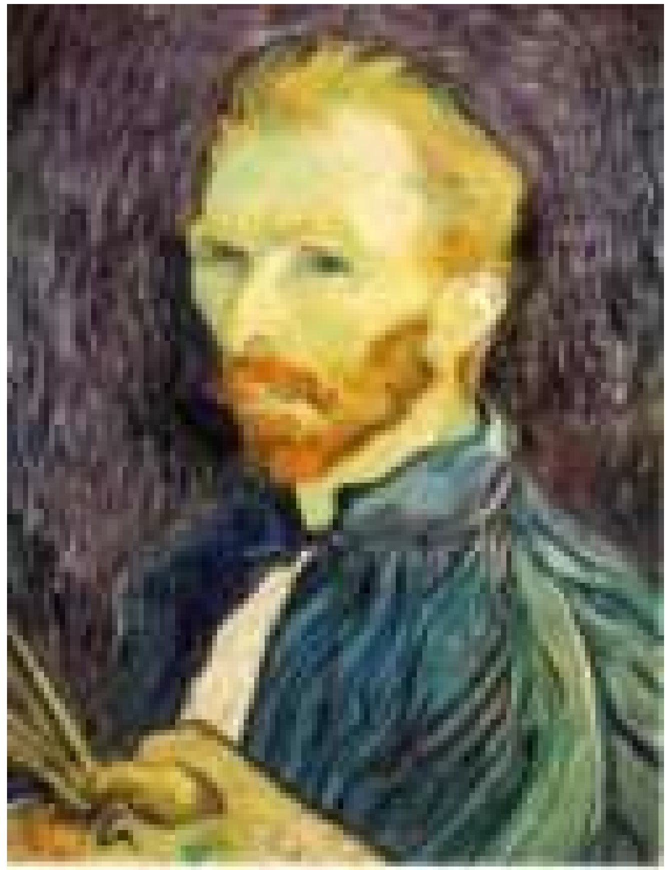
G 1/8

Solution: filter the image, *then* subsample

- Filter size should double for each ½ size reduction. Why?

Subsampling with Gaussian pre-filtering

Gaussian 1/2



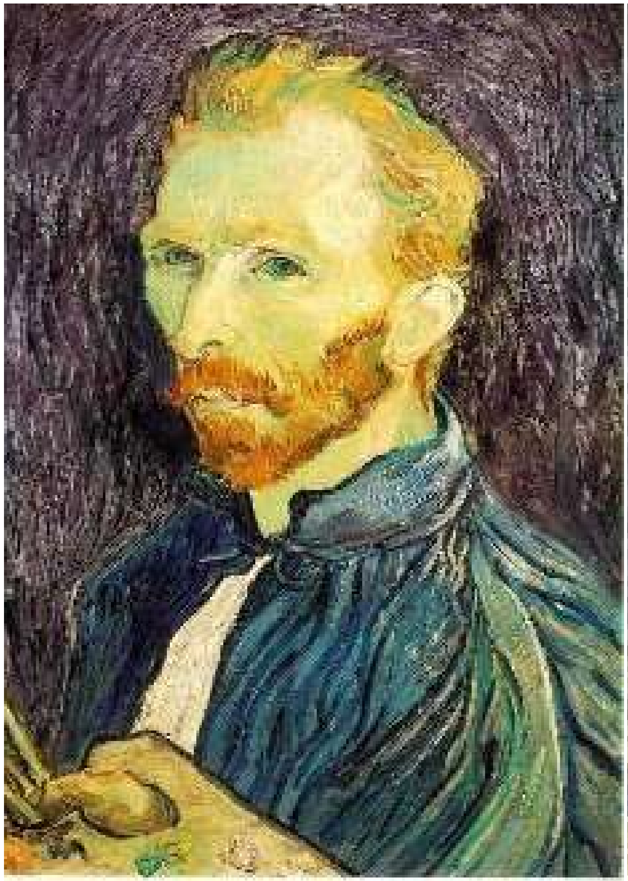
G 1/4

G 1/8

Solution: filter the image, *then* subsample

- Filter size should double for each ½ size reduction. Why?
- How can we speed this up?

Compare with...



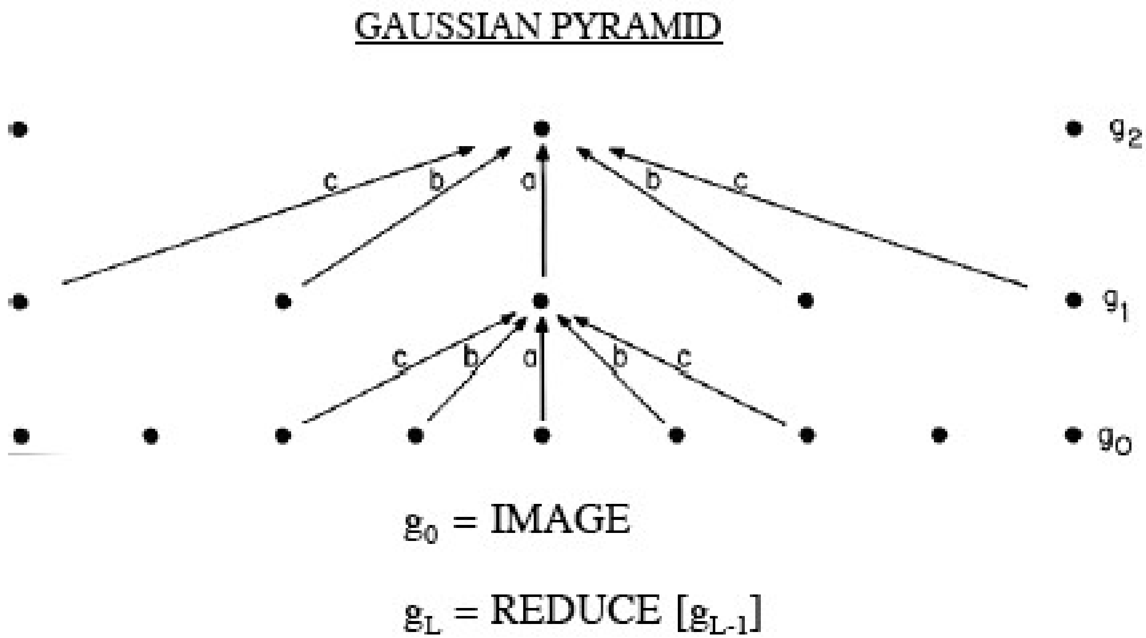
1/2

1/4 (2x zoom)

1/8 (4x zoom)

Gaussian Pyramid for encoding

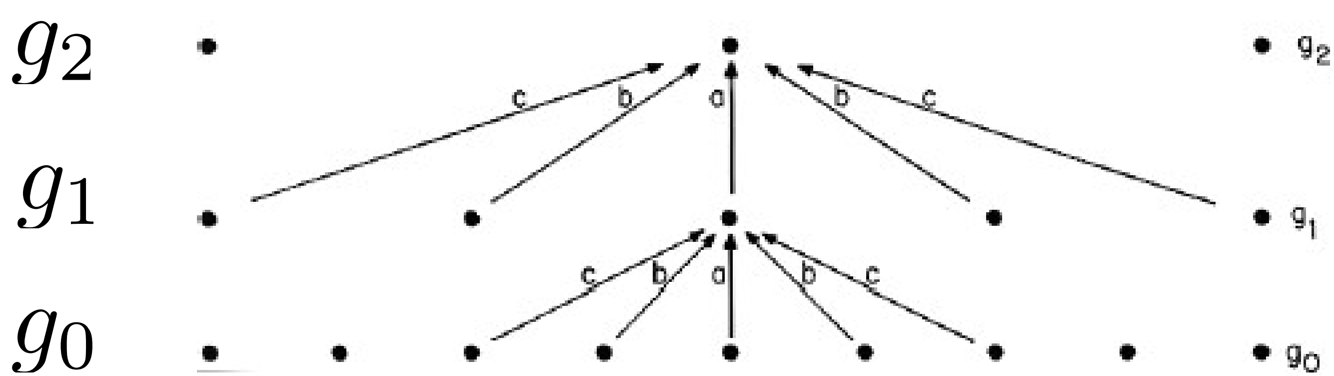
[Burt & Adelson, 1983]



1) **Prediction using weighted local Gaussian average**
2) **Encode the difference as the Laplacian**
3) **Both Laplacian and the Averaged image is easy to encode**

Gaussian pyramid

GAUSSIAN PYRAMID



g_0 = IMAGE

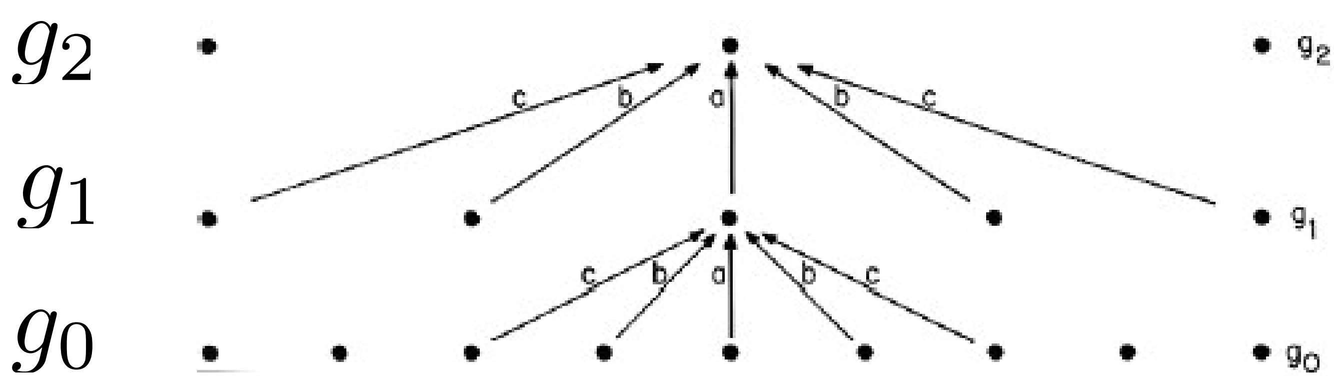
g_L = REDUCE [g_{L-1}]

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

$$\mathrm{w}(m, n) = \hat{w}(m)\ \hat{w}(n). \qquad \sum_{m=-2}^{2} \hat{w}(m) = 1 \qquad \hat{w}(i) = \hat{w}(-i)$$

Gaussian pyramid

GAUSSIAN PYRAMID



g_0 = IMAGE

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

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

$$g_l = [g_{l-1} \otimes w] \downarrow 2$$

Choice in weighting function

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

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

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

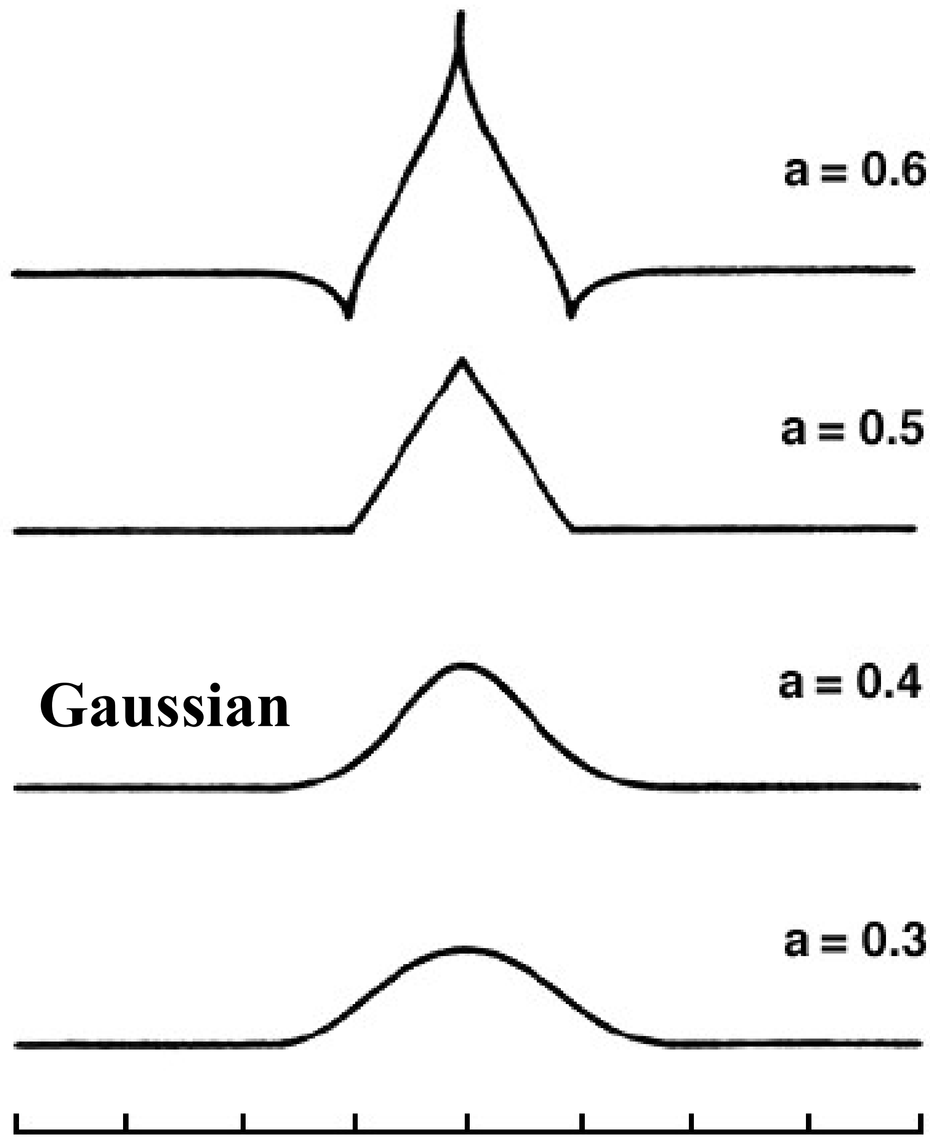
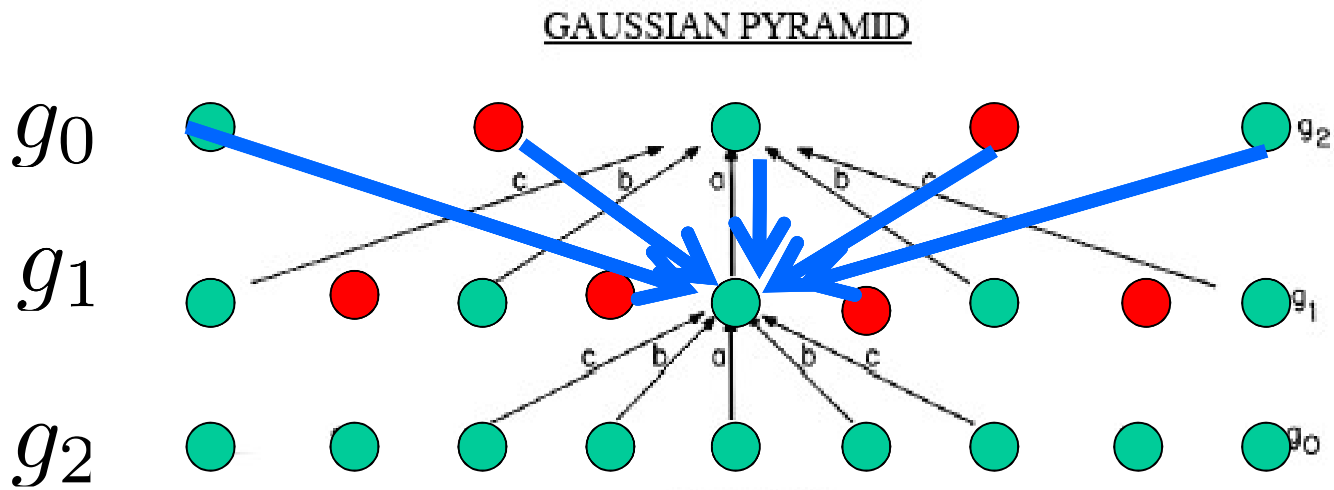


Image Expansion



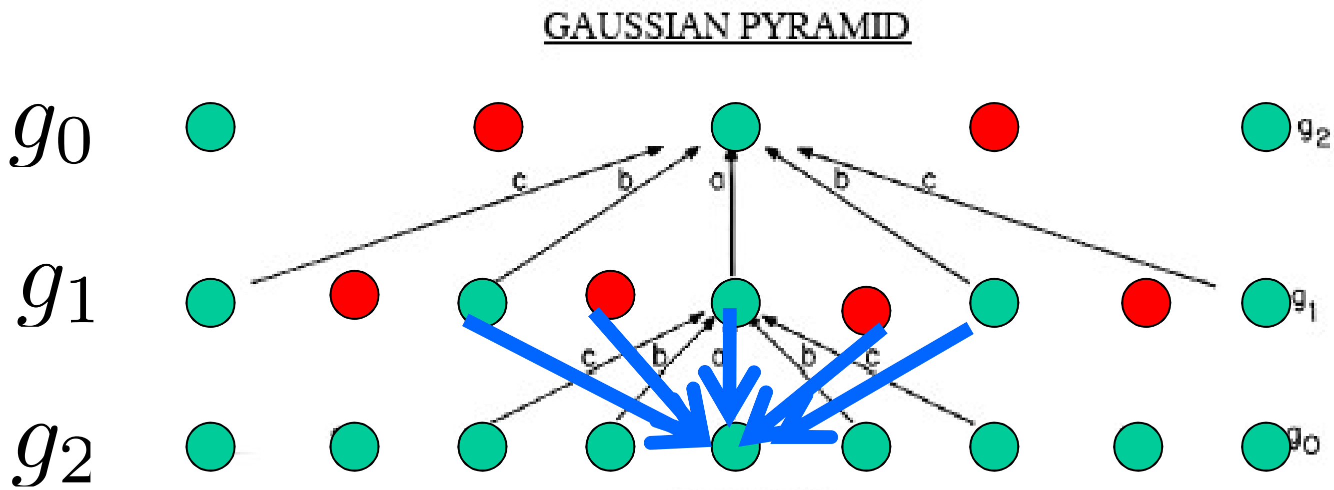
g_0 = IMAGE

g_L = REDUCE [g_{L-1}]

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

$(i-m)/2$ and $(j-n)/2$ are integers

Image Expansion



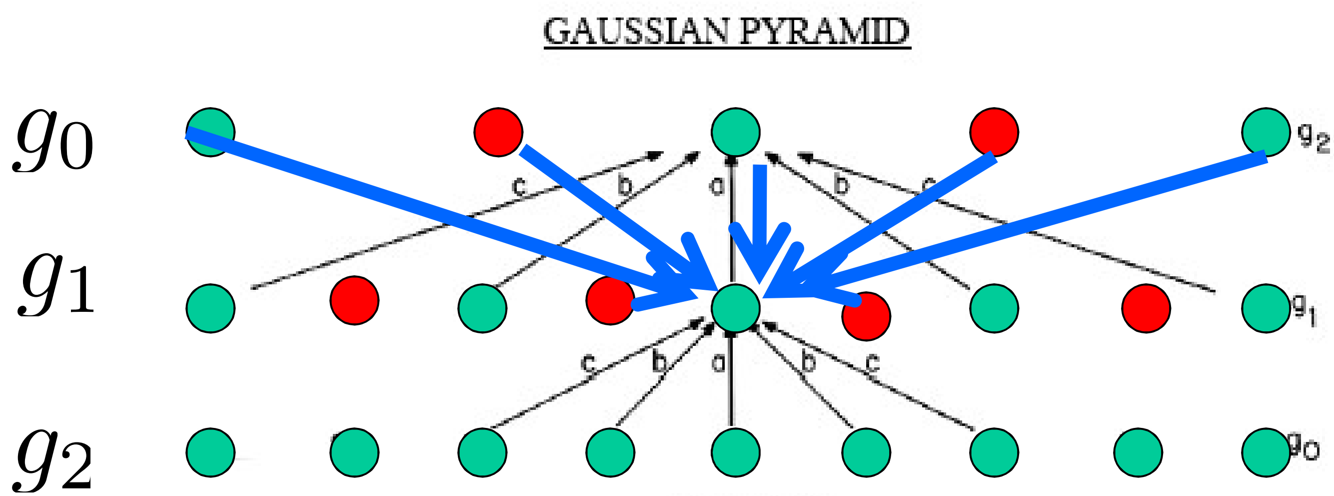
g_0 = IMAGE

g_L = REDUCE [g_{L-1}]

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

$(i-m)/2$ and $(j-n)/2$ are integers

Image Expansion



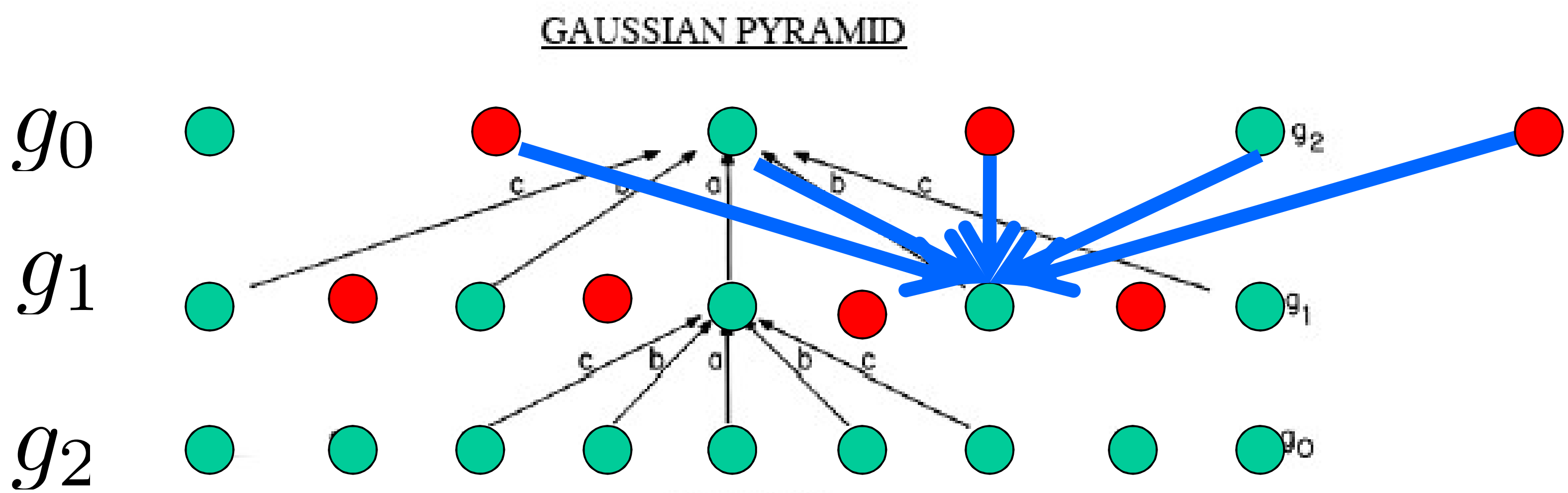
g_0 = IMAGE

g_L = REDUCE [g_{L-1}]

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

$(i-m)/2$ and $(j-n)/2$ are integers

Image Expansion



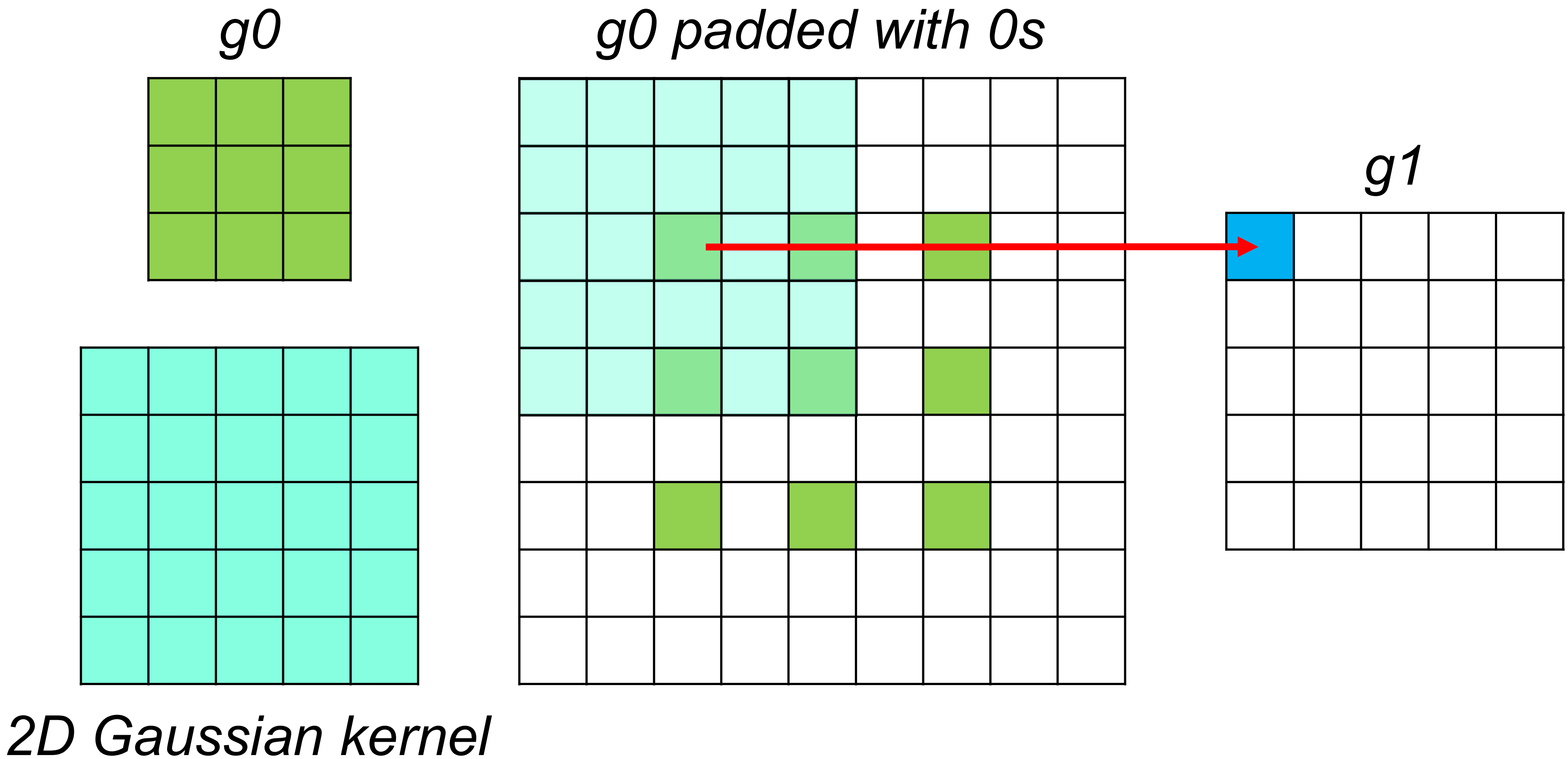
g_0 = IMAGE

g_L = REDUCE [g_{L-1}]

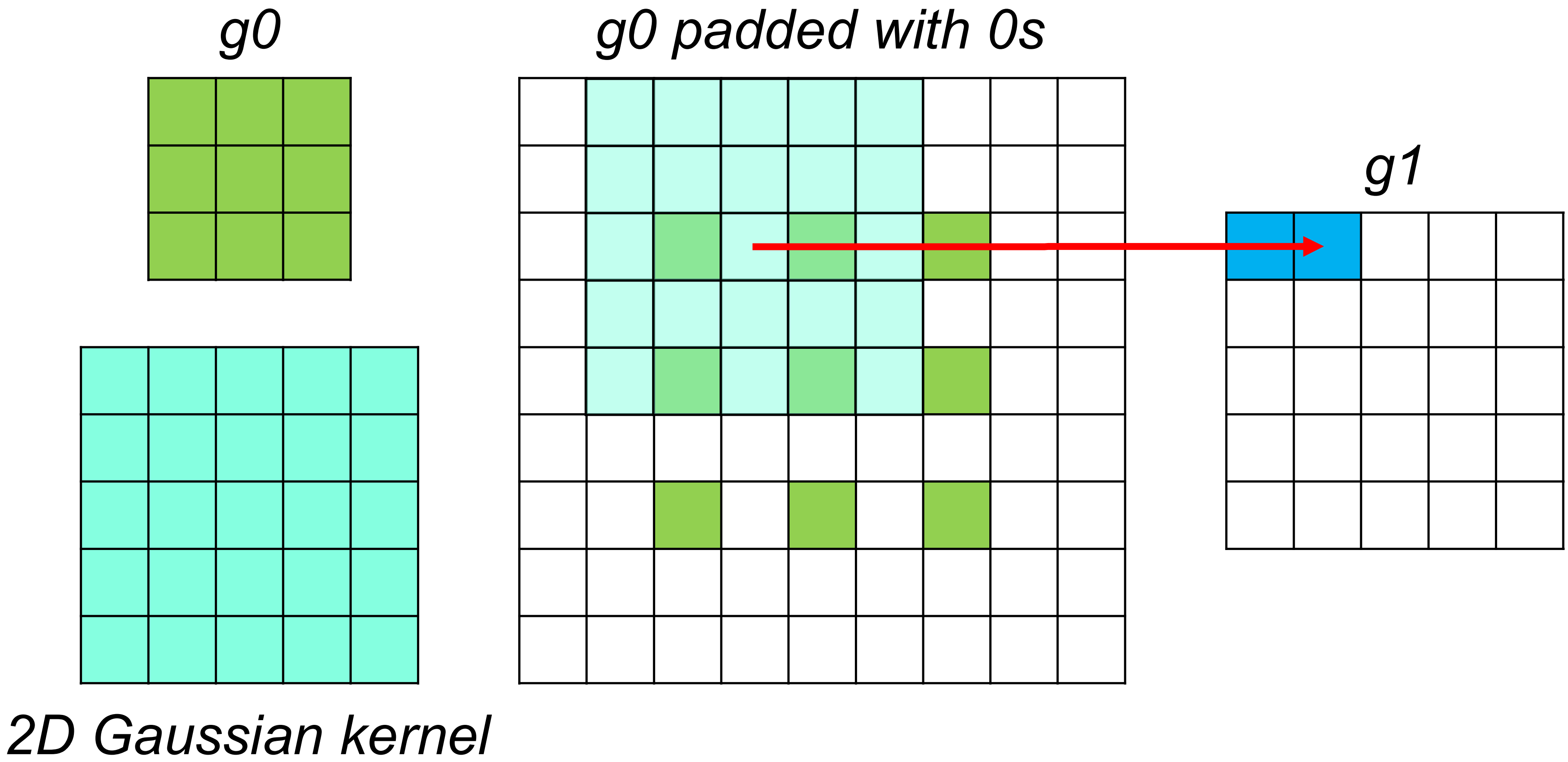
$$g_{l,n}(ij) = 4 \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) \bullet g_{l,n-1}\left(\frac{i-m}{2}, \frac{j-n}{2}\right).$$

$(i-m)/2$ and $(j-n)/2$ are integers

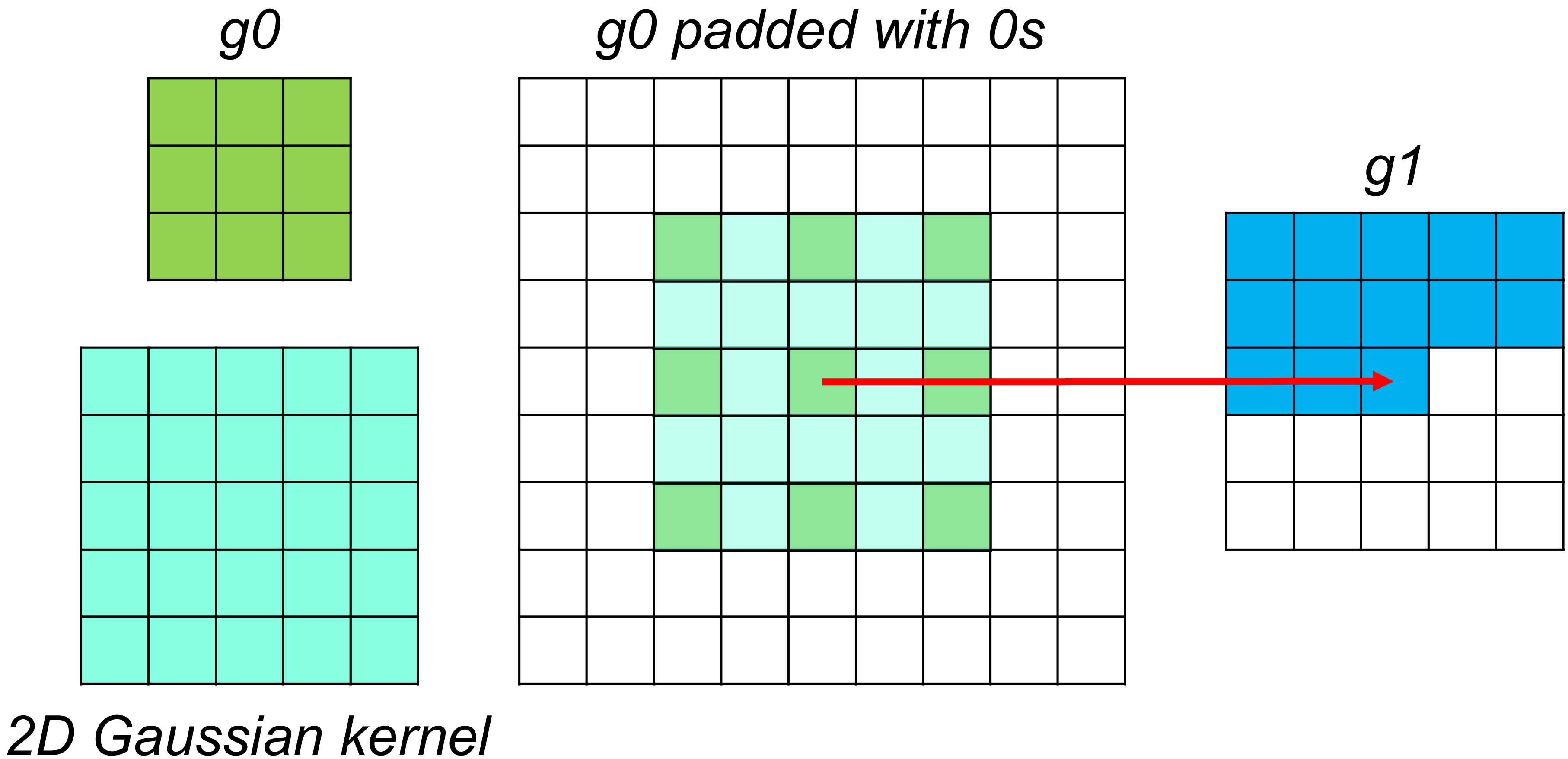
2D Image Expansion (part1)



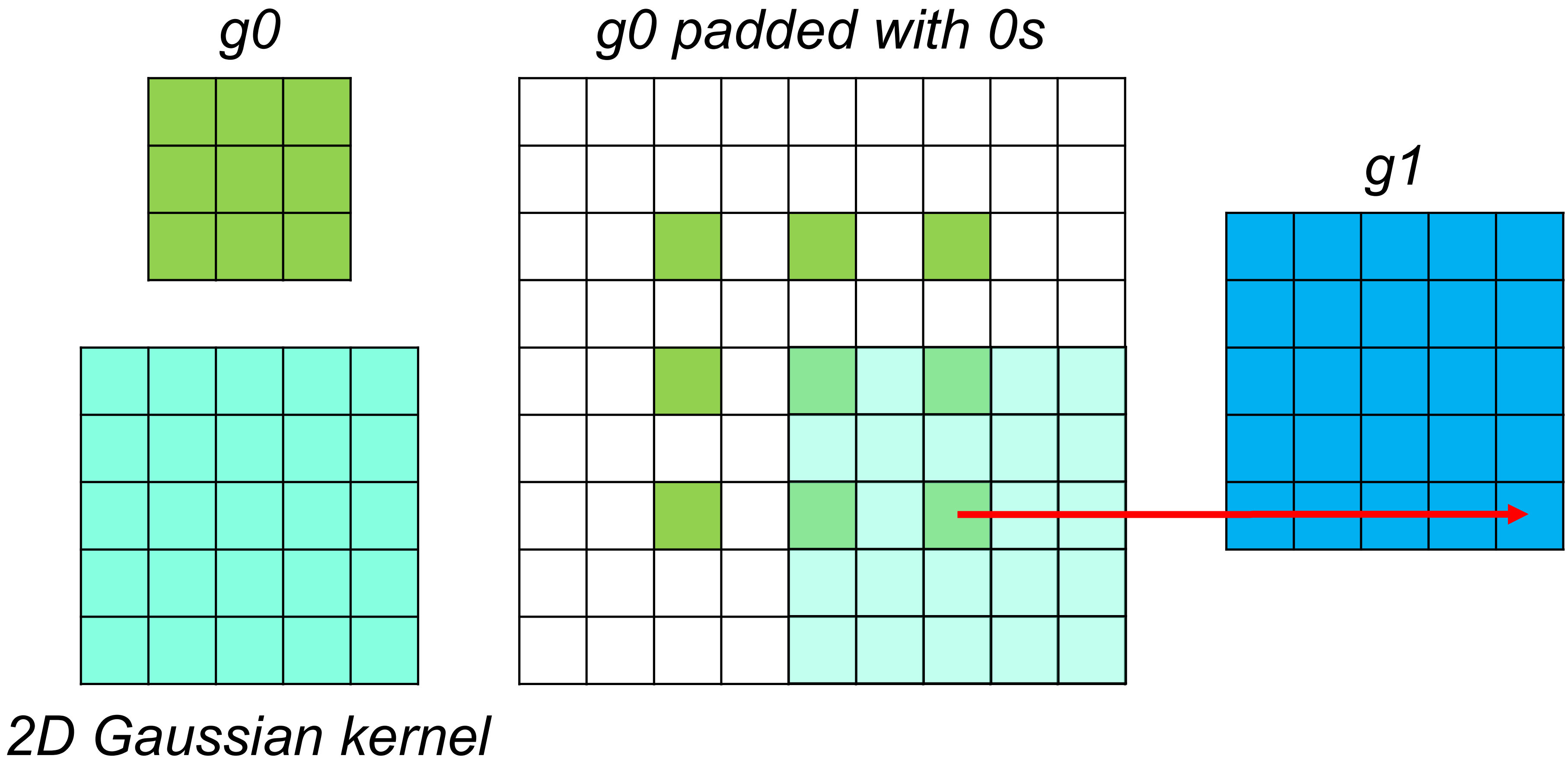
2D Image Expansion (part2)



2D Image Expansion (part3)



2D Image Expansion (part4)



What does blurring take away?

original

What does blurring take away?

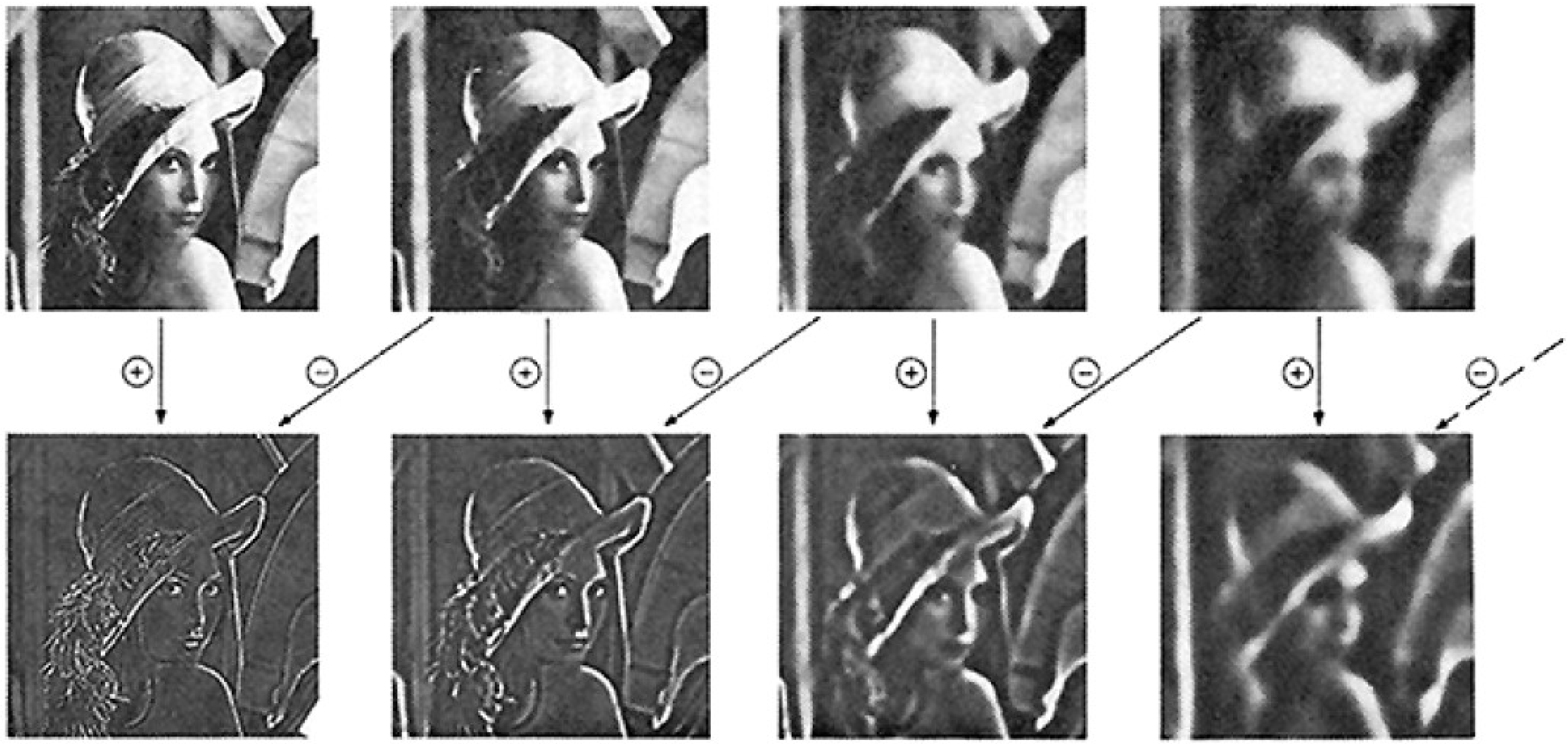
smoothed (5x5 Gaussian)

High-Pass filter

smoothed – original

Laplacian Image

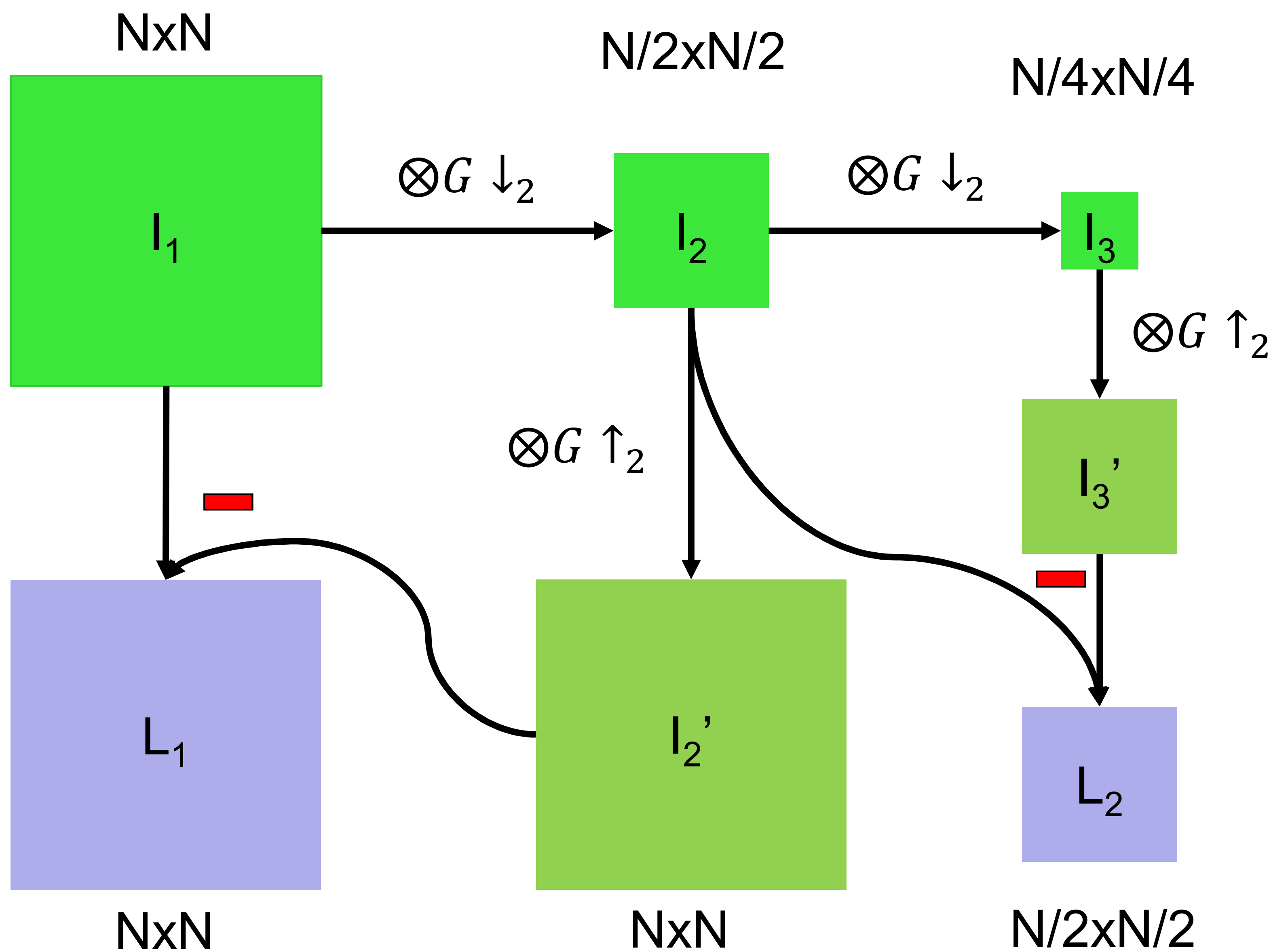
$$L_l = g_l - \text{EXPAND}(g_{l+1})$$

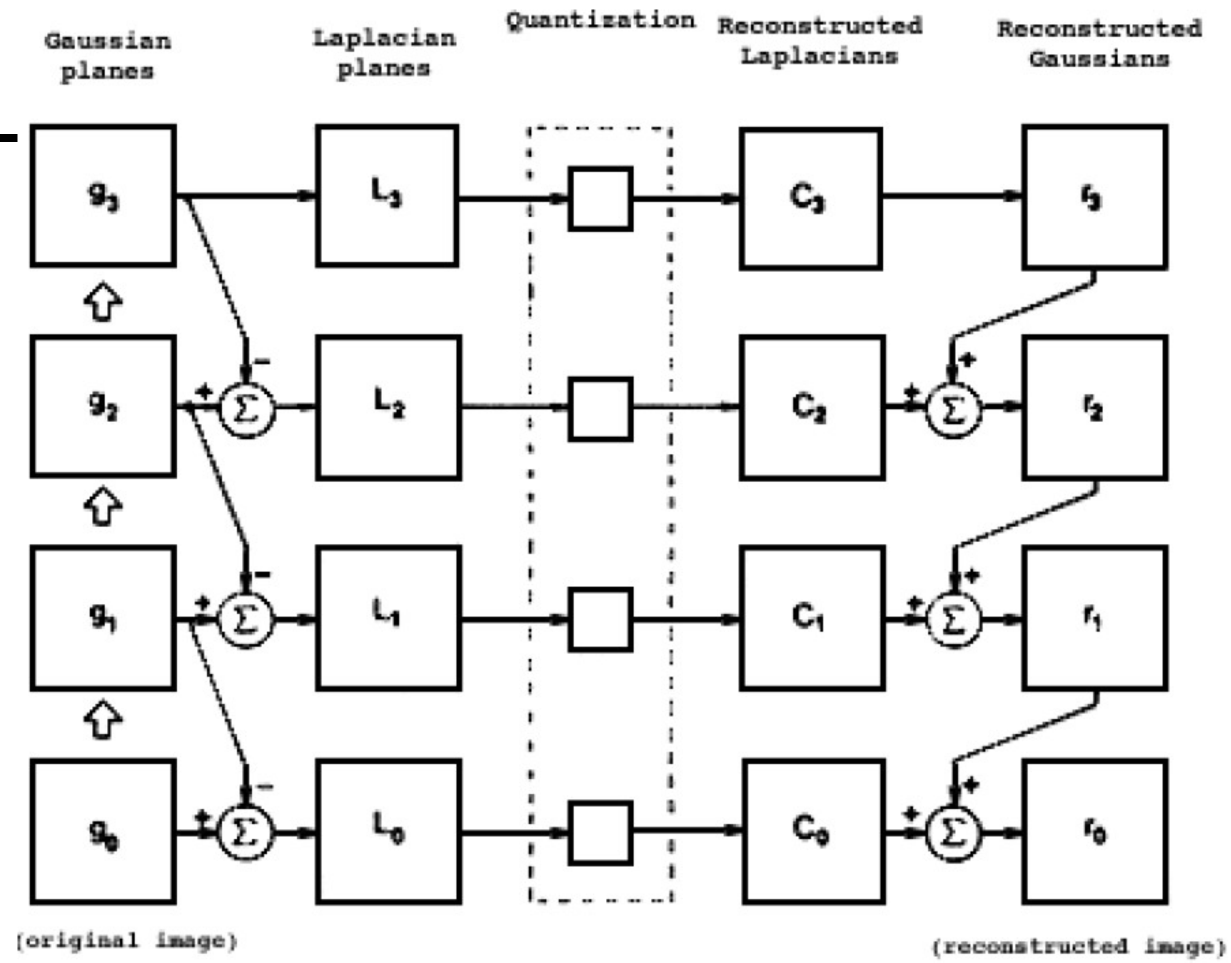


Gaussian pyramid is smooth=> can be subsampled

Laplacian pyramid has narrow band of frequency=> compressed

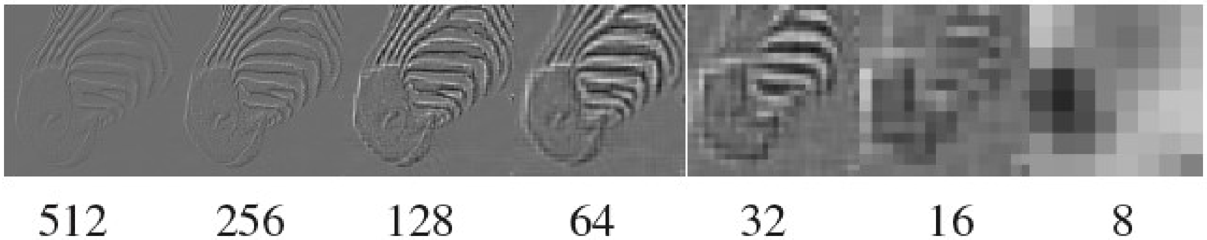
Pyramid extraction of Laplacian

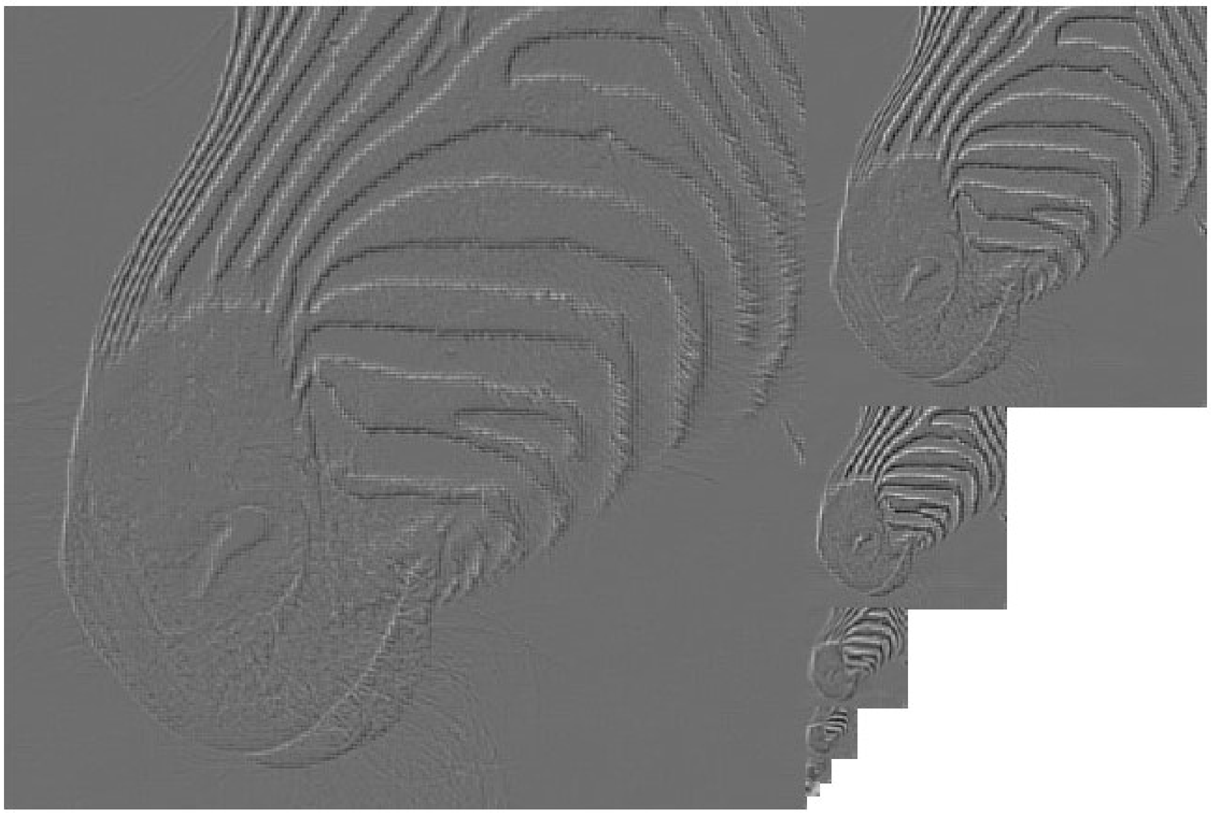




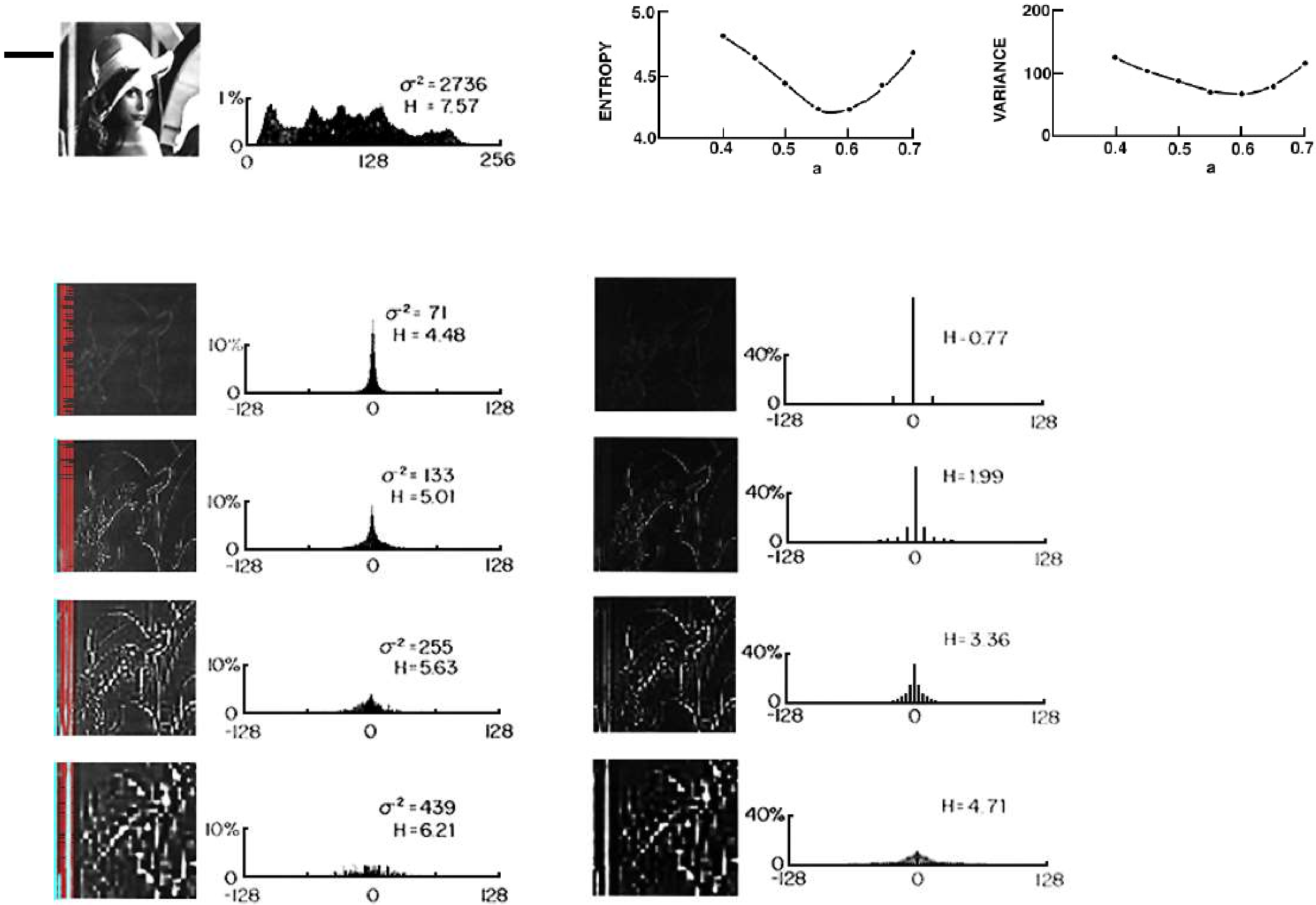
$$g_N = L_N$$

$$g_l = L_l + \mathrm{EXPAND}(g_{l+1}).$$



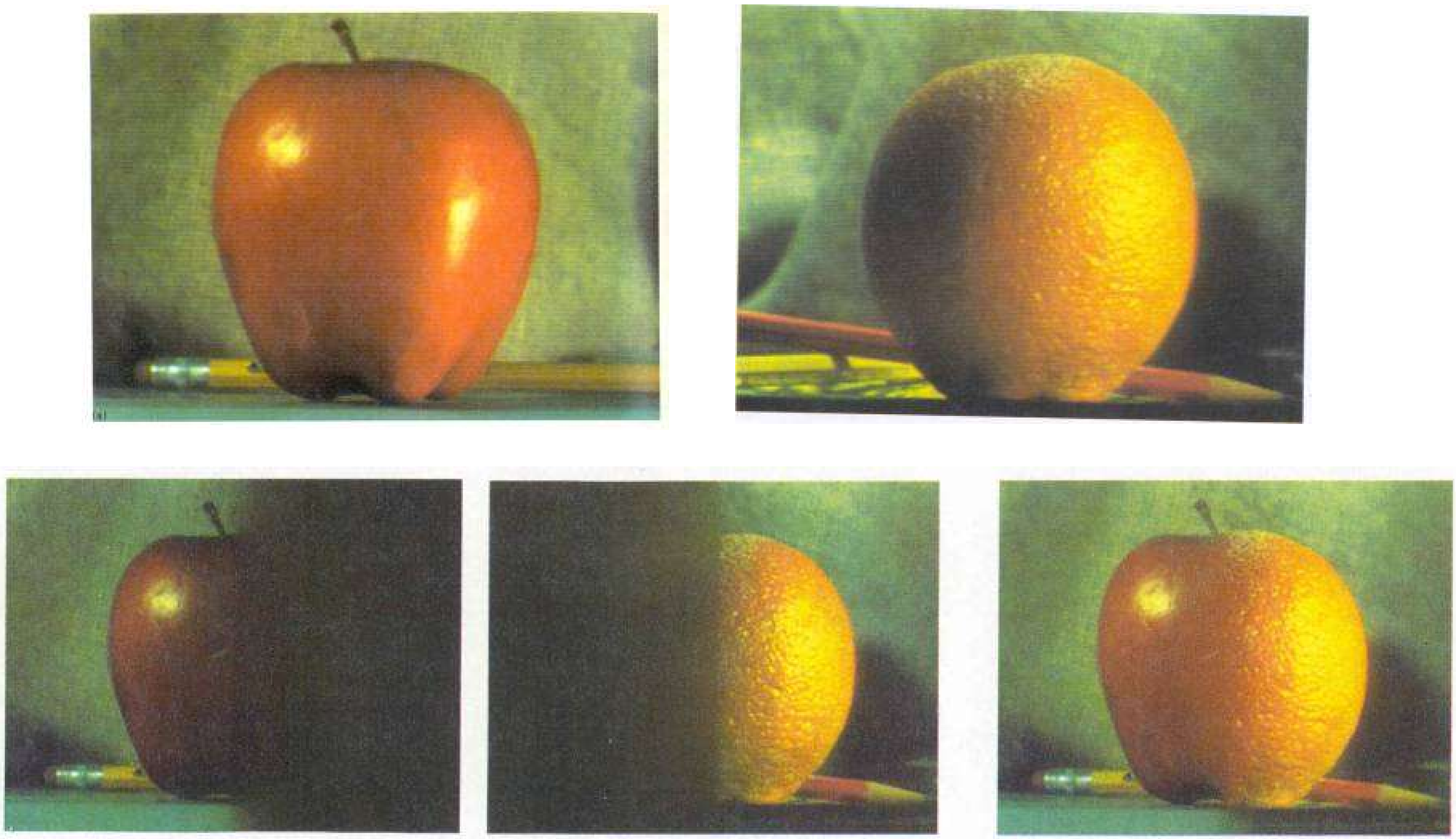


Ln = Gn



σ² = 2736
H = 7.57
1%
0
0
128
256
5.0
4.5
4.0
ENTROPY
0.4
0.5
0.6
0.7
a
200
100
0
VARIANCE
σ² = 71
H = 4.48
10%
-128
0
128
H = 0.77
40%
σ² = 133
H = 5.01
H = 1.99
σ² = 255
H = 5.63
H = 3.36
σ² = 439
H = 6.21
H = 4.71

Pyramid Blending

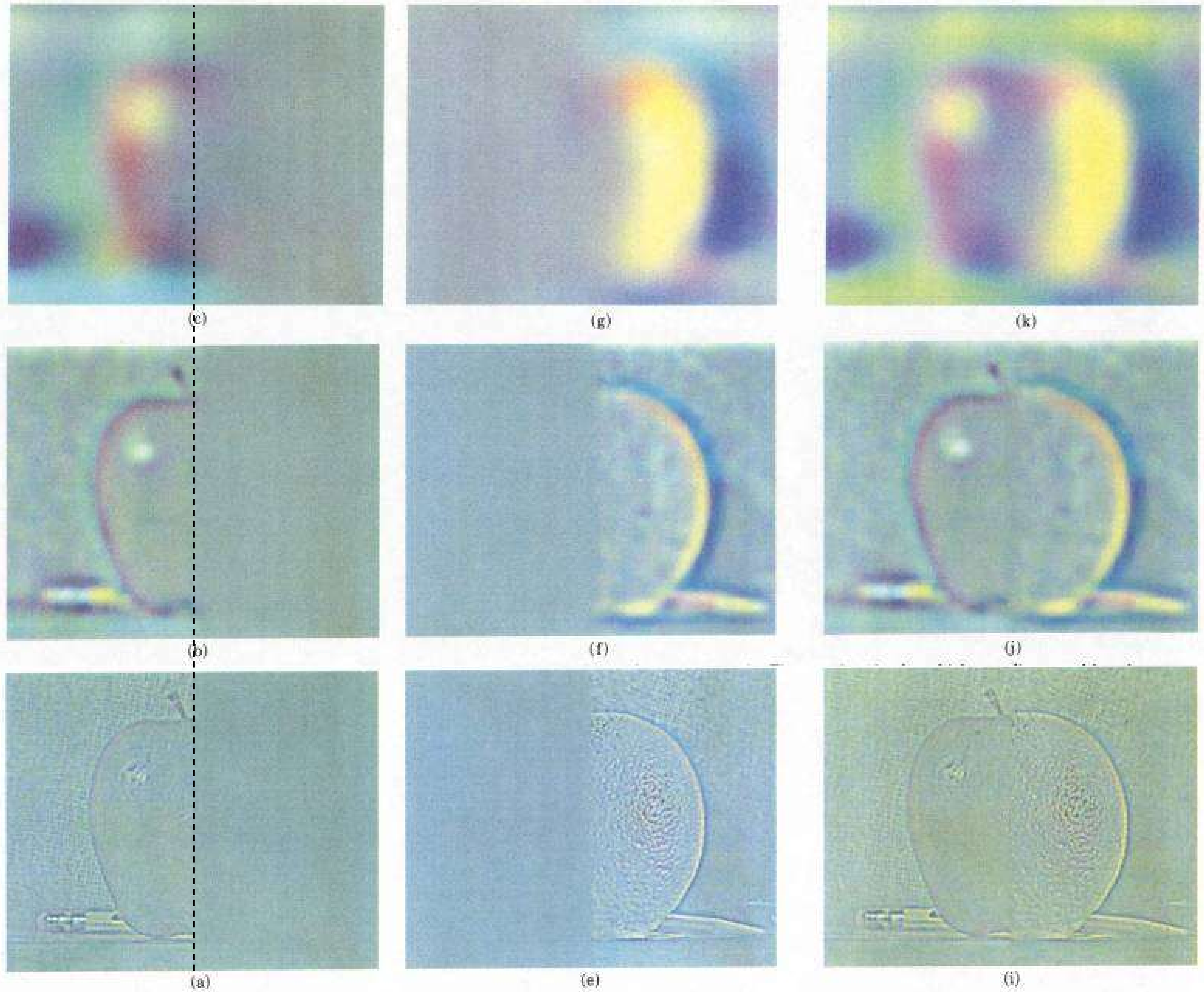


(d) (h) (l)

laplacian level 4

laplacian level 2

laplacian level 0



left pyramid

right pyramid

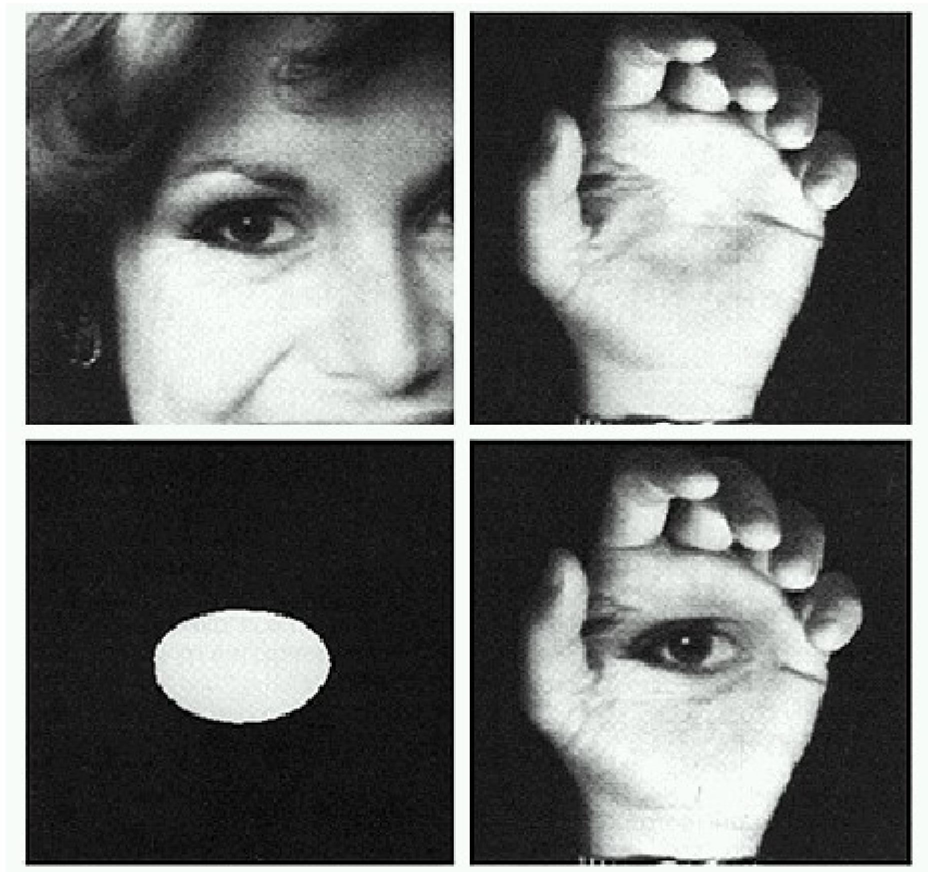
blended pyramid

Laplacian Pyramid: Blending

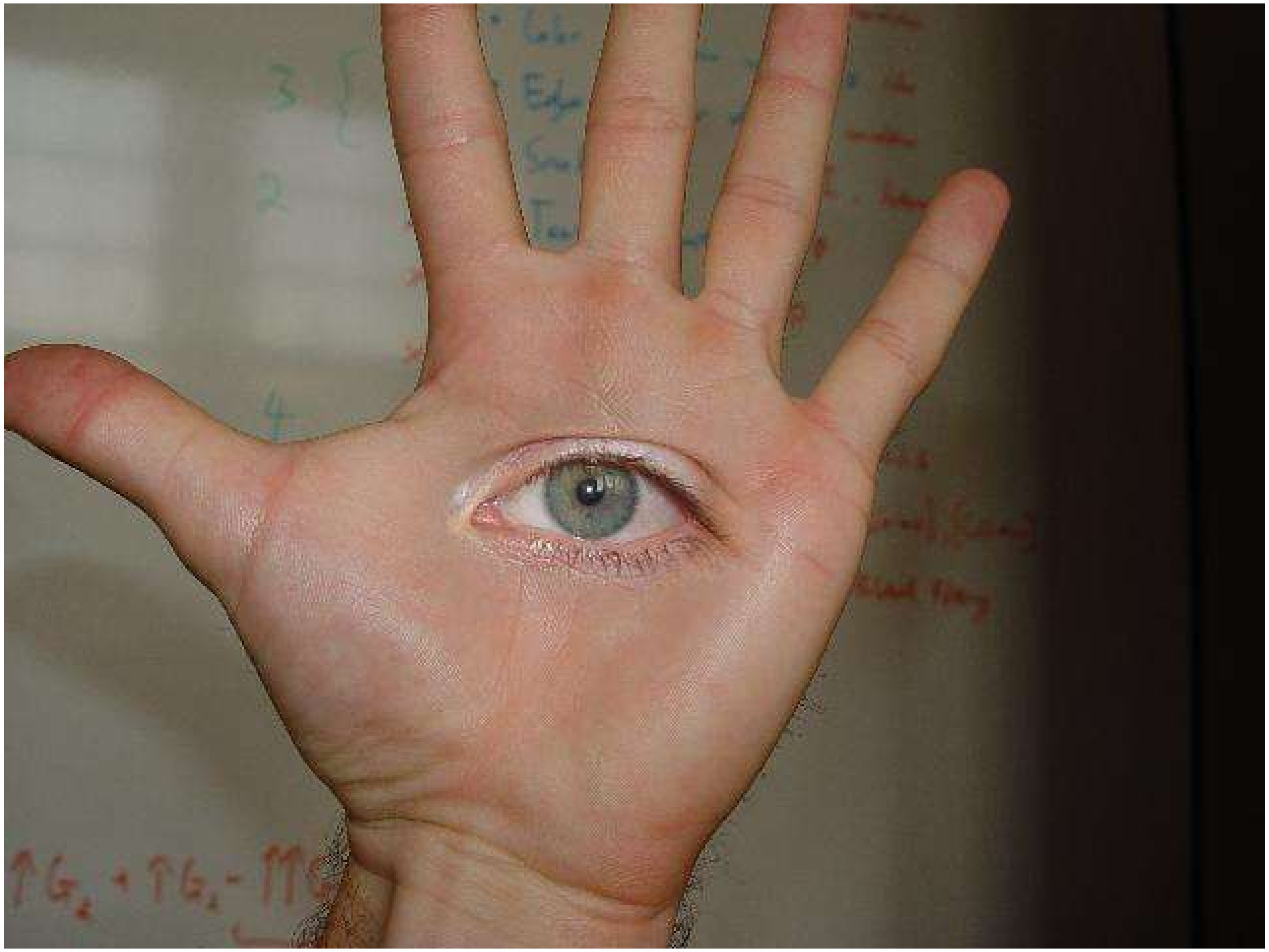
General Approach:

1. Build Laplacian pyramids *LA* and *LB* from images *A* and *B*
2. Build a Gaussian pyramid *GR* from selected region *R*
3. Form a combined pyramid *LS* from *LA* and *LB* using nodes of *GR* as weights:
   - *LS(i,j) = GR(I,j,)*LA(I,j) + (1-GR(I,j))*LB(I,j)*
4. Collapse the *LS* pyramid to get the final blended image

Blending Regions



Horror Photo



© prof. dmartin