What are image pyramids used for?

image compression

multi-scale texture mapping

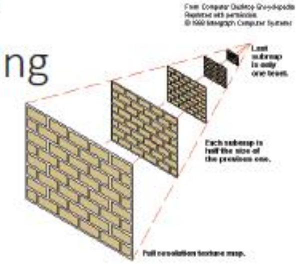


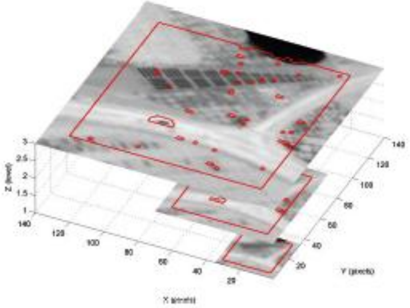
image blending

focal stack compositing

denoising

multi-scale detection

multi-scale registration

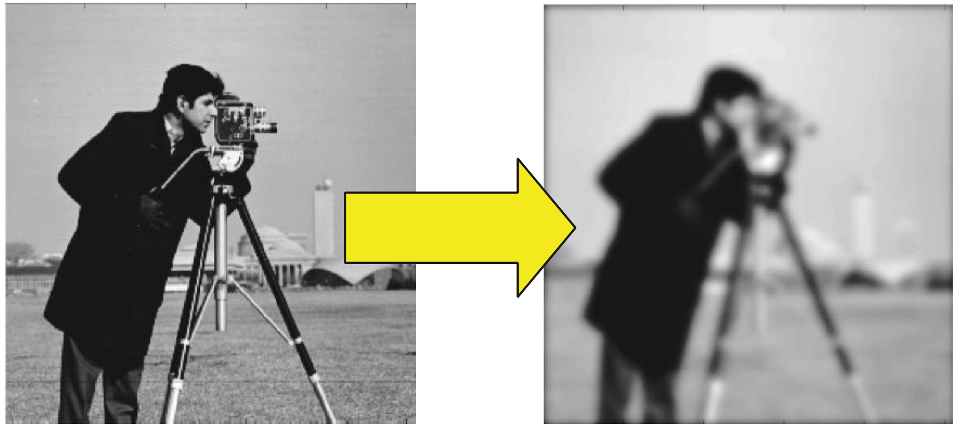


Generating a Gaussian Pyramid

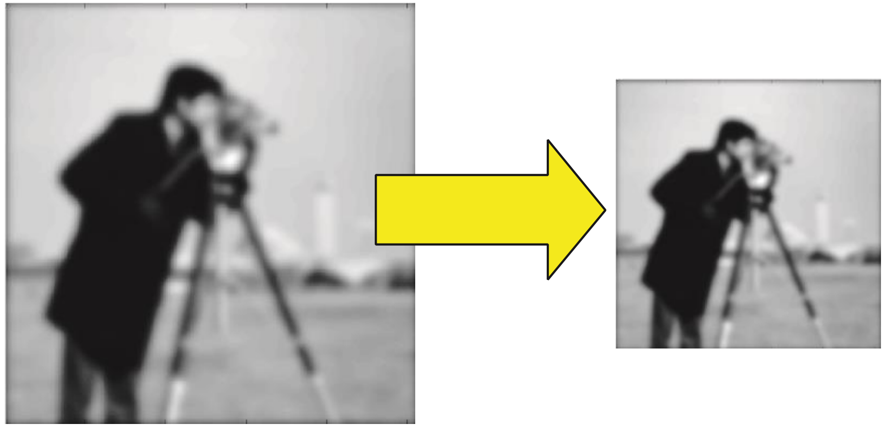
Basic Functions:

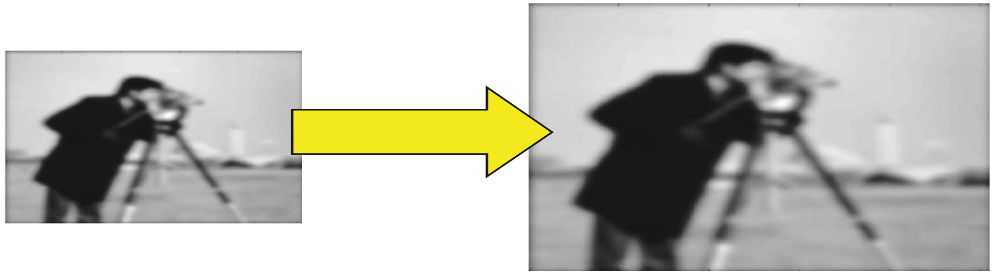
Blur (convolve with Gaussian to smooth image)

We've talked about blur already



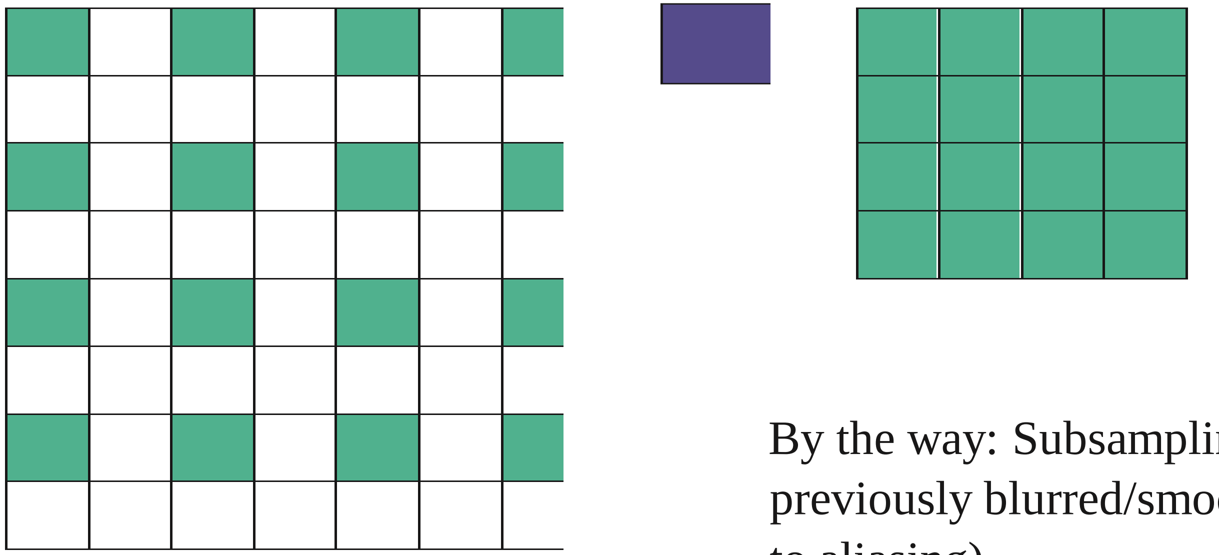
DownSample (reduce image size by half)





<u>Upsample</u> (double image size)

Downsample



By the way: Subsampling is a bad idea unless you have previously blurred/smoothed the image! (because it leads to aliasing)

To Elaborate: Thumbnails

131x97

65x48

32x24

original image
262x195

downsampled (left)
vs. smoothed then
downsampled (right)

To Elaborate: Thumbnails

original image
262x195

downsampled (left)
vs. smoothed then
downsampled (right)

131x97

To Elaborate: Thumbnails

original image
262x195

downsampled (left)
vs. smoothed then
downsampled (right)

65x48

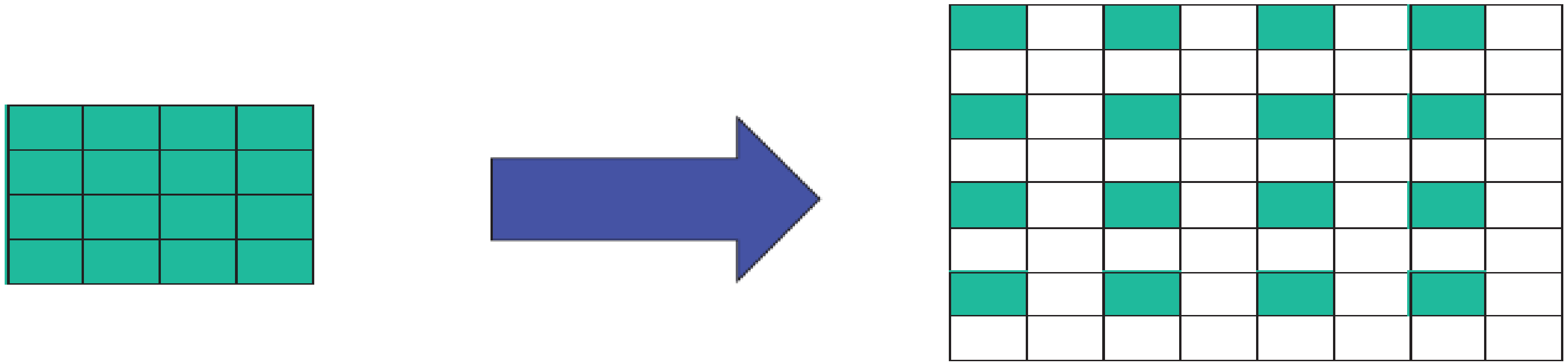
To Elaborate: Thumbnails

original image
262x195

downsampled (left)
vs. smoothed then
downsampled (right)

32x24

Upsample



How to fill in the empty values?
Interpolation:

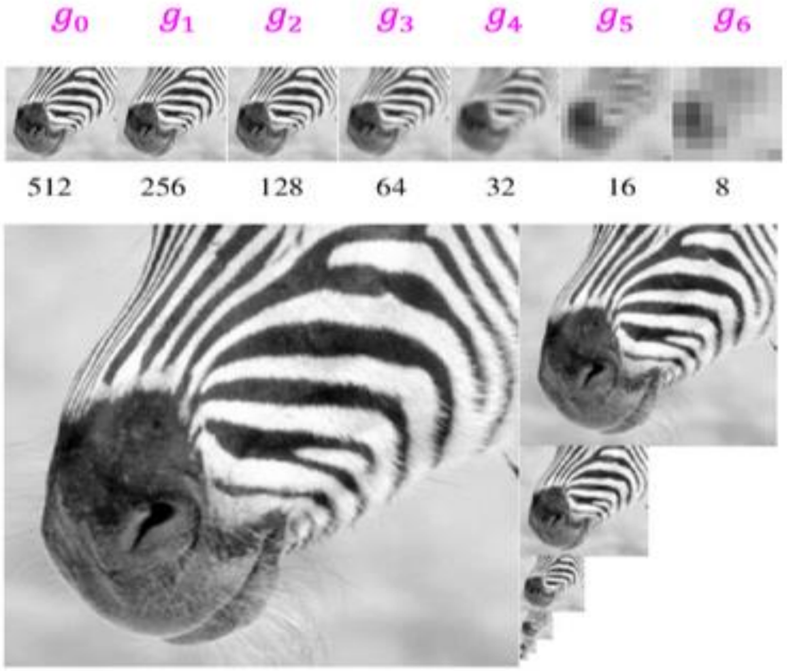
- initially set empty pixels to zero
- convolve upsampled image with Gaussian filter!
  e.g. 5x5 kernel with sigma = 1.
- **Must also multiply by 4. Explain why.**

Gaussian Image Pyramid

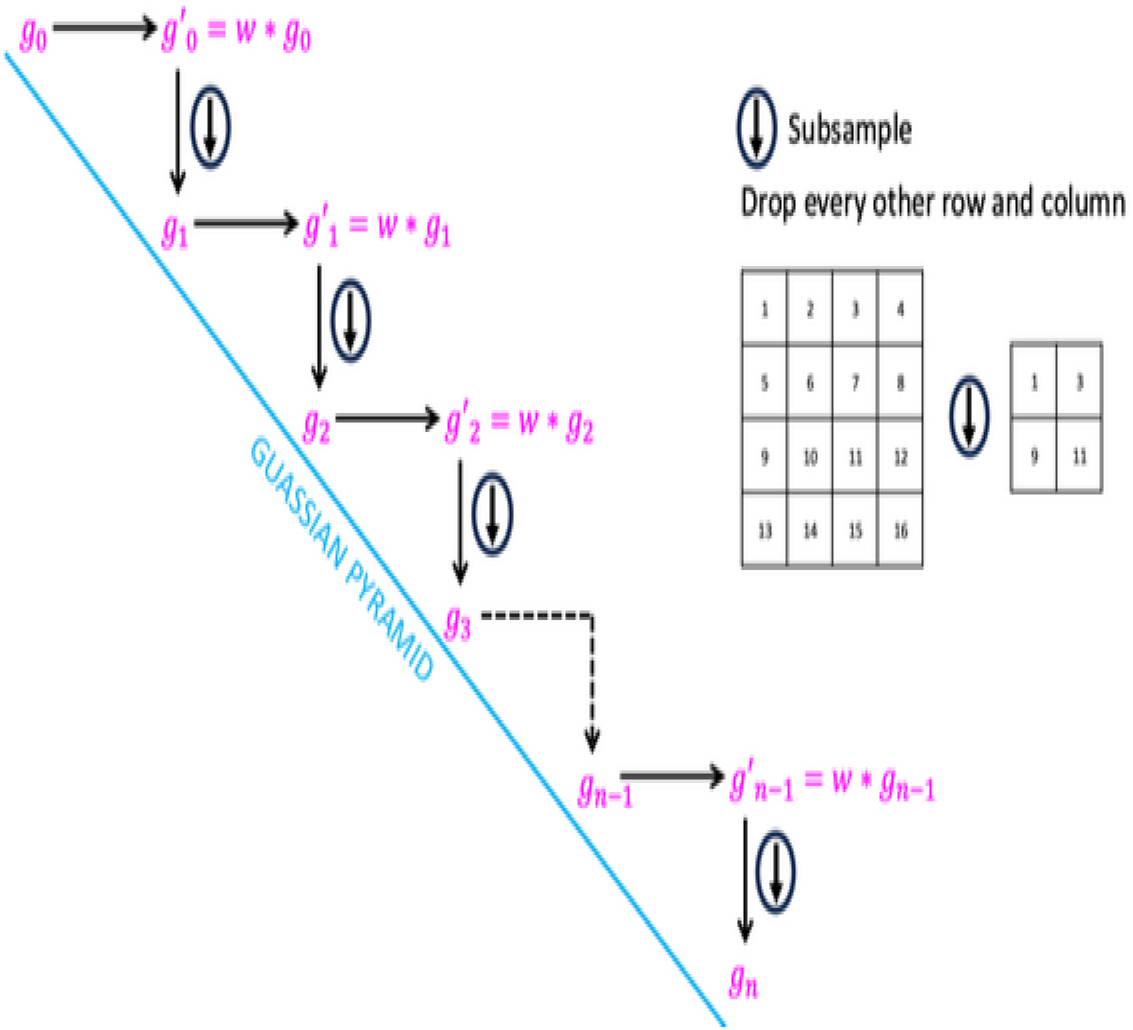
The basic idea for constructing Gaussian image pyramid is as follows:

Let $g_0 = I$ and $i = 0$

1. Blur g_i with a Gaussian kernel w to create g_i'
2. Reduce g_i' dimensions by half by discarding every other row and and every other column to construct g_{i+1}
3. Set $i = i + 1$ and repeat this process until desired numbers levels are achieved or the image is reduced to size 1×1



Starting with $g_0 = I$ the following figure depicts the construction of a Gaussian Pyramid.



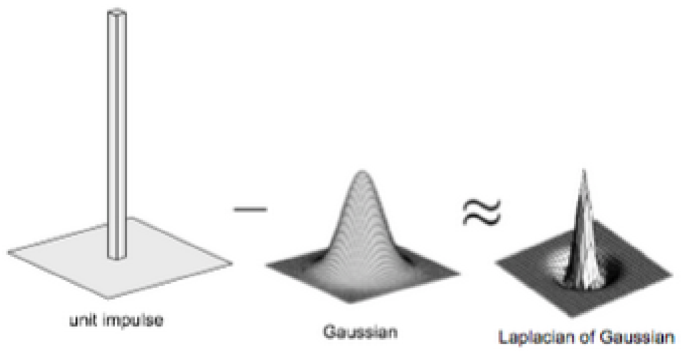
Laplacian operator

We define Laplacian operator as follows:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Approximation

In addition we can approximate the Laplacian of a Gaussian as follows:



Source: Lazebnik

We use this property when constructing Laplacian image pyramids above.

Laplacian of a Gaussian

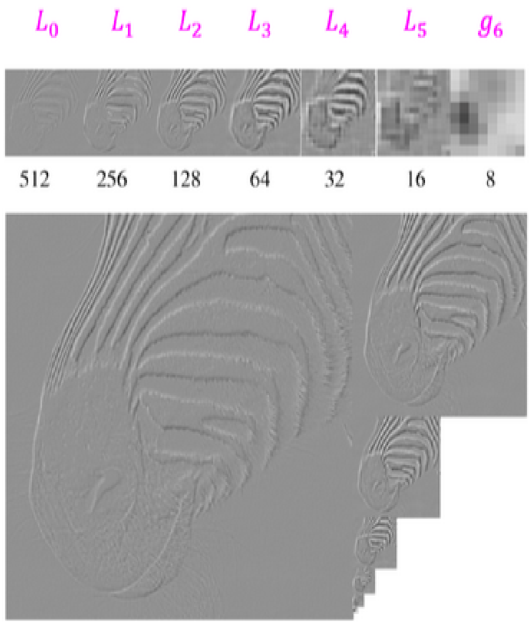
Laplacian Image Pyramid

Proposed by Burt and Adelson is a bandpass image decomposition derived from the Gaussian pyramid. Each level encodes information at a particular spatial frequency. The basic steps for constructing Laplacian pyramids are:

Construction (for image I)

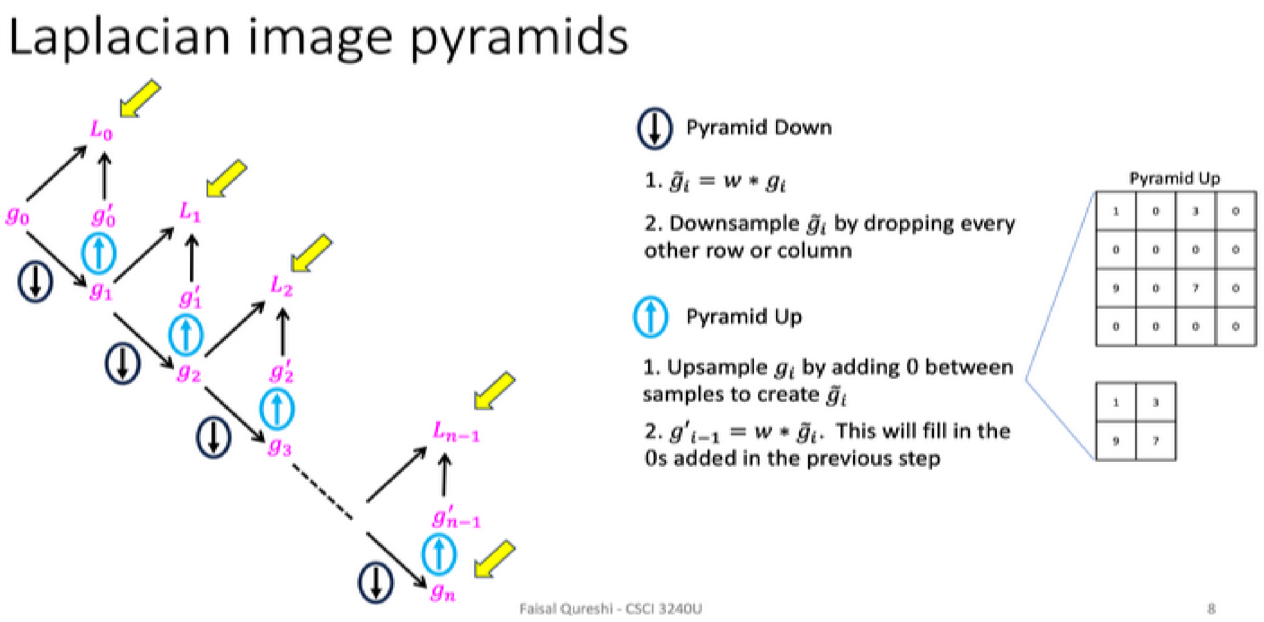
Let $g_0 = I$ and set $i = 0$

1. Convolve g_i with Gaussan kernel (low-pass filter w) and down-sample by half to construct g_{i+1}. Downsample by discarding every other row and column.
2. Upsample g_{i+1} by inserting 0s between each row and column and interpolating the missing values by convolving it with w and create g_i'
3. Compute $L_i = g_i - g_i'$
4. Set $i = i + 1$ and repeat till the desired levels are reached.



Courtesy D. Forsyth

Starting with image I, the following figure illustrates the construction of Laplacian pyramid.



Reconstructing the original image

It is possible to reconstruct the original image I from its Laplacian image pyramid consisting of n levels: $\{L_0, L_1, \cdots, L_{n-1}, g_n\}$.

Set $i = n$

1. Upsample g_i by inserting zeros between sample values and interpolate the missing values by convolving it with a filter w to get g'_{i-1}.
2. Set $g_{i-1} = L_{i-1} + g'_{i-1}$.
3. Set $i = i - 1$ and repeat till $i = 0$
4. Set $I = g_0$

Image Blending


Source: Torralba, Freeman, Isola

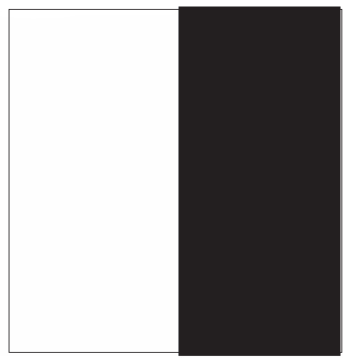
Image Blending

Source: Torralba, Freeman, Isola

Image Blending

I^A

I^B



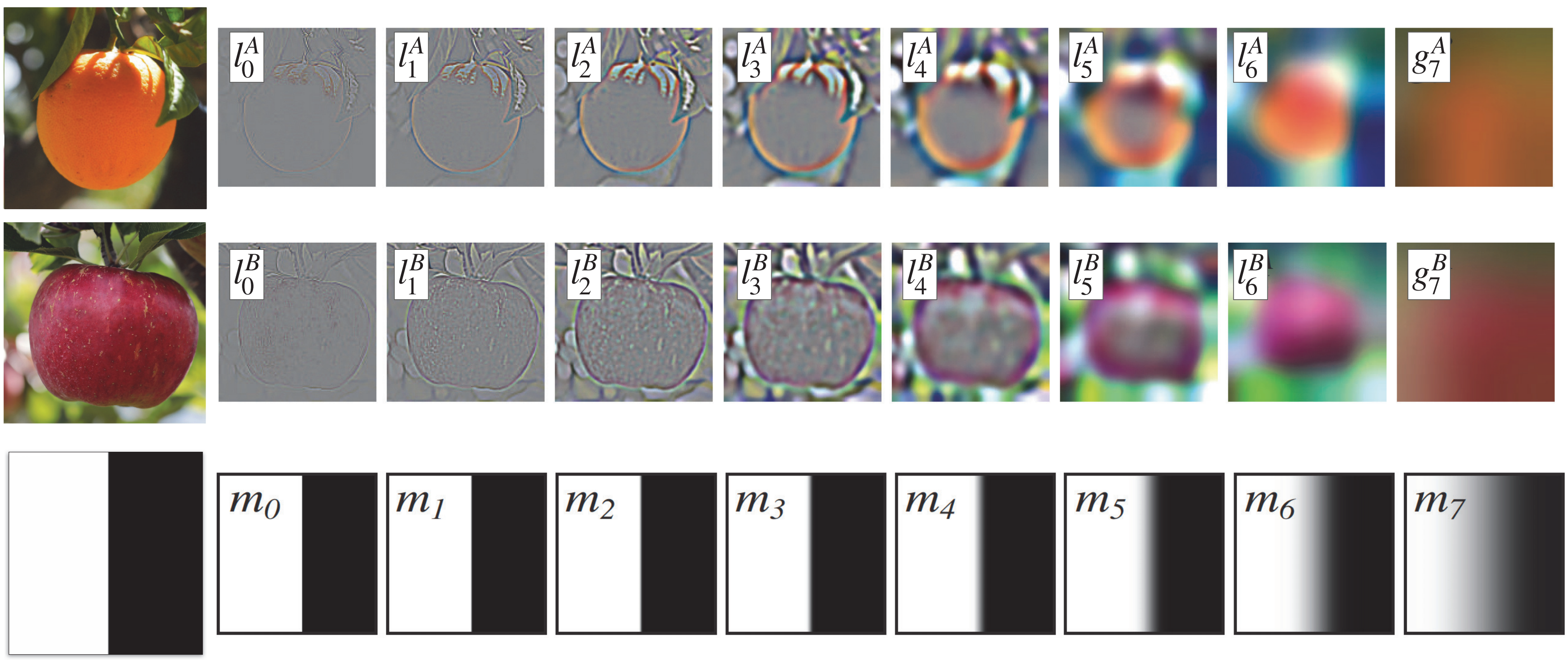
m

I

$$I = m * I^A + (1 - m) * I^B$$

Source: Torralba, Freeman, Isola

Image Blending with the Laplacian Pyramid



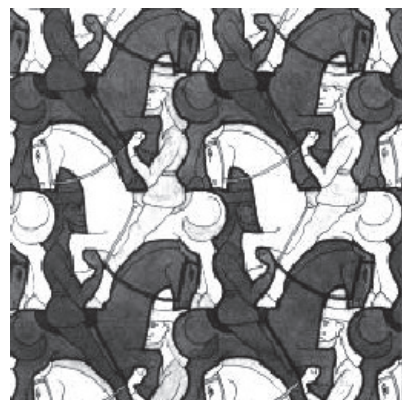
$$l_k = l_k^A * m_k + l_k^B * (1 - m_k)$$

Source: Torralba, Freeman, Isola

Image Blending with the Laplacian Pyramid

Source: Torralba, Freeman, Isola

+



=

Simple blend

With Laplacian pyr.

Source: A. Efros

Photo credit: Chris Cameron

Source: A. Efros