Lecture 5: Multi-scale pyramids

- PS1 due at midnight
- PS2 out tonight, due next Weds.
- Reading: Szeliski 3.5
- If you joined class late, make sure you have access to Gradescope and Canvas

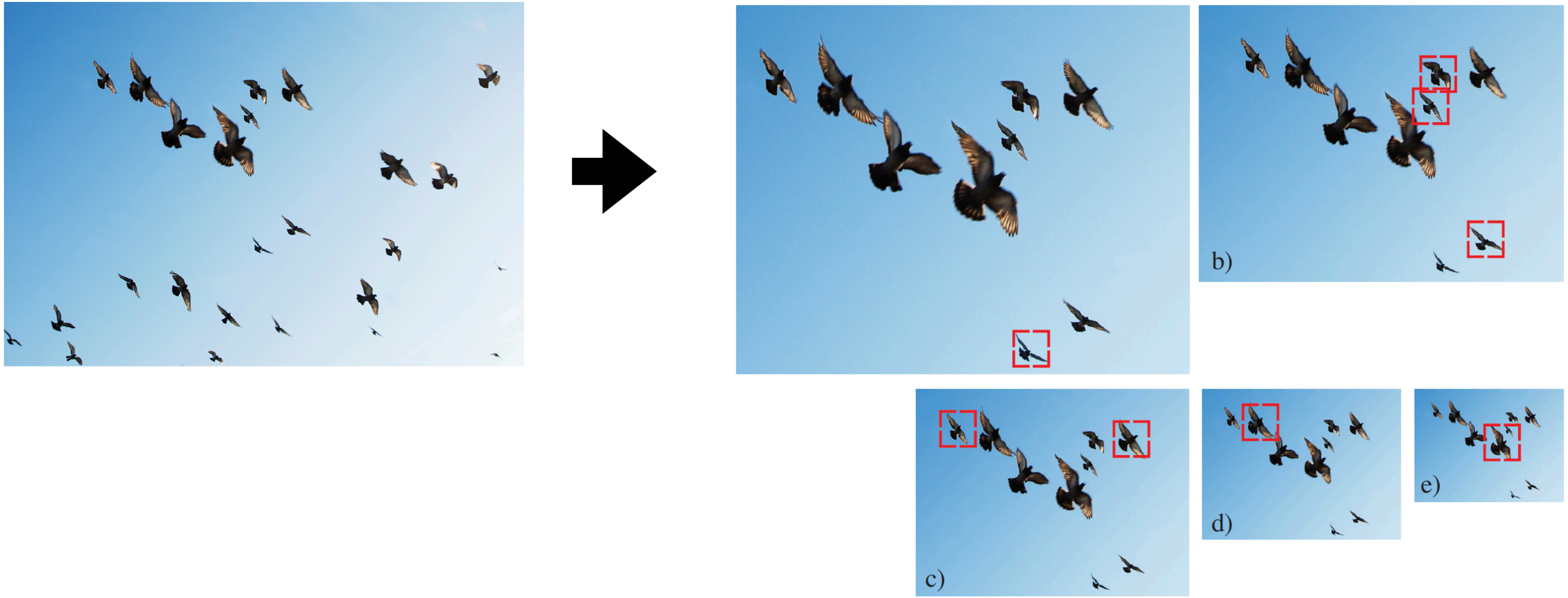
Today

- Image pyramids
- Sampling
- Texture

We want **scale** invariance.

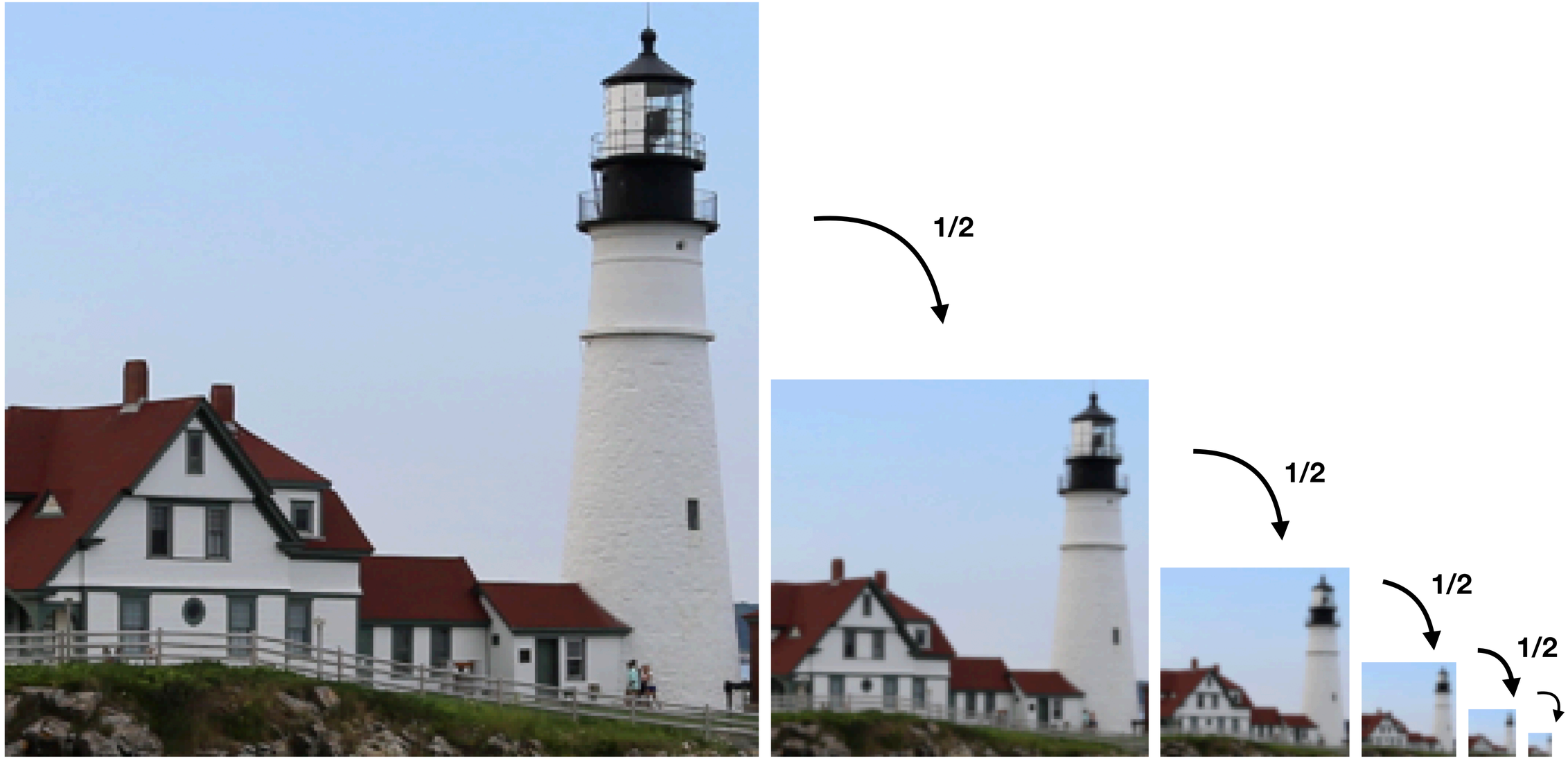
Source: Torralba, Freeman, Isola

Image pyramids



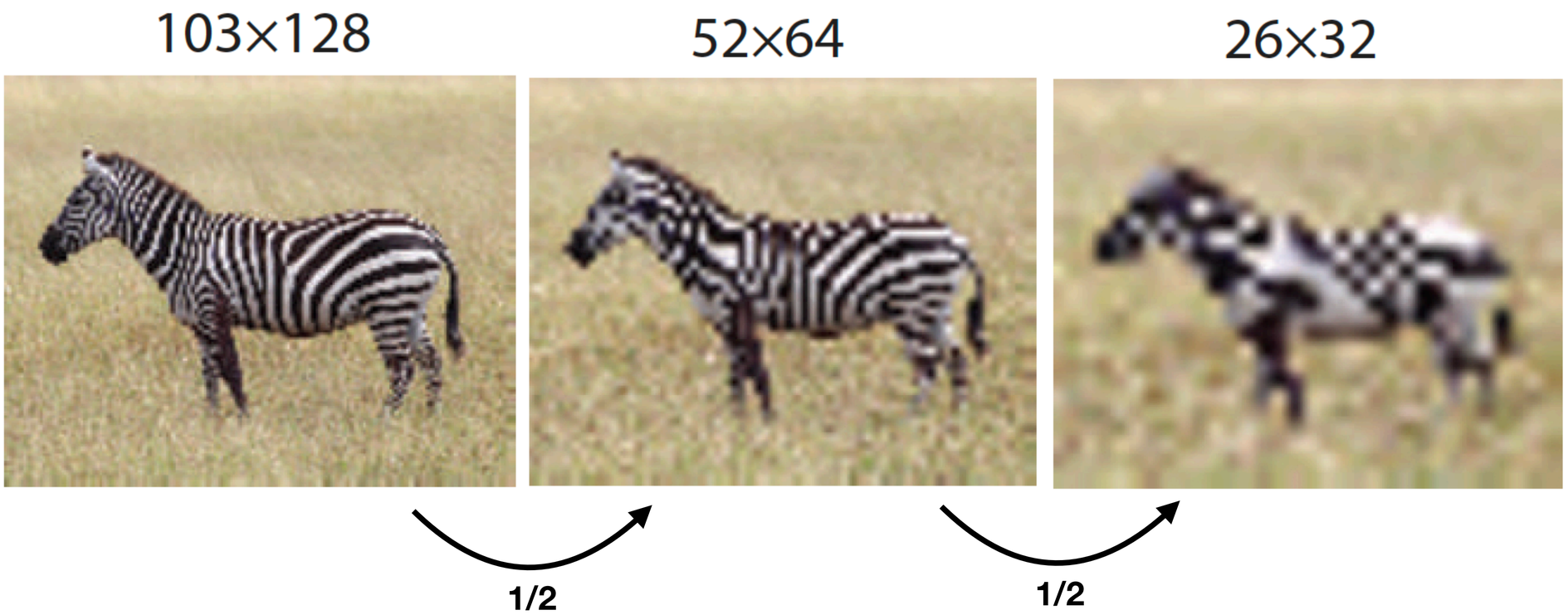
Source: Torralba, Freeman, Isola

Image pyramid



Source: Torralba, Freeman, Isola

Subsampling and aliasing

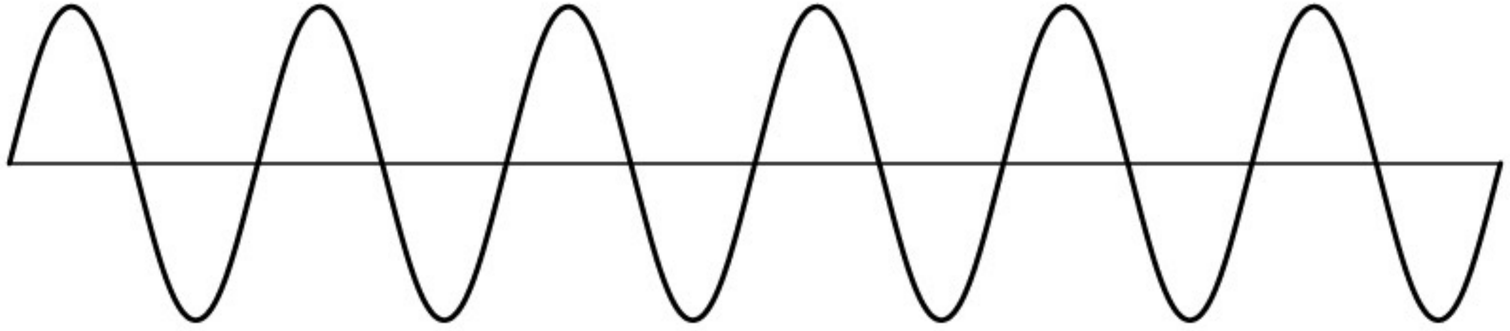


Idea #1: Throw away every other pixel.

Source: Torralba, Freeman, Isola

What’s happening?

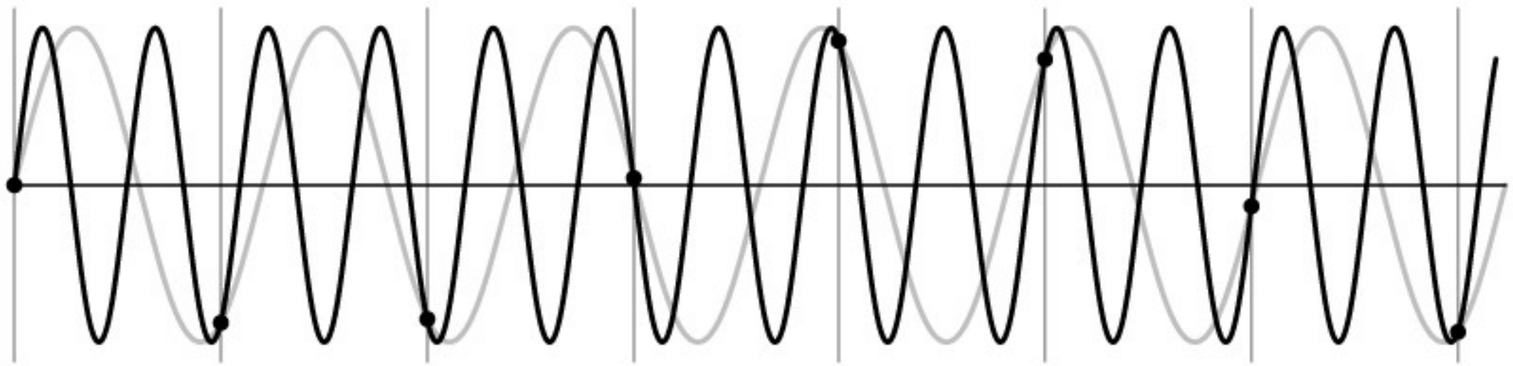
Consider a sinusoid:



Source: S. Marschner

Undersampling

- What if we “missed” things between the samples?
- As expected, information is lost
- Unexpectedly: indistinguishable from low-frequency sinusoid!
- Also indistinguishable from higher frequencies
- **Aliasing:** signals “traveling in disguise” as other frequencies



Source: S. Marschner

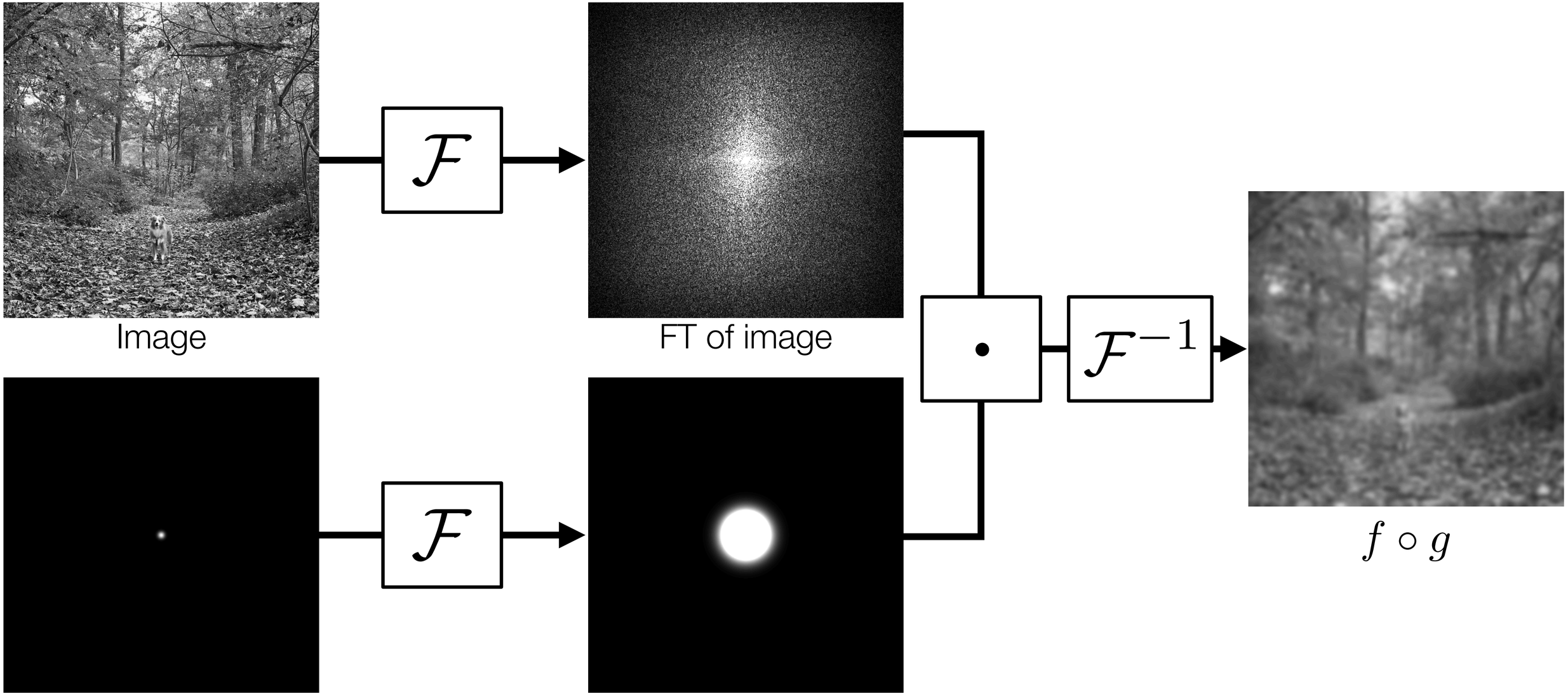
Removing aliasing

- Remove the high frequencies first!
- Blur the image before downsampling

Blur →

Source: Torralba, Freeman, Isola

Recall: blurring removes high frequencies



Image

FT of image

$f \circ g$

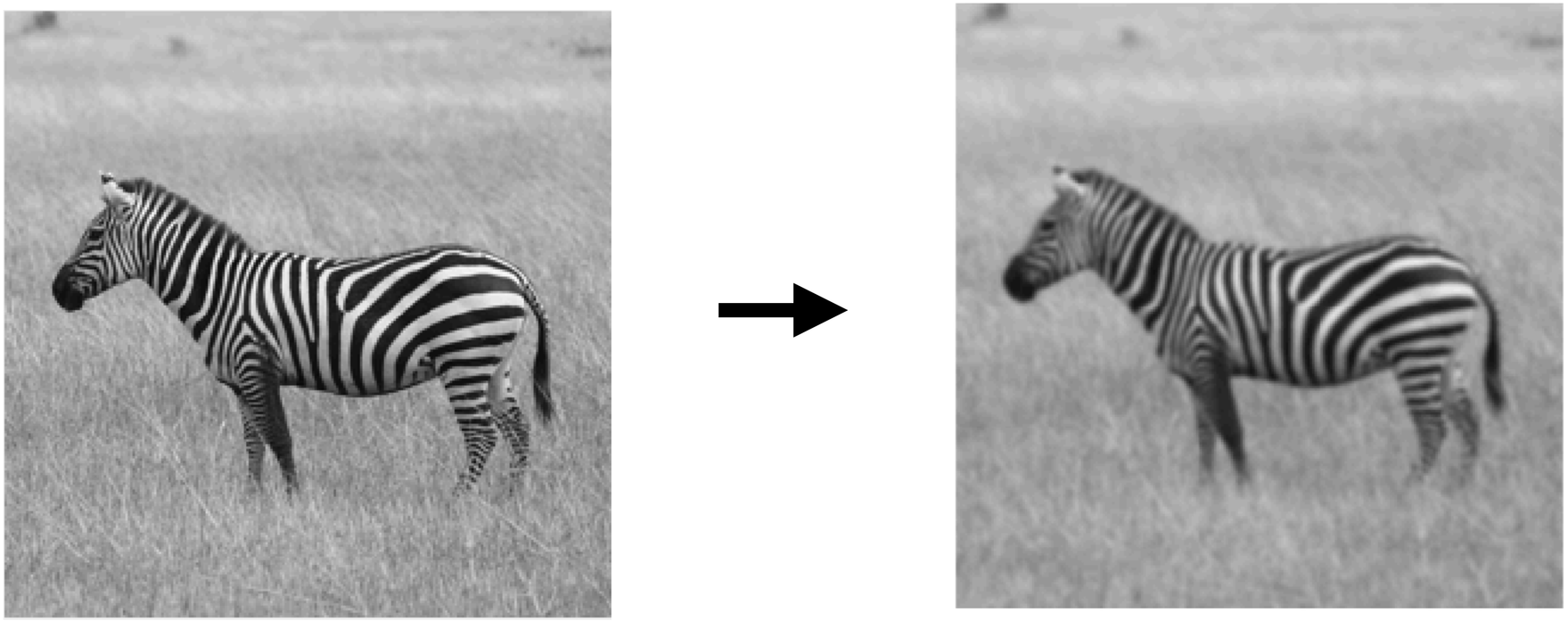
Gaussian $\sigma = 4$

FT of Gaussian is another Gaussian

Gaussian pyramid

For each level:

1. Blur input image with a Gaussian (or binomial) filter

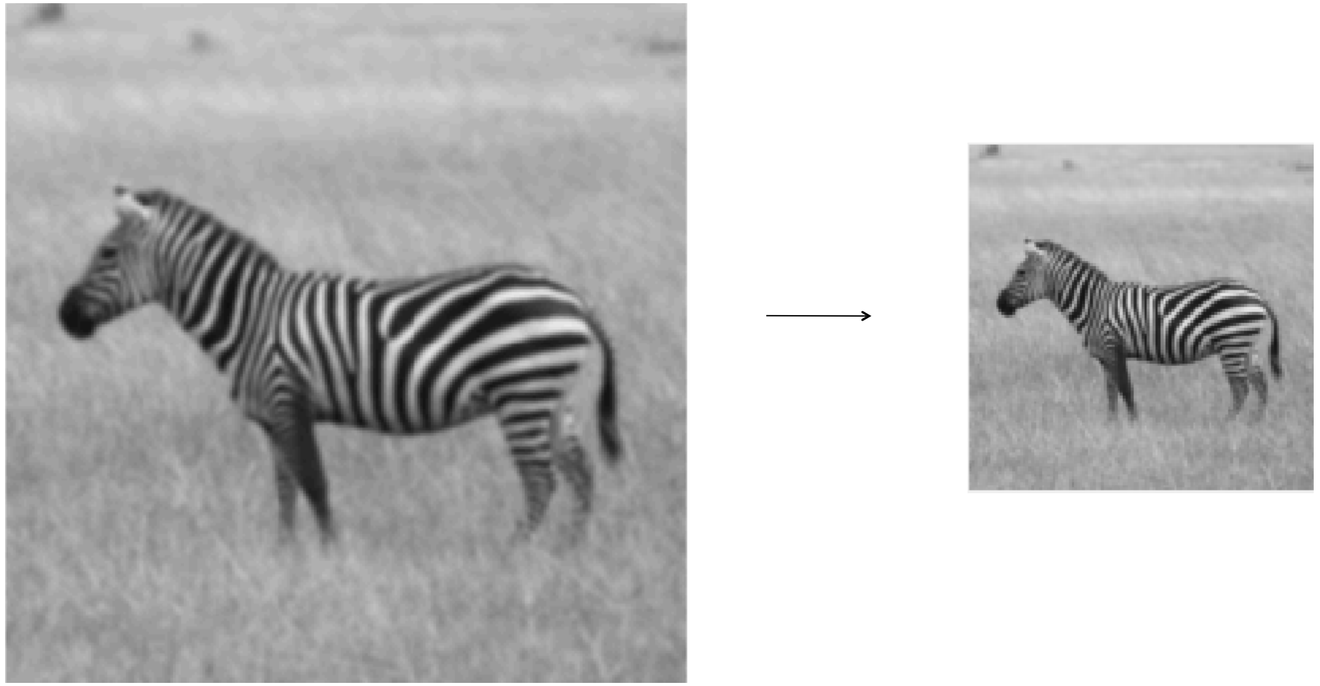


Source: Torralba, Freeman, Isola

Gaussian pyramid

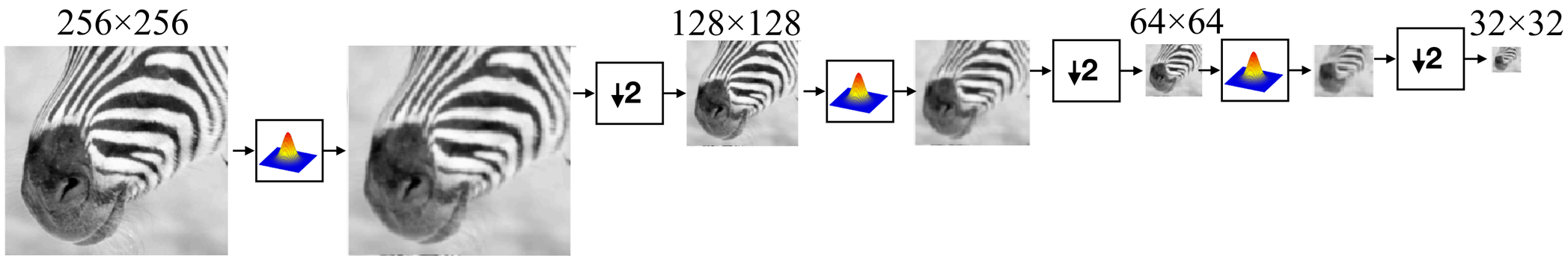
For each level:

1. Blur input image with a Gaussian (or binomial) filter
2. Downsample (throw away every other pixel)



Source: Torralba, Freeman, Isola

Gaussian pyramid



Source: Torralba, Freeman, Isola

Gaussian pyramid

512×512

(original image)

256×256

128×128

64×64

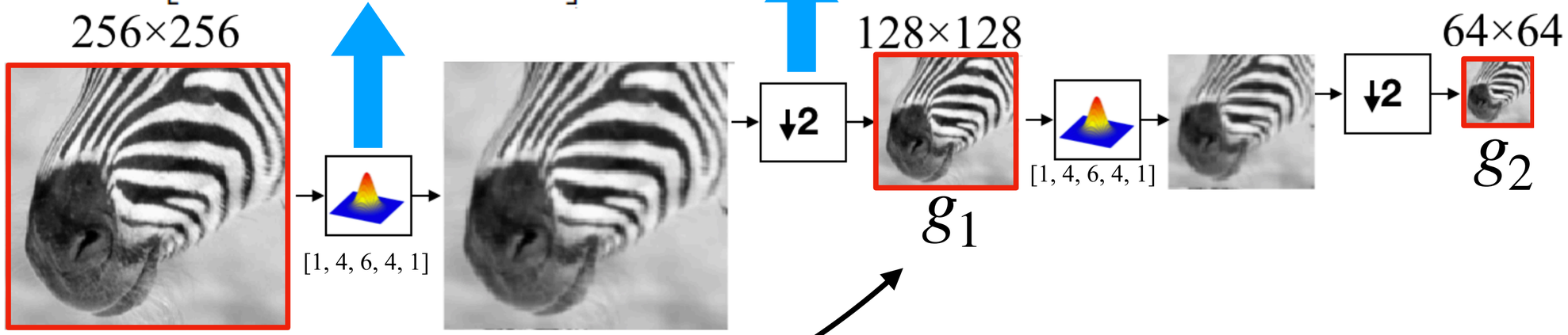
32×32

Source: Torralba, Freeman, Isola. Image from Forsyth & Ponce

Gaussian pyramid

$$\frac{1}{16}\begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$



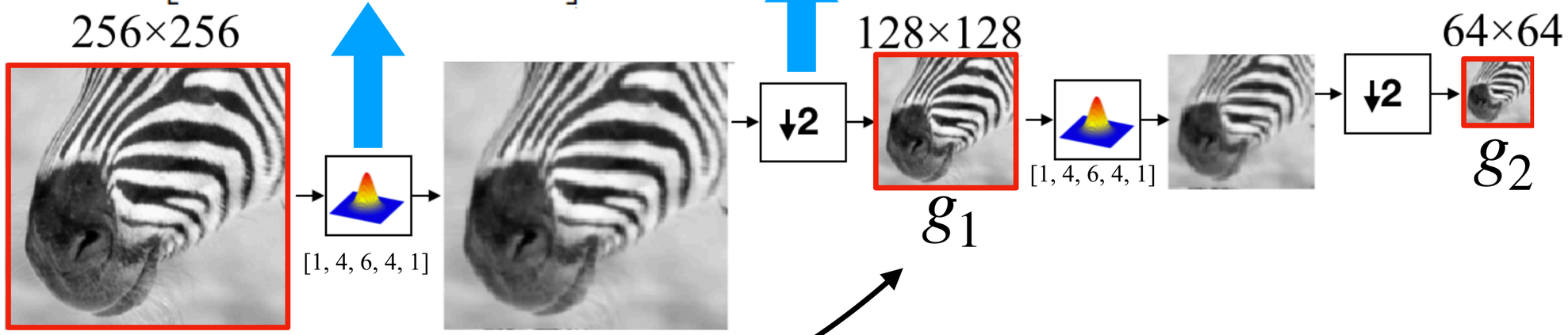
$$g_1 = G_0 g_0 \qquad G_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \frac{1}{16}\begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{bmatrix}$$

Source: Torralba, Freeman, Isola

Gaussian pyramid

$$\frac{1}{16}\begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{bmatrix}$$

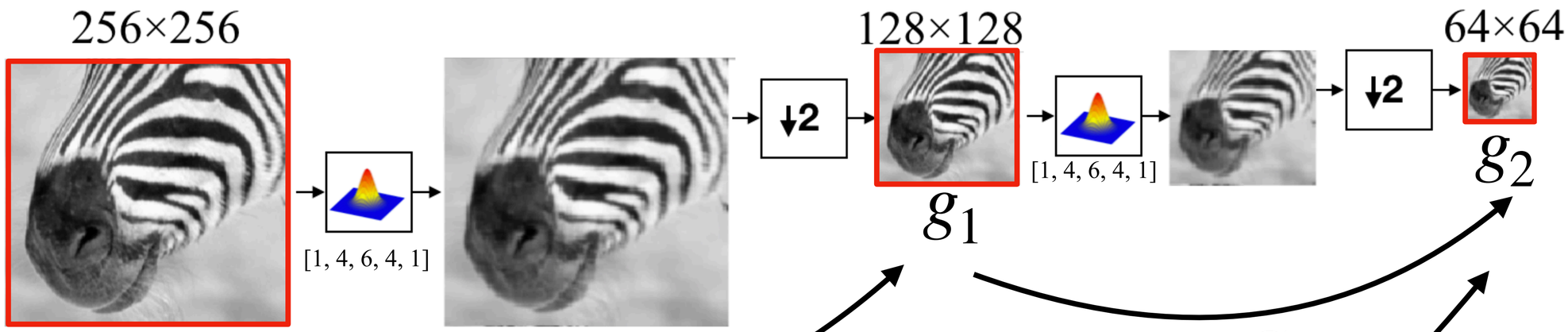
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$



$$g_1 = G_0 g_0 \qquad G_0 = \frac{1}{16}\begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \end{bmatrix}$$

Source: Torralba, Freeman, Isola

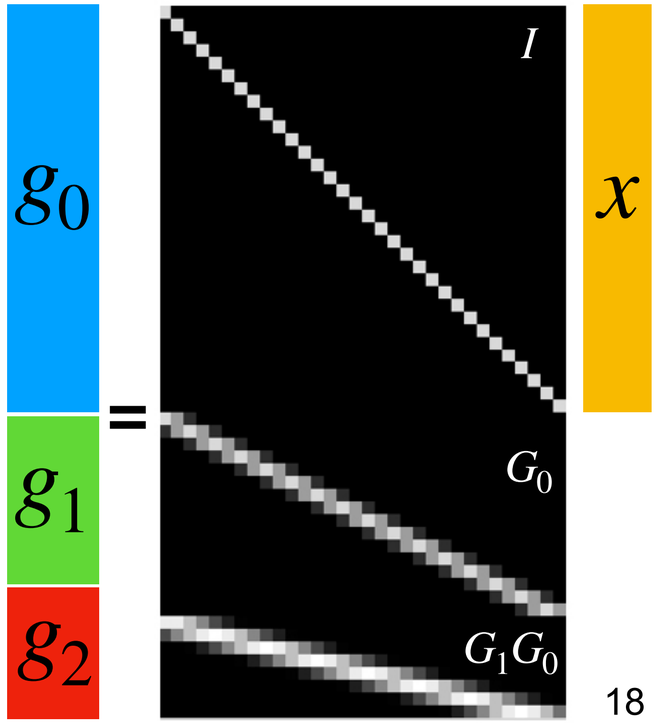
Gaussian pyramid



$$g_1 = G_0 g_0$$

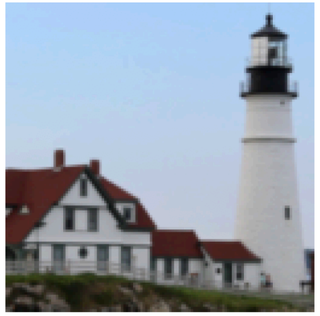
$$g_2 = G_1 g_1$$

$$g_2 = G_1 G_0 g_0$$



Source: Torralba, Freeman, Isola

Gaussian pyramid



$$g_k \longrightarrow \boxed{G_k} \longrightarrow g_{k+1}$$

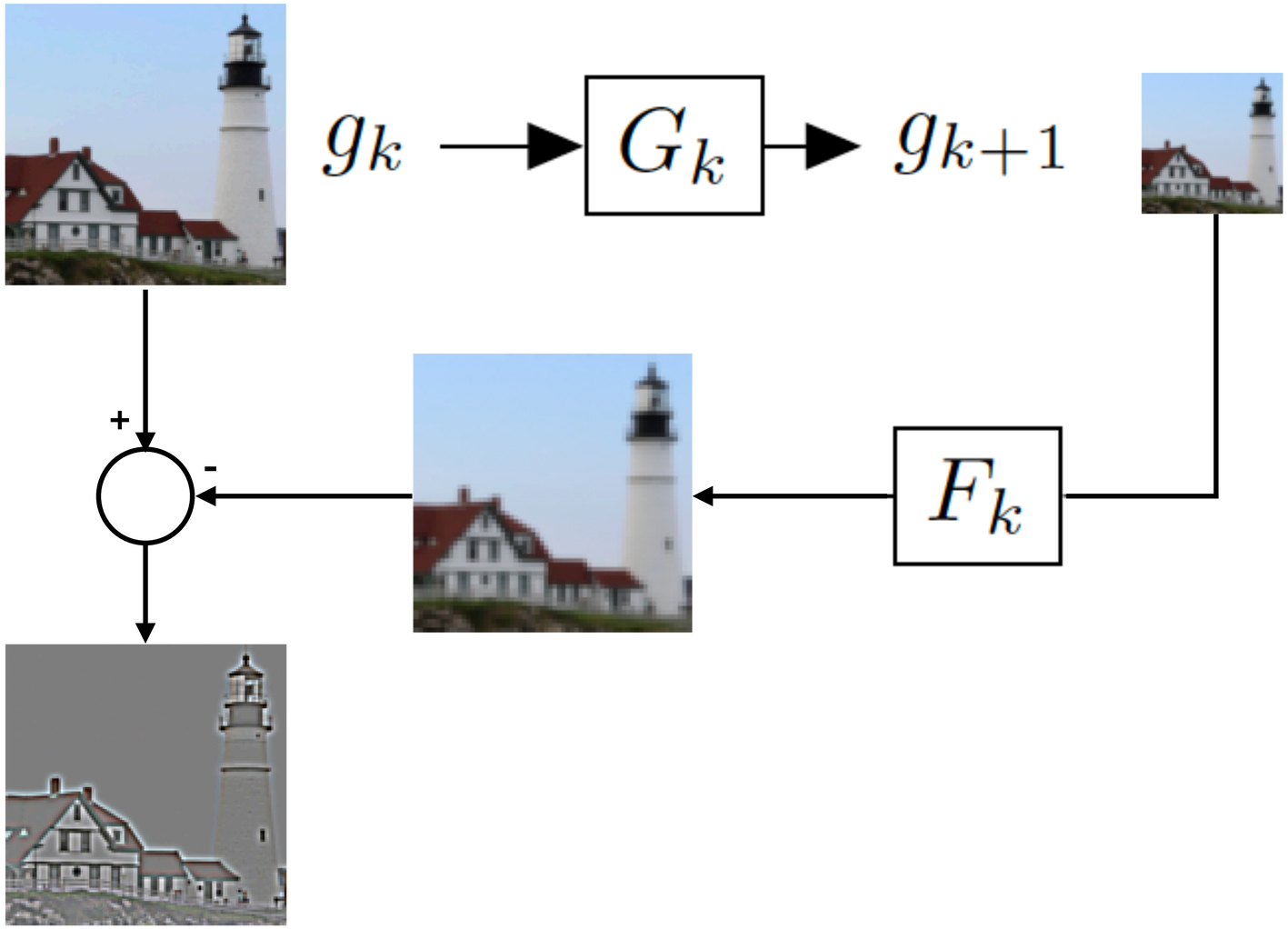
For each level

1. Blur input image with a Gaussian filter
2. Downsample image

Source: Torralba, Freeman, Isola

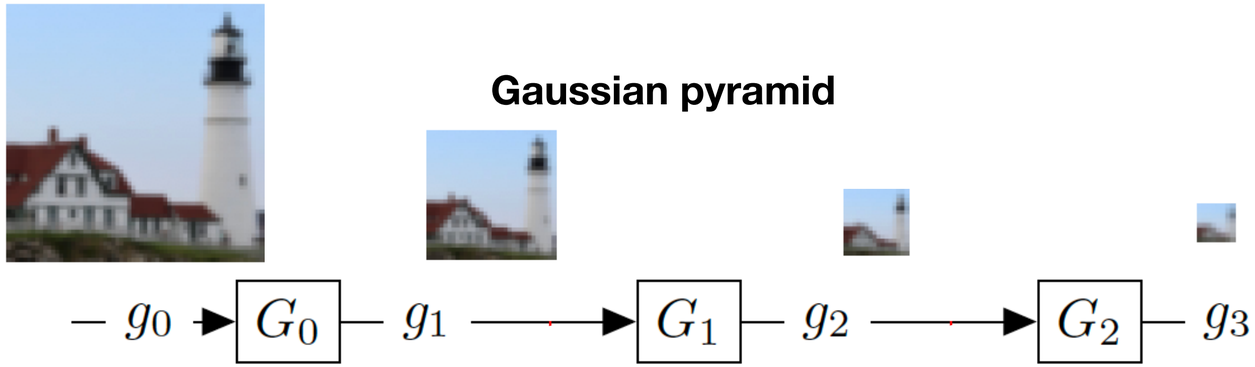
Laplacian Pyramid

Compute the difference between upsampled Gaussian pyramid level k+1 and Gaussian pyramid level k. Recall that this approximates the blurred Laplacian.



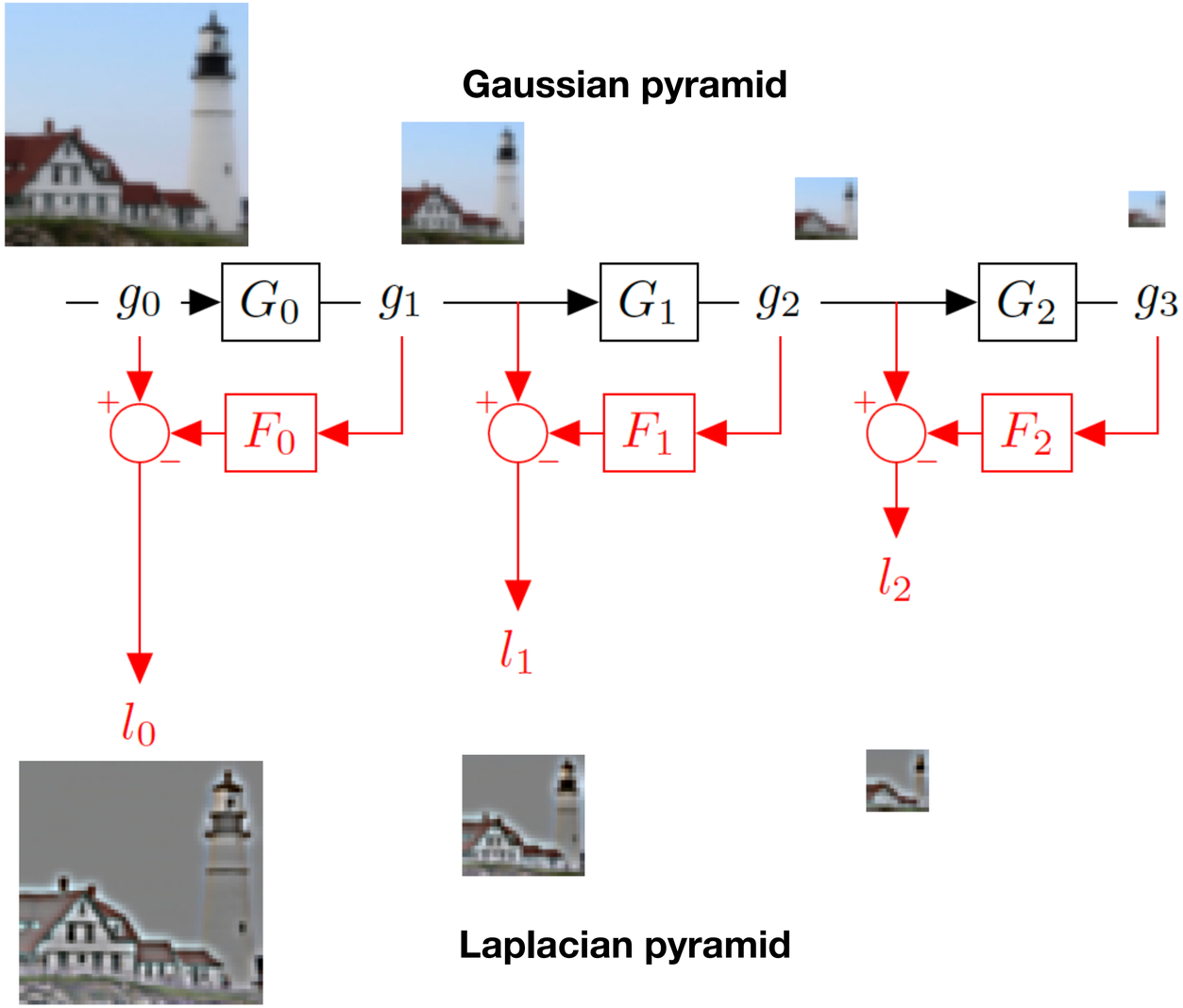
Source: Torralba, Freeman, Isola

Laplacian Pyramid



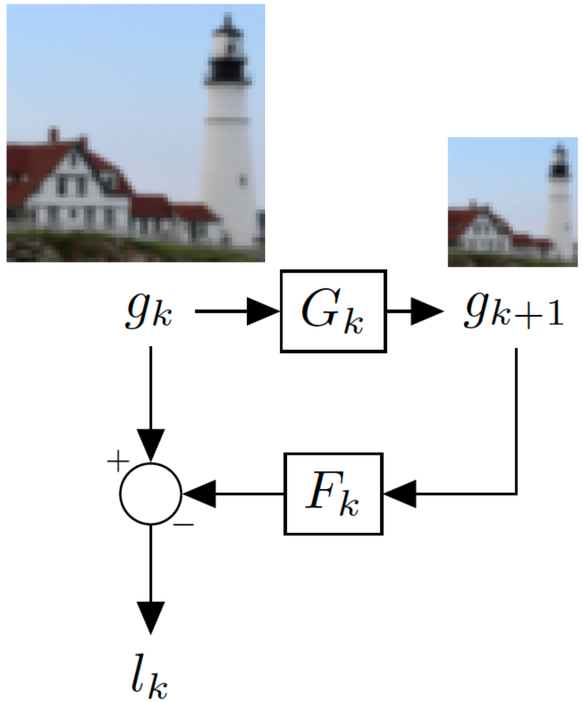
Source: Torralba, Freeman, Isola

Laplacian Pyramid



Source: Torralba, Freeman, Isola

Laplacian Pyramid



Blurring and downsampling:

$$G_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \frac{1}{16} \begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{bmatrix}$$

(Downsampling by 2) (blur)

Upsampling and blurring:

$$F_0 =$$

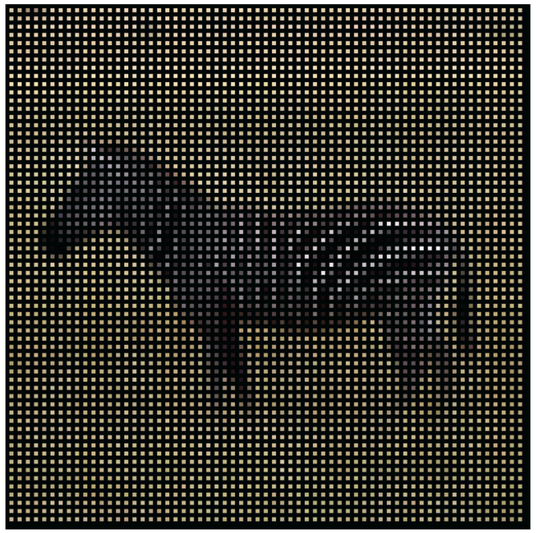
Source: Torralba, Freeman, Isola

Upsampling

64x64

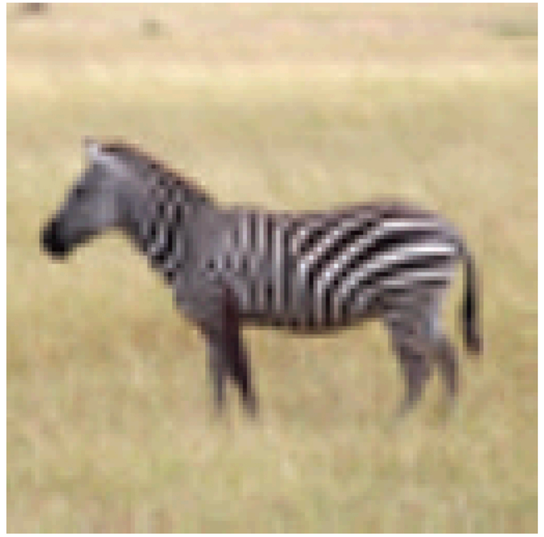
Insert zeros

128x128



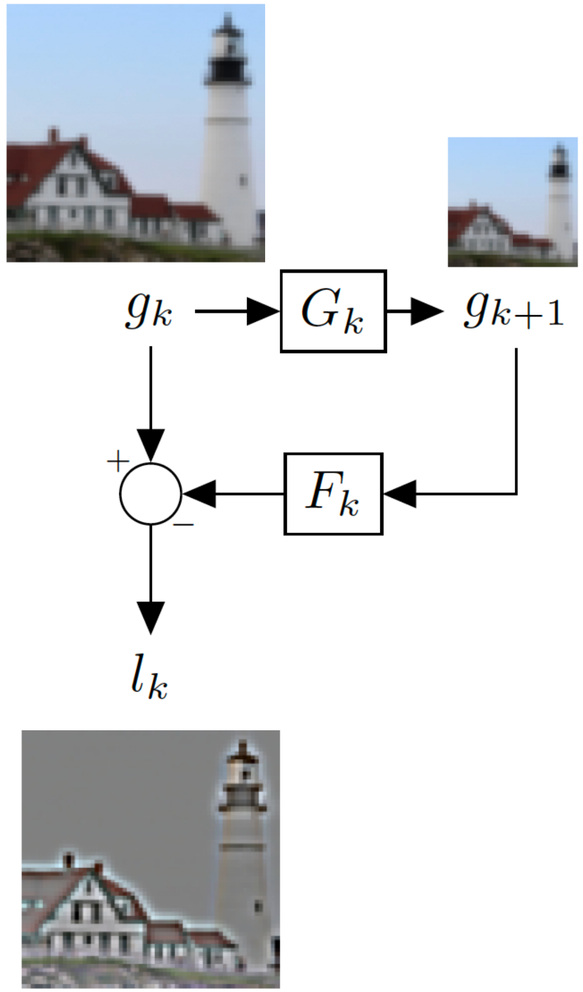
$$\bigcirc \begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.5 & 1 & 0.5 \\ 0.25 & 0.5 & 0.25 \end{bmatrix} =$$

128x128



Source: Torralba, Freeman, Isola

Laplacian Pyramid



Blurring and downsampling:

$$G_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \frac{1}{16} \begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{bmatrix}$$

(Downsampling by 2) (blur)

Upsampling and blurring:

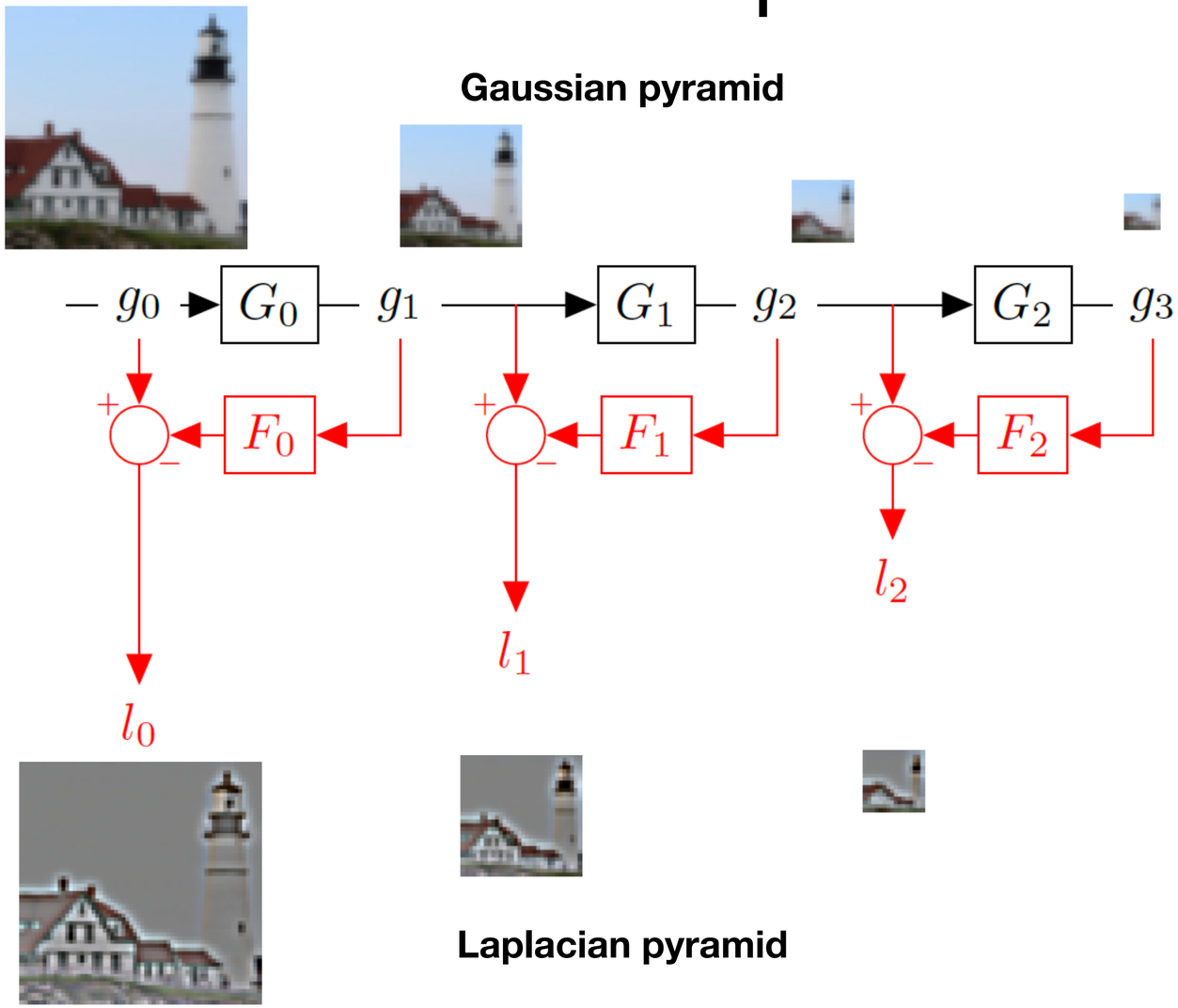
$$F_0 = \frac{1}{8} \begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

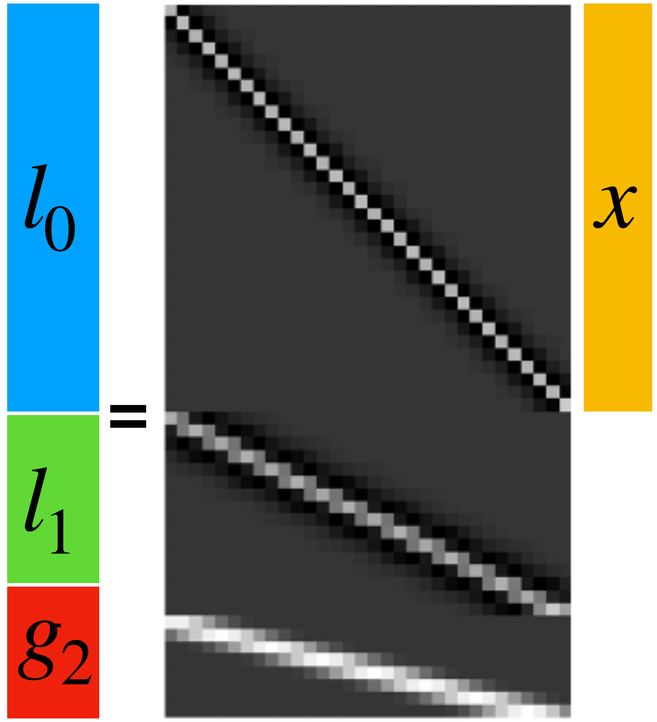
(blur) (Upsampling by 2)

$$\boxed{l_0 = (I_0 - F_0 G_0) g_0}$$

Source: Torralba, Freeman, Isola

Laplacian Pyramid





Source: Torralba, Freeman, Isola

Laplacian Pyramid

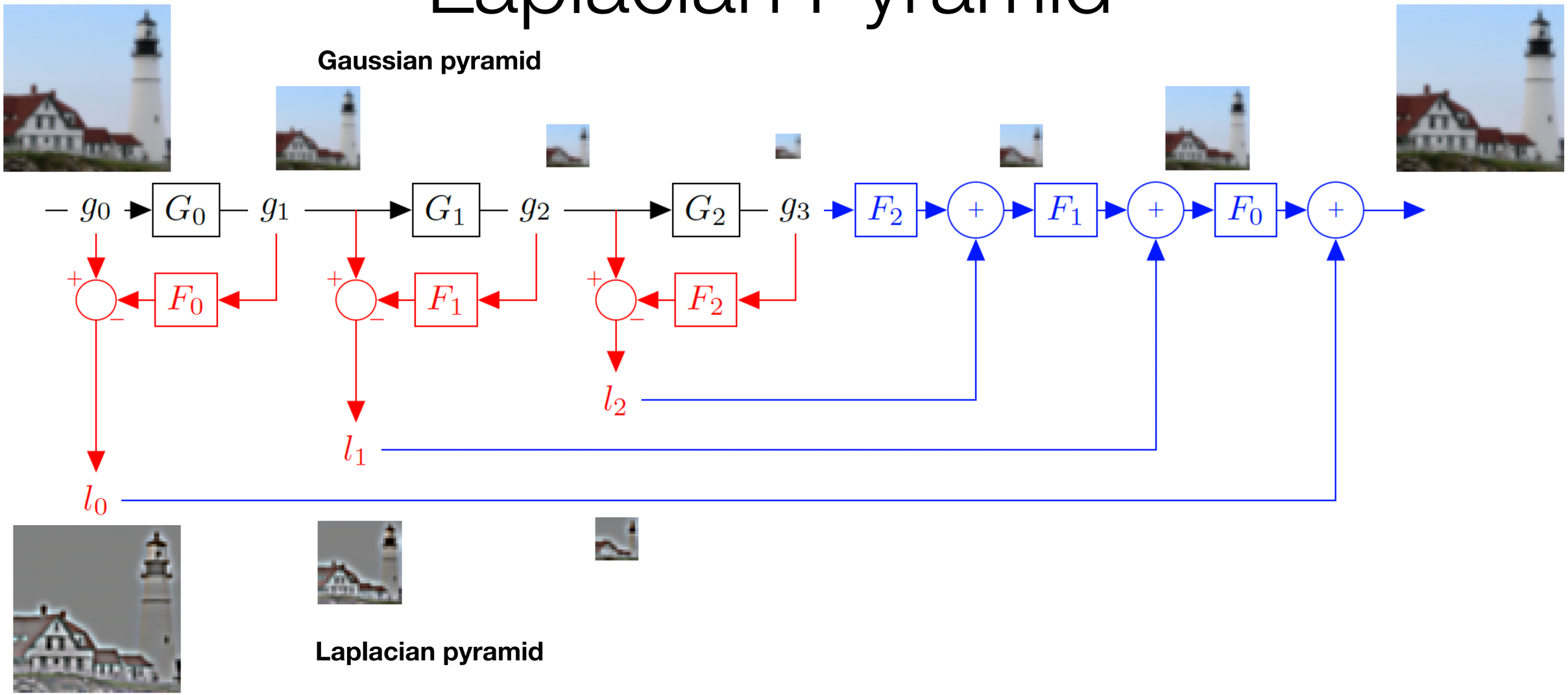
Gaussian residual

Laplacian pyramid

Can we invert the Laplacian Pyramid?

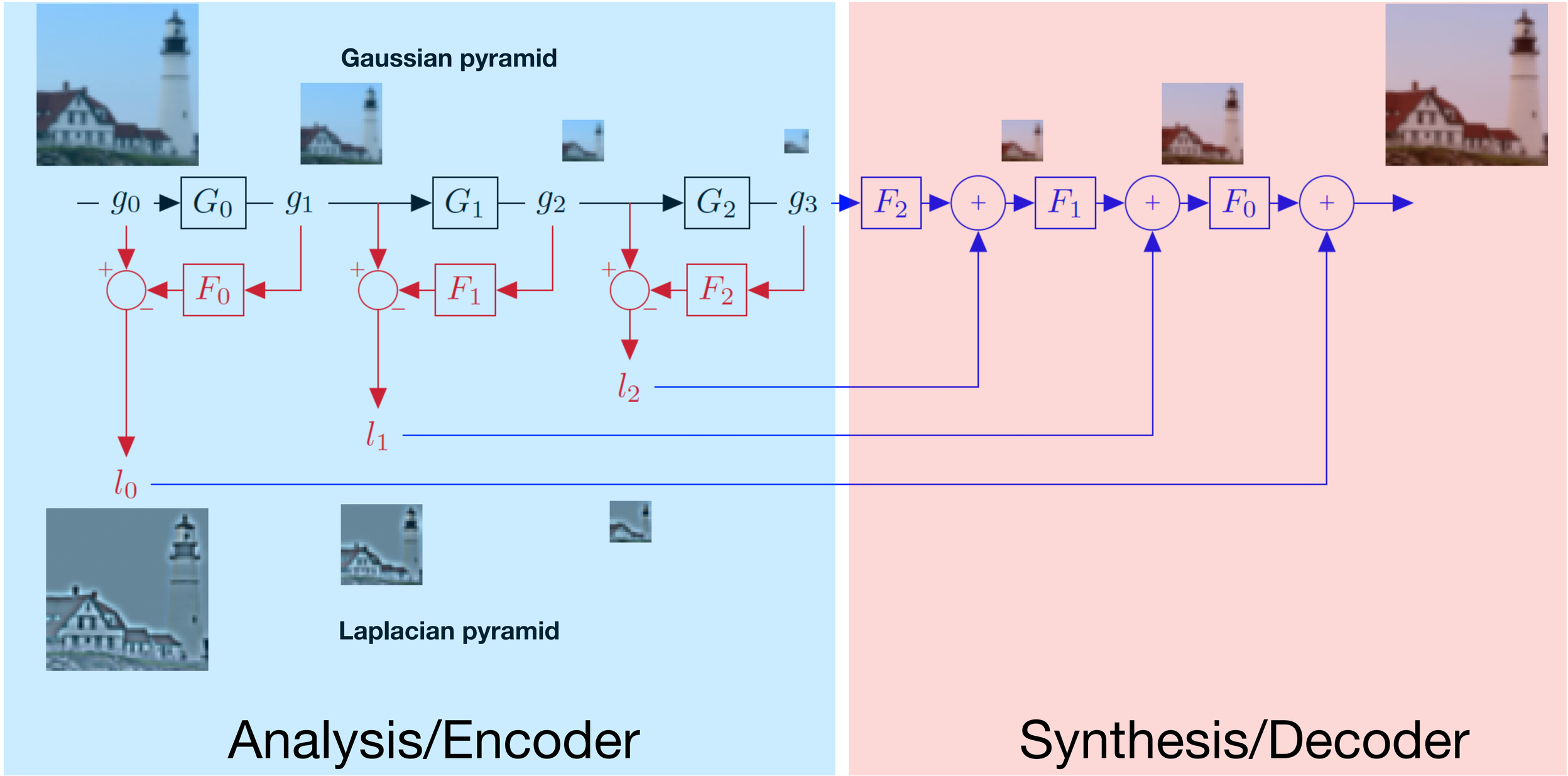
Source: Torralba, Freeman, Isola

Laplacian Pyramid



Source: Torralba, Freeman, Isola

Laplacian Pyramid



Source: Torralba, Freeman, Isola

Laplacian pyramid applications

- Texture synthesis
- Image compression
- Noise removal
- Computing image “keypoints”

Source: Torralba, Freeman, Isola

Image Blending

Source: Torralba, Freeman, Isola

Image Blending

Source: Torralba, Freeman, Isola

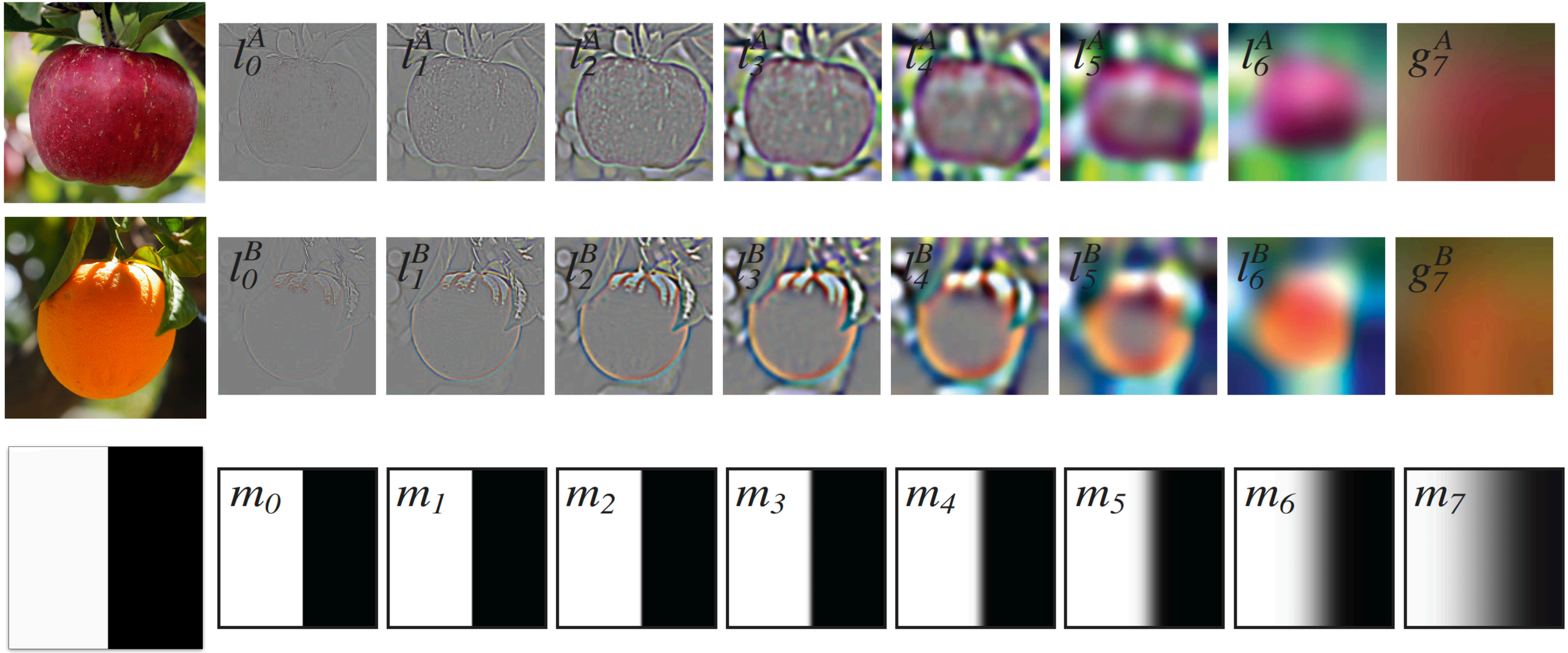
Image Blending

I^A I^B m I

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

Source: Torralba, Freeman, Isola

Image Blending with the Laplacian Pyramid



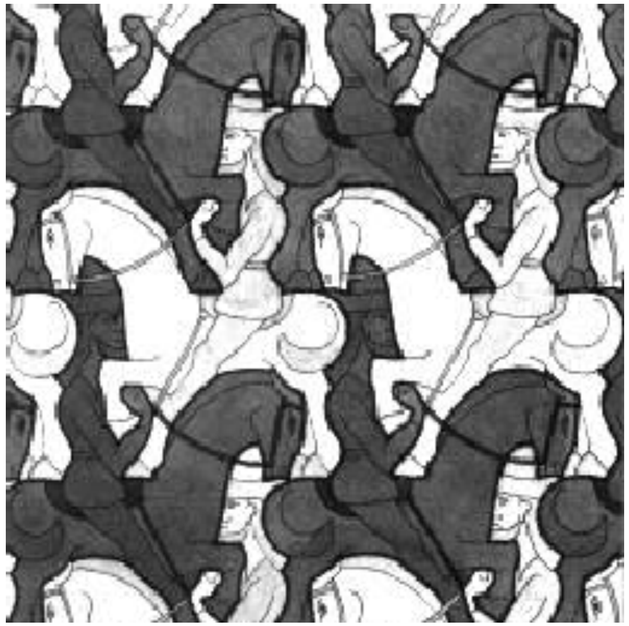
$$l_k = l_k^A * m_k + l_i^B * (1 - m_k)$$

Source: Torralba, Freeman, Isola

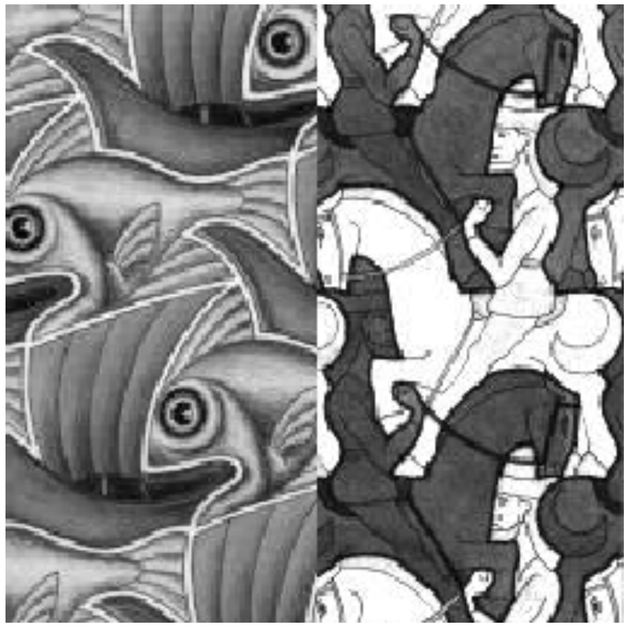
Image Blending with the Laplacian Pyramid

Source: Torralba, Freeman, Isola

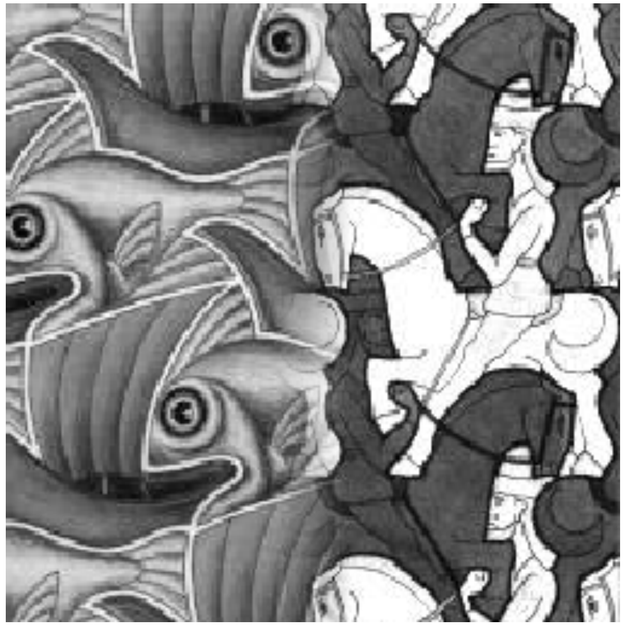
+



=



Simple blend



With Laplacian pyr.

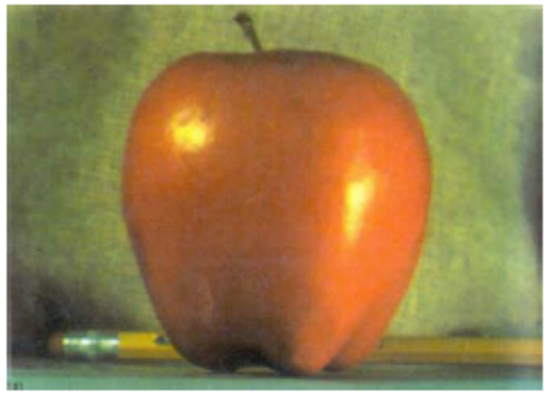
Source: A. Efros

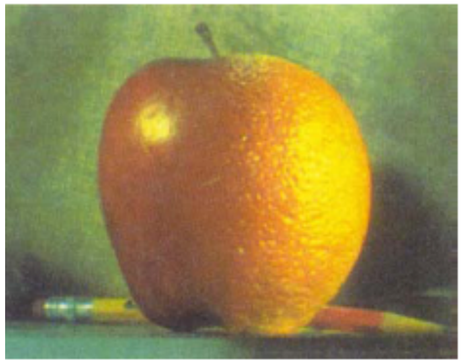
Photo credit: Chris Cameron

Source: A. Efros

Image Blending (PS2 problem)

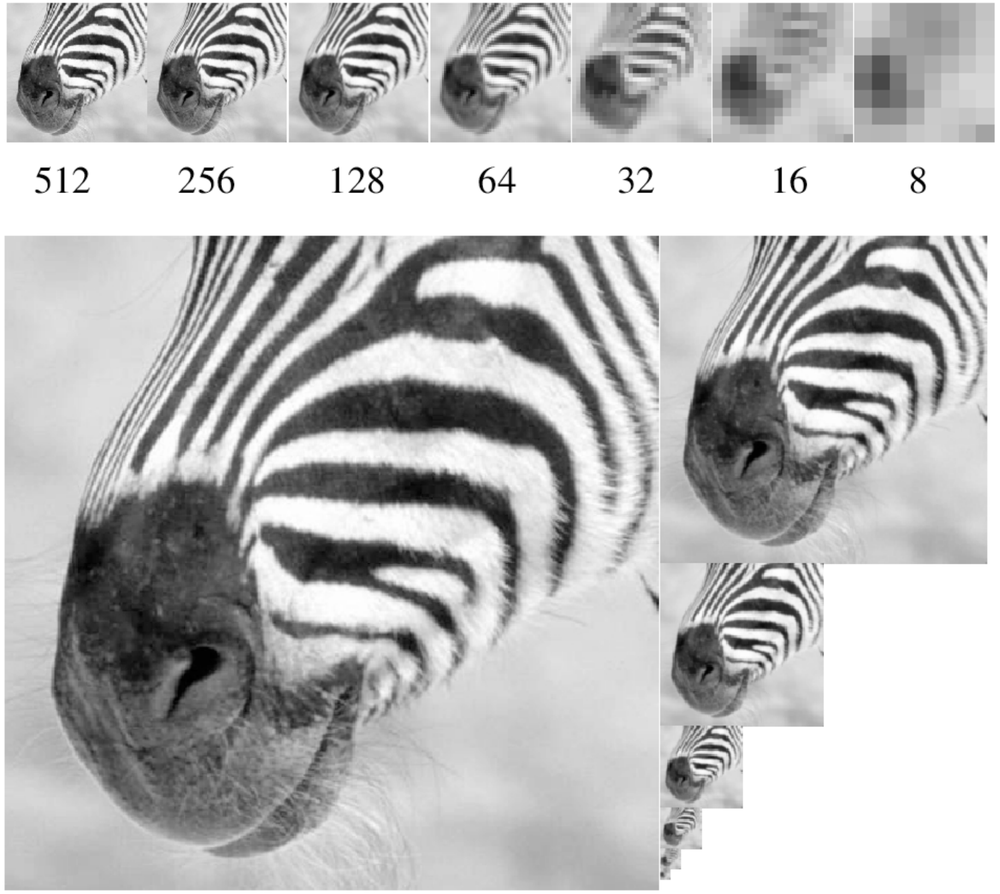
- Build Laplacian pyramid for both images: L_A, L_B
- Build Gaussian pyramid for mask: G
- Build a combined Laplacian pyramid
- Collapse L to obtain the blended image



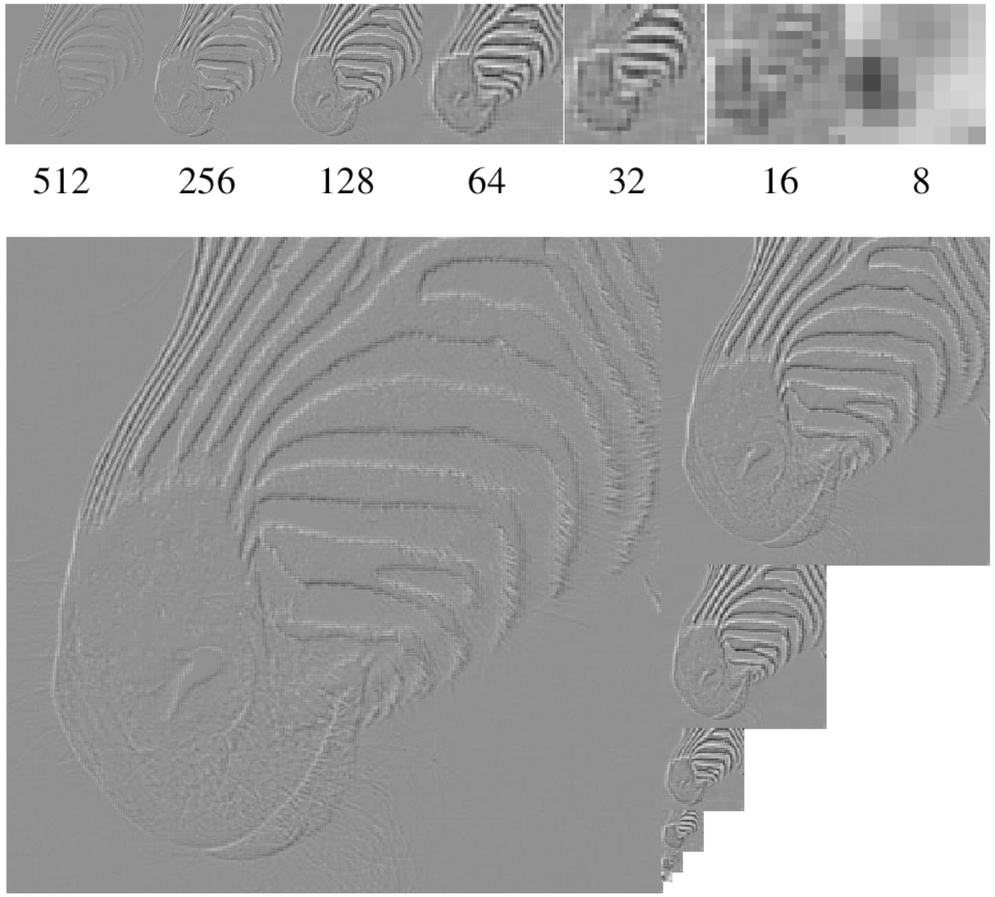


Source: Torralba, Freeman, Isola

Image pyramids



Gaussian Pyramid



Laplacian Pyramid

And many more: steerable filters, wavelets, … (and later) convolutional networks!

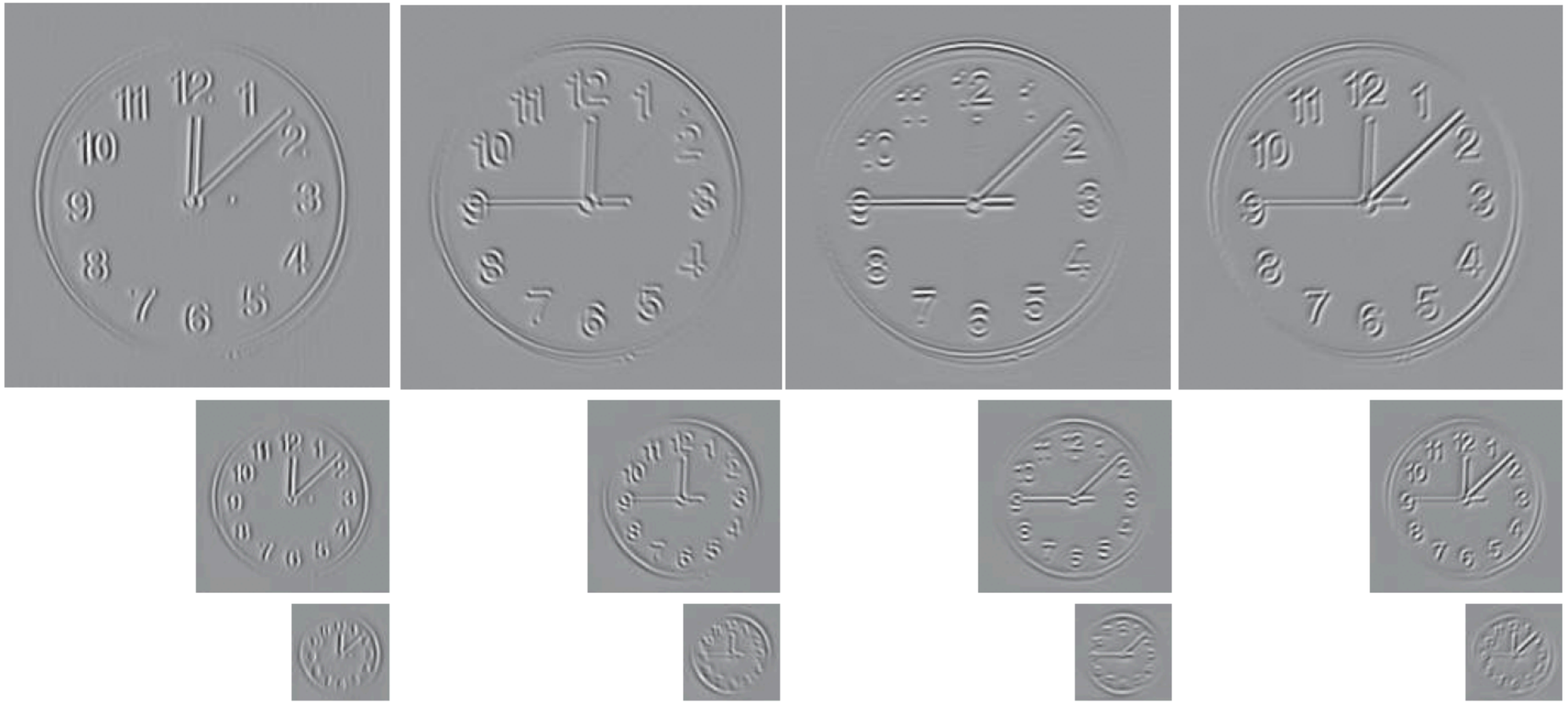
Source: Torralba, Freeman, Isola

Orientations

Source: Torralba, Freeman, Isola

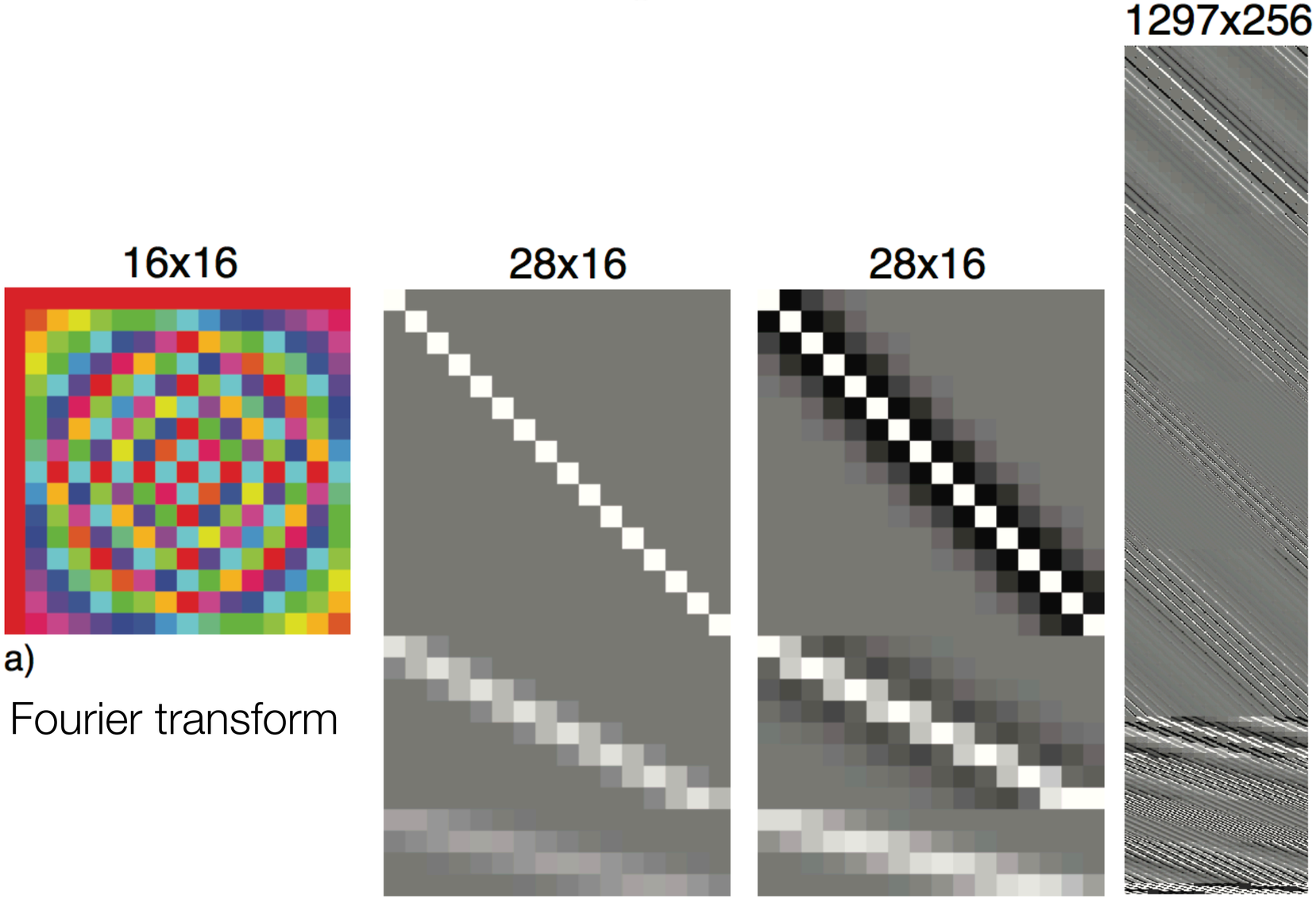
Steerable Pyramid

Oriented gradient



Source: Torralba, Freeman, Isola

Linear Image Transforms



Fourier transform

Gaussian pyr. Laplacian pyr. Steerable pyr.

Source: Torralba, Freeman, Isola

Source: Torralba, Freeman, Isola

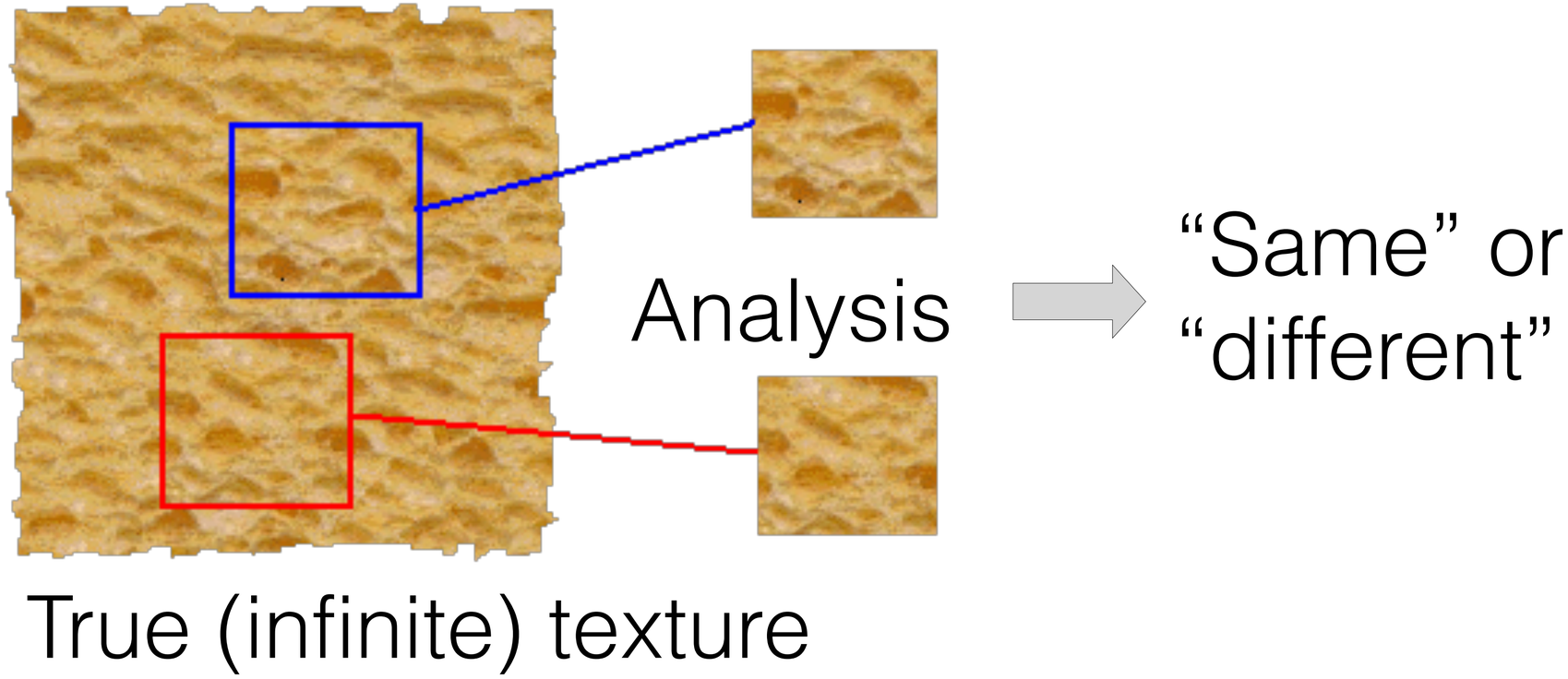
Texture

Stationary

Stochastic

Source: Torralba, Freeman, Isola

Texture analysis



What we'd like: are they made of the same "stuff". Are these textures similar?

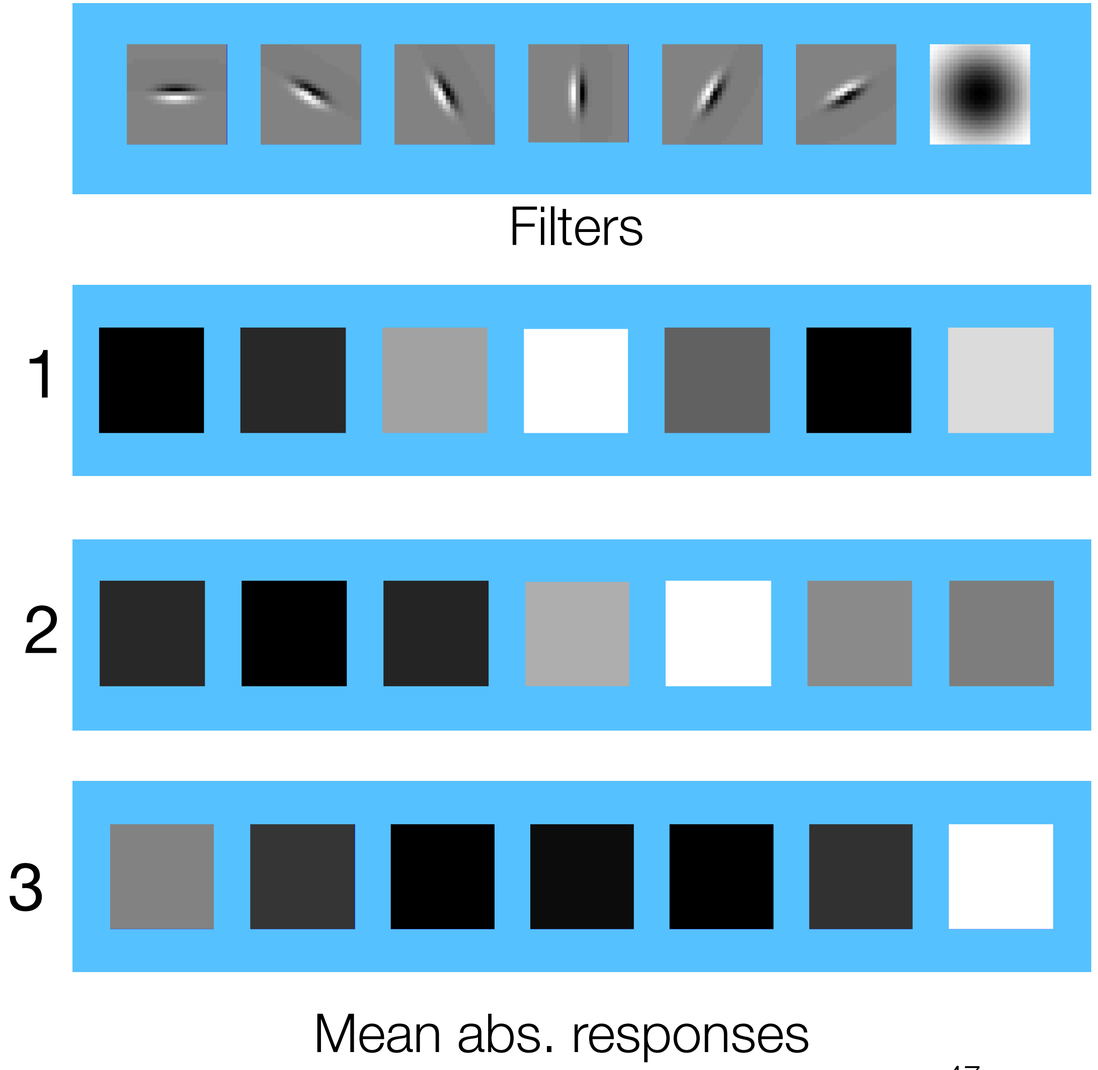
Source: A. Efros

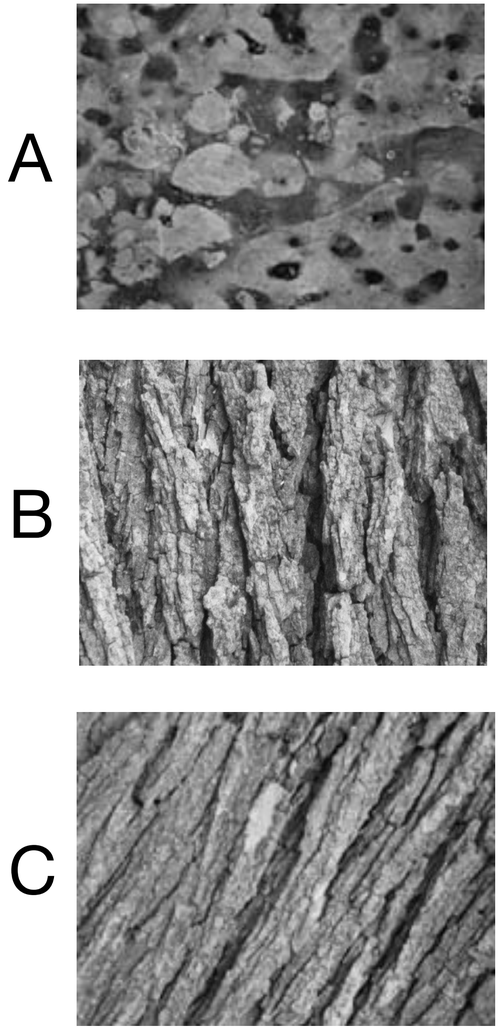
How can we represent texture in natural images?

<u>Idea #1</u>: Record simple statistics (e.g., mean, std.) of absolute filter responses

Source: A. Efros

Can you match the texture to the response?





Source: A. Efros

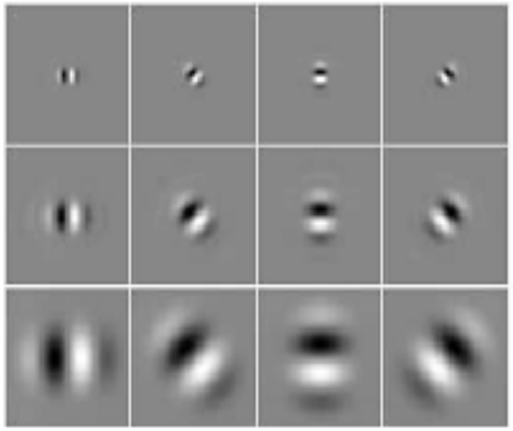
How can we represent texture?

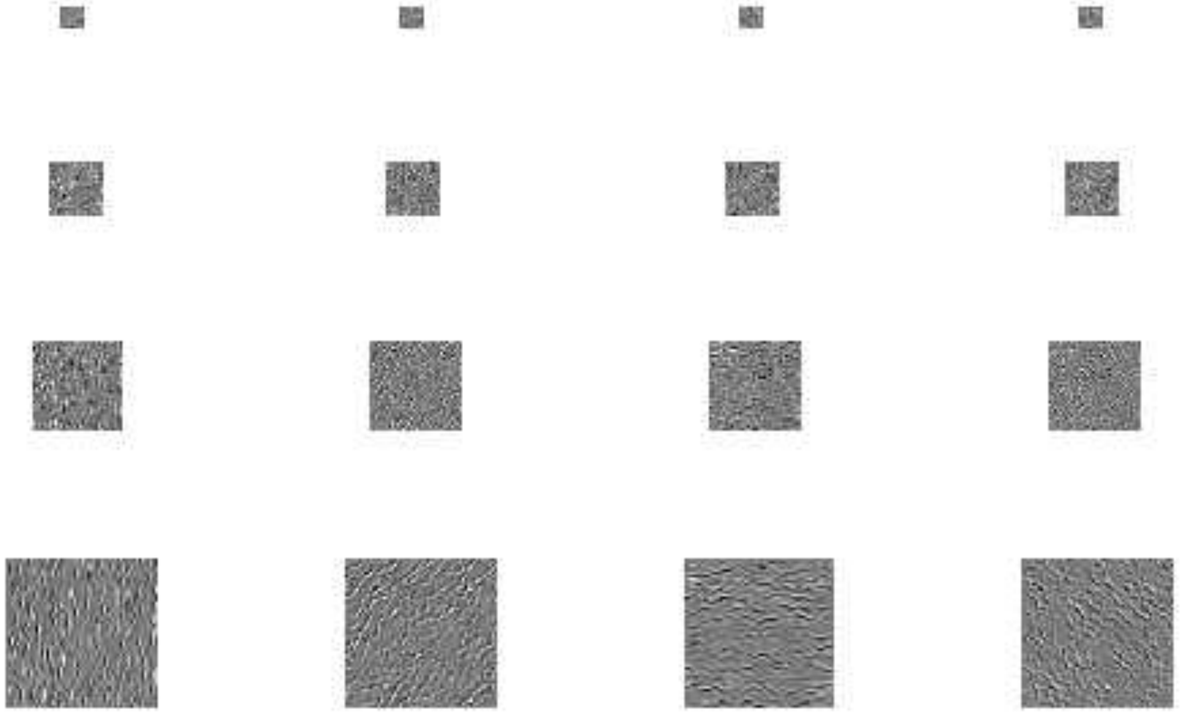
- Generalize this to “orientation histogram”
- <u>Idea #2</u>: Histograms of filter responses
  - One histogram per filter

Source: A. Efros

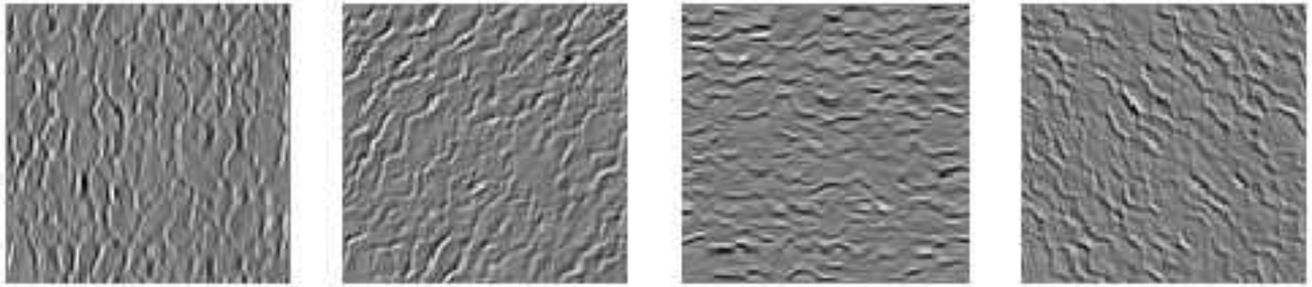
Steerable filter decomposition

Filter bank



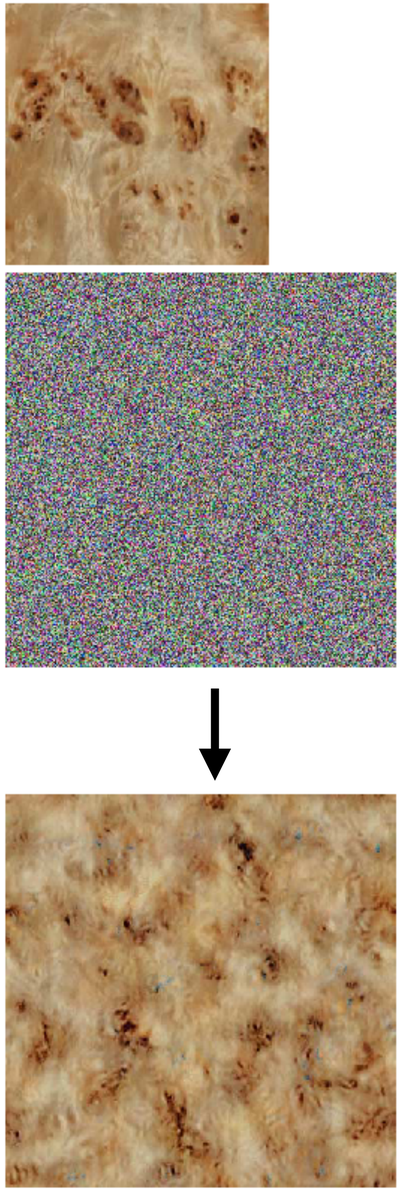


Input image



Source: A. Efros

Texture synthesis



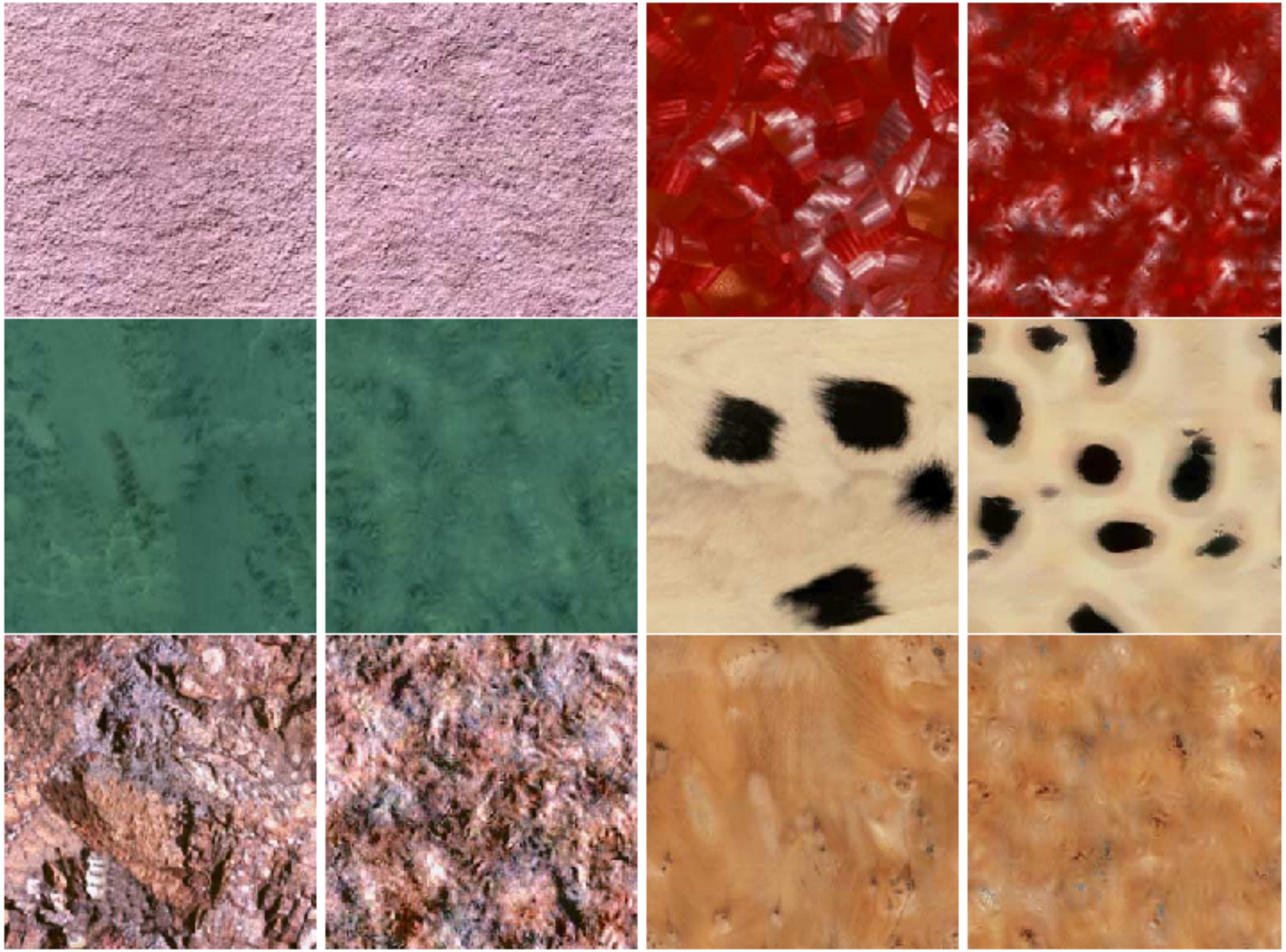
Start with a noise image as output.

Iterative algorithm [Heeger & Bergen, 95]:

- Match pixel histogram of output image to input
- Decompose input/output images using a Steerable Pyramid
- Match histograms of input and output pyramids
- Reconstruct image and repeat

Later in the class we'll see a simpler optimization method on neural nets [Gatys et al. 2015]

Source: A. Efros



Source: A. Efros

Source: A. Efros

Failure cases

Figure 7: (Left pair) Inhomogeneous input texture produces blotchy synthetic texture. (Right pair) Homogenous input.

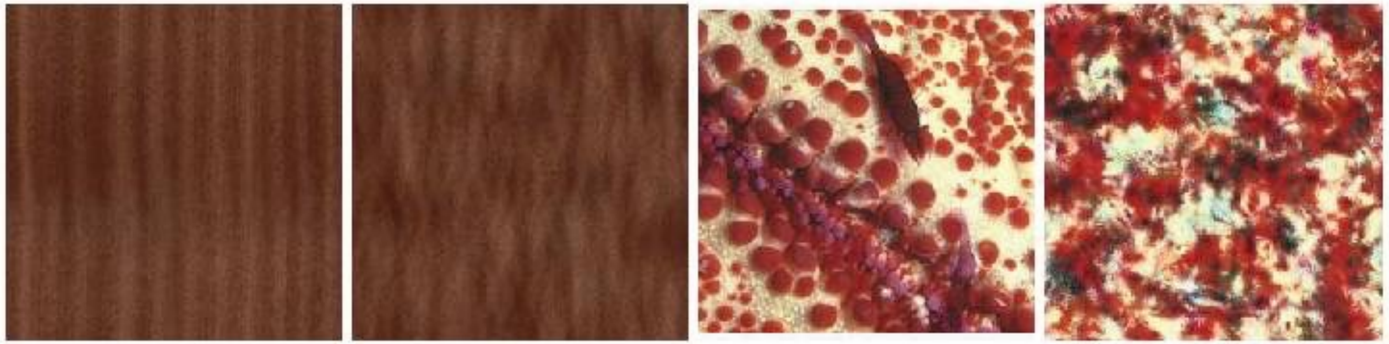


Figure 8: Examples of failures: wood grain and red coral.

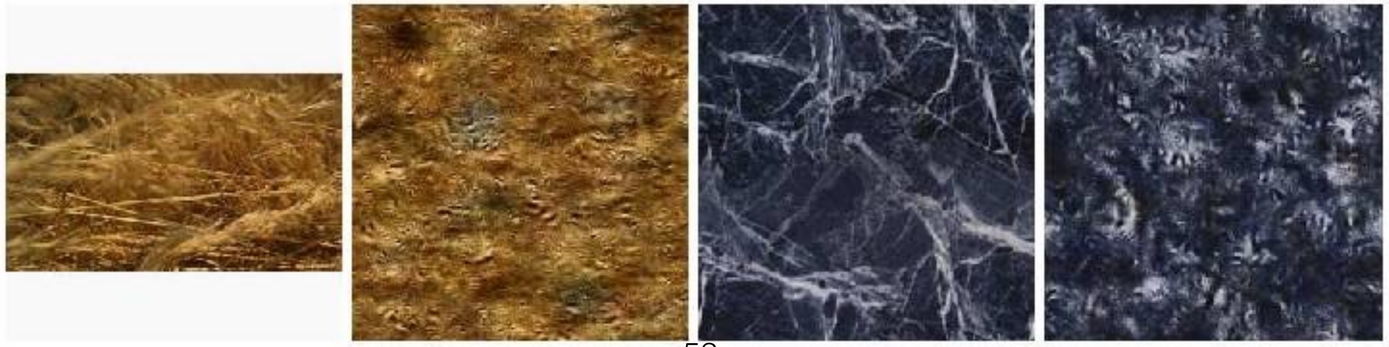


Figure 9: More failures: hay and marble.

Source: A. Efros

Nonparametric texture synthesis: who needs pyramids or filters?

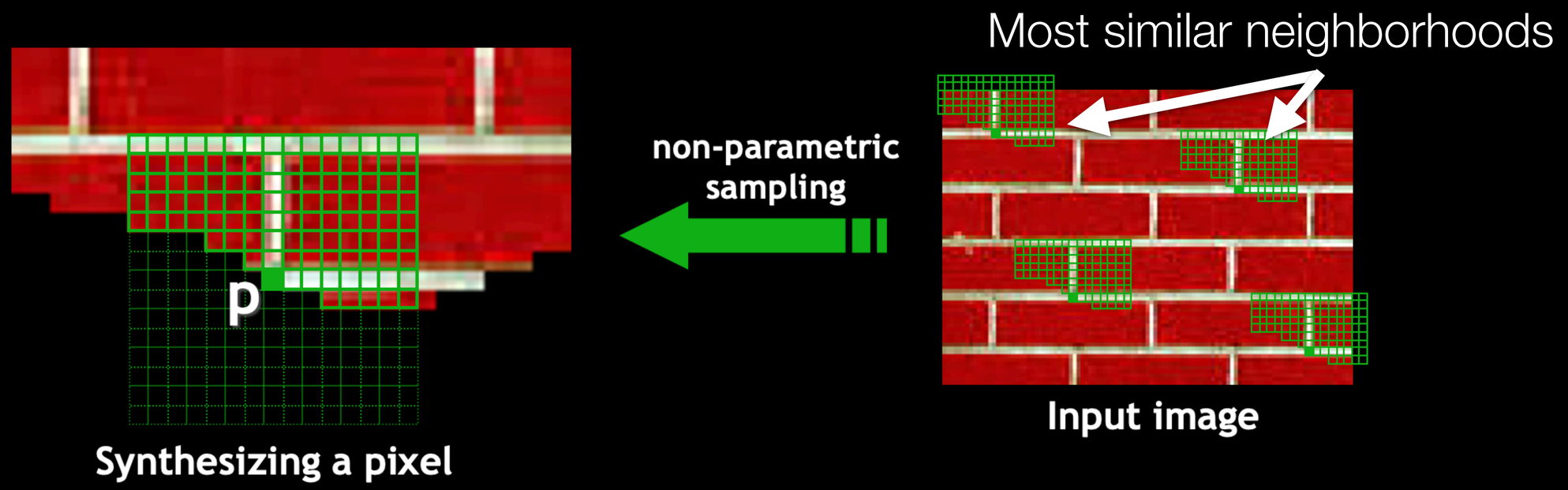
Modeling local neighborhoods

Model $p(\mathbf{p} \mid N(\mathbf{p}))$,

Probability of pixel given its neighbors

Source: A. Efros

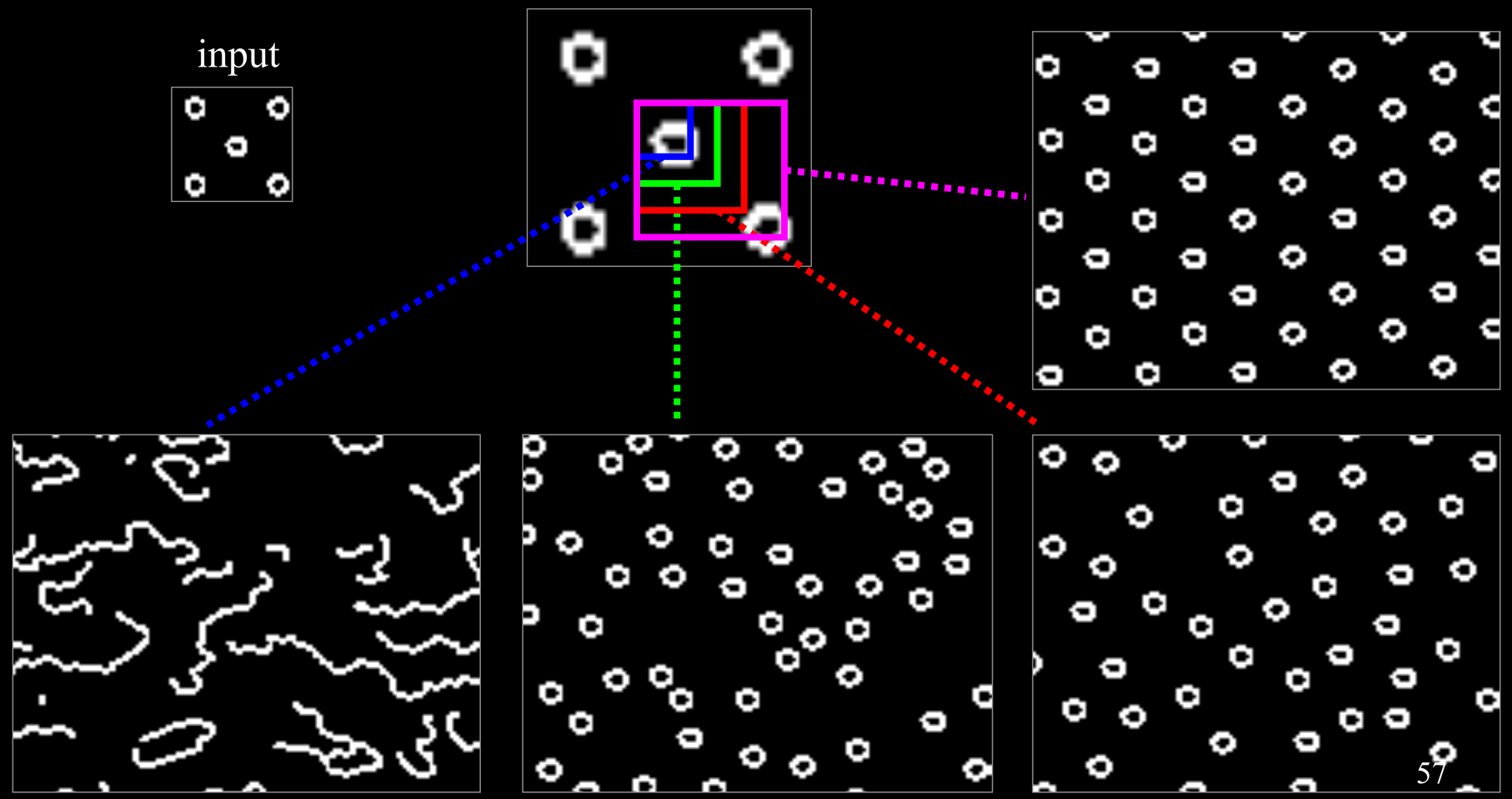
Efros & Leung Algorithm



- Synthesize one pixel at a time. Want to sample: P(**p**|N(**p**)), where N(p) are the already filled-in neighbors
  - Building explicit probability tables is hard
- Instead, we *search the input image* for all similar neighborhoods — that's our distribution for **p**
- To sample from this distribution, just pick one match at random

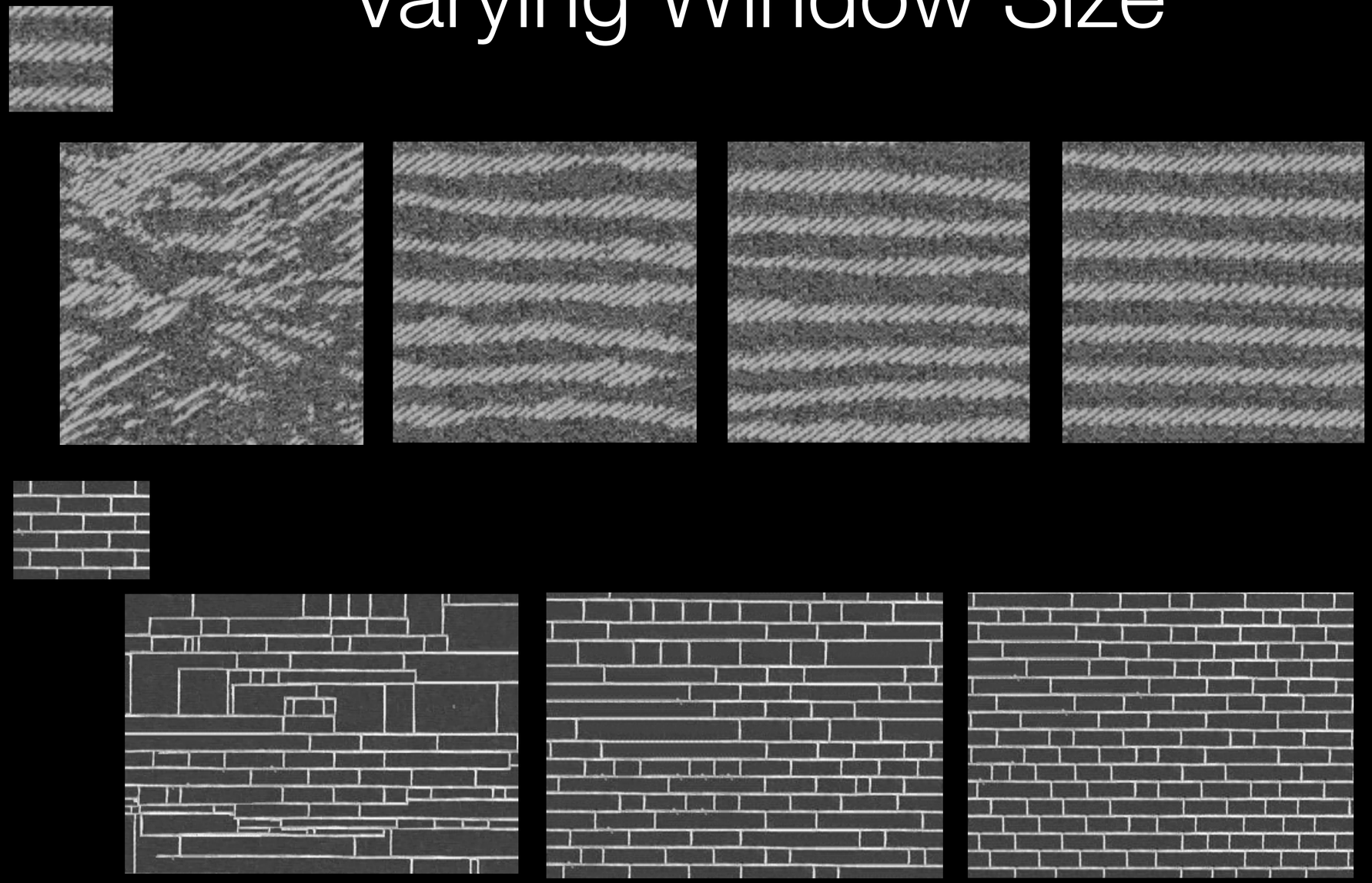
Source: A. Efros

Neighborhood Window



Source: A. Efros

Varying Window Size

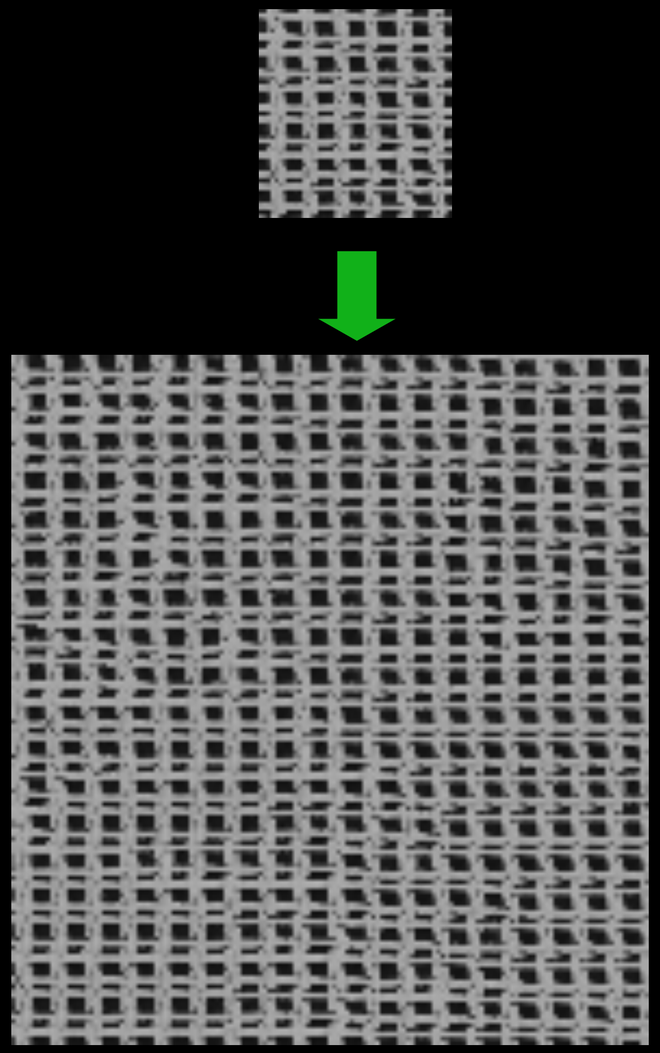


Increasing window size

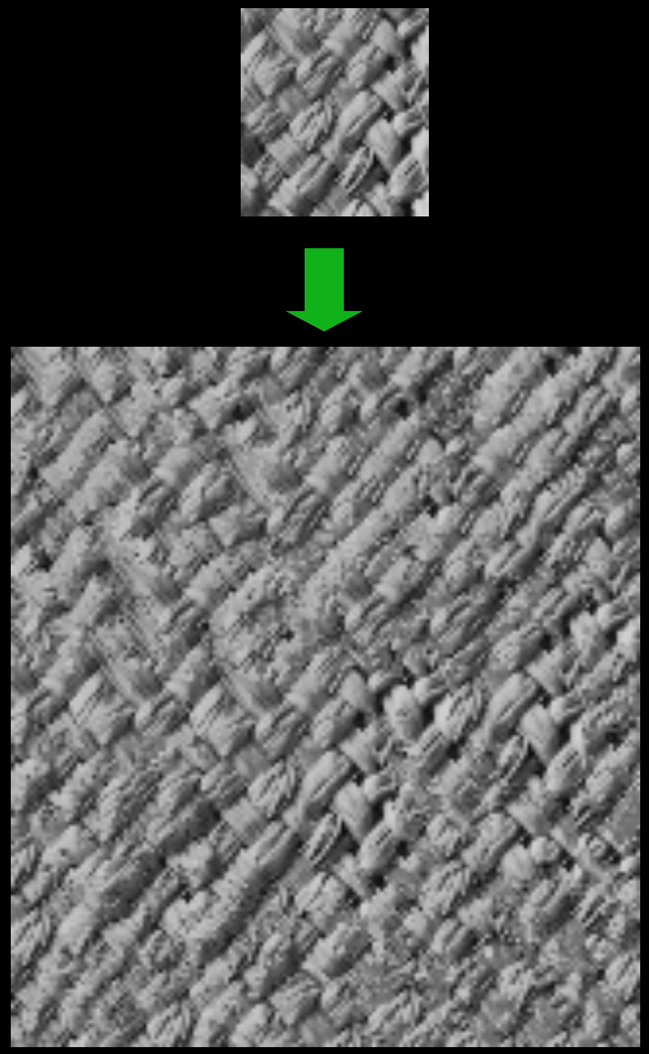
Source: A. Efros

Synthesis Results

french canvas



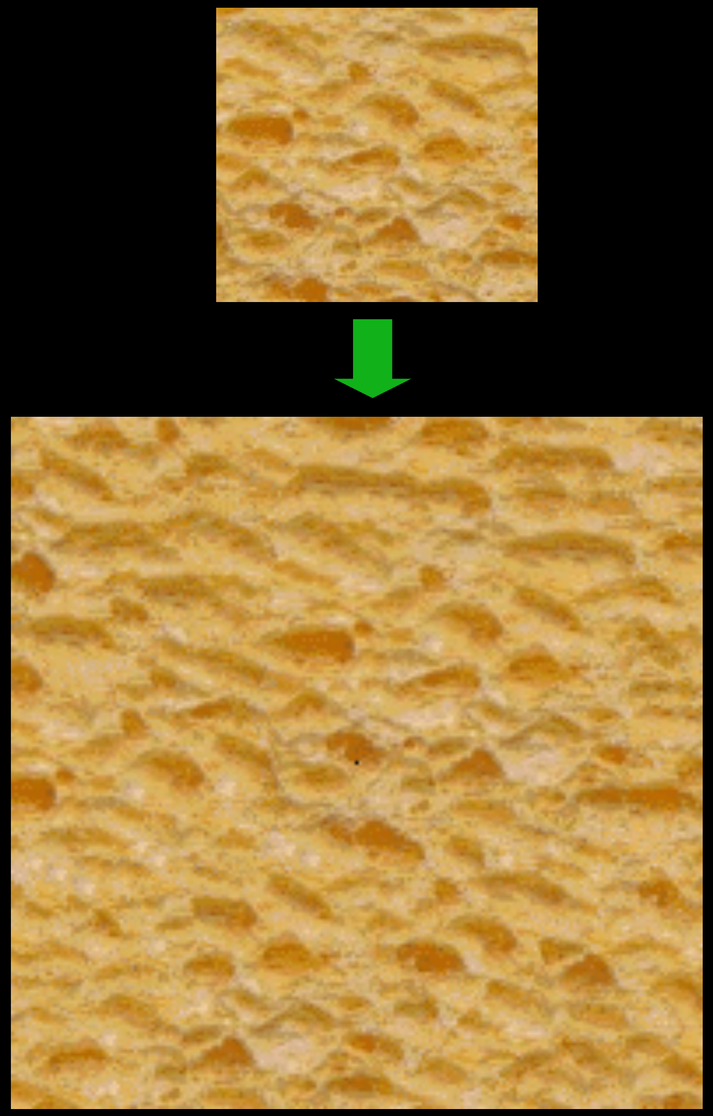
rafia weave



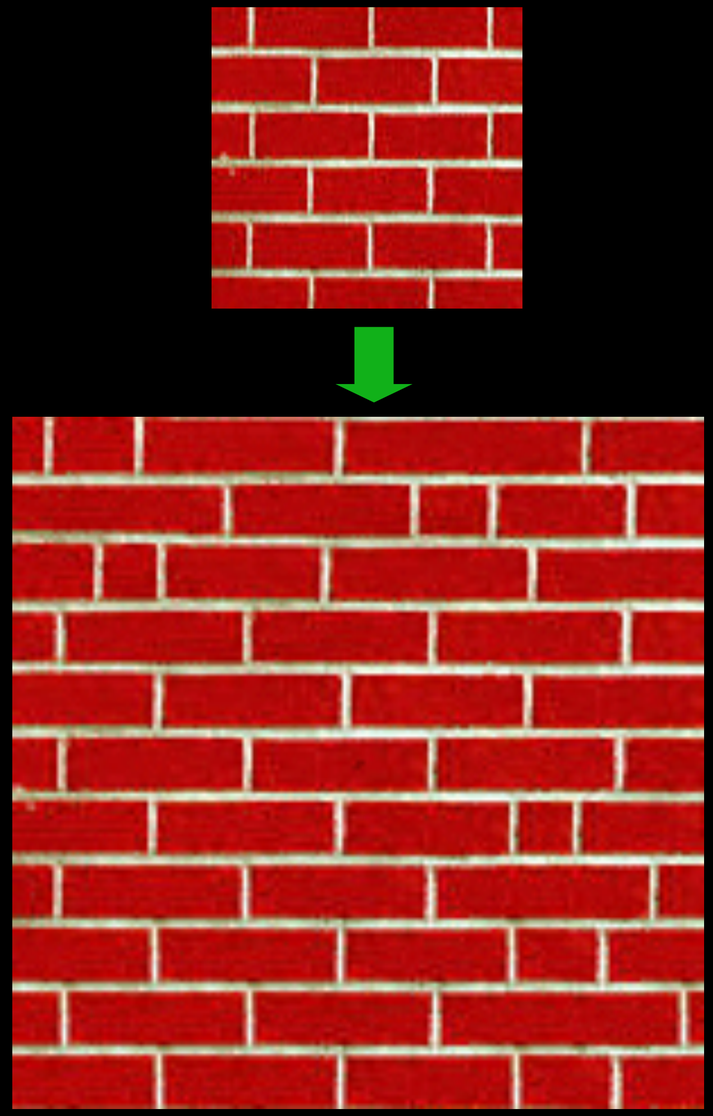
Source: A. Efros

More Results

white bread

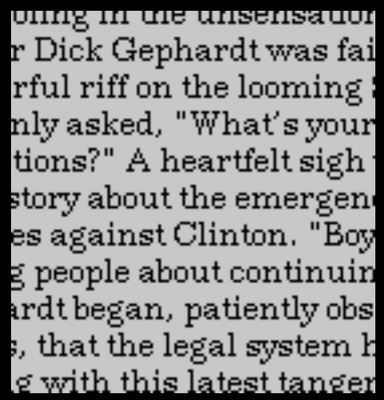


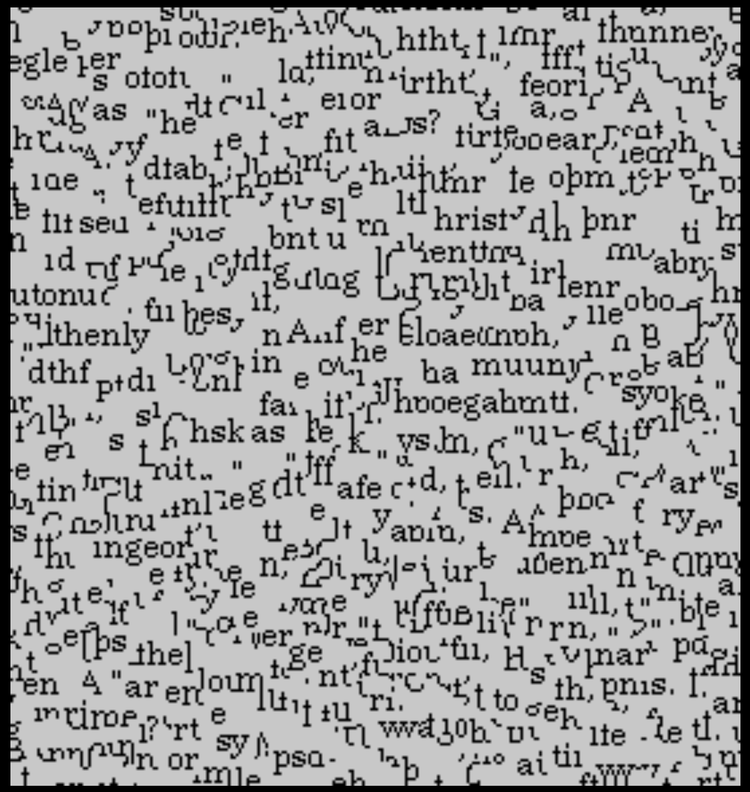
brick wall

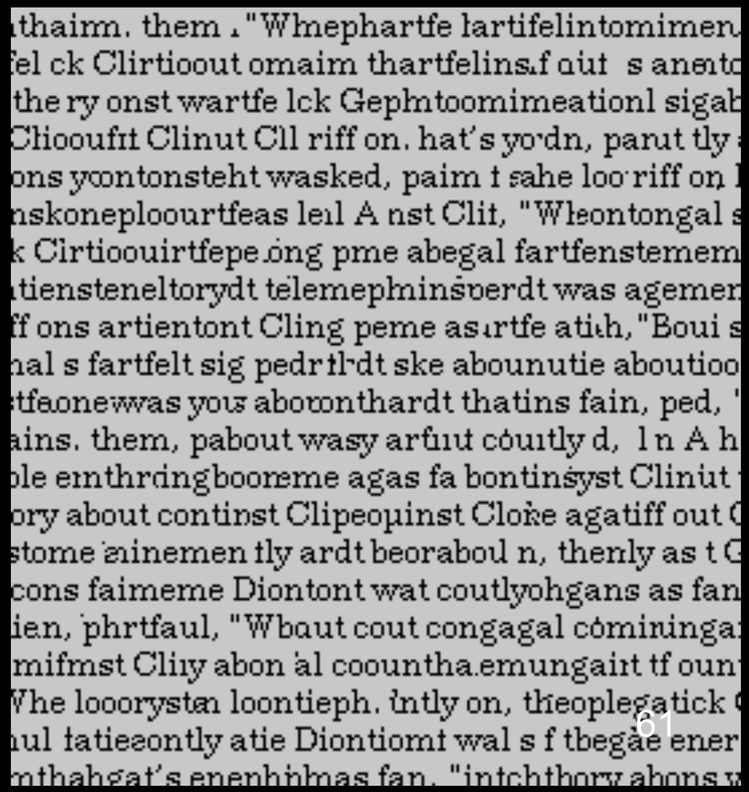


Source: A. Efros

Homage to Shannon

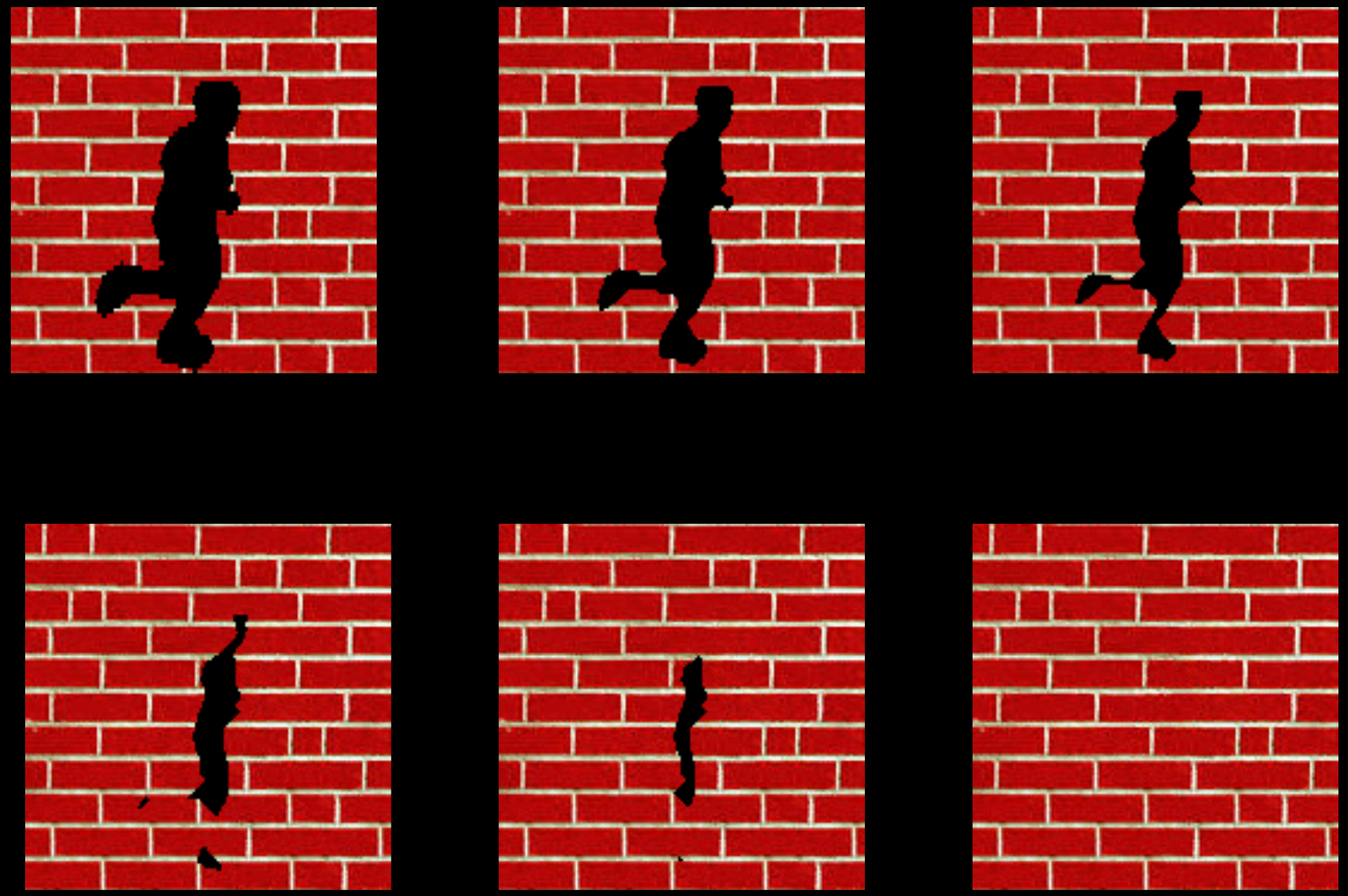






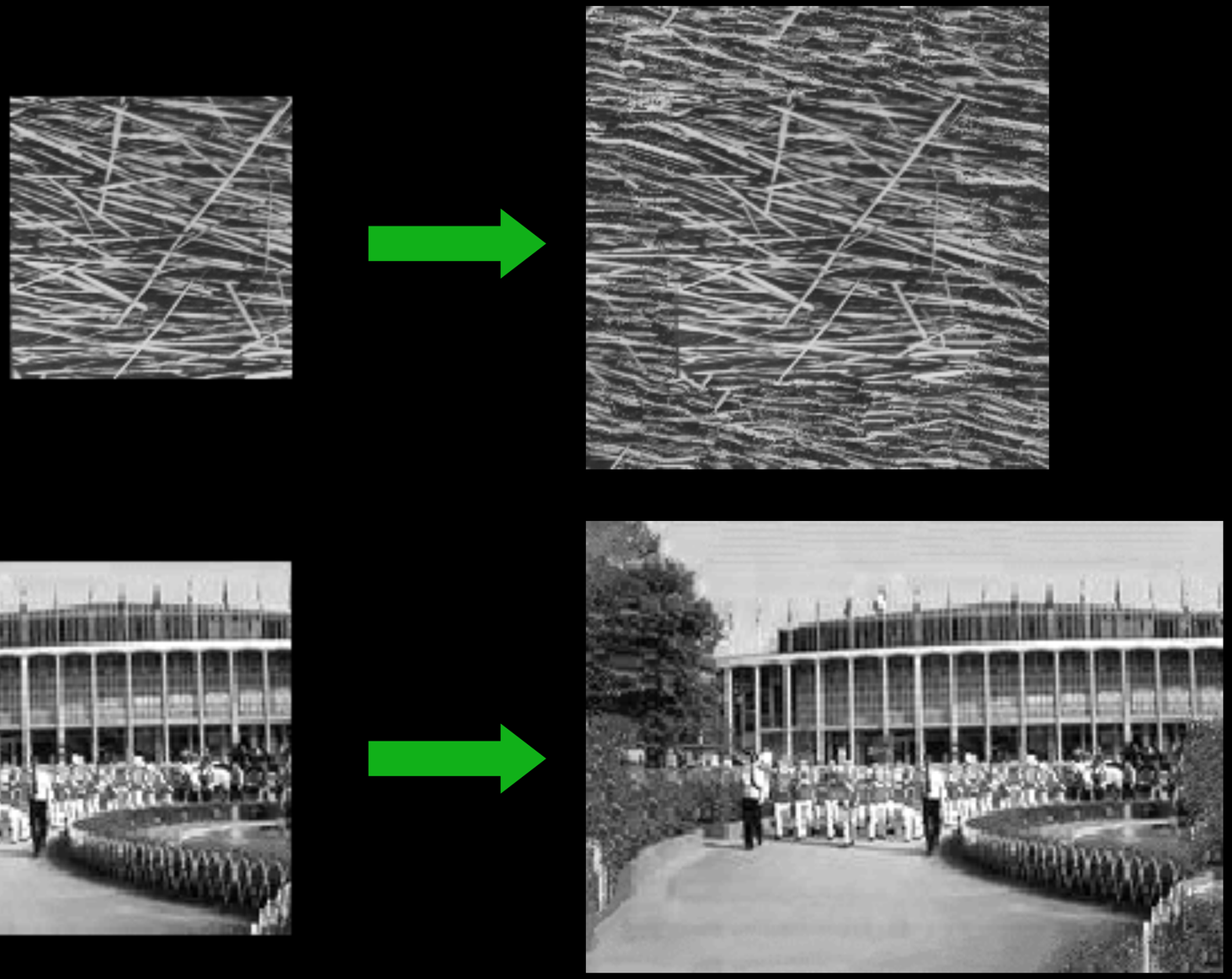
Source: A. Efros

Hole Filling (a.k.a. Inpainting)



Source: A. Efros

Extrapolation



Source: A. Efros