

Image Pyramids

Computational Photography (CSCI 3240U)

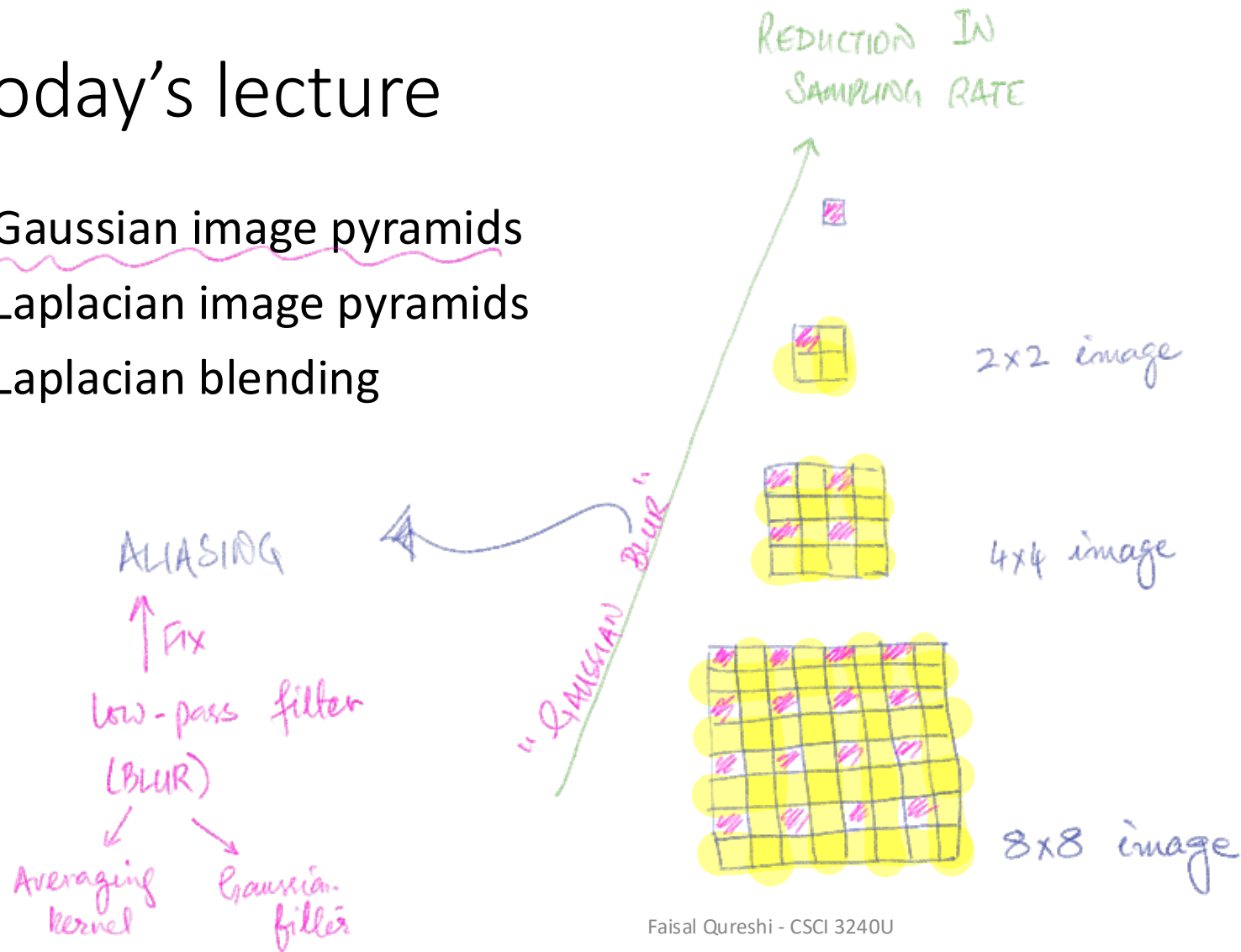
Faisal Z. Qureshi

<http://vclab.science.ontariotechu.ca>



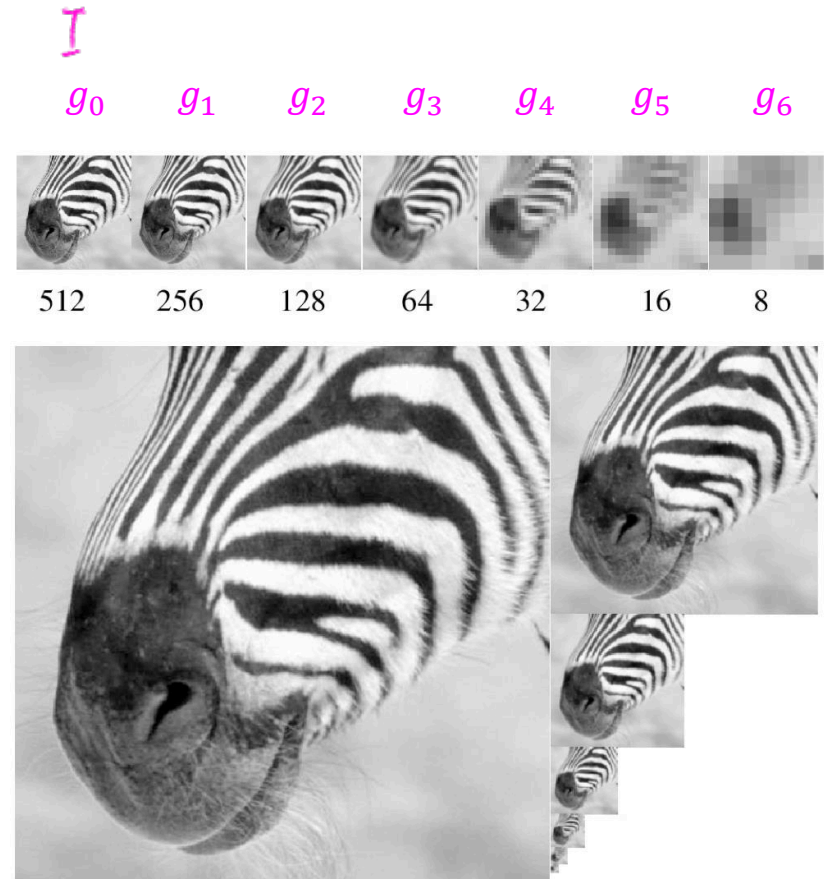
Today's lecture

- Gaussian image pyramids
- Laplacian image pyramids
- Laplacian blending



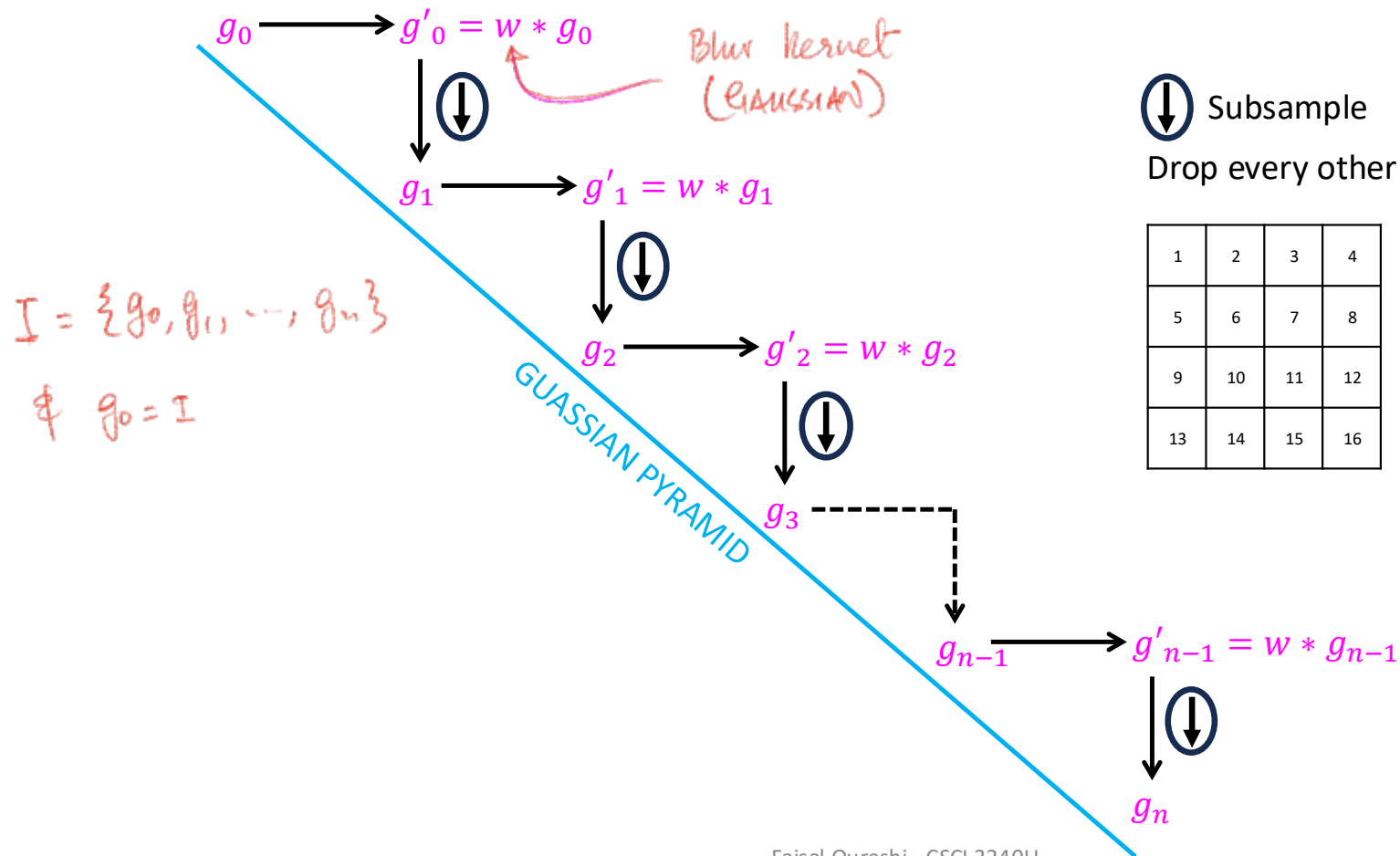
Gaussian image pyramids

- Blur the image with a Gaussian kernel
- Reduce image dimensions by half
 - Discard every other row and column
- Repeat
 - Till the desired numbers of levels have been reached or till the image because 1x1



Courtesy: Forsyth

Gaussian image pyramids



⬇ Subsample

Drop every other row and column

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16



1	3
9	11

Laplace operator

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

Compute the Laplace operator output for the shaded pixel

$I =$

1	1	9	8	1
8	8	8	8	8
1	3	5	8	1
5	3	2	8	6

$$f(x,y) = 3x^3y^2 + x^2y^2$$

$$\frac{\partial f}{\partial x} = 9x^2y^2 + 2xy^2$$

$$\frac{\partial^2 f}{\partial x^2} = 18xy^2 + 2y^2$$

$$\frac{\partial f}{\partial y} = 6x^3y + 2x^2y$$

$$\frac{\partial^2 f}{\partial y^2} = 6x^3 + 2x^2$$

$$\nabla^2 f = 18xy^2 + 2y^2 + 6x^3 + 2x^2$$

Sobel

$$H_x =$$

1	0	-1
2	0	-2
1	0	-1

$$H_y =$$

1	2	1
0	0	0
-1	-2	-1

3x3 Gaussian kernel (approx.)

$$G = 1/16$$

1	2	1
2	4	2
1	2	1

Compute the Laplace operator output for the shaded pixel

$I =$

1	1	9	8	1
8	8	8	8	8
1	3	5	8	1
5	3	2	8	6

Filter is already flipped.

-1	0	1
-2	0	2
-1	0	1

compute

$$I * H_x \rightarrow I_x, \quad I_x * H_x \rightarrow I_{xx}$$

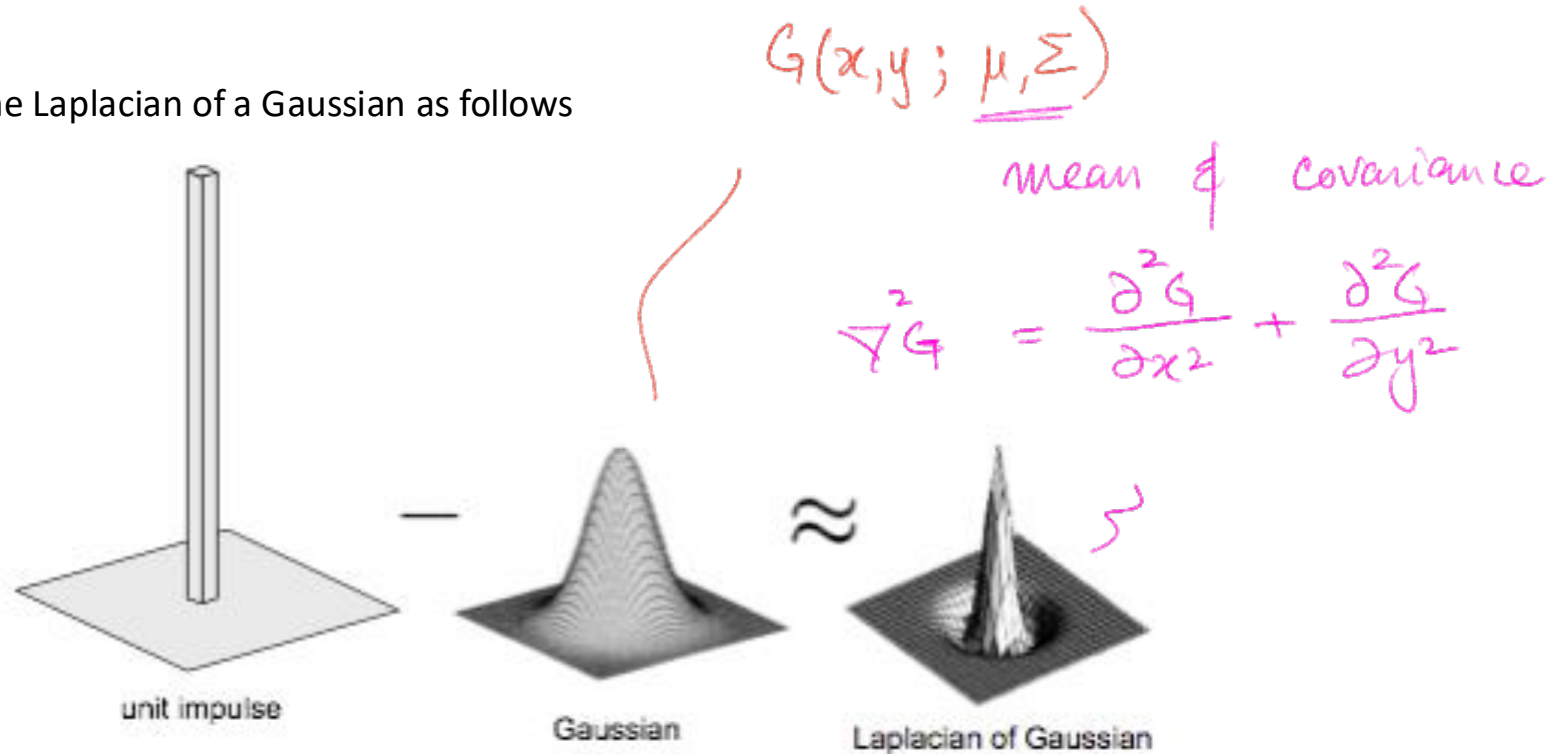
$$I * H_y \rightarrow I_y, \quad I_y * H_y \rightarrow I_{yy}$$

$$\nabla^2 I = I_{xx} + I_{yy}$$

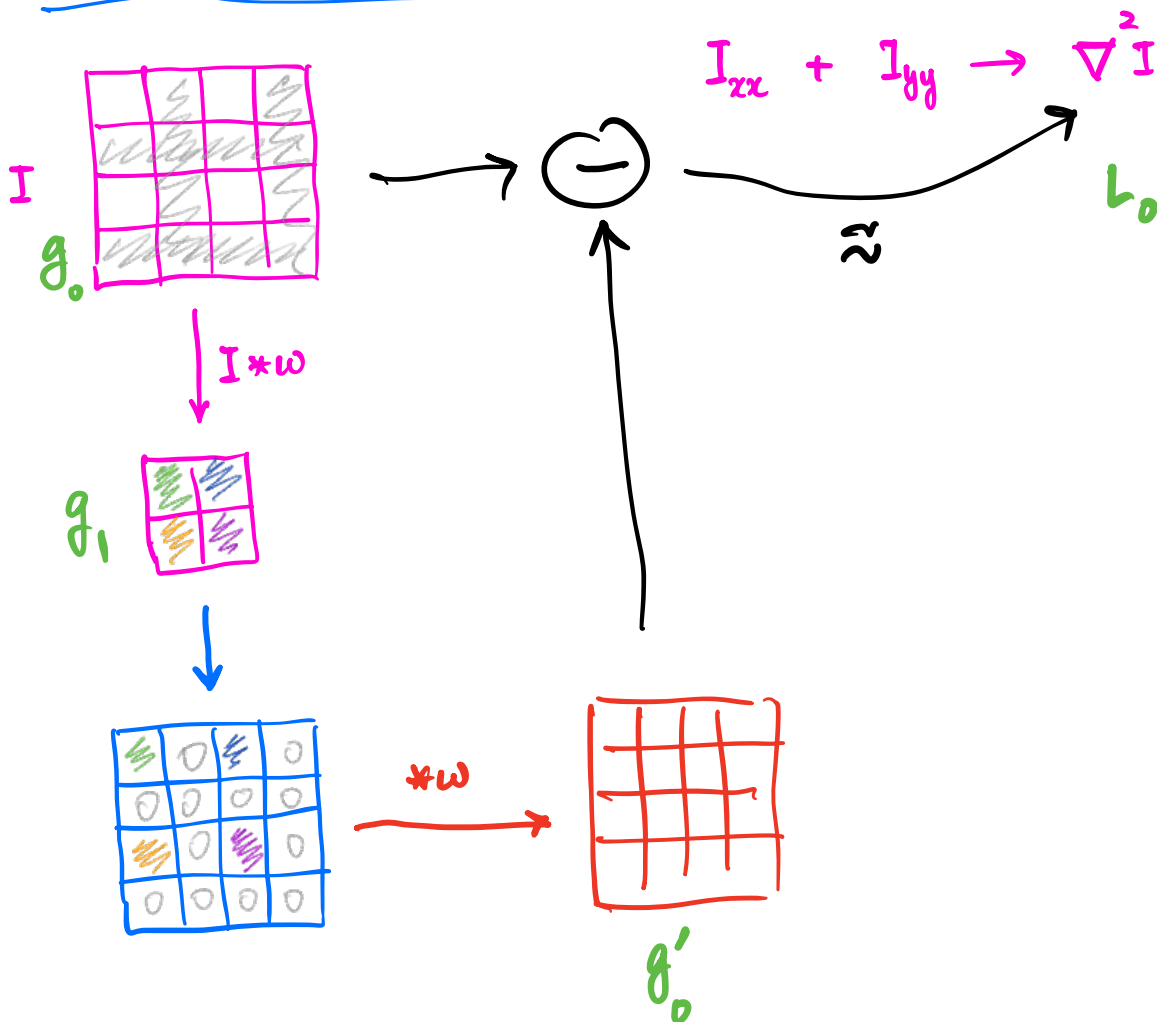
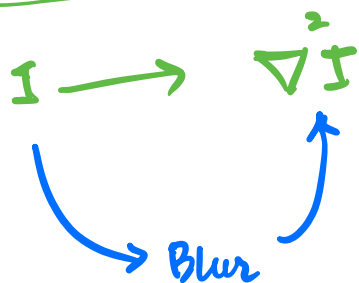
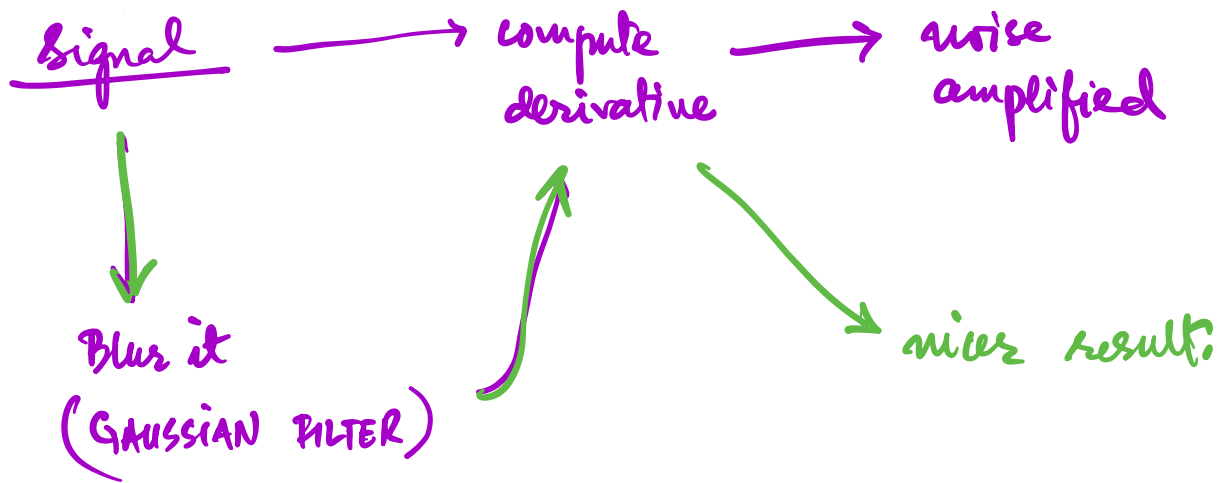
Laplacian of the image at every pixel.

Laplacian of a Gaussian

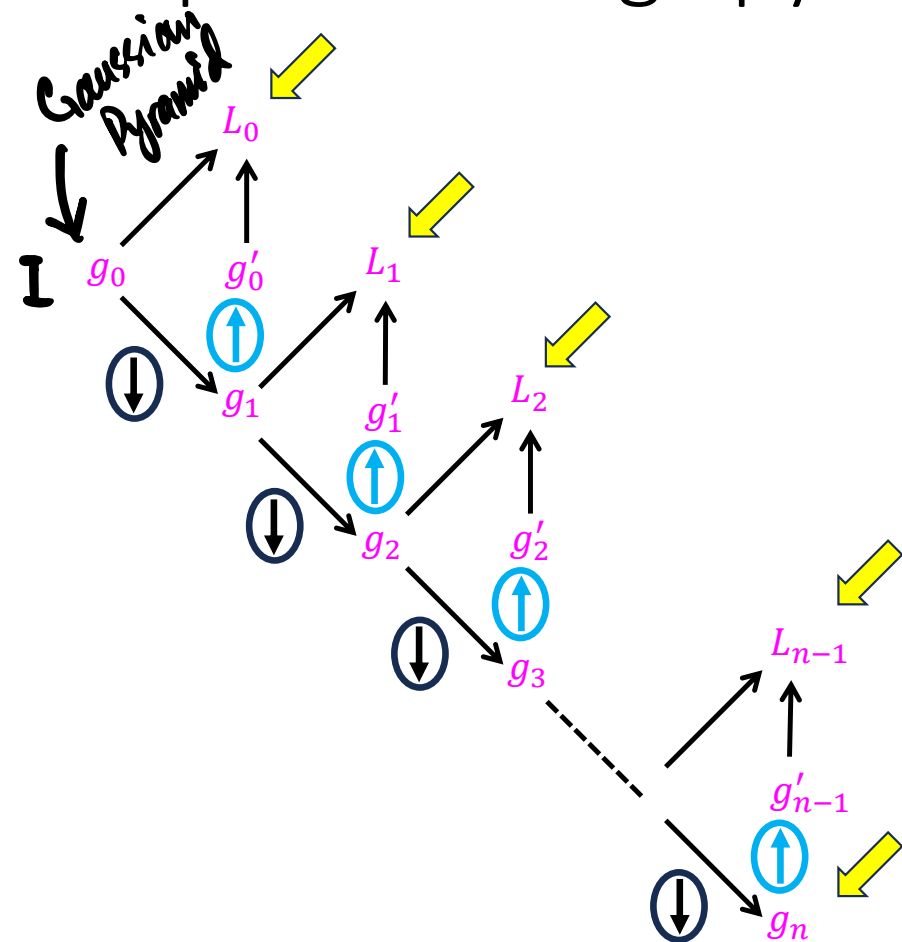
We can approximate the Laplacian of a Gaussian as follows



Source: Lazebnik



Laplacian image pyramids



Pyramid Down

1. $\tilde{g}_i = w * g_i$
2. Downsample $\tilde{g}_i$ by dropping every other row or column

Pyramid Up

1. Upsample g_i by adding 0 between samples to create $\tilde{g}_i$
2. $g'_{i-1} = w * \tilde{g}_i$. This will fill in the 0s added in the previous step

1	0	3	0
0	0	0	0
5	0	7	0
0	0	0	0

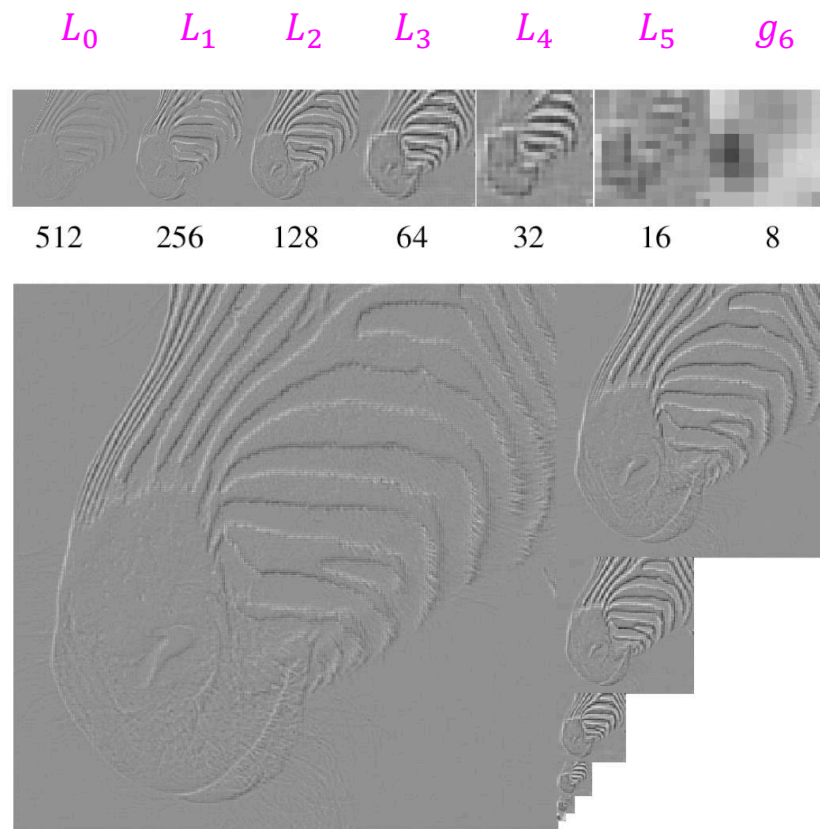
1	3
9	11

$$I = \{L_0, L_1, \dots, L_{n-1}, g_n\}$$

Laplacian image pyramid

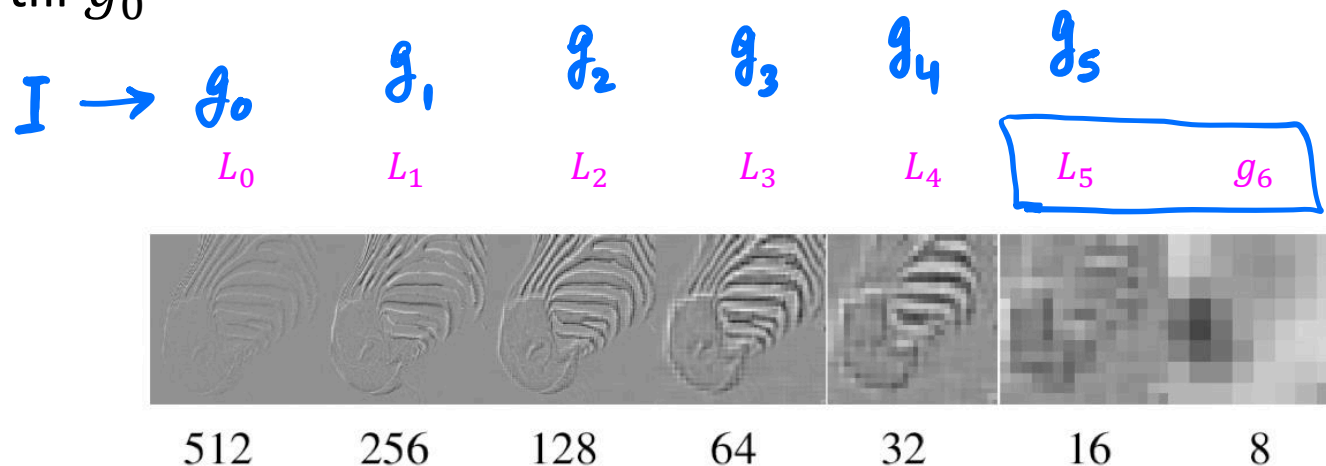
Given image $I = g_0$

- Convolve g_i with low-pass filter w and down-sample by half to construct g_{i+1}
 - Downsample by discarding every other row and column
- Upsample g_{i+1} by double by inserting 0s and interpolating the missing values by convolving it with w and create g'_i
- Compute $L_i = g_i - g'_i$
- Repeat

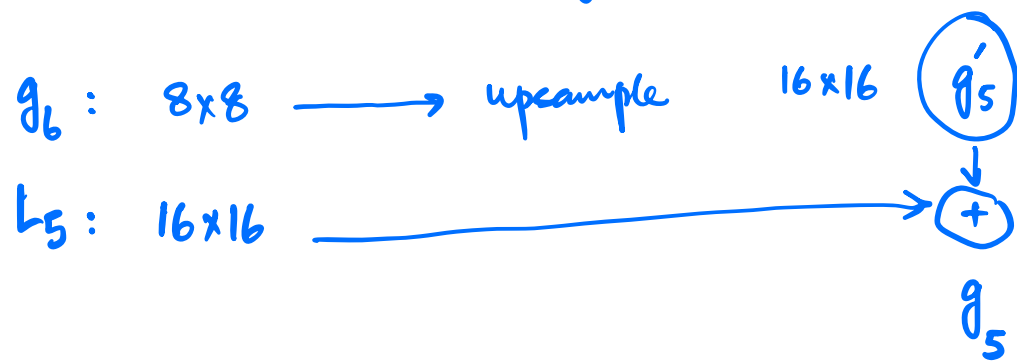


Reconstructing the original image

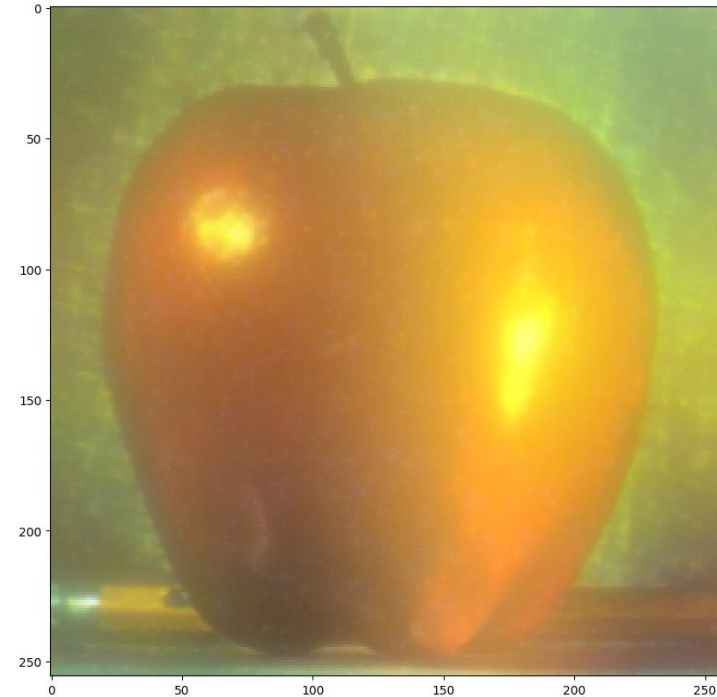
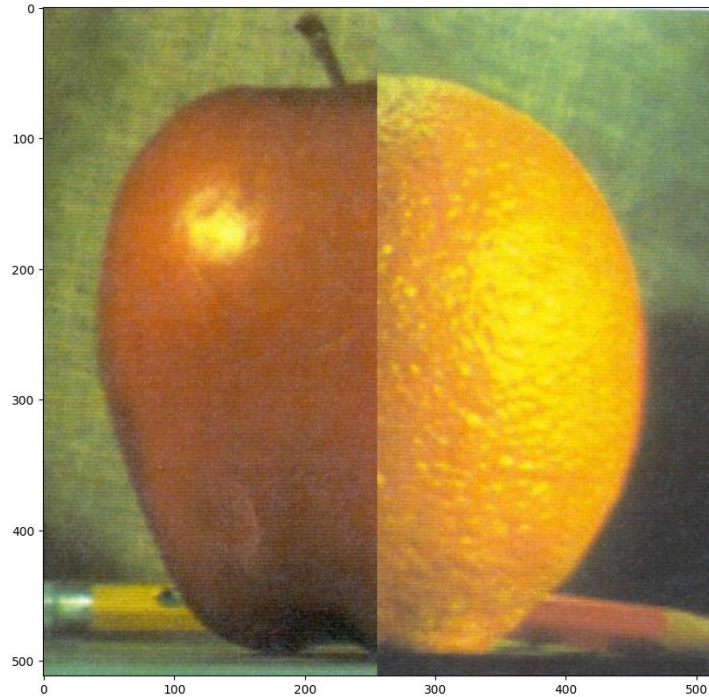
- g_n is upsampled by inserting 0s and interpolating the missing value by convolving with w to get g'_{n-1}
- Compute $g_{n-1} = g'_{n-1} + L_{n-1}$
- Repeat till g_0



Given $L_5, g_6 \longrightarrow g_5$

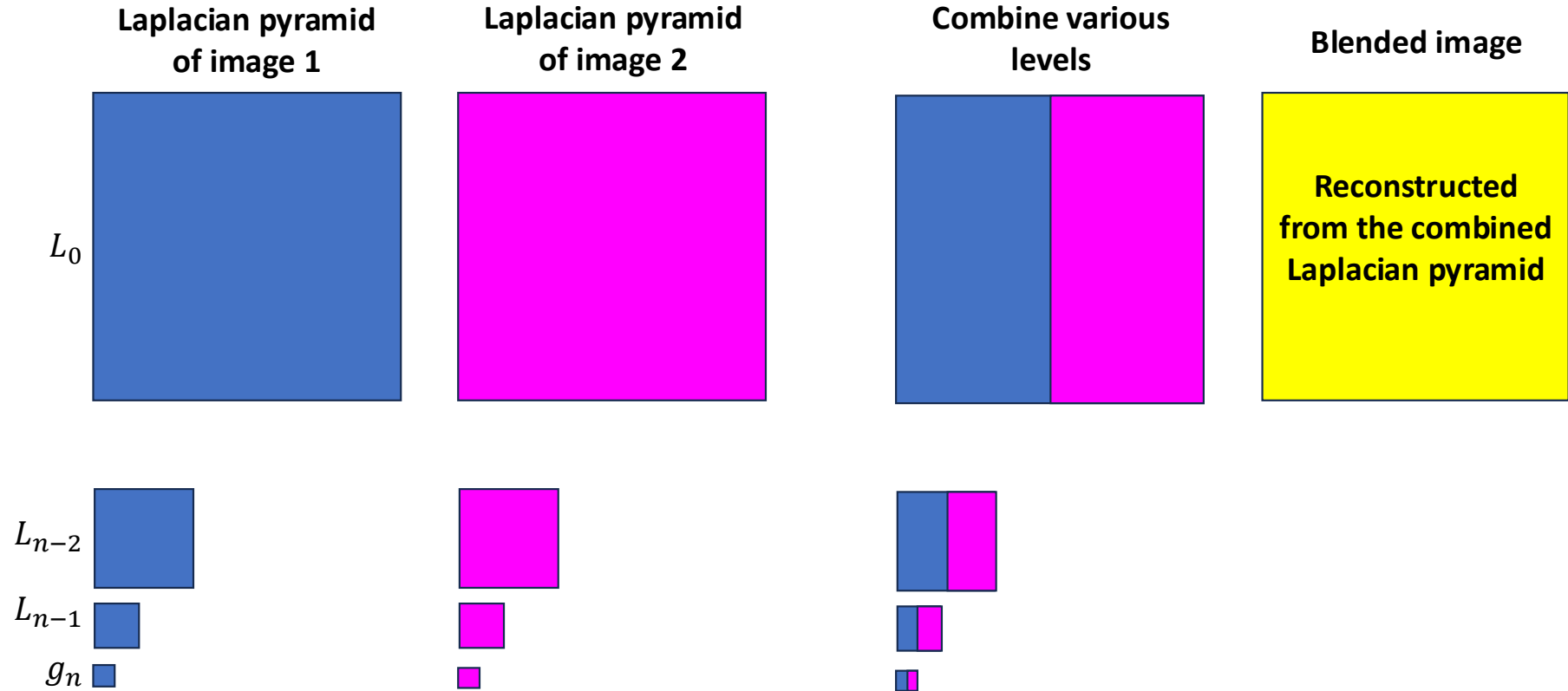


An Example of Laplacian Blending



Laplacian Blending

Laplacian Blending



Uses

- Scale-invariant image analysis
- Template matching
- Image registration
- Image enhancement
- Interest point detection
- Object detection
- Image compression
- ...

Summary

- Gaussian pyramid
 - Coarse-to-fine search
 - Multi-scale image analysis (hold this thought)
- Laplacian pyramid
 - More compact image representation
 - Can be used for image compositing (computation photography)
- Downsampling
 - Nyquist limit: The Nyquist limit gives us a theoretical limit to what rate we have to sample a signal that contains data at a certain maximum frequency. Once we sample below that limit, not only can we not accurately sample the signal, but the data we get out has corrupting artifacts. These artifacts are "aliases".
 - Need to sufficiently low-pass before downsampling

Various image representations

- Pixels
 - Great for spatial processing, poor access to frequency
- Fourier transform
 - Great for frequency analysis, poor spatial info
- Pyramids
 - Trade-off between spatial and frequency information