Image Pyramids

Computational Photography (CSCI 3240U)

Faisal Z. Qureshi
http://vclab.science.ontariotechu.ca
OntarioTech UNIVERSITY

Today’s lecture

- Gaussian image pyramids
- Laplacian image pyramids
- Laplacian blending

2	0	1
6	$\boxed{5}$	7
8	9	0

$$f(x,y) \xrightarrow[\text{Laplacian}]{} \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \nabla^2 f$$

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

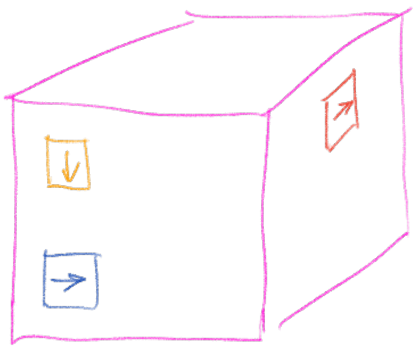
$$\frac{\partial f}{\partial x} = 3x^2 y + 6xy^2 \qquad \frac{\partial f}{\partial y} = ?$$

$$\frac{\partial^2 f}{\partial x^2} = 6xy + 6y^2 \qquad \frac{\partial^2 f}{\partial y^2} = ?$$

$$f(x,y,z) = x^4 y + 3yz^2$$

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

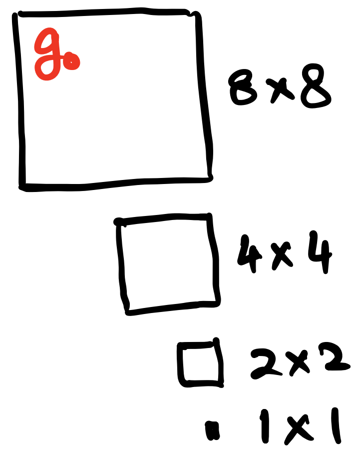
$$\left.\begin{aligned} \frac{\partial f}{\partial x} &\longrightarrow H_x * f \\ \frac{\partial f}{\partial y} &\longrightarrow H_y * f \end{aligned}\right\} \text{Taylor Series}$$



$I \in \mathbb{R}^{8\times 8}$

low-pass filter (Gaussian)

$g_0 = I$



$g_0 \rightarrow g_0' = w * g_0$

$\downarrow$ subsample (drop every other row & column)

$g_1 \longrightarrow g_1' = w * g_1$

$\downarrow$ subsample

g_2

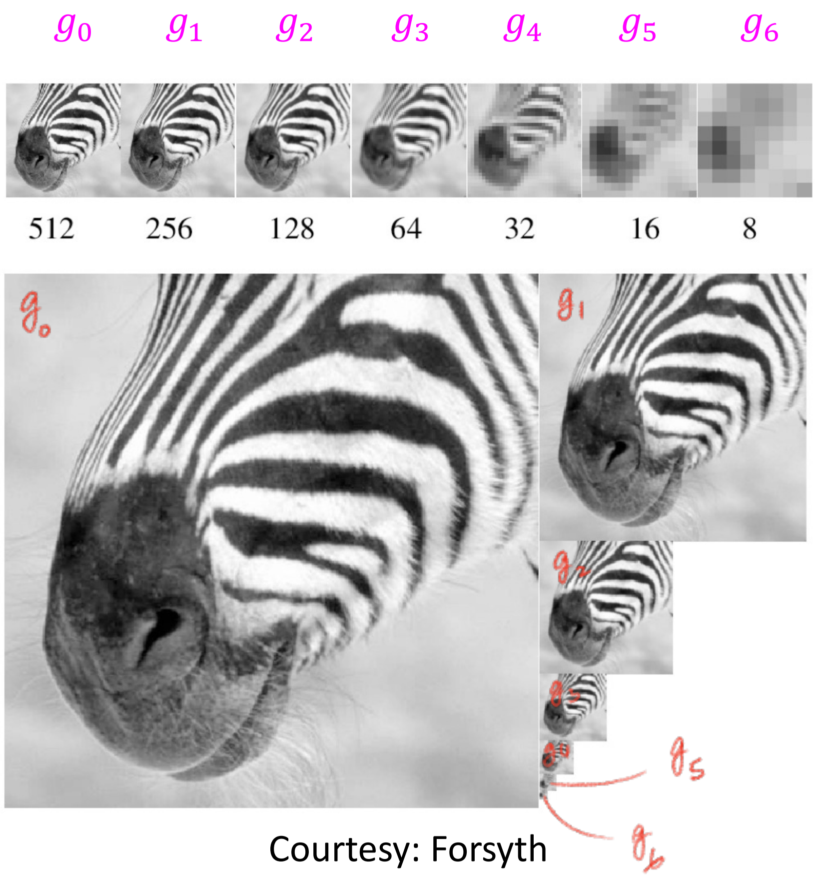
$g_n \in \mathbb{R}^{1\times 1}$

$I = \{g_0, g_1, g_2, \ldots, g_n\}$

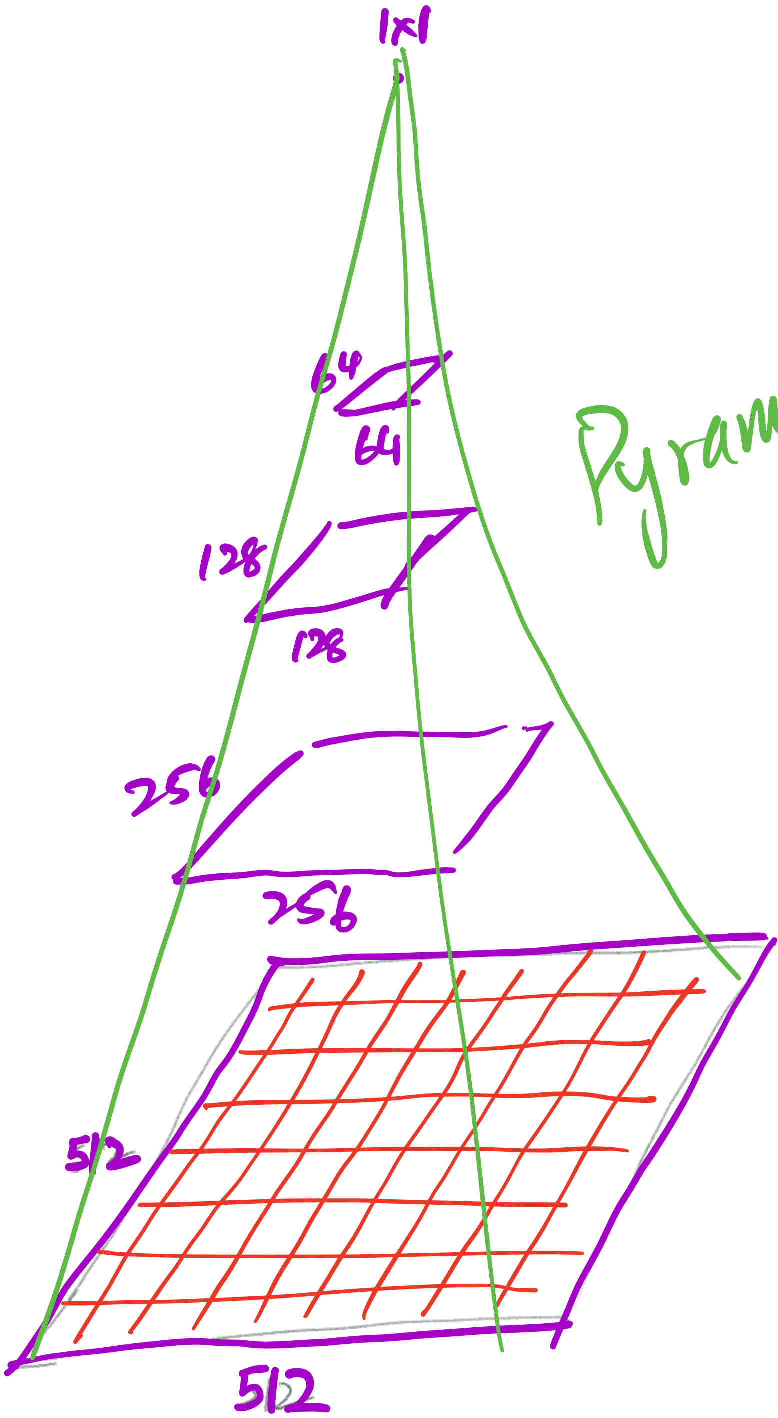
Construction of a Gaussian Pyramids.

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



Pyramid

3	1
1	2

3	2	1	6
6	7	5	1
1	1	2	3
3	9	8	1

!!! WITHOUT BLURRING VISUAL QUALITY DROPS !!!

$$I = g_0$$

$$g_0' = G_\sigma * g_0$$

$$g_1 = \text{subsample}(g_0')$$

drop alternate rows & columns.

Laplace operator

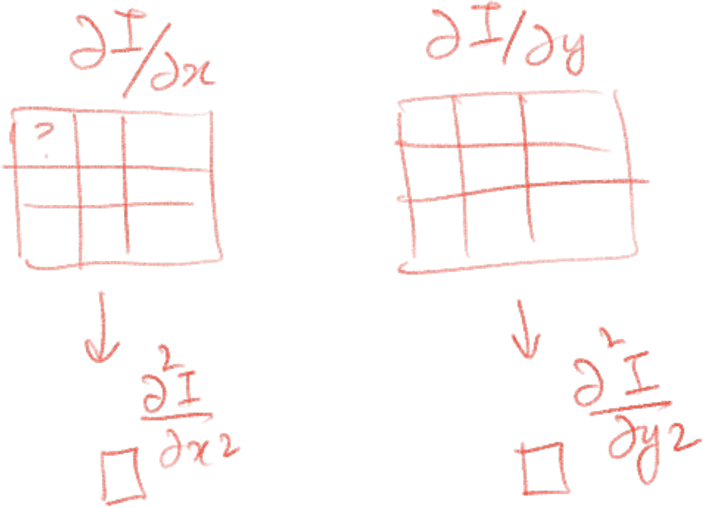
second partial derivative

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

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



Sobel

$H_x =$

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

$H_y =$

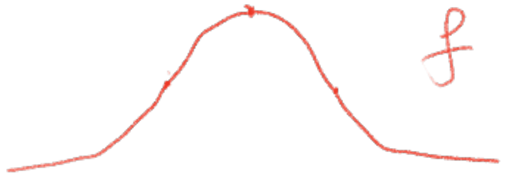
1	2	1
0	0	0
-1	-2	-1

3x3 Gaussian kernel (approx.)

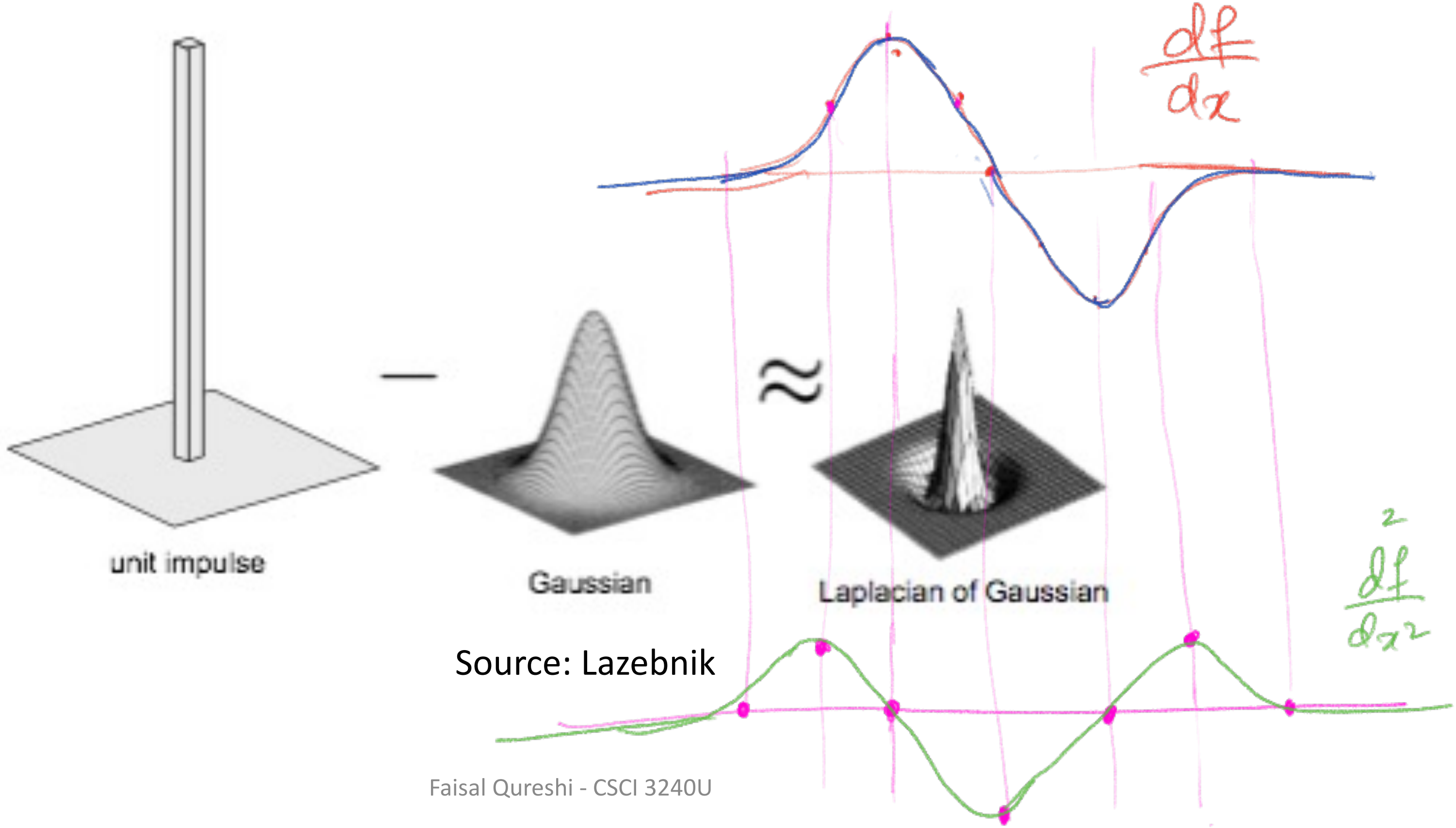
$G = 1/16$

1	2	1
2	4	2
1	2	1

Laplacian of a Gaussian



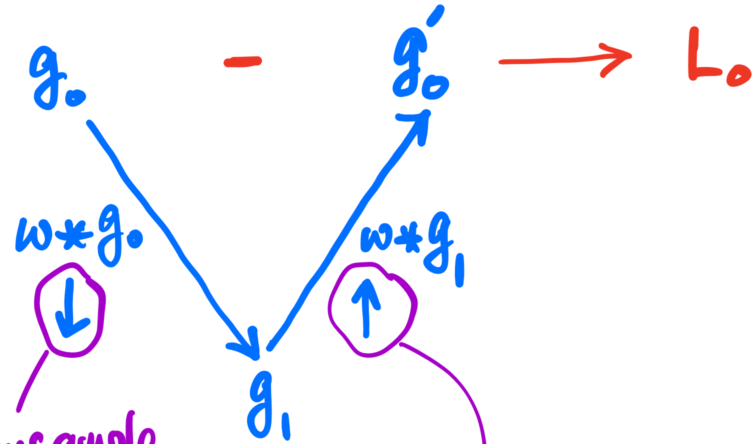
We can approximate the Laplacian of a Gaussian as follows



Source: Lazebnik

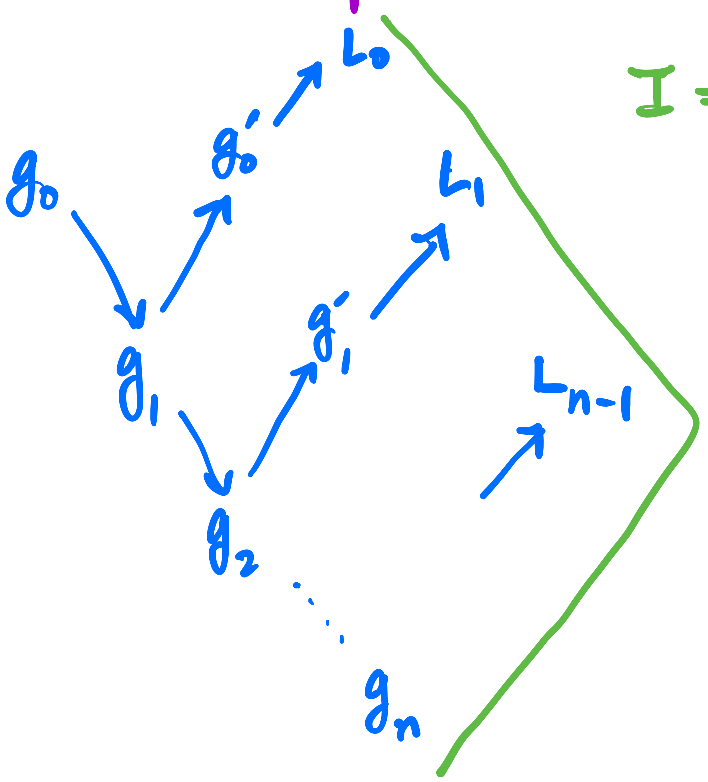
Construction of a Laplacian Pyramid

$I \rightarrow g_0$



downsample.
discard every other row & column.

upsample
adding 0's between samples.



$$I = \{\underbrace{L_0, L_1, L_2, \ldots,}_{\text{Detailed structure}} g_n\}$$

- Most of the pixels are close to zero
- Quantization can lead to compression.

1 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17

18 bytes

$\boxed{1}$ $\boxed{1}$ 1 2 16 3

0 1 2 3 4 5

6 bytes

Reconstruction from a Laplacian Pyramid.

$$\{L_0, L_1, L_2, \ldots, g_n\} \overset{?}{\rightsquigarrow} I$$

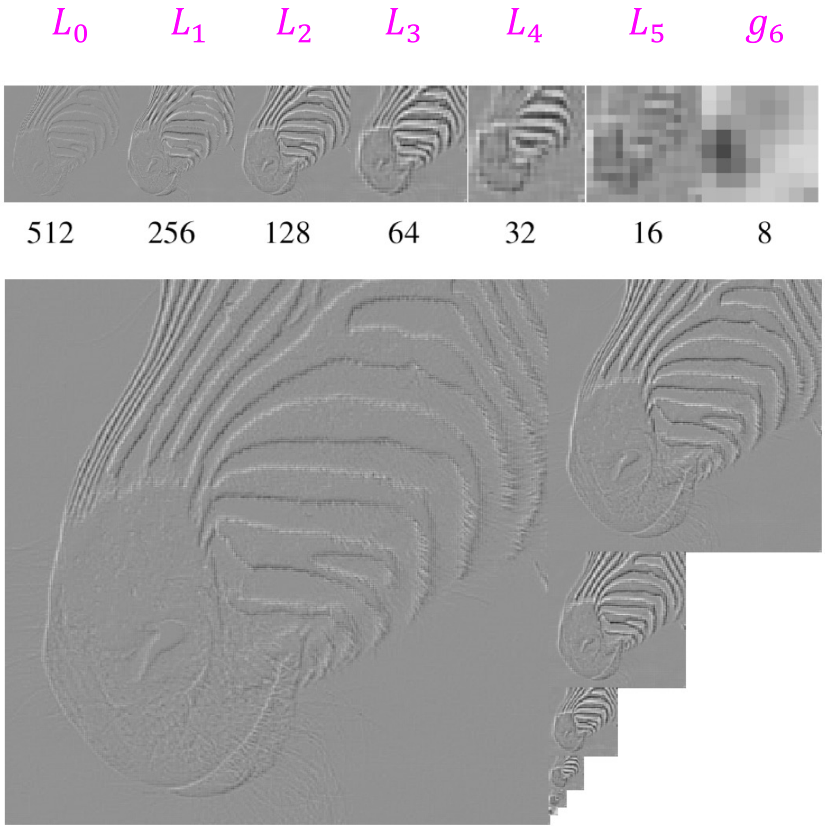
How do you reconstruct from Gaussian Pyramid?

$$\{\underset{I}{\underline{\underline{g_0}}}, g_1, g_2, \ldots, g_n\}$$

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 and interpolating the missing values by convolving it with w and create g'_i
- Compute $L_i = g_i - g'_i$
- Repeat



$\{L_0, L_1, L_2, \ldots, g_n\}$.

$$g_n \xrightarrow{\uparrow (w*)} g'_{n-1} \xrightarrow{+\,L_{n-1}} g_{n-1}$$

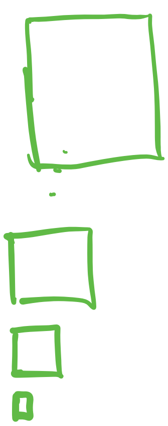
upsample.
Add 0s between samples

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \uparrow \quad \begin{bmatrix} 1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 \\ 3 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * w$$

Laplacian Blending

A

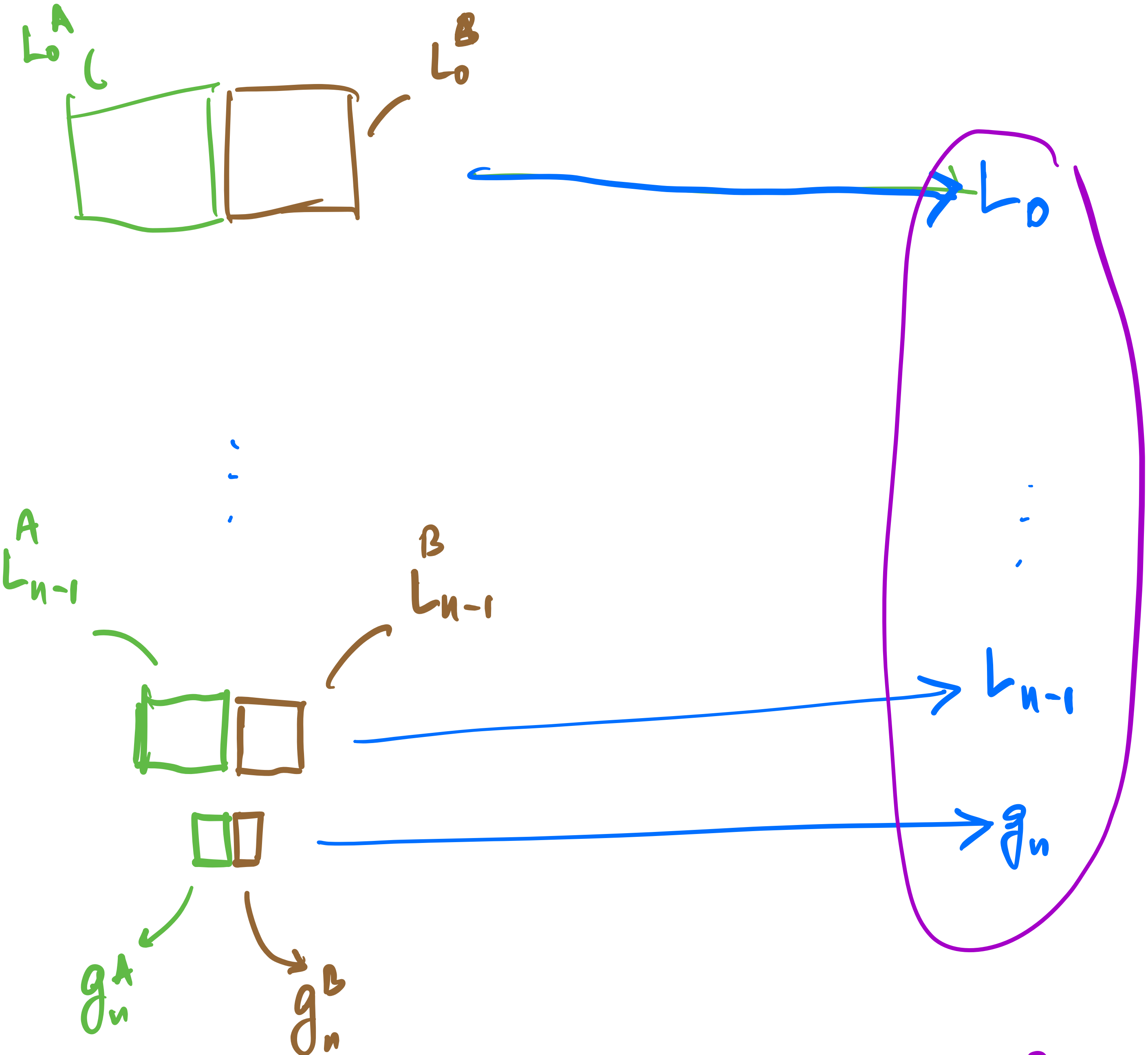
$$\begin{bmatrix} L_0^A \\ L_1^A \\ \vdots \\ g_n^A \end{bmatrix}$$



B

$$\begin{bmatrix} L_0^B \\ L_1^B \\ \vdots \\ g_n^B \end{bmatrix}$$

MERGE THE TWO LAPLACIAN PYRAMIDS.



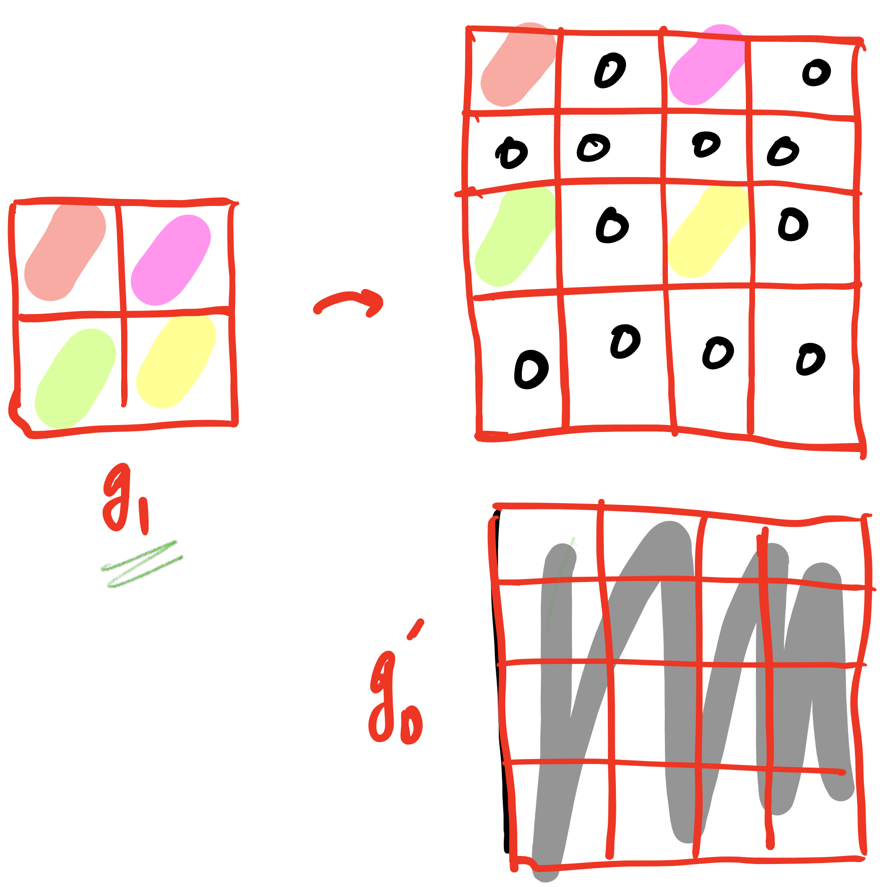
$$\{L_0, L_1, \ldots, g_n\}$$

$$\downarrow$$

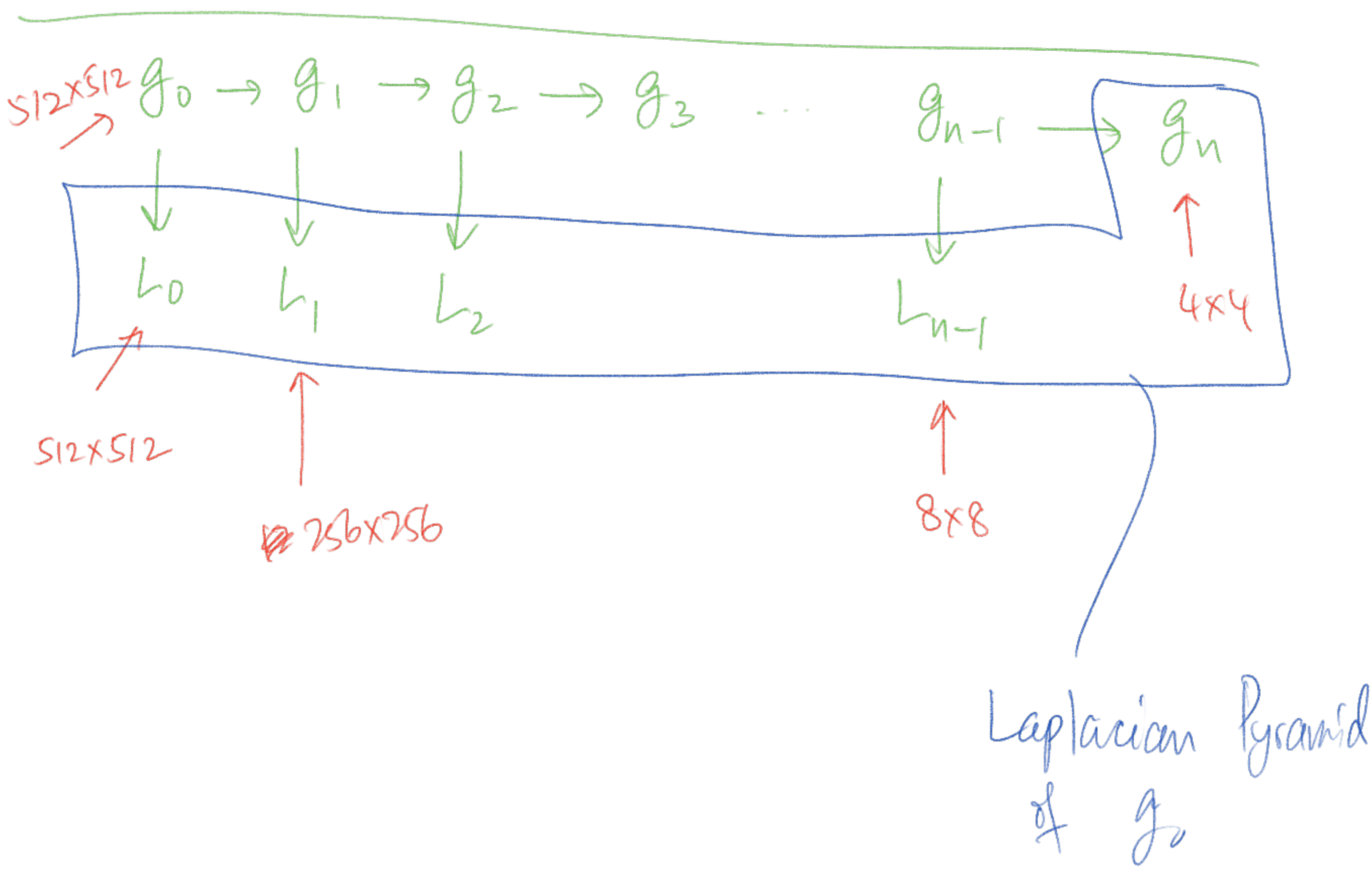
$$I$$

3	4	1	5
1	0	9	7
3	2	4	6
1	6	1	8

g_0

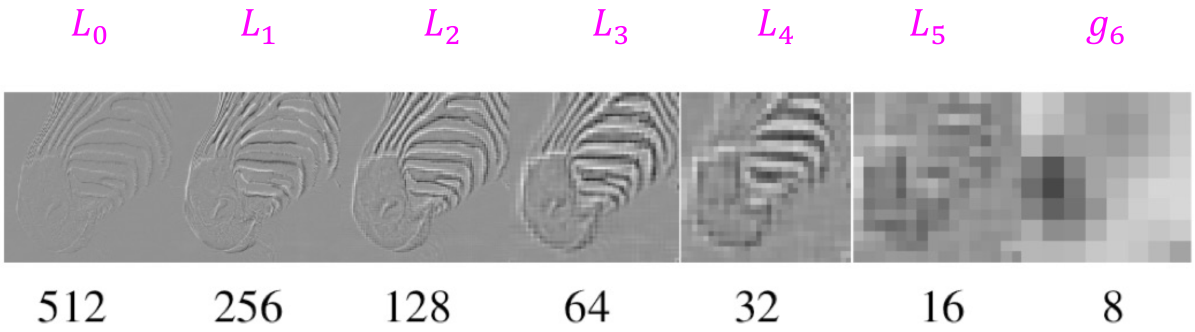


$$L_0 = g_0 - g_0'$$



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



Laplacian blending

$f(x,y)$ $\qquad \frac{\partial f}{\partial x} = f_x \qquad \frac{\partial f}{\partial y} = f_y.$

$$\nabla f = [f_x \quad f_y]^T$$

$$\nabla^2 f = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix}$$

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