Computational Photography

Dr. Irfan Essa
Professor
School of Interactive Computing

Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Digital Images: Merging and Blending Images using Image Pyramids

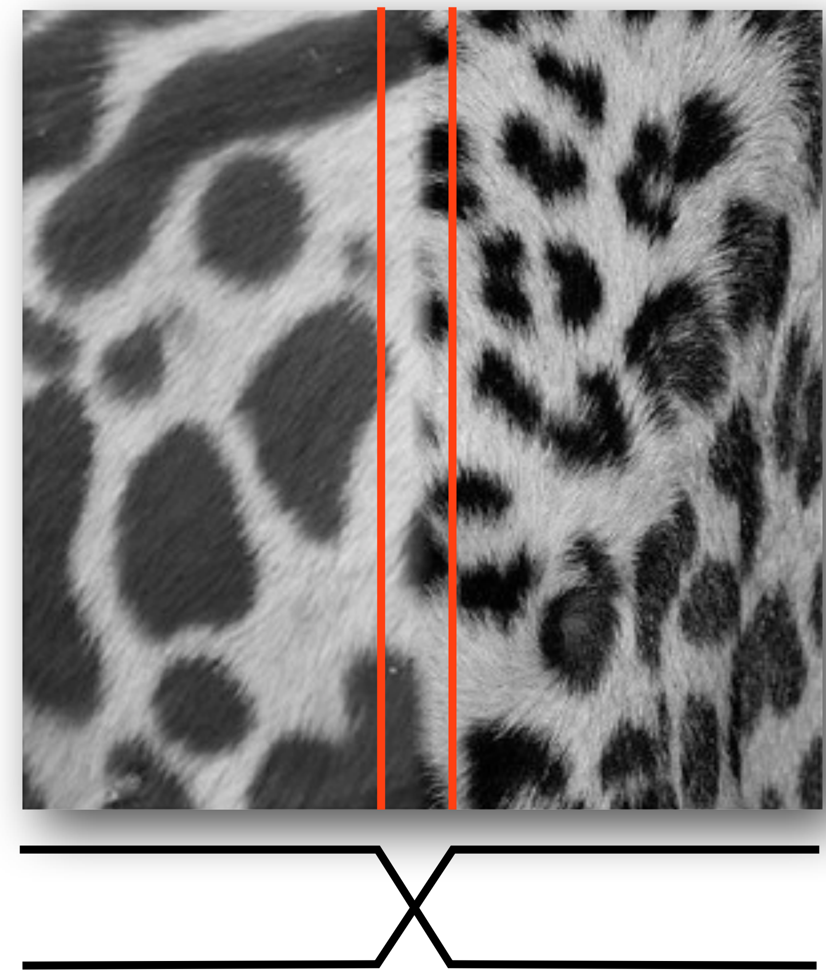
Dr. Irfan Essa
Professor
School of Interactive Computing

Different methods for Combing Multiple Images to Generate a Novel Image: Use of Image Pyramids

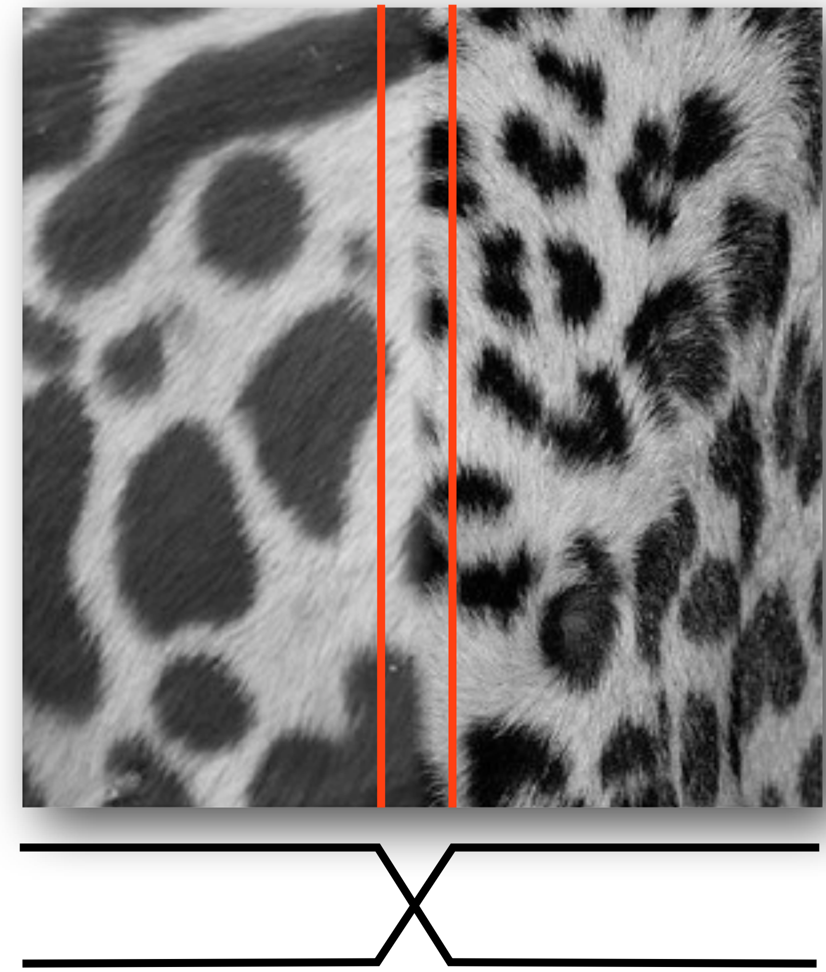
Lesson Objectives

- ★ Explain in your own words the Gaussian and the Laplacian Pyramids
- ★ Explain in your own words, how these Pyramids encode the Frequency domain
- ★ Write the equation for computing a Laplacian Pyramid form a Gaussian Pyramid
- ★ Describe in your own words the specific process to blend two images using Pyramids

REVIEW: Optimal Window Size



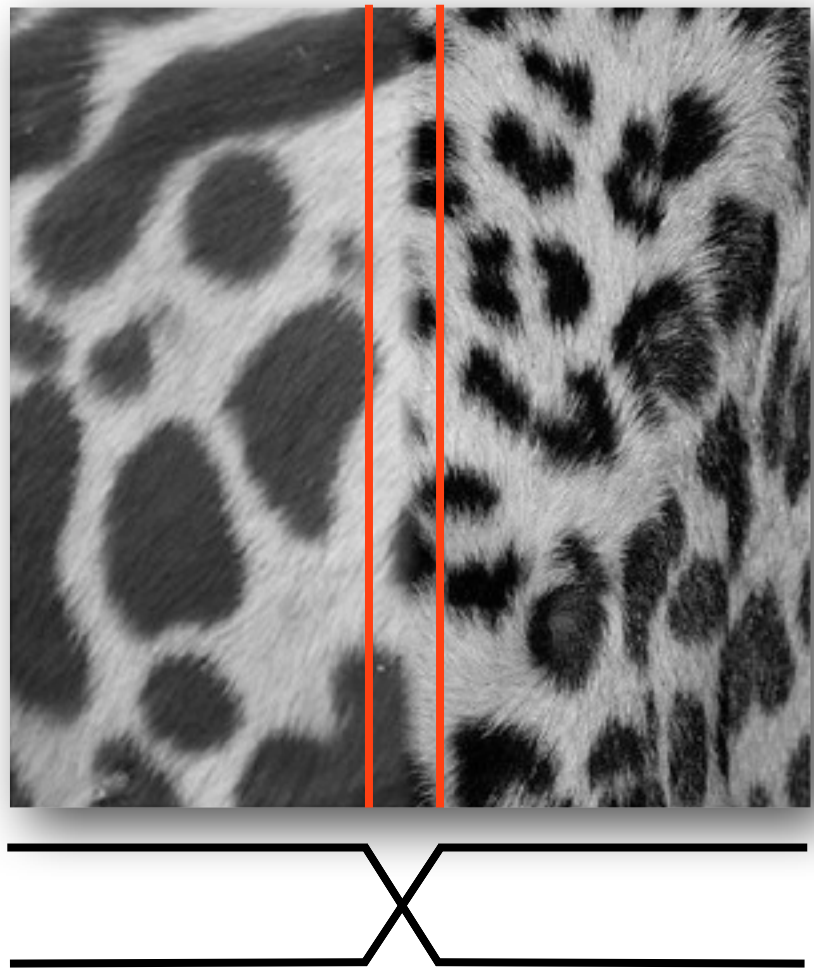
REVIEW: Optimal Window Size



★ To avoid seams: Window = size of largest prominent “feature”

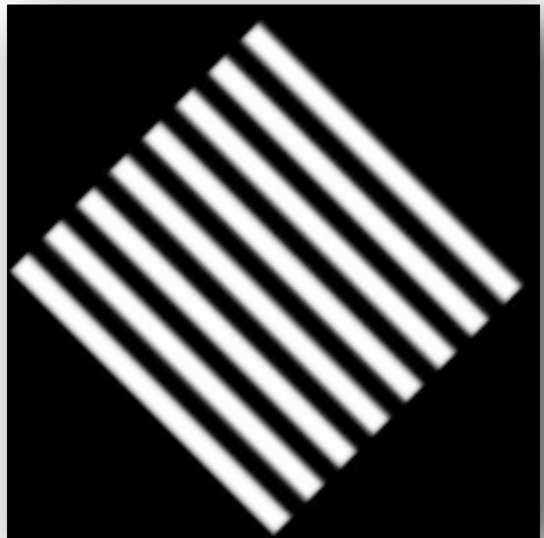
4

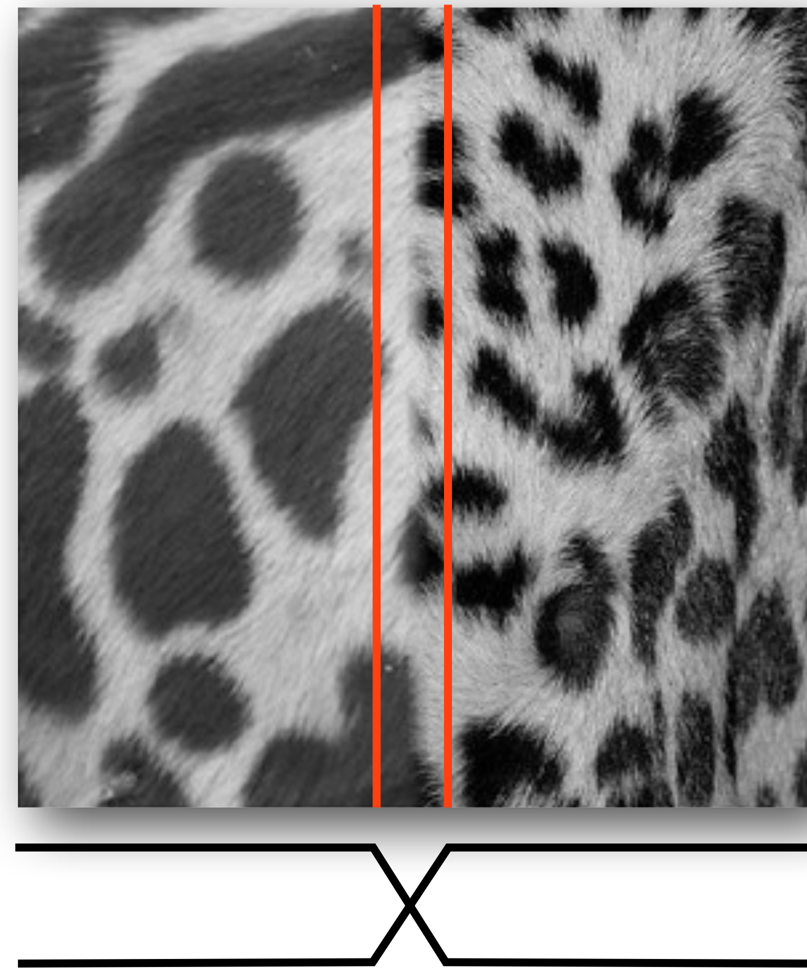
REVIEW: Optimal Window Size



- ★ To avoid seams: Window = size of largest prominent “feature”
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent “feature”

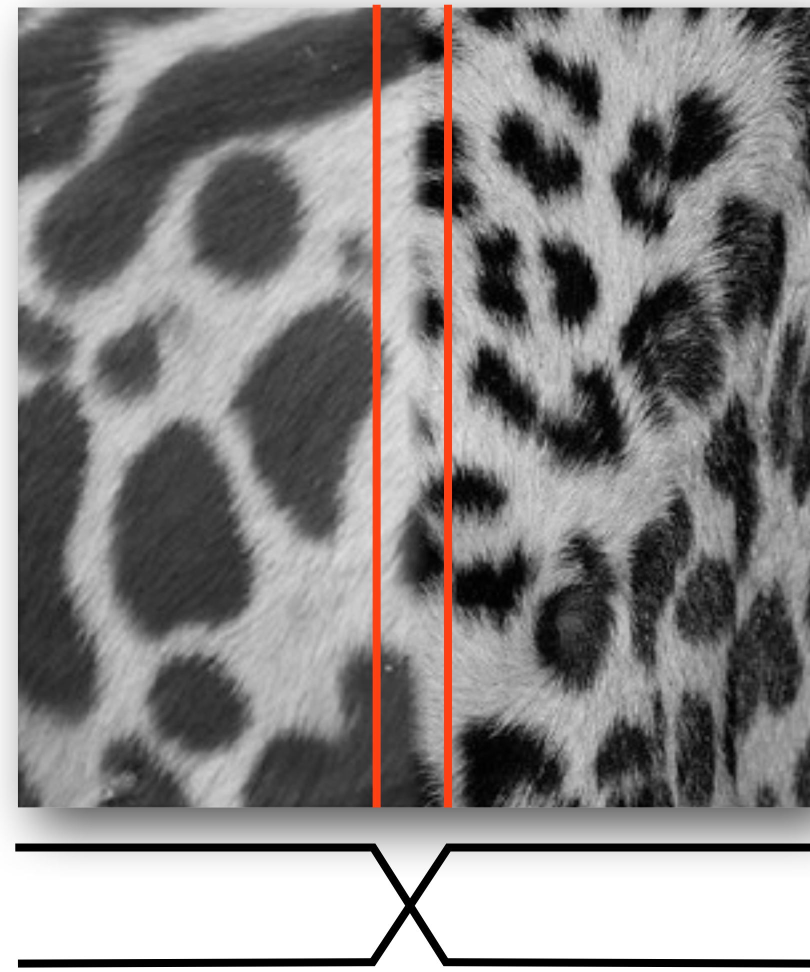
REVIEW: Optimal Window Size





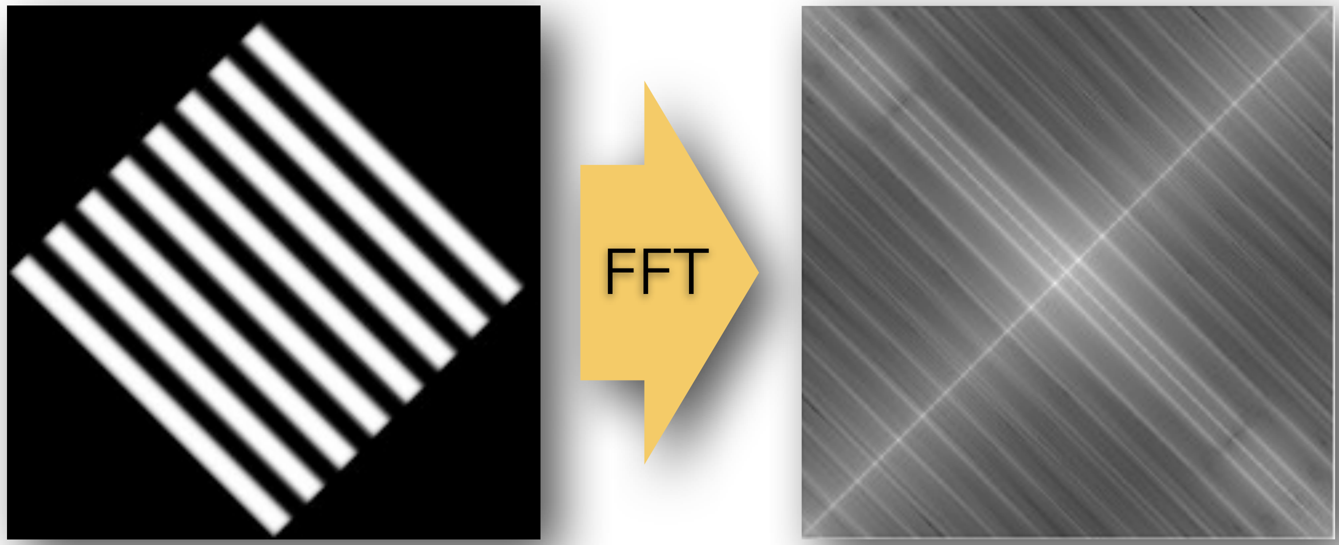
- ★ To avoid seams: Window = size of largest prominent "feature"
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent "feature"

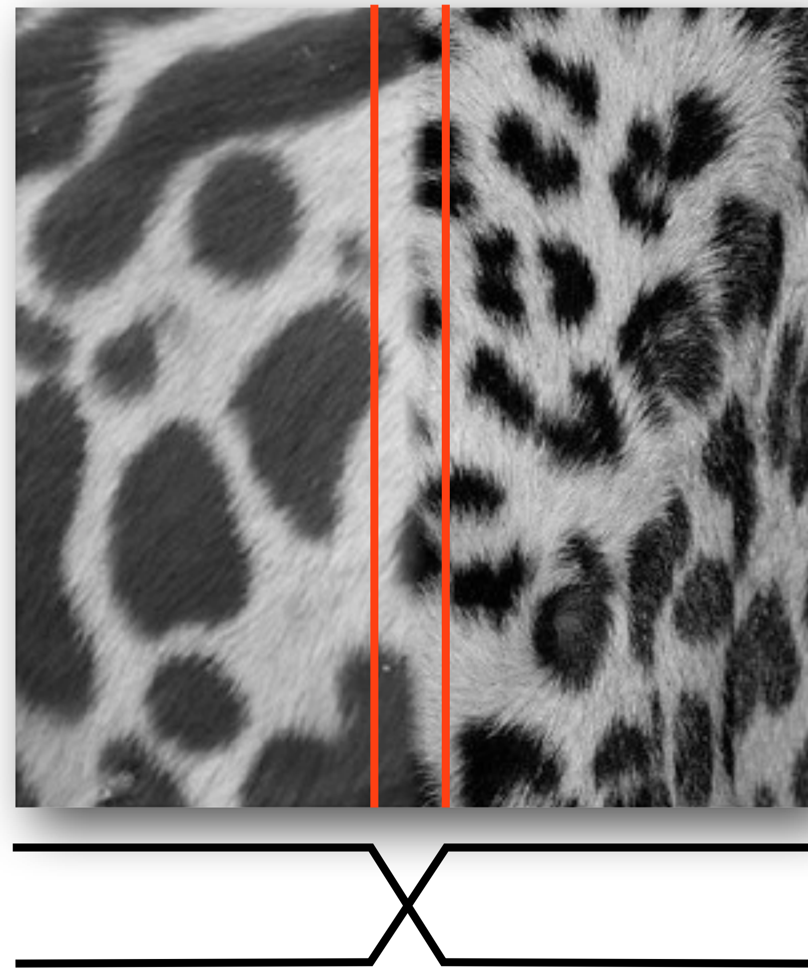
REVIEW: Optimal Window Size



- ★ To avoid seams: Window = size of largest prominent “feature”
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent “feature”

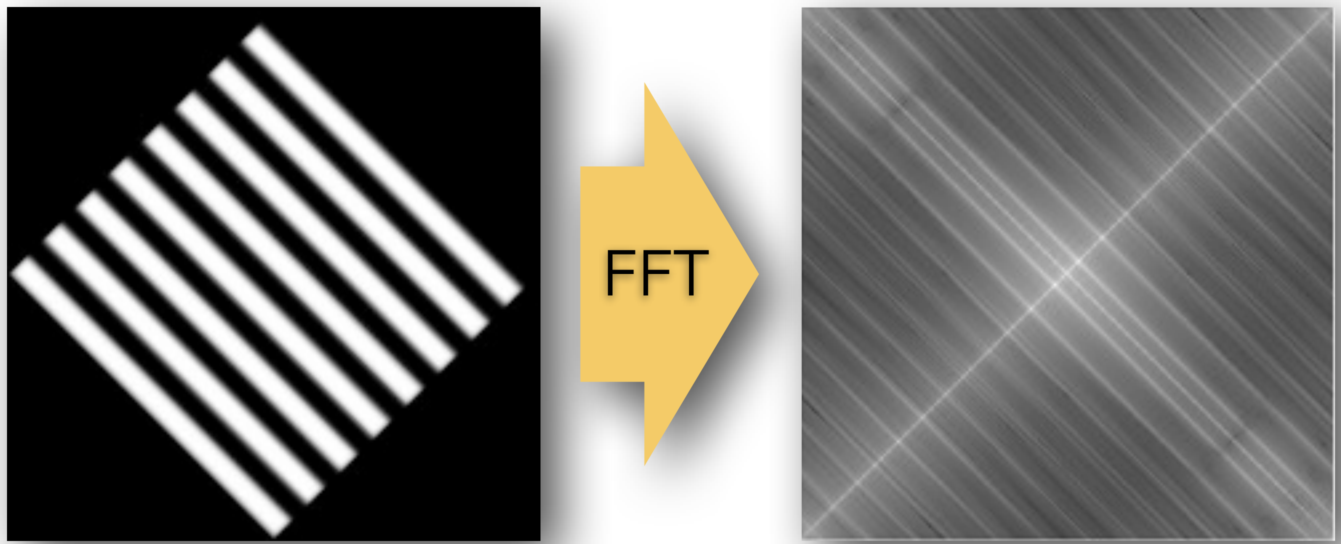
REVIEW: Optimal Window Size

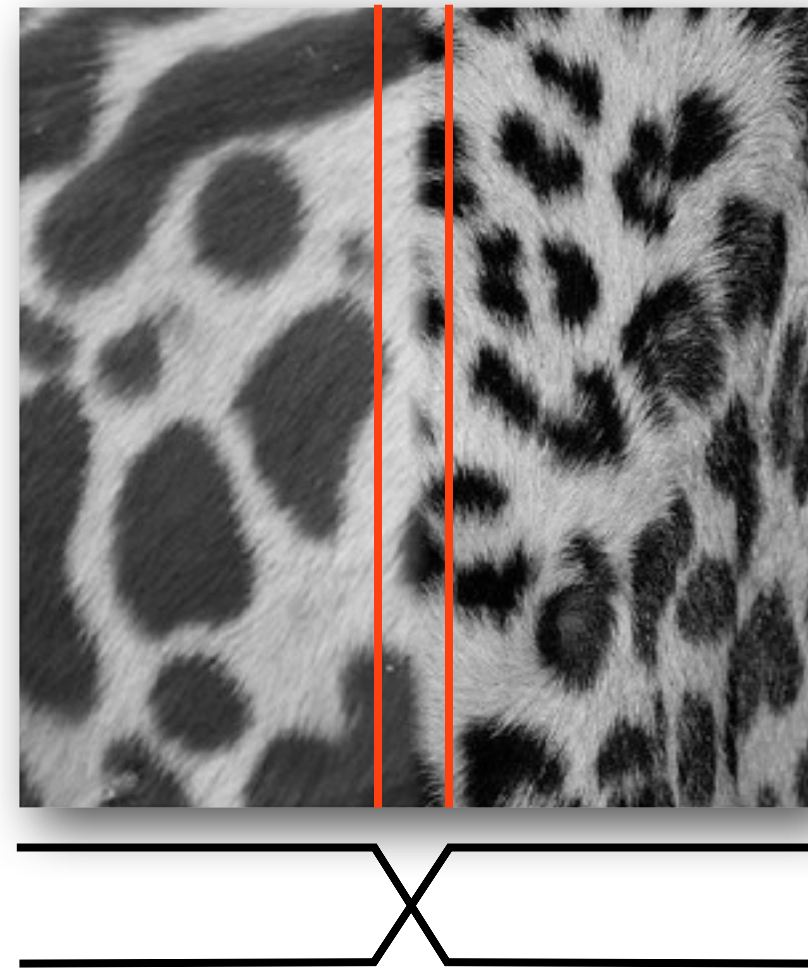




- ★ To avoid seams: Window = size of largest prominent "feature"
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent "feature"

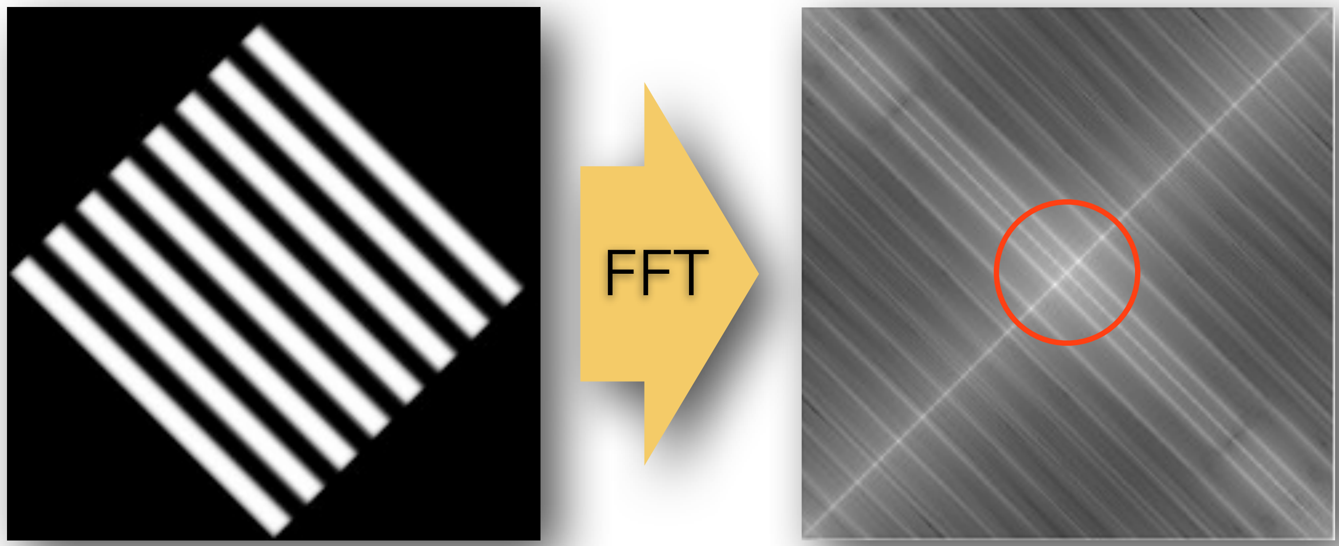
REVIEW: Optimal Window Size

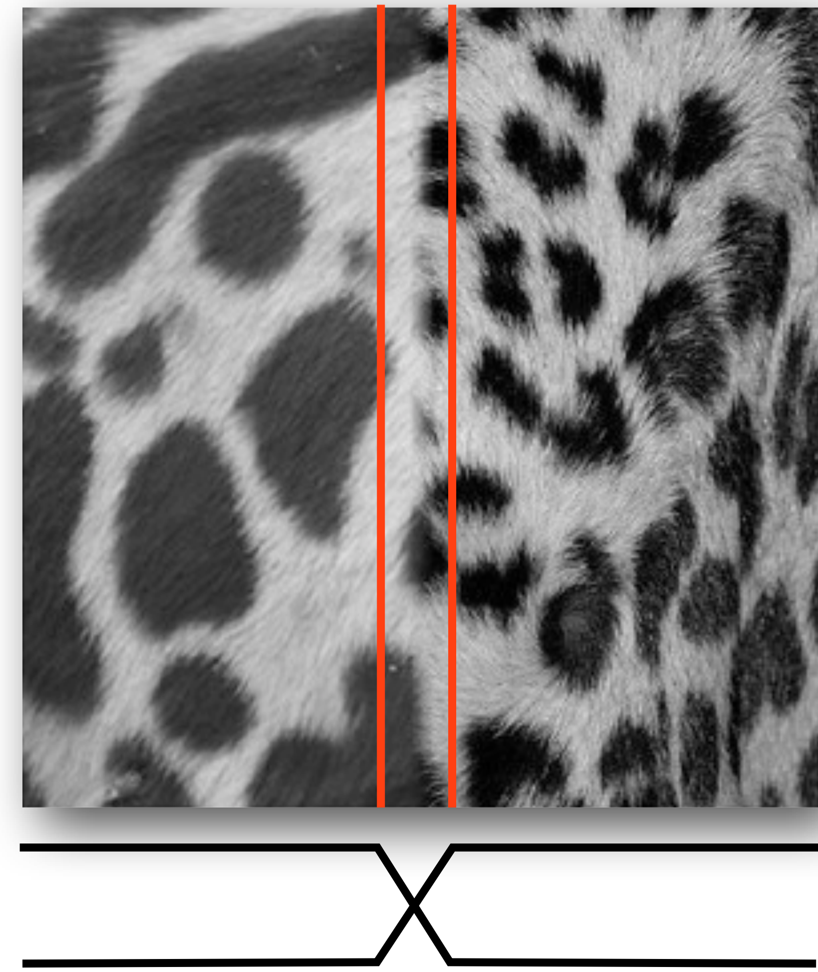




- ★ To avoid seams: Window = size of largest prominent "feature"
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent "feature"
- ★ Use Fourier domain
  - Largest frequency ≤ 2 × size of smallest frequency
  - Image frequency content should occupy one "octave" (power of two)

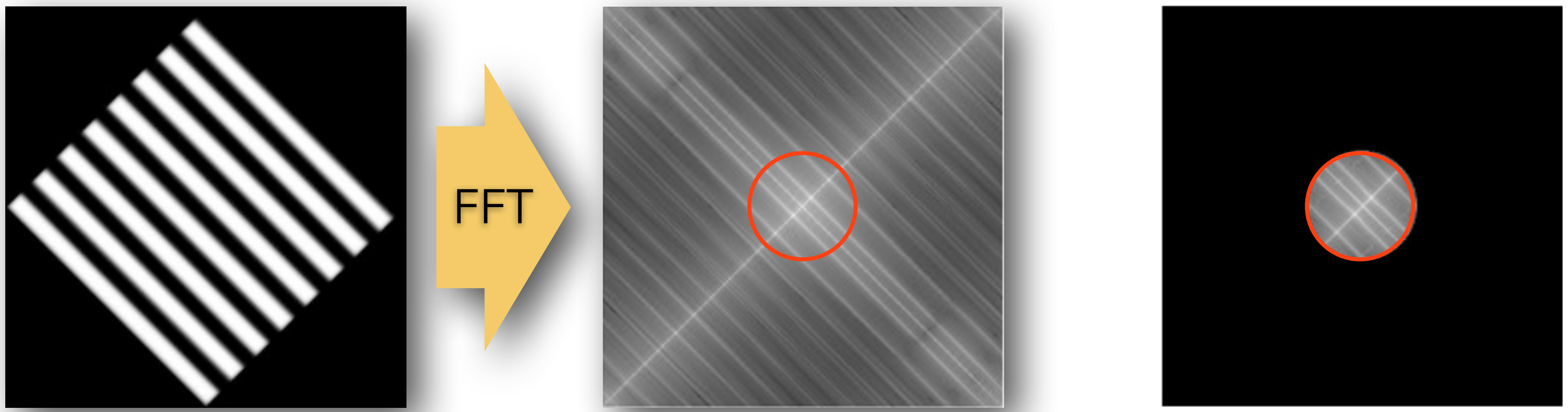
REVIEW: Optimal Window Size

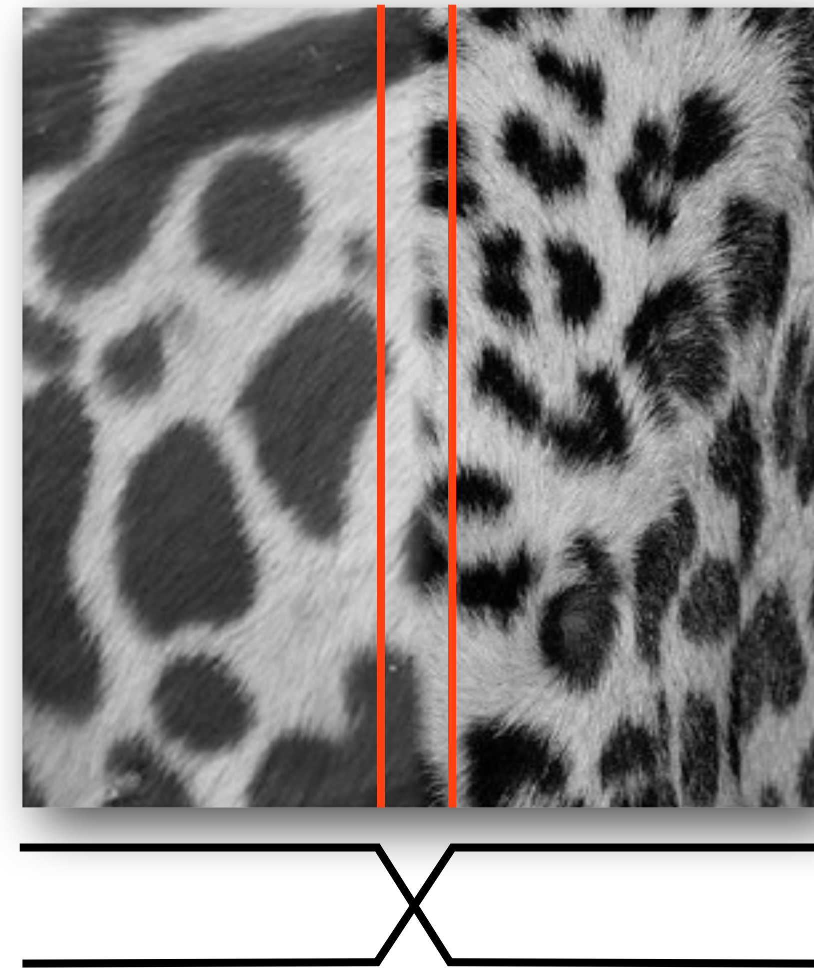




- ★ To avoid seams: Window = size of largest prominent "feature"
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent "feature"
- ★ Use Fourier domain
  - Largest frequency ≤ 2 × size of smallest frequency
  - Image frequency content should occupy one "octave" (power of two)

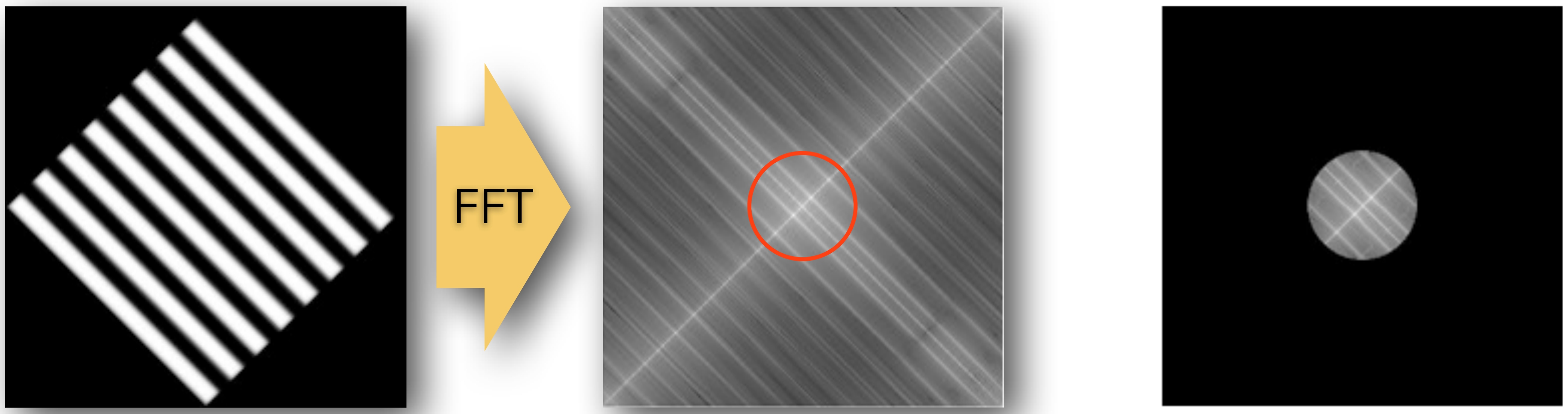
REVIEW: Optimal Window Size

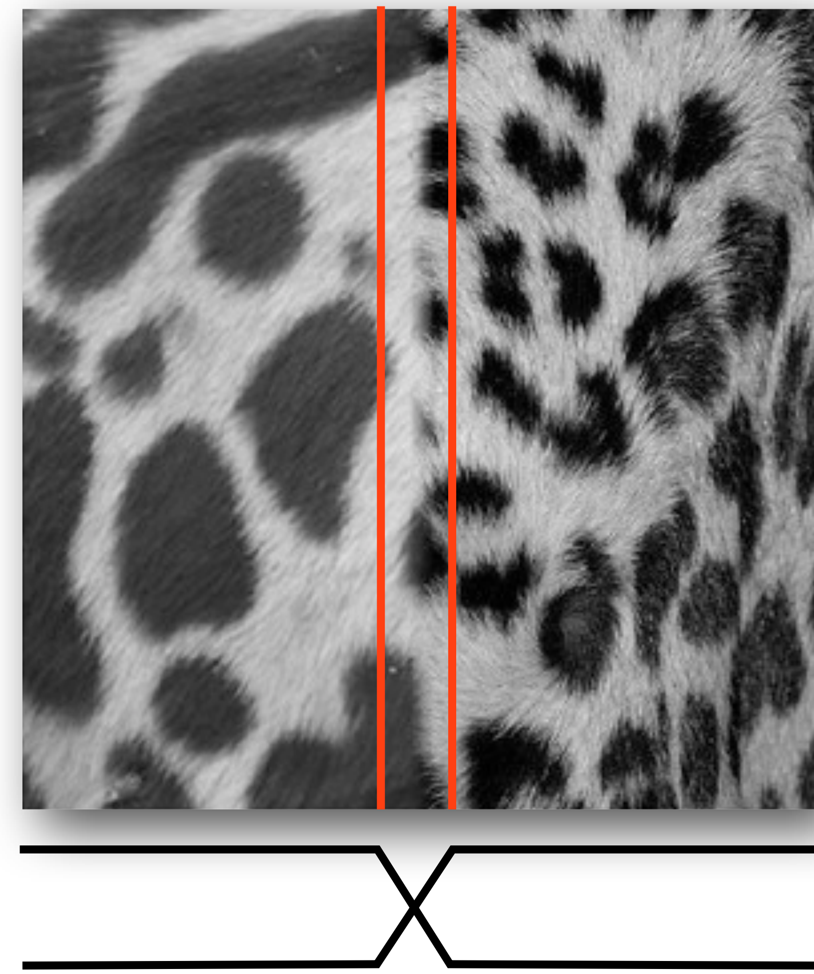




- To avoid seams: Window = size of largest prominent "feature"
- To avoid ghosting: Window ≤ 2×size of smallest prominent "feature"
- Use Fourier domain
  - Largest frequency ≤ 2 × size of smallest frequency
  - Image frequency content should occupy one "octave" (power of two)

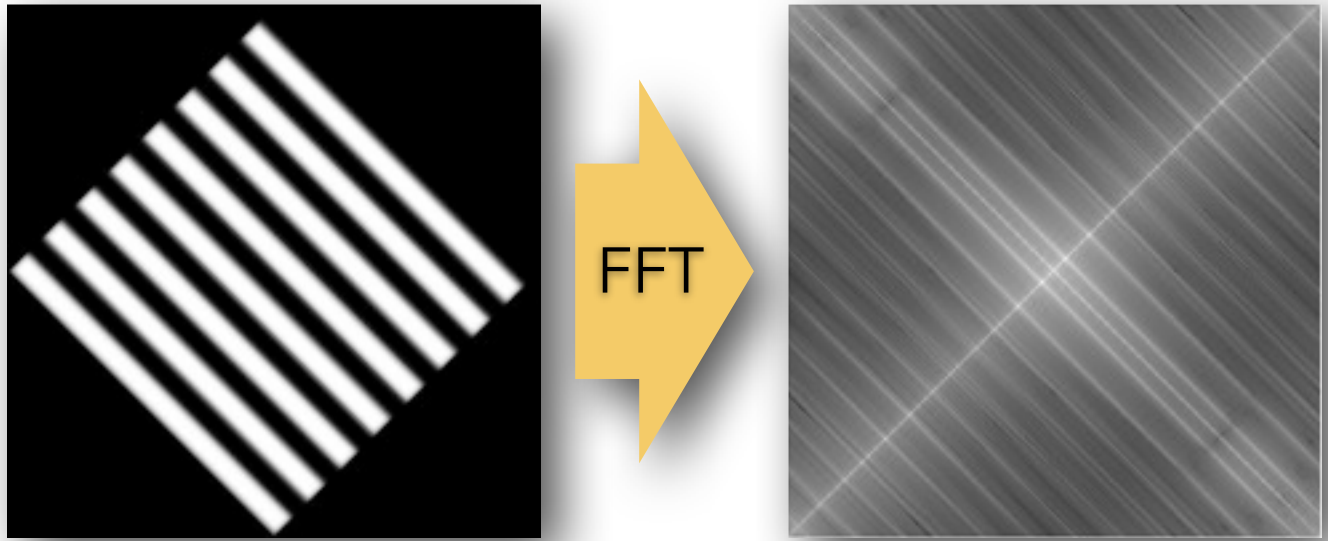
REVIEW: Optimal Window Size



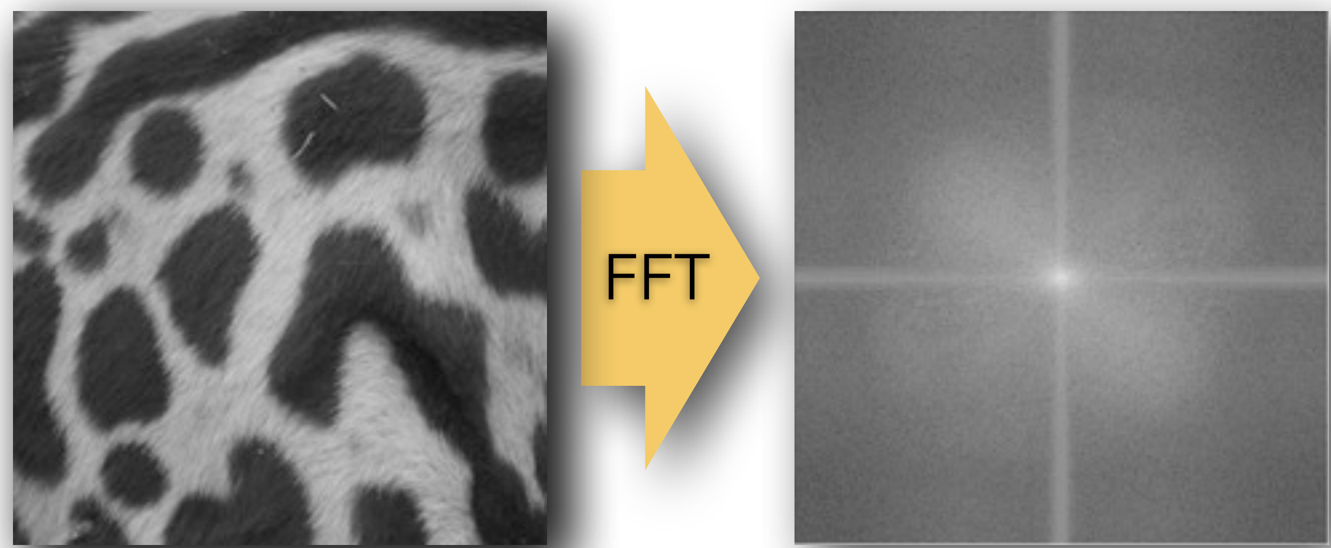


- ★ To avoid seams: Window = size of largest prominent "feature"
- ★ To avoid ghosting: Window ≤ 2×size of smallest prominent "feature"
- ★ Use Fourier domain
  - Largest frequency ≤ 2 × size of smallest frequency
  - Image frequency content should occupy one "octave" (power of two)

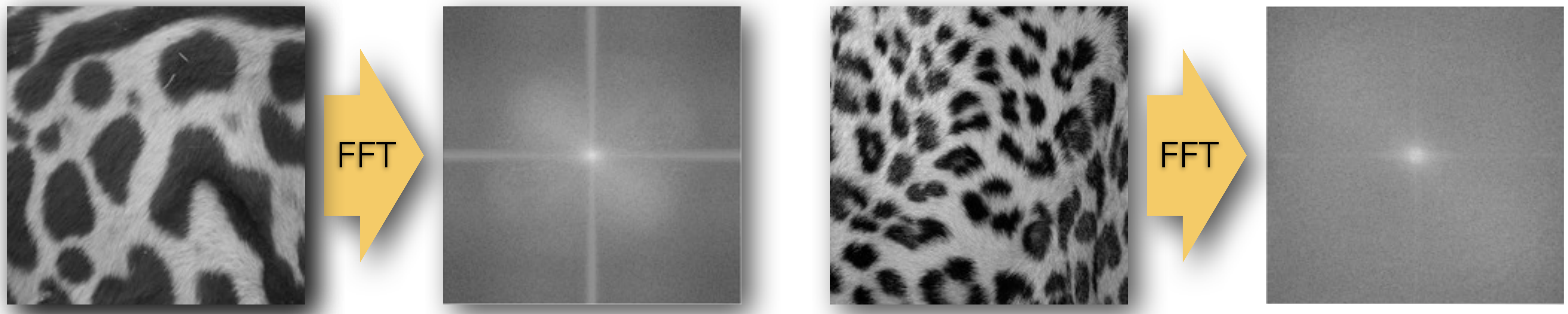
Frequency Spread needs to be Modeled



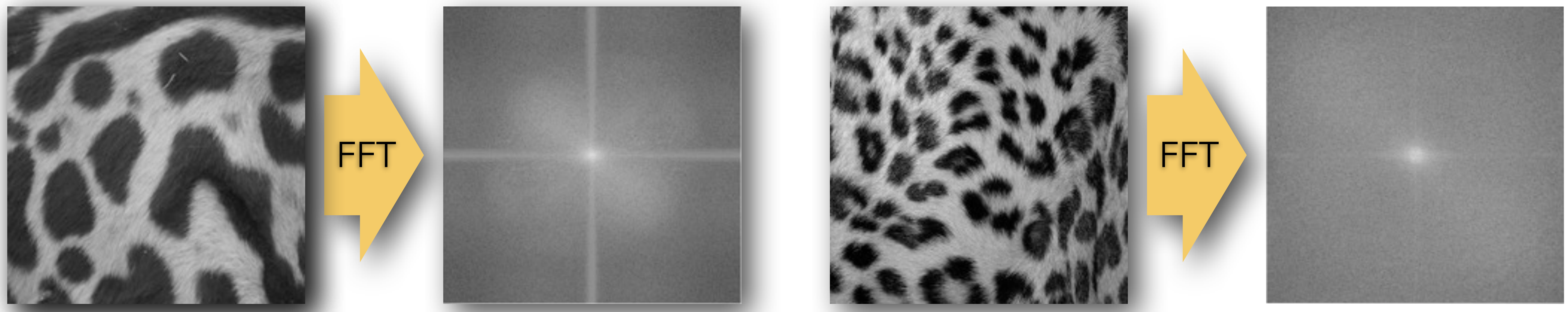
Frequency Spread needs to be Modeled



Frequency Spread needs to be Modeled

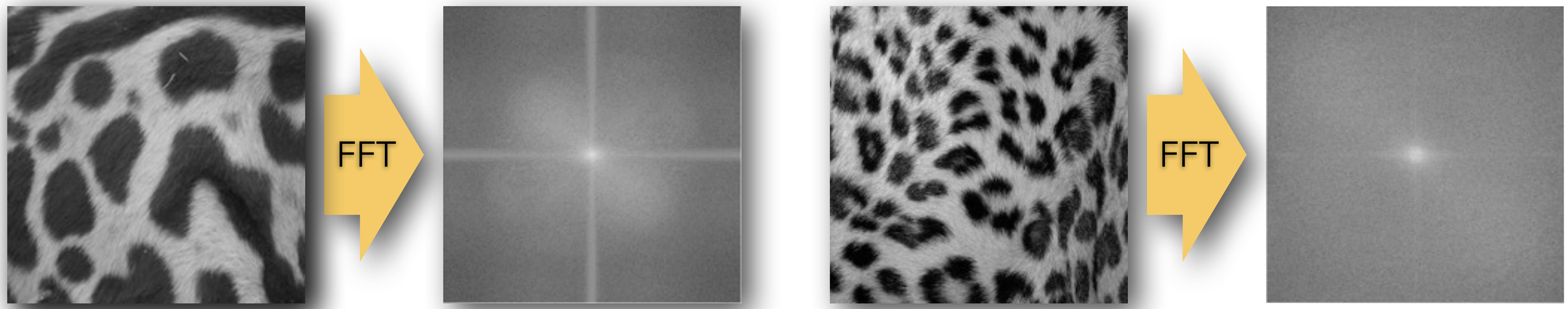


Frequency Spread needs to be Modeled



- ★ Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$

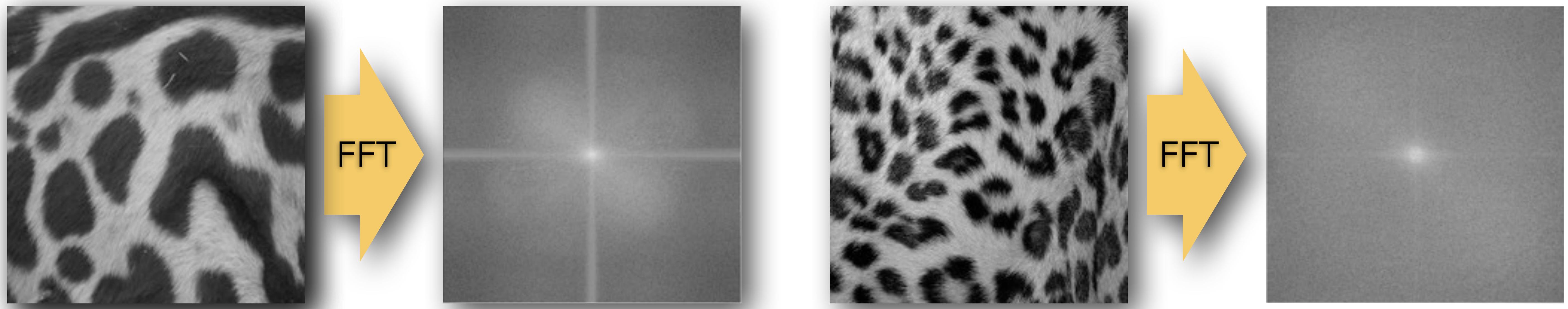
Frequency Spread needs to be Modeled



- Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$
- Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \ldots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \ldots$$

Frequency Spread needs to be Modeled

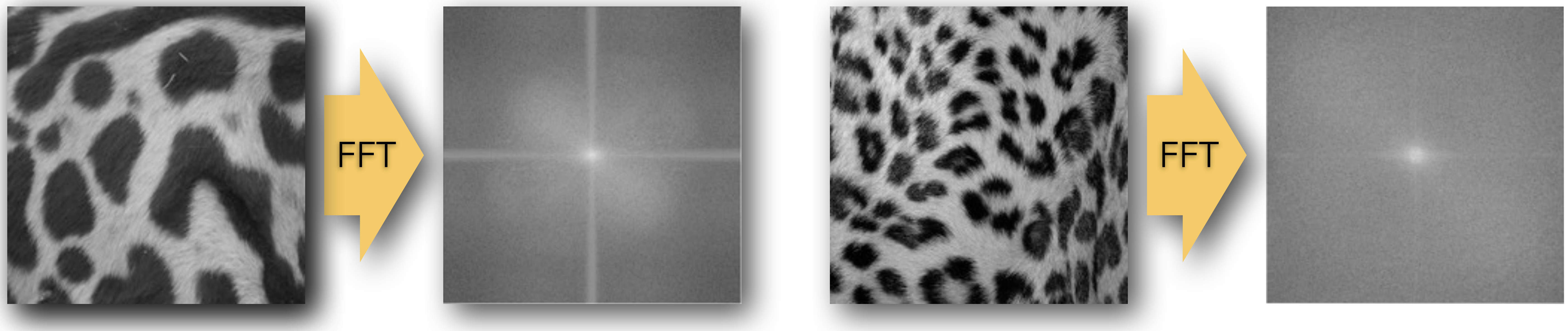


- Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$
- Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \ldots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \ldots$$

- "Feather" corresponding octaves of: $F_l \quad F_r$

Frequency Spread needs to be Modeled

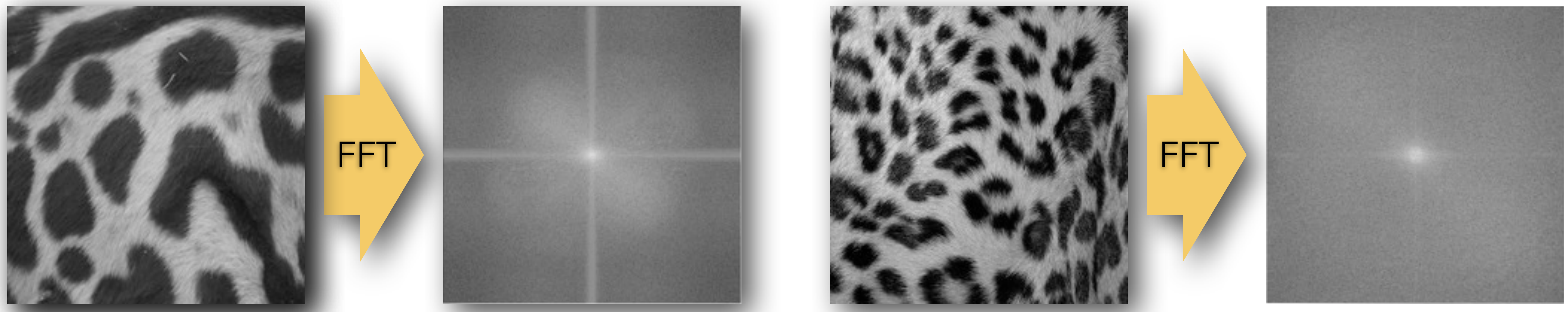


- ★ Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$
- ★ Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \ldots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \ldots$$

- ★ "Feather" corresponding octaves of: $F_l \quad F_r$
- ★ Compute inverse FFT and feather in spatial domain

Frequency Spread needs to be Modeled

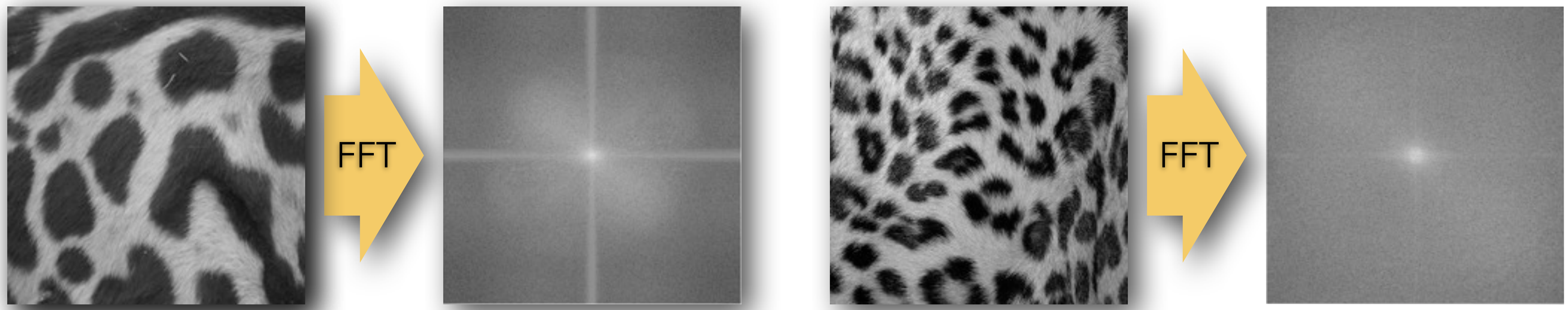


- Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$
- Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \dots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \dots$$

- "Feather" corresponding octaves of: $F_l \quad F_r$
- Compute inverse FFT and feather in spatial domain
- Sum feathered octave images in frequency domain

Frequency Spread needs to be Modeled



- ★ Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$
- ★ Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \ldots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \ldots$$

- ★ "Feather" corresponding octaves of: $F_l \quad F_r$
- ★ Compute inverse FFT and feather in spatial domain
- ★ Sum feathered octave images in frequency domain
- ★ Burt and Adelson (1983)

g_0

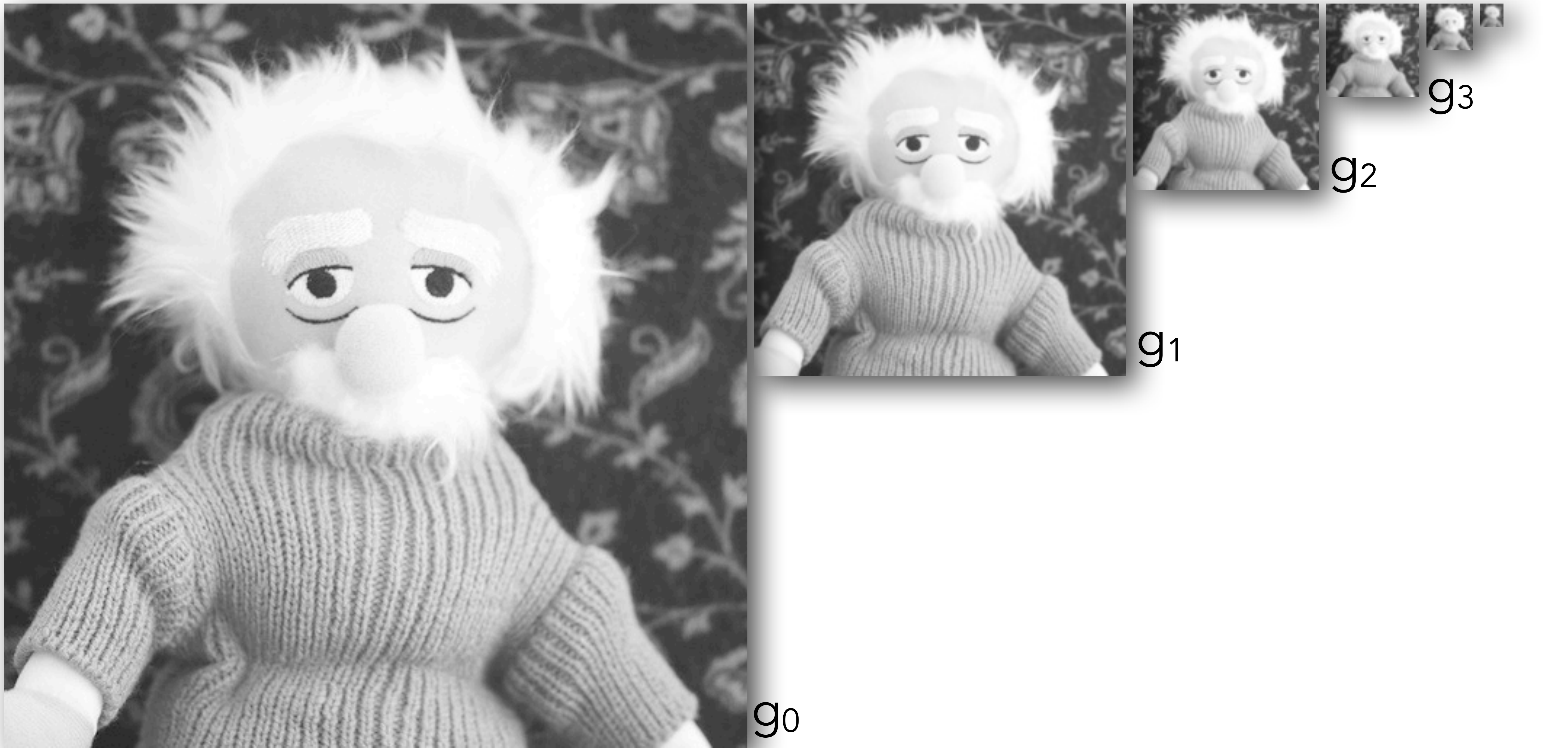
Pyramid Representation of Images (A Gaussian Pyramid)

g_0

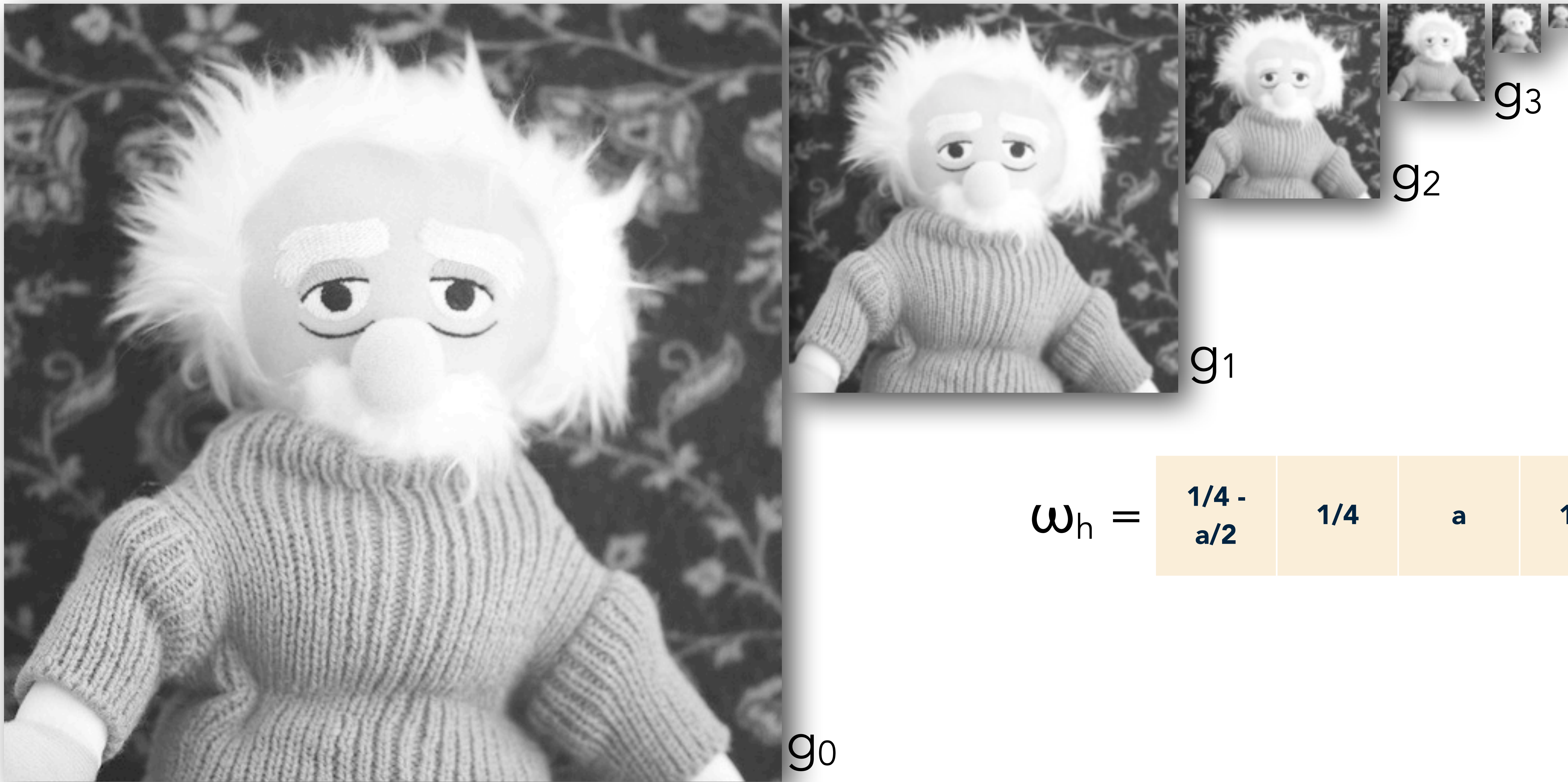
g_1

Pyramid Representation of Images (A Gaussian Pyramid)

Pyramid Representation of Images (A Gaussian Pyramid)



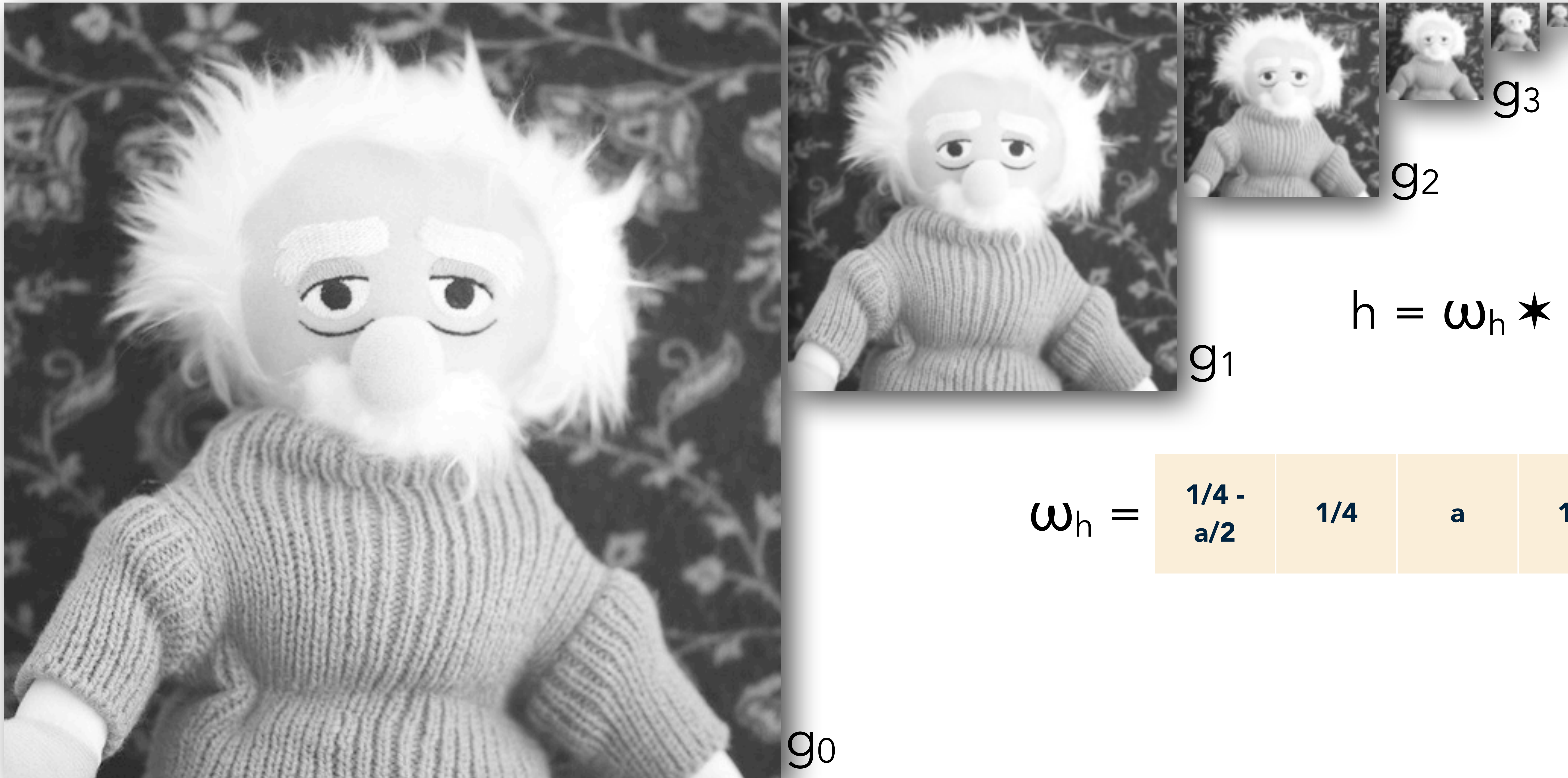
Pyramid Representation of Images
(A Gaussian Pyramid)



$$\omega_v = \begin{bmatrix} \frac{1}{4} - \frac{a}{2} \\ \frac{1}{4} \\ a \\ \frac{1}{4} \\ \frac{1}{4} - \frac{a}{2} \end{bmatrix}$$

$$\omega_h = \begin{bmatrix} \frac{1}{4} - \frac{a}{2} & \frac{1}{4} & a & \frac{1}{4} & \frac{1}{4} - \frac{a}{2} \end{bmatrix}$$

Pyramid Representation of Images (A Gaussian Pyramid)



$\omega_v =$

1/4 - a/2
1/4
a
1/4
1/4 - a/2

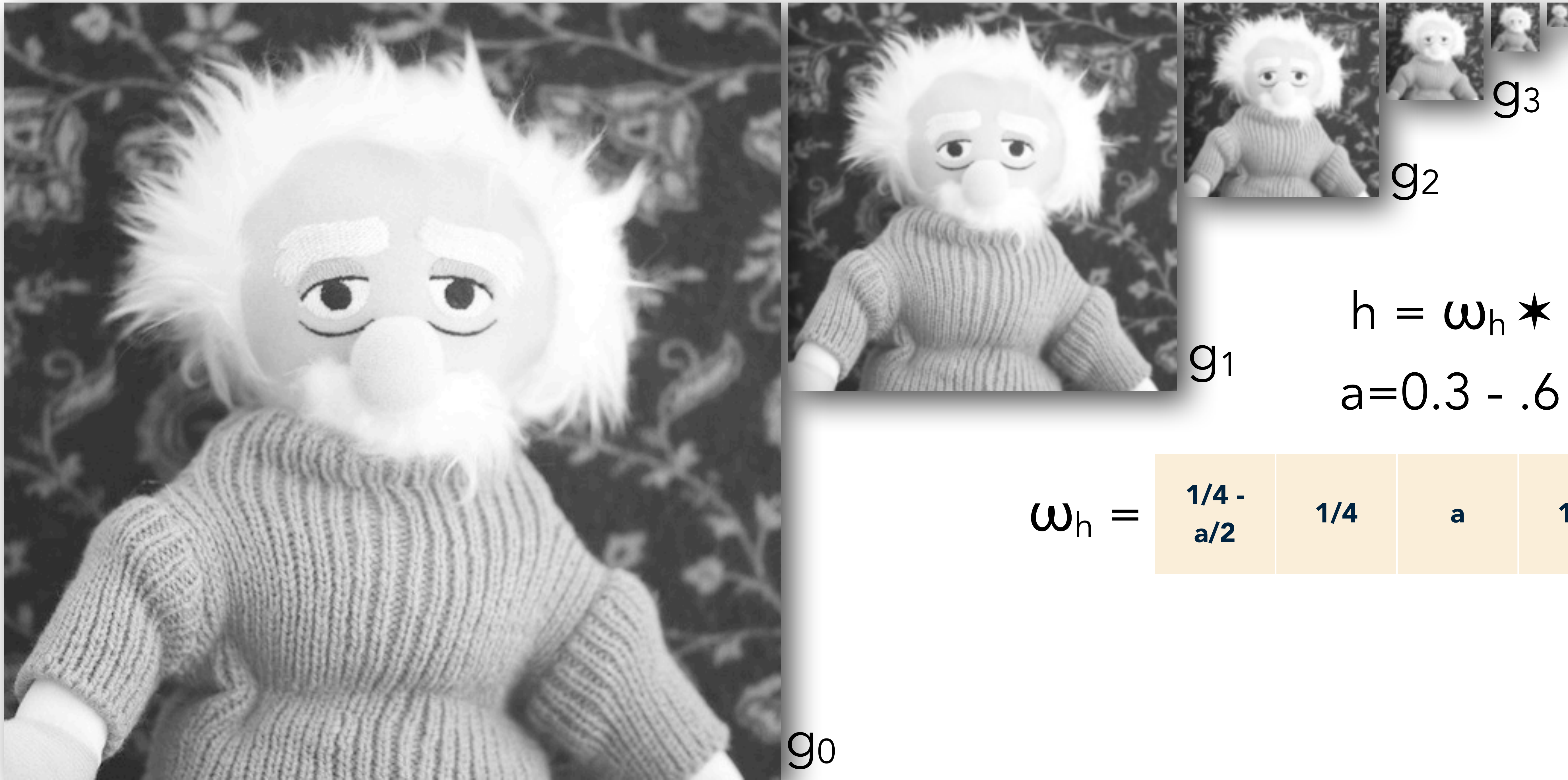
$$h = \omega_h \star \omega_v$$

$\omega_h =$

1/4 - a/2	1/4	a	1/4	1/4 - a/2

Pyramid Representation of Images (A Gaussian Pyramid)

Pyramid Representation of Images (A Gaussian Pyramid)



$$h = \omega_h \star \omega_v$$

$$a = 0.3 - .6\ (.38)$$

$$\omega_v = \begin{bmatrix} 1/4 - a/2 \\ 1/4 \\ a \\ 1/4 \\ 1/4 - a/2 \end{bmatrix}$$

$\omega_h =$

1/4 - a/2	1/4	a	1/4	1/4 - a/2

g_0

g_1

g_2

g_3

$$\omega_v = \begin{bmatrix} 1/4 - a/2 \\ 1/4 \\ a \\ 1/4 \\ 1/4 - a/2 \end{bmatrix}$$

$$h = \omega_h \star \omega_v$$

$$a = 0.3 - .6\ (.38)$$

$$\omega_h = \begin{bmatrix} 1/4 - a/2 & 1/4 & a & 1/4 & 1/4 - a/2 \end{bmatrix}$$

$$g_k = h \star g_{k-1}$$

Pyramid Representation of Images (A Gaussian Pyramid)

$$\omega_v = \begin{bmatrix} 1/4 - a/2 \\ 1/4 \\ a \\ 1/4 \\ 1/4 - a/2 \end{bmatrix}$$

$$h = \omega_h \star \omega_v$$

$$a = 0.3 - .6\ (.38)$$

$$\omega_h = \begin{bmatrix} 1/4 - a/2 & 1/4 & a & 1/4 & 1/4 - a/2 \end{bmatrix}$$

$$g_k = h \star g_{k-1}$$

$$g_k = \text{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)

$$g_k = h \ast g_{k-1}$$

$$g_k = \mathrm{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)

g_1

$$g_k = h \star g_{k-1}$$

$$g_k = \text{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)

g_1 g_2

$$g_k = h \ast g_{k-1}$$

$$g_k = \text{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)

g_1 g_2 g_3

$$g_k = h \ast g_{k-1}$$

$$g_k = \mathrm{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)

g_1 g_2 g_3 g_4

$$g_k = h \ast g_{k-1}$$

$$g_k = \mathrm{REDUCE}(g_{k-1})$$

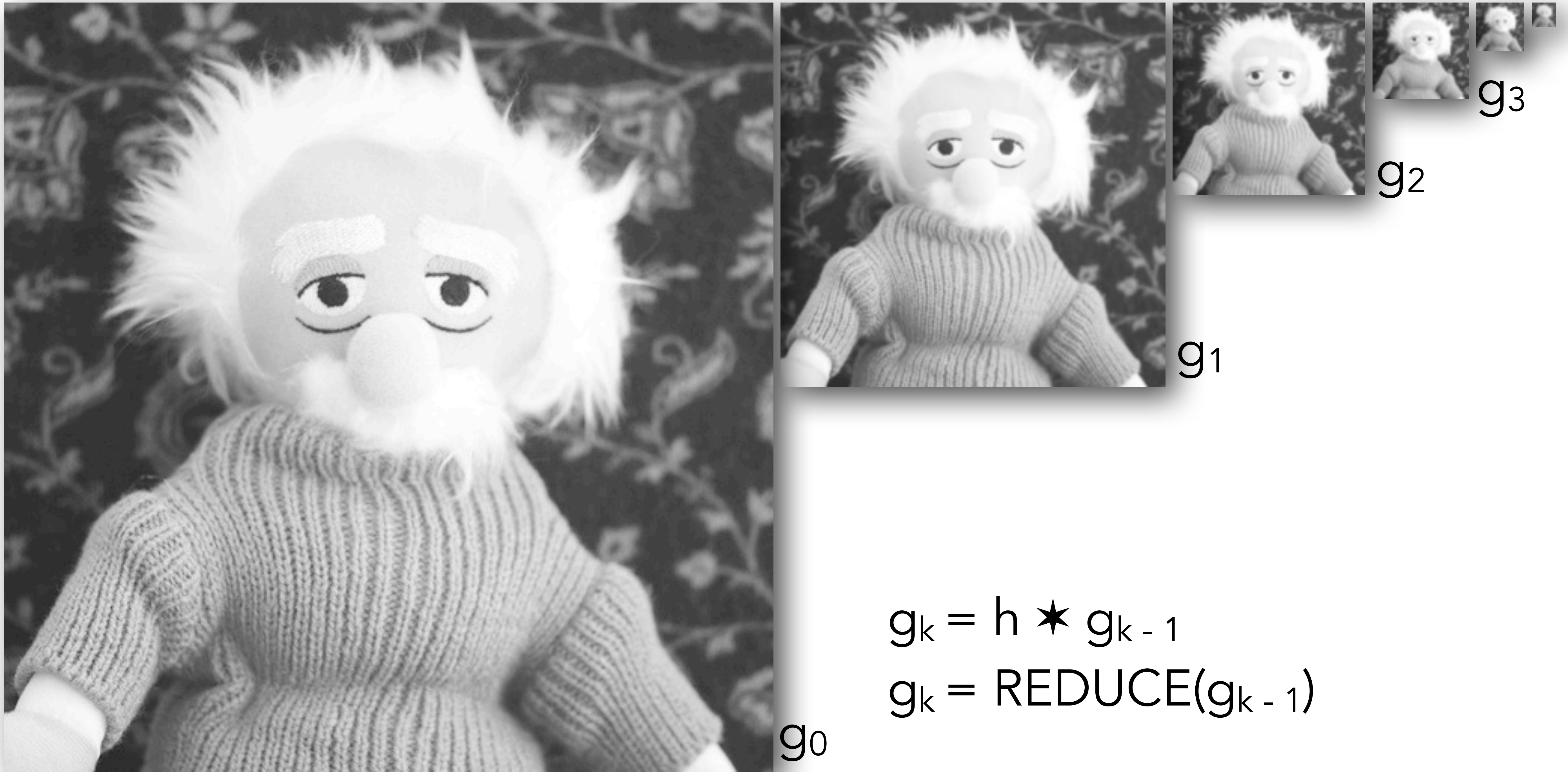
Pyramid Representation of Images (A Gaussian Pyramid)

g_1 g_2 g_3 g_4 g_5

$$g_k = h \star g_{k-1}$$

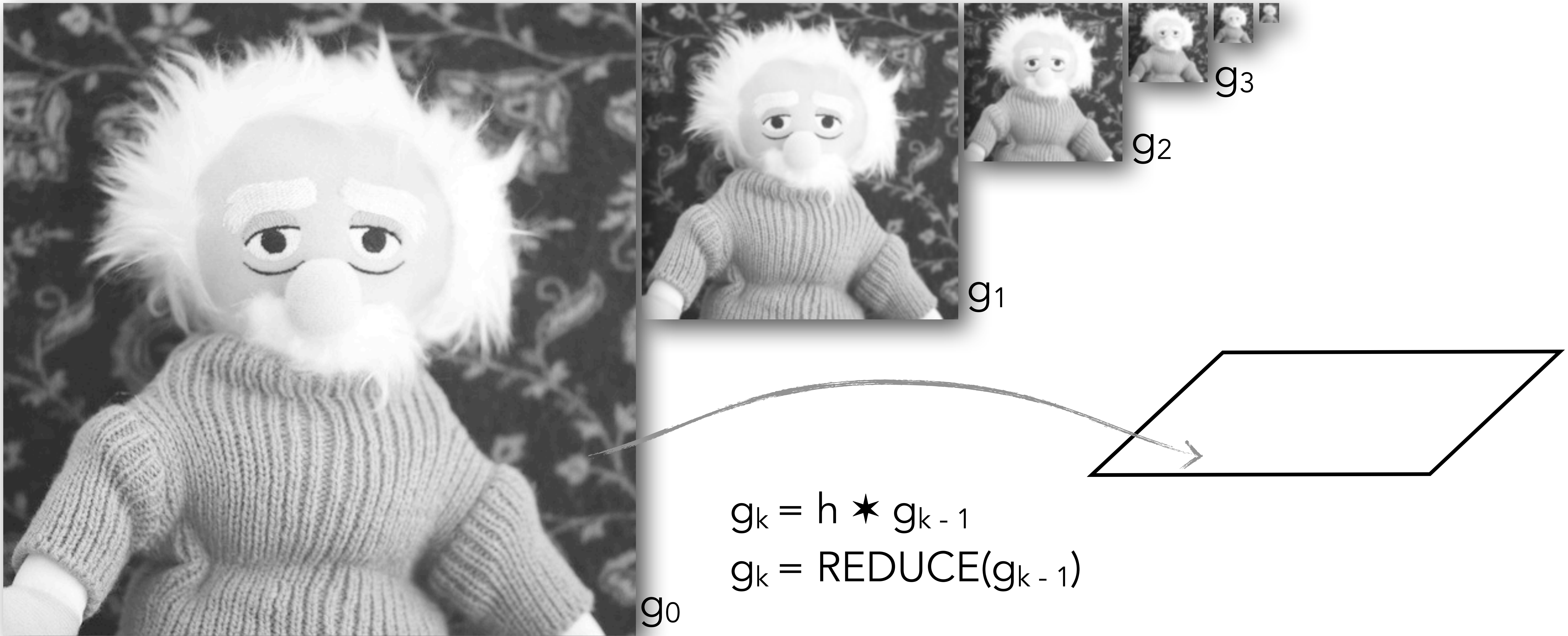
$$g_k = \mathrm{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)



$$g_k = h \star g_{k-1}$$
$$g_k = \mathrm{REDUCE}(g_{k-1})$$

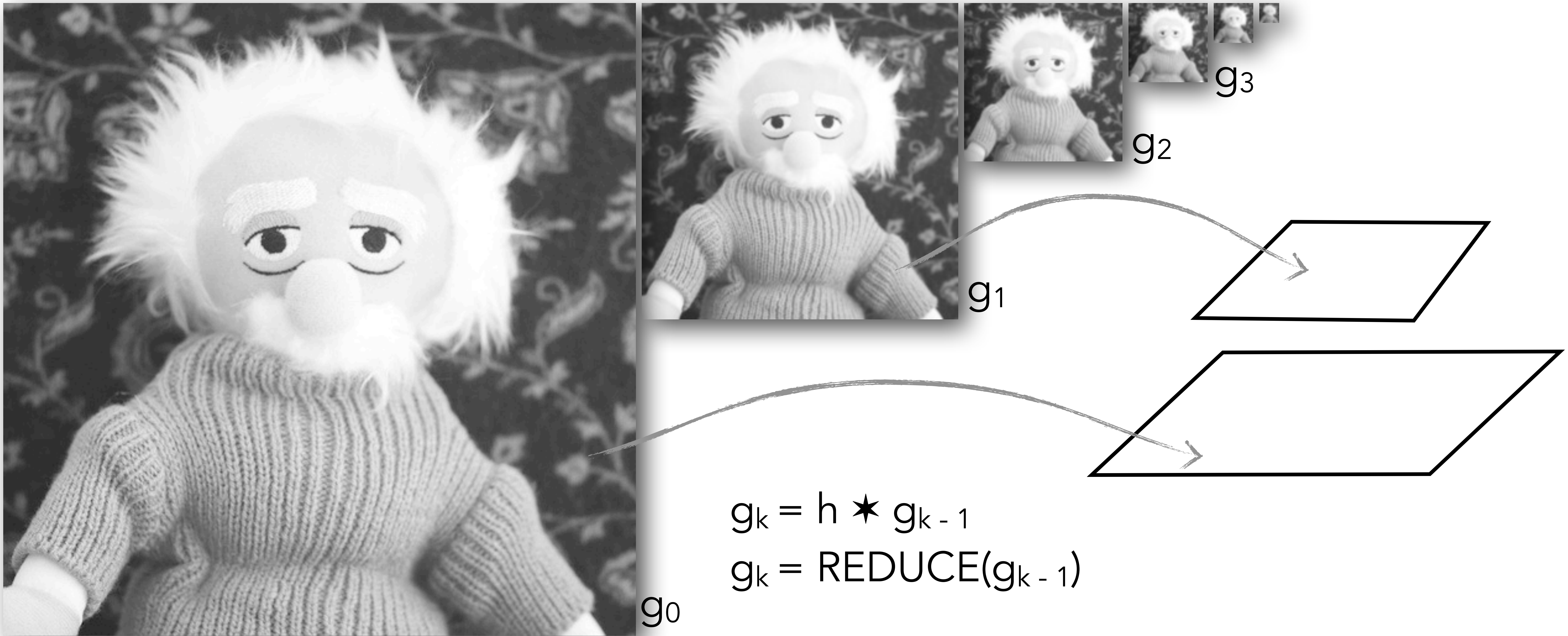
Pyramid Representation of Images (A Gaussian Pyramid)



$$g_k = h \star g_{k-1}$$

$$g_k = \mathrm{REDUCE}(g_{k-1})$$

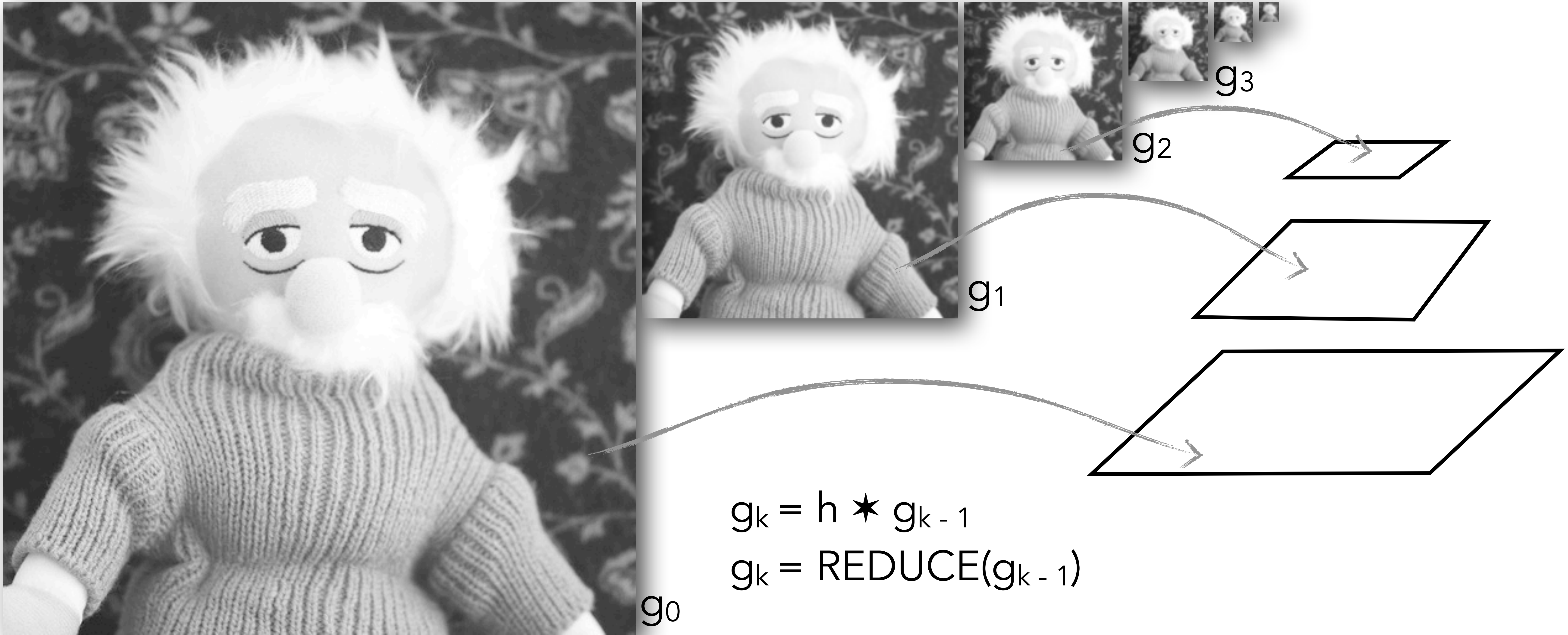
Pyramid Representation of Images (A Gaussian Pyramid)



$$g_k = h \star g_{k-1}$$

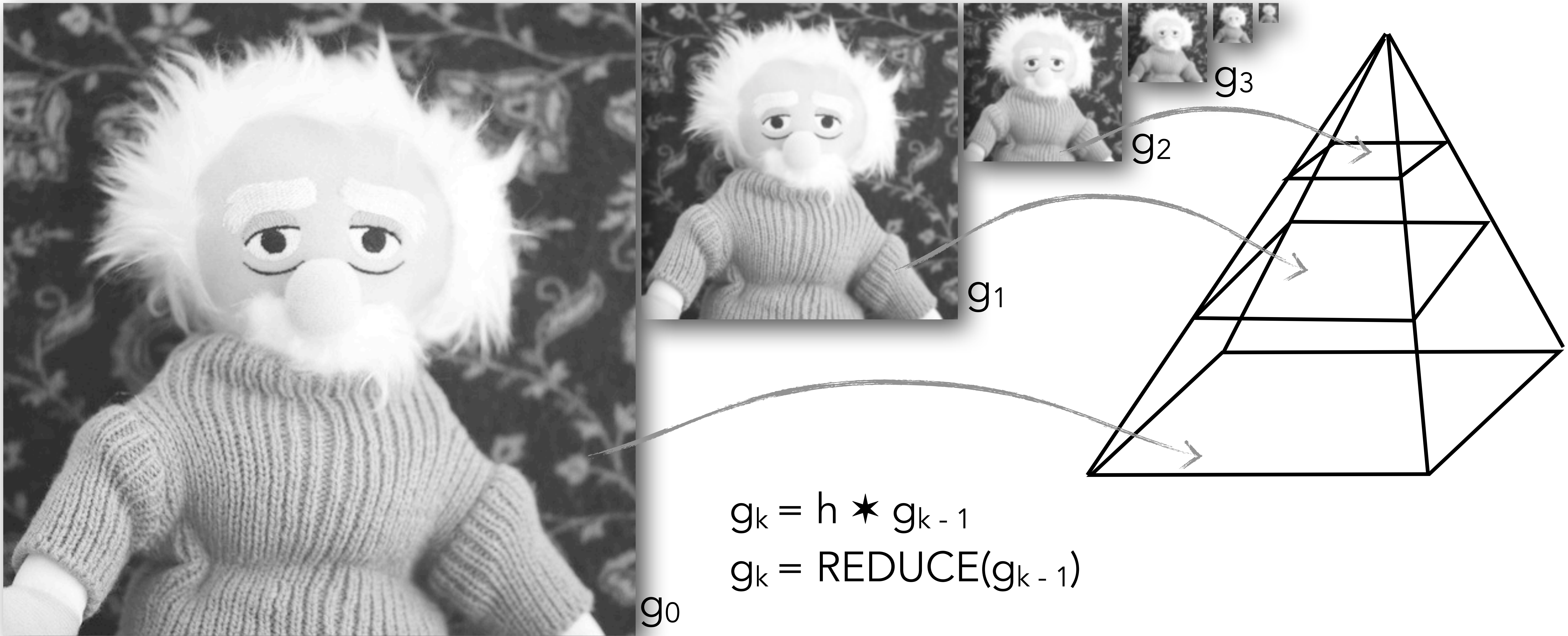
$$g_k = \text{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)



$$g_k = h \star g_{k-1}$$
$$g_k = \mathrm{REDUCE}(g_{k-1})$$

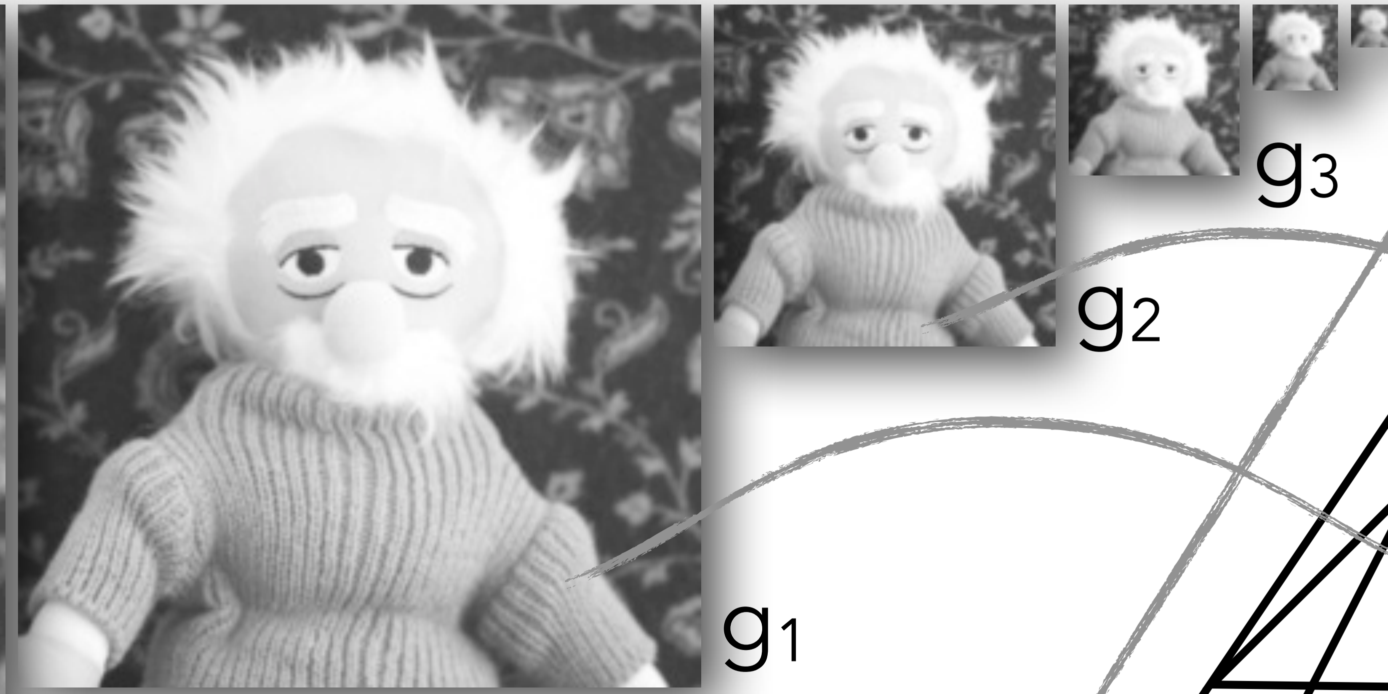
Pyramid Representation of Images (A Gaussian Pyramid)

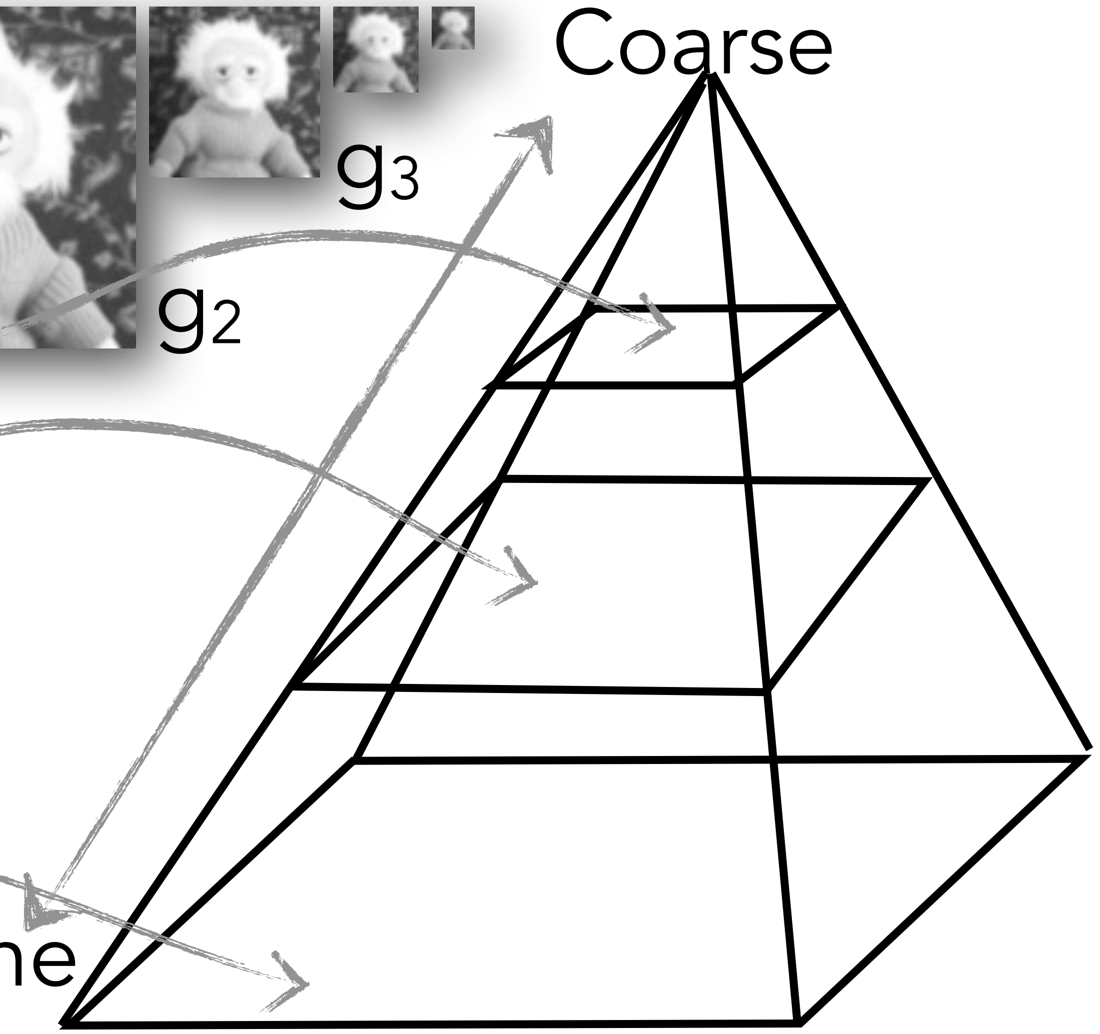


$$g_k = h \star g_{k-1}$$

$$g_k = \mathrm{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)





$$g_k = h \star g_{k-1}$$
$$g_k = \mathrm{REDUCE}(g_{k-1})$$

Pyramid Representation of Images (A Gaussian Pyramid)

$g_k = \mathrm{REDUCE}(g_{k-1})$

g_0

g_1

Pyramid Representation of Images (A Gaussian Pyramid)

g_0

g_1

$g_k = \mathrm{REDUCE}(g_{k-1})$

$g_{l,n} = \mathrm{EXPAND}(g_l, n-1)$

Pyramid Representation of Images (A Gaussian Pyramid)

g_0

g_1

$g_k = \text{REDUCE}(g_{k-1})$

$g_{l,n} = \text{EXPAND}(g_l, n\text{-}1)$

EXPAND is inverse of REVERSE, as it seeks to add new values in between knowns ones. $g_{l,n}$ is g_l expanded n times

Pyramid Representation of Images (A Gaussian Pyramid)

g_1

$g_k = \text{REDUCE}(g_{k-1})$

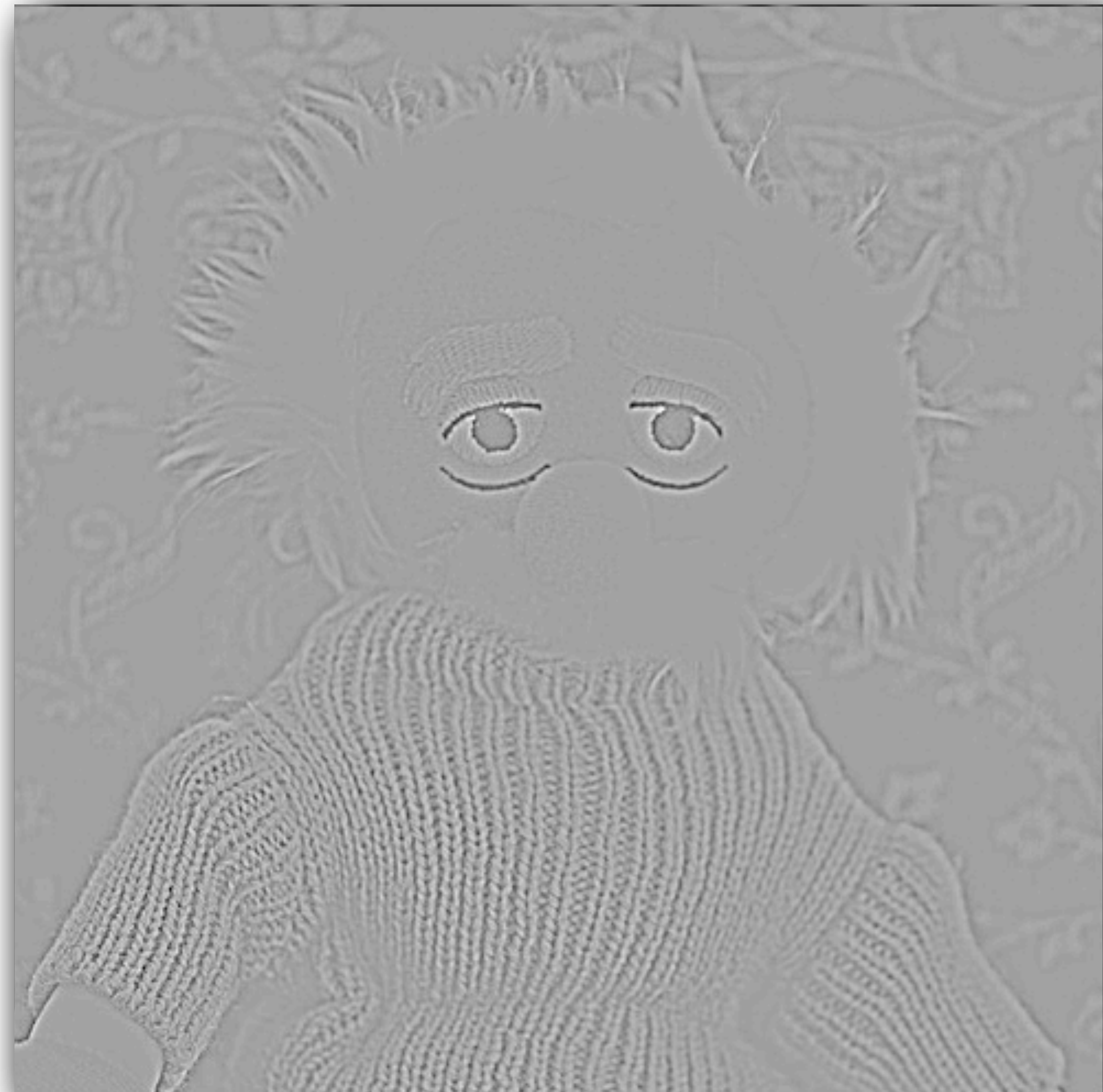
$g_{l,n} = \text{EXPAND}(g_l, n\text{-}1)$

EXPAND is inverse of REVERSE, as it seeks to add new values in between knowns ones. $g_{l,n}$ is g_l expanded *n* times

$g_{0,1}$

g_0

Pyramid Representation of Images (A Gaussian Pyramid)



g_1

$g_k = \text{REDUCE}(g_{k-1})$

$g_{l,n} = \text{EXPAND}(g_l, n\text{-}1)$

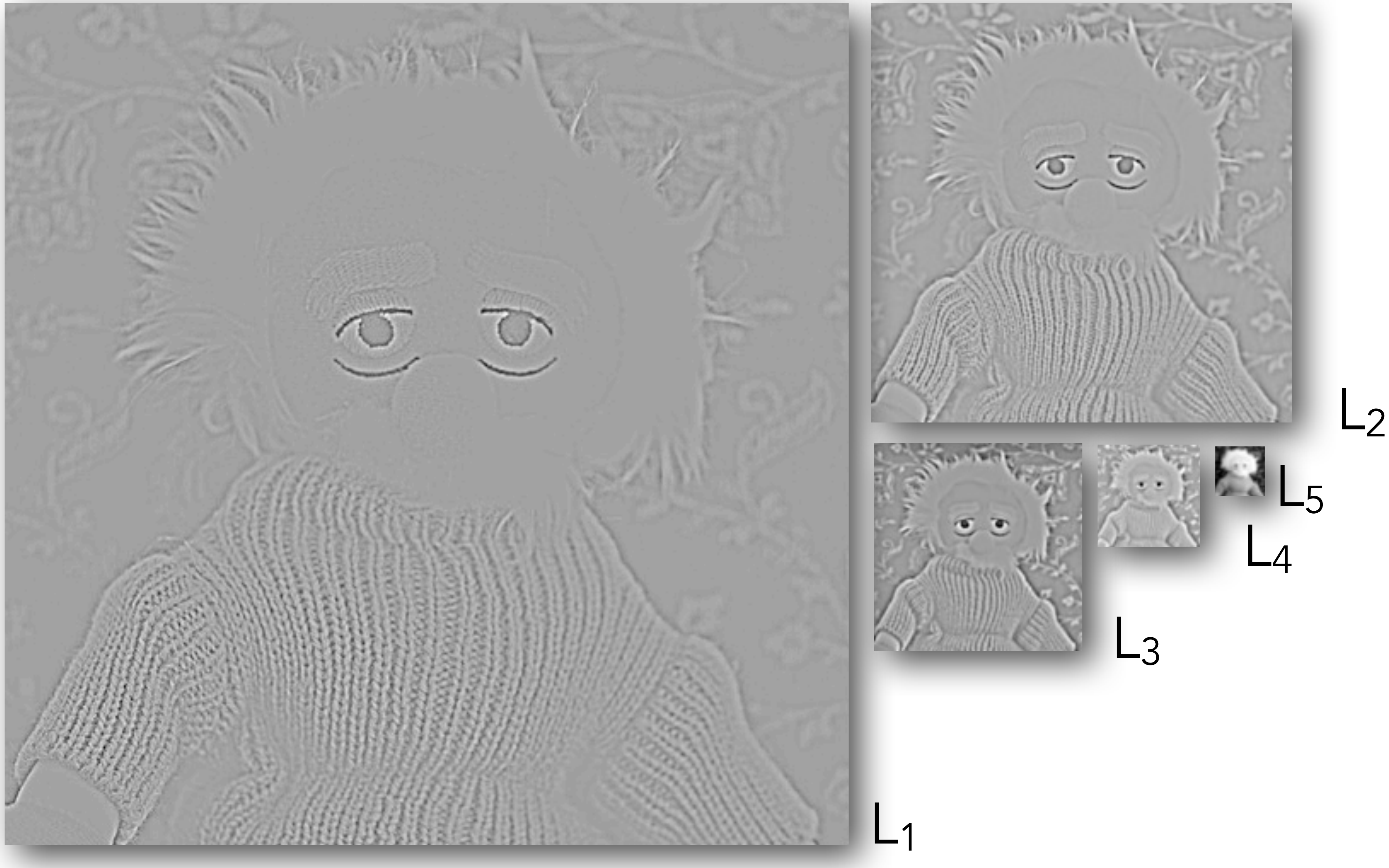
EXPAND is inverse of REVERSE, as it seeks to add new values in between knowns ones. $g_{l,n}$ is g_l expanded n times

$g_0 - g_{0,1}$

$g_{0,1}$

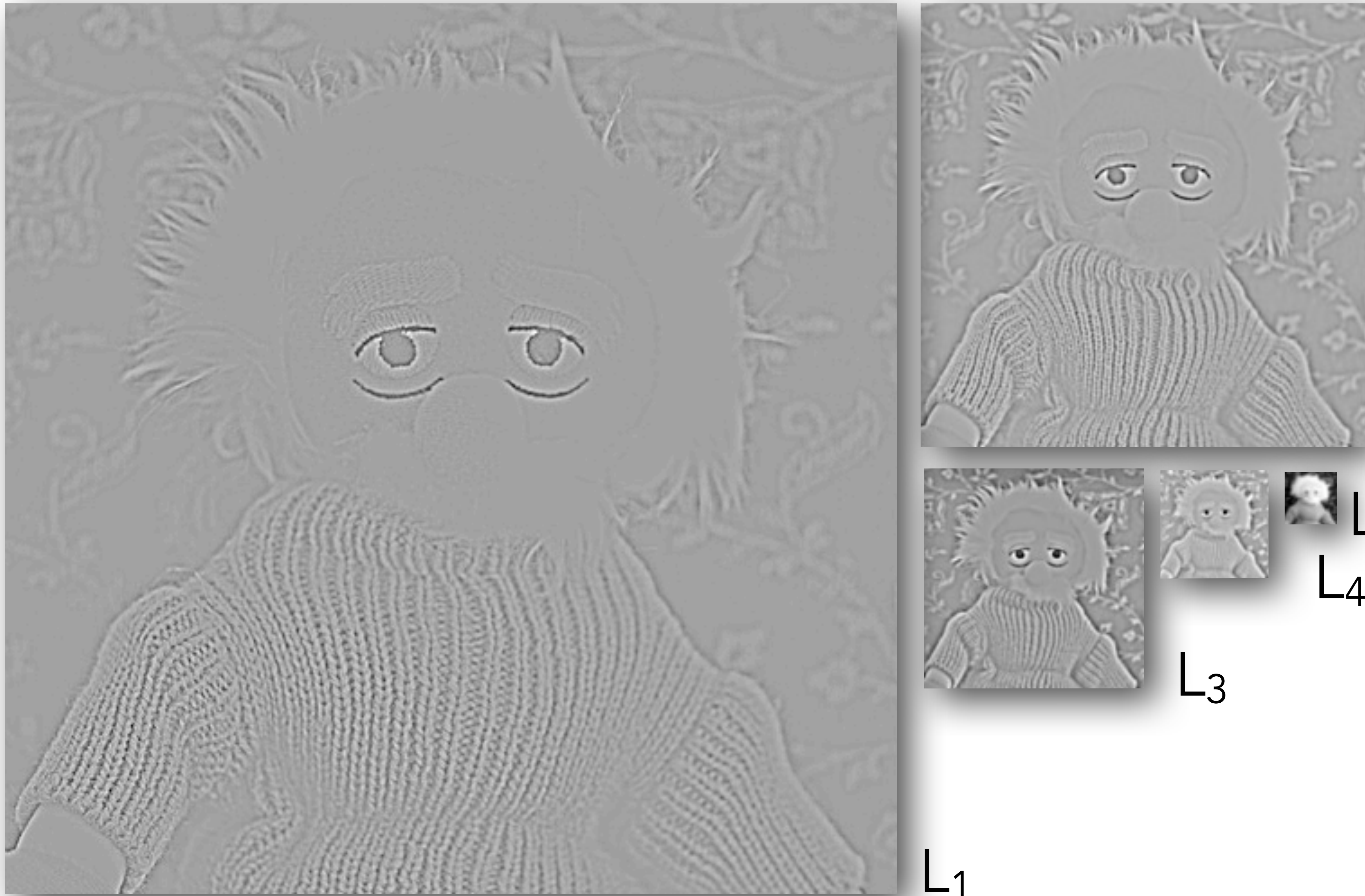
g_0

Pyramid Representation of Images (A Gaussian Pyramid)



- A Laplacian Pyramid is a series of "error" images, L_0, L_1, L_2, ...
- Each is a difference between two levels of a Gaussian Pyramid

Pyramid Representation of Images (A Laplacian Pyramid)



L_2 L_5 L_4 L_3 L_1

- A Laplacian Pyramid is a series of "error" images, L_0, L_1, L_2, ...
- Each is a difference between two levels of a Gaussian Pyramid

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

Pyramid Representation of Images (A Laplacian Pyramid)

10

g_1 $g_{1,1}$

L_1

Computing Gaussian and Laplacian Pyramids

Georgia Institute of Technology

g_1

$g_{1,1}$

L_1

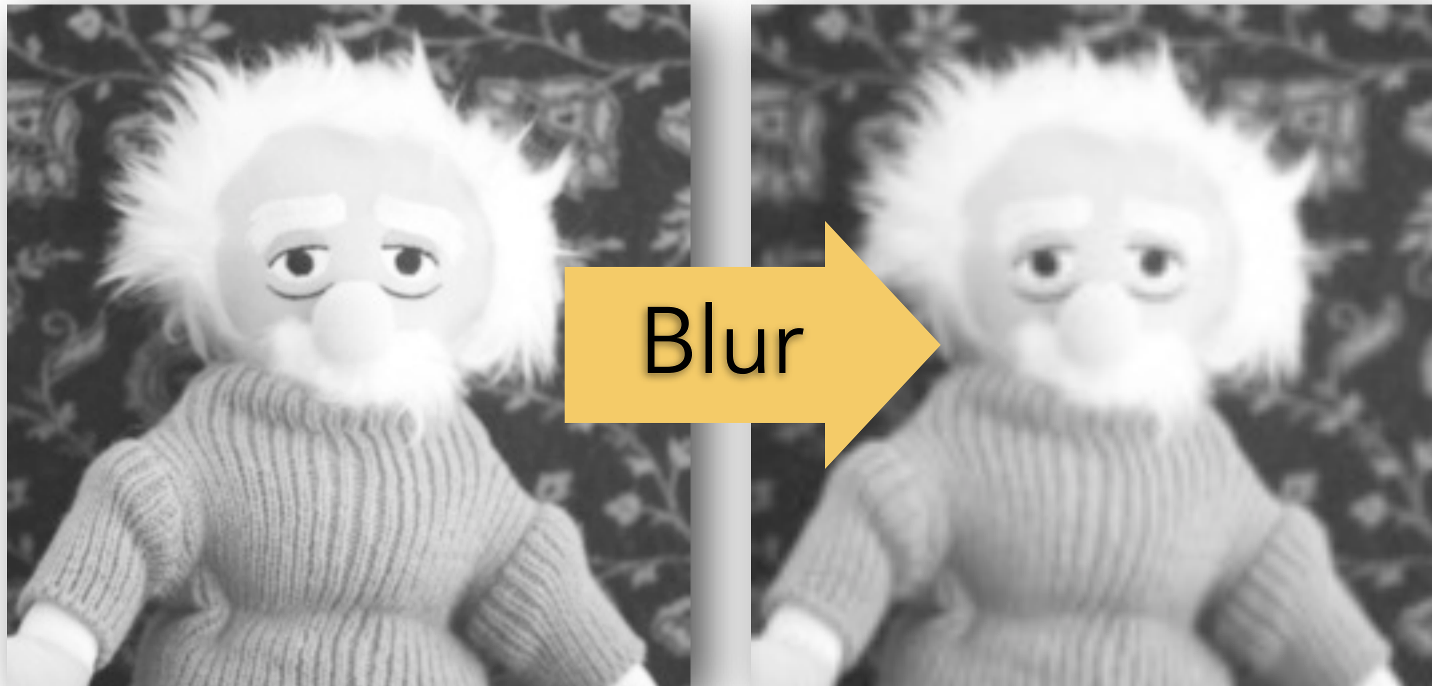
Computing Gaussian and Laplacian Pyramids

g_1

$g_{1,1}$

L_1

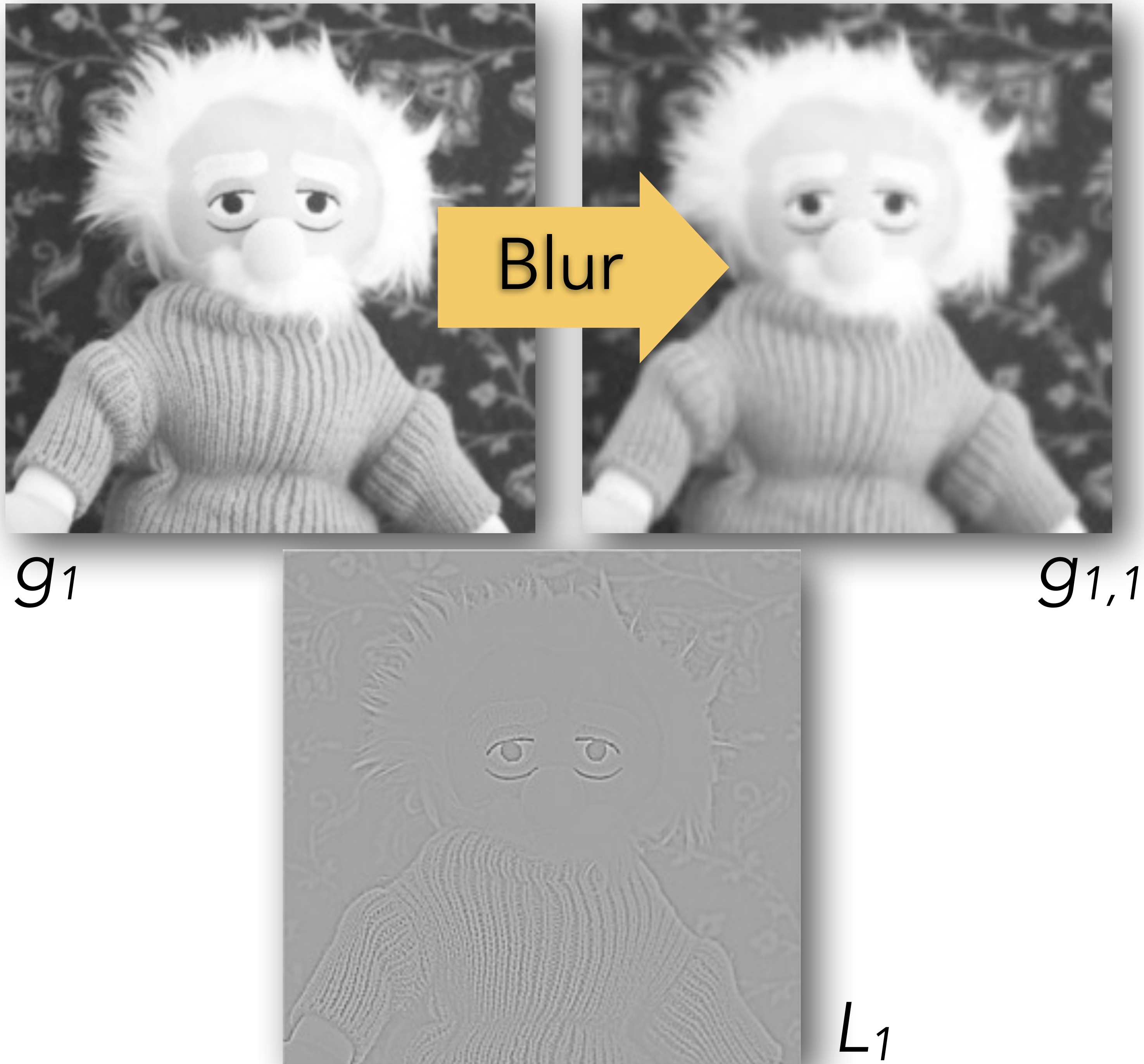
Computing Gaussian and Laplacian Pyramids



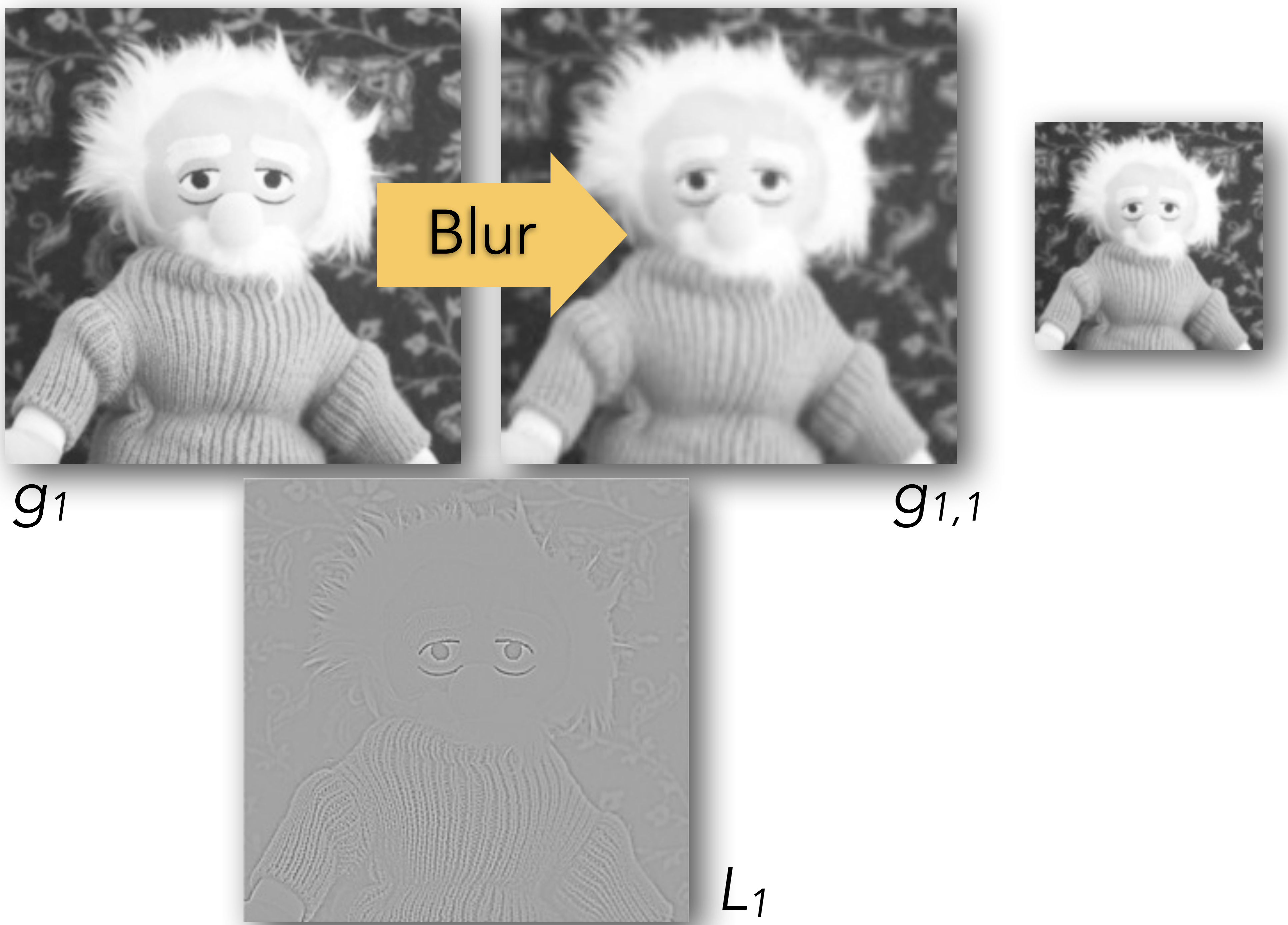
g_1 $g_{1,1}$

L_1

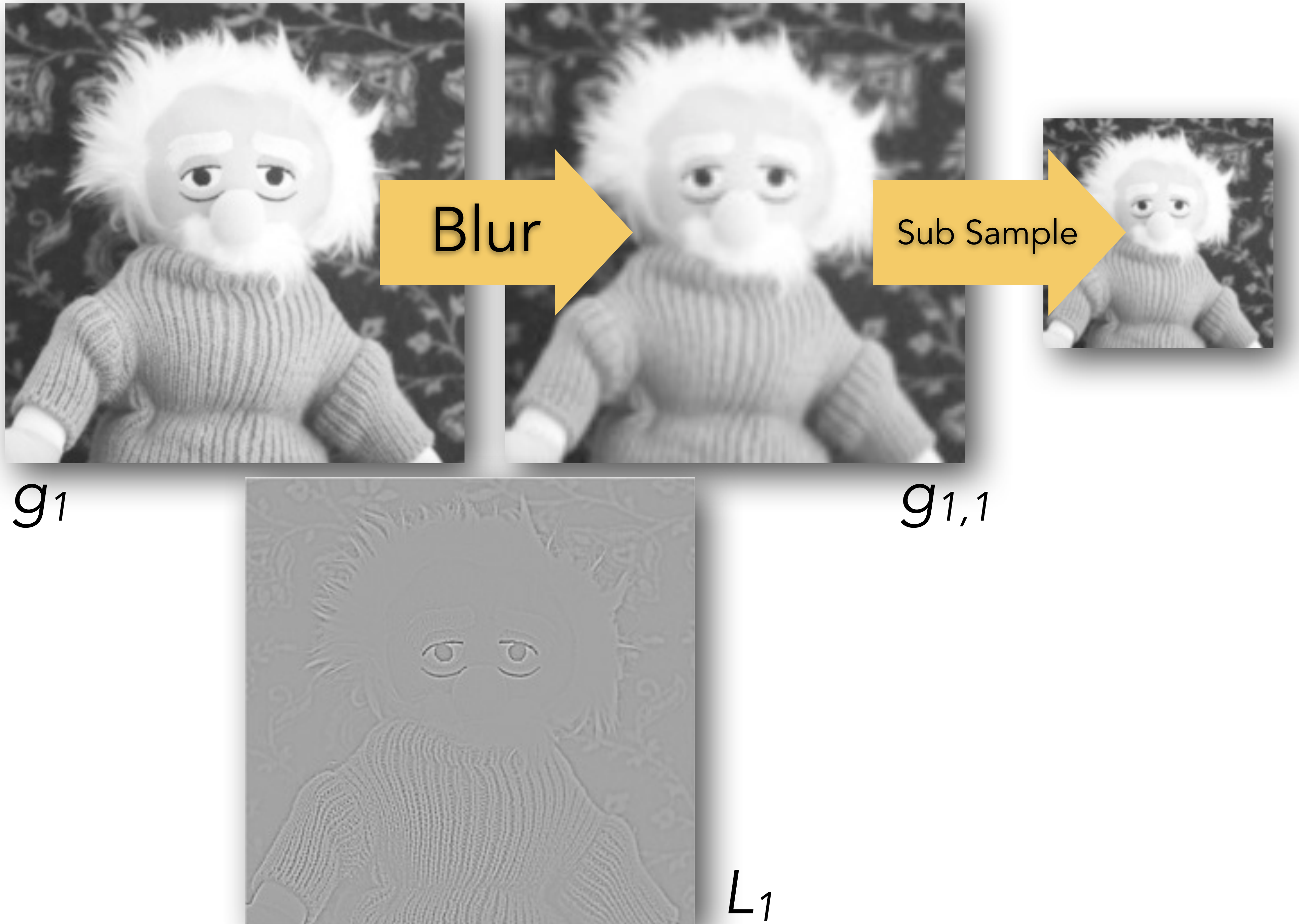
Computing Gaussian and Laplacian Pyramids



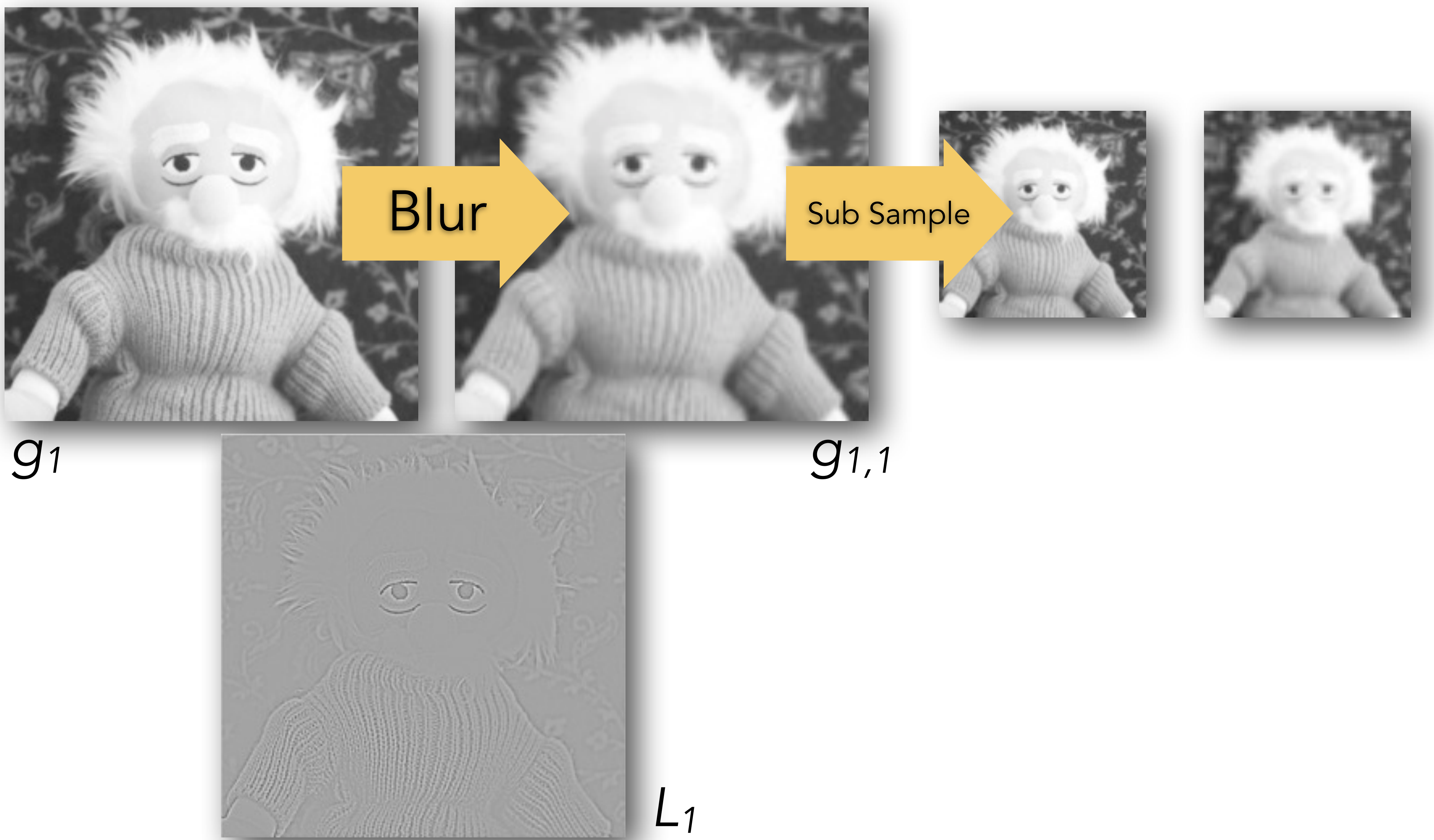
Computing Gaussian and Laplacian Pyramids



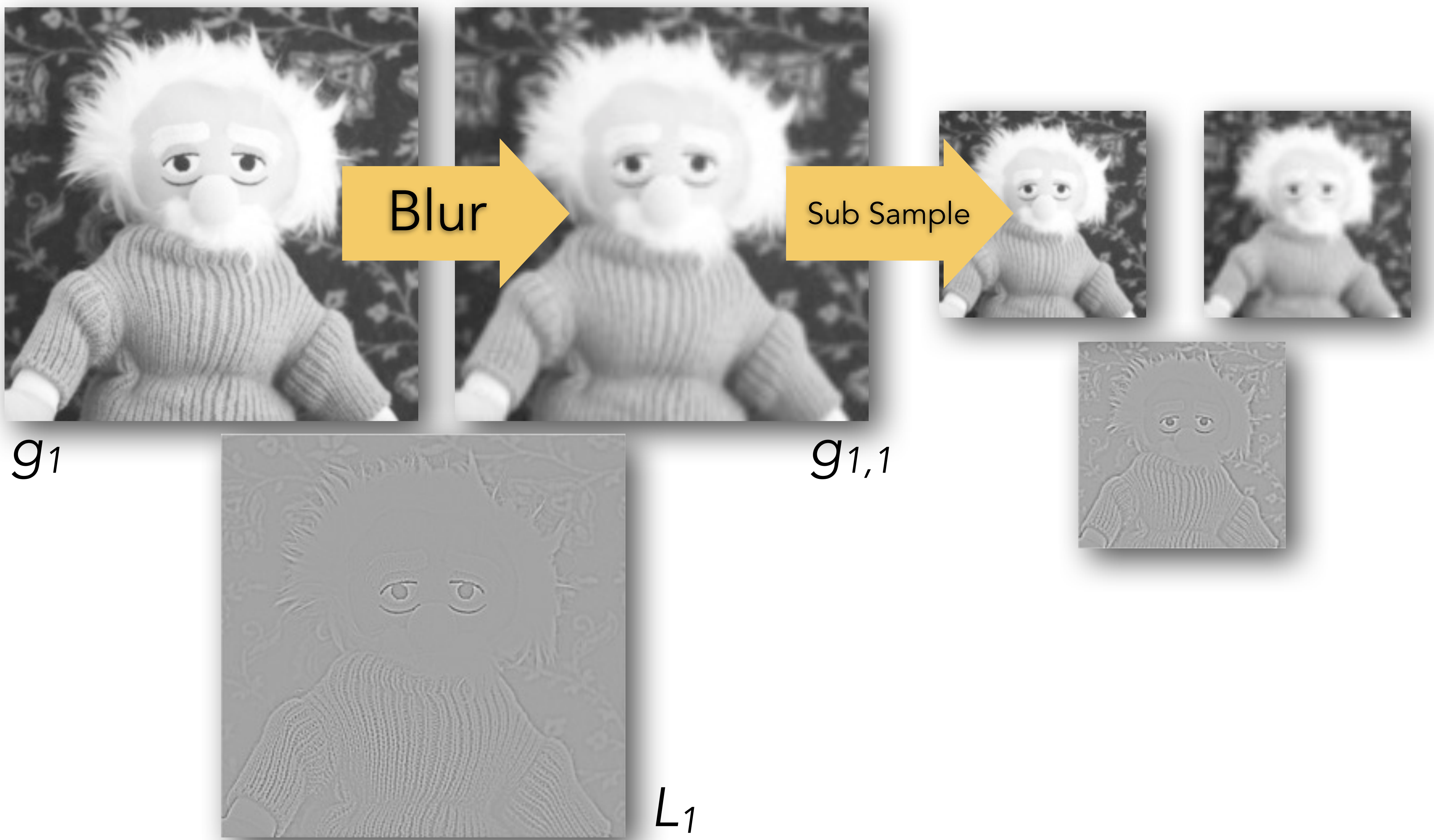
Computing Gaussian and Laplacian Pyramids



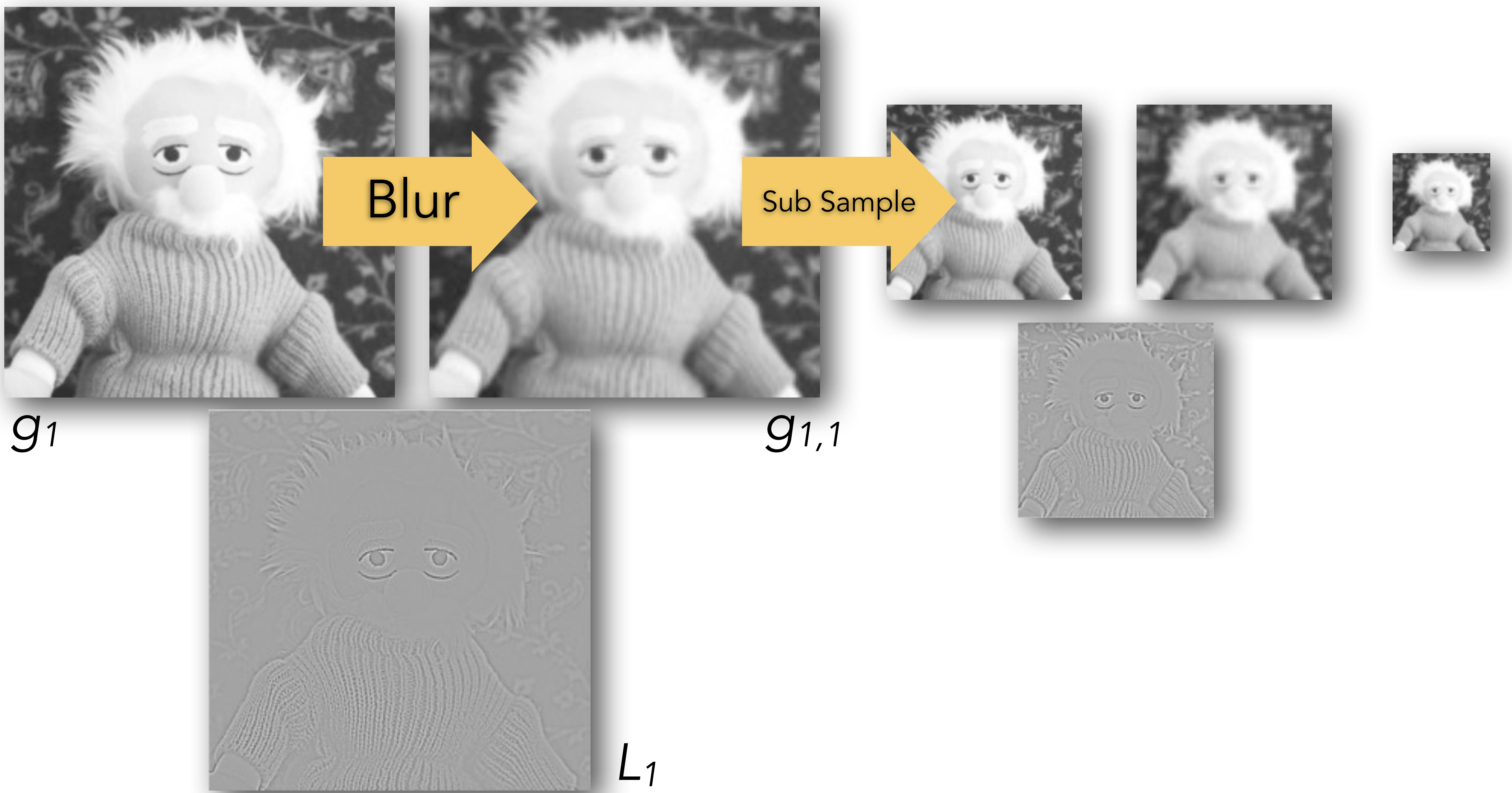
Computing Gaussian and Laplacian Pyramids



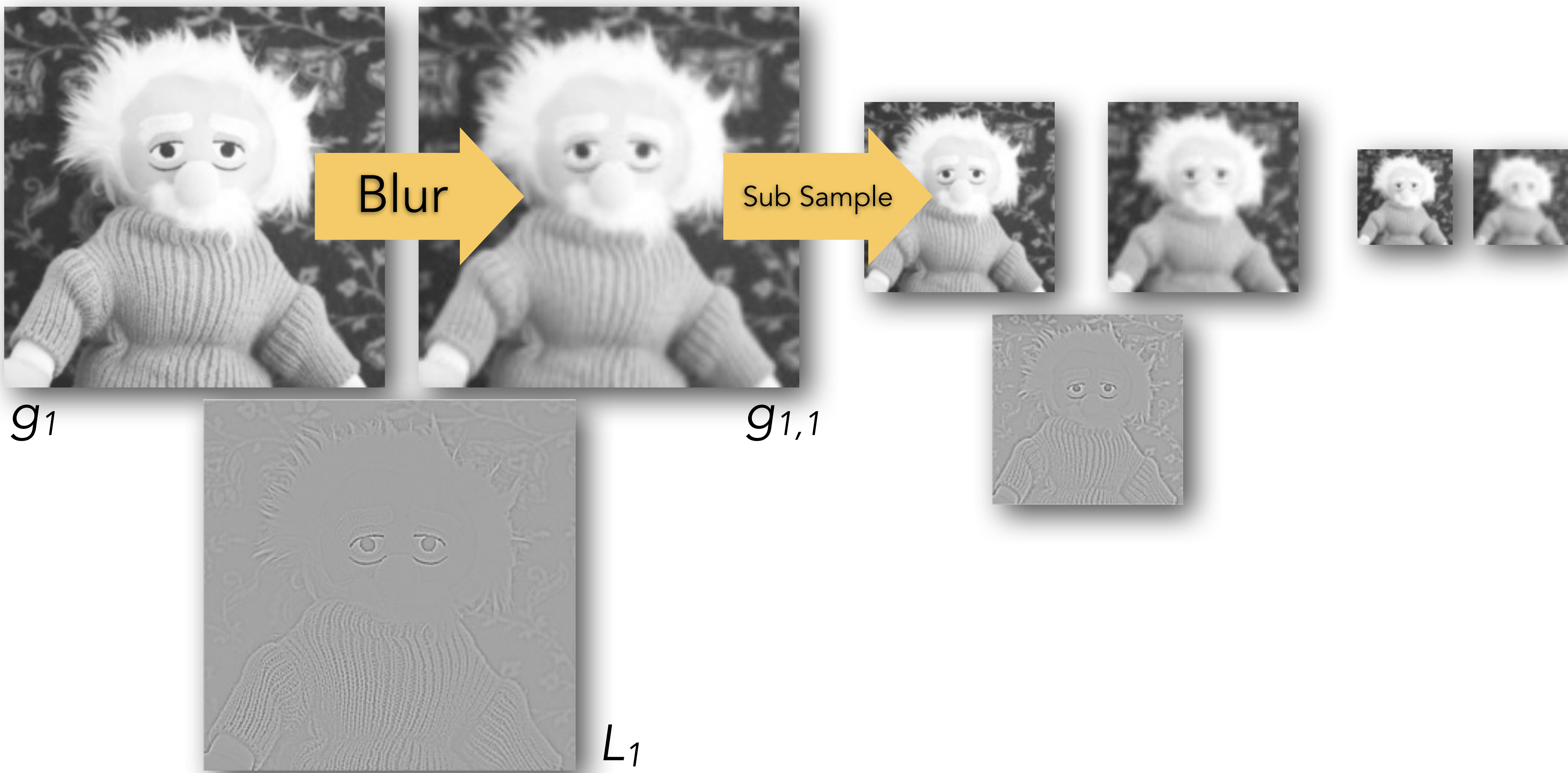
Computing Gaussian and Laplacian Pyramids



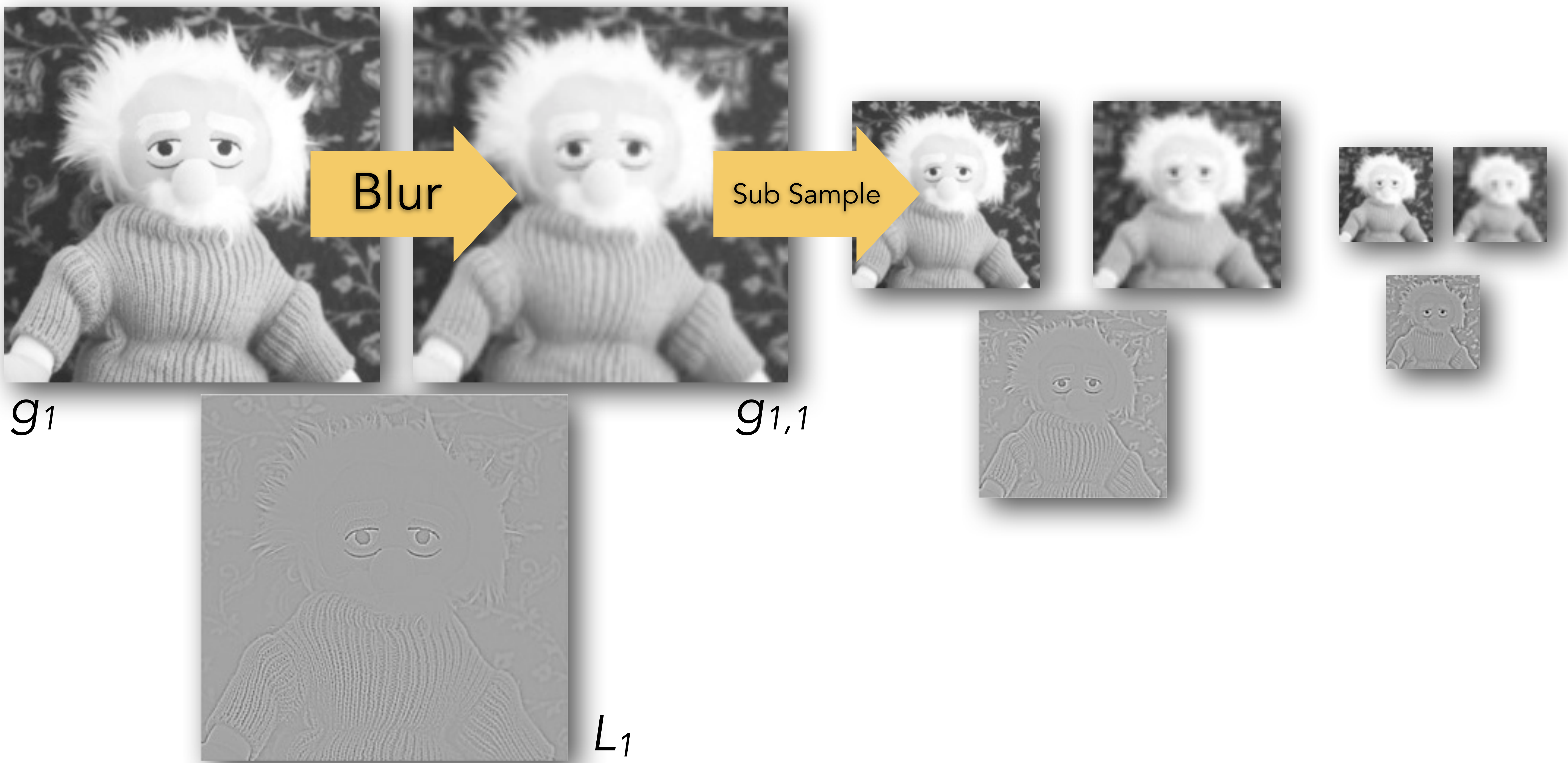
Computing Gaussian and Laplacian Pyramids



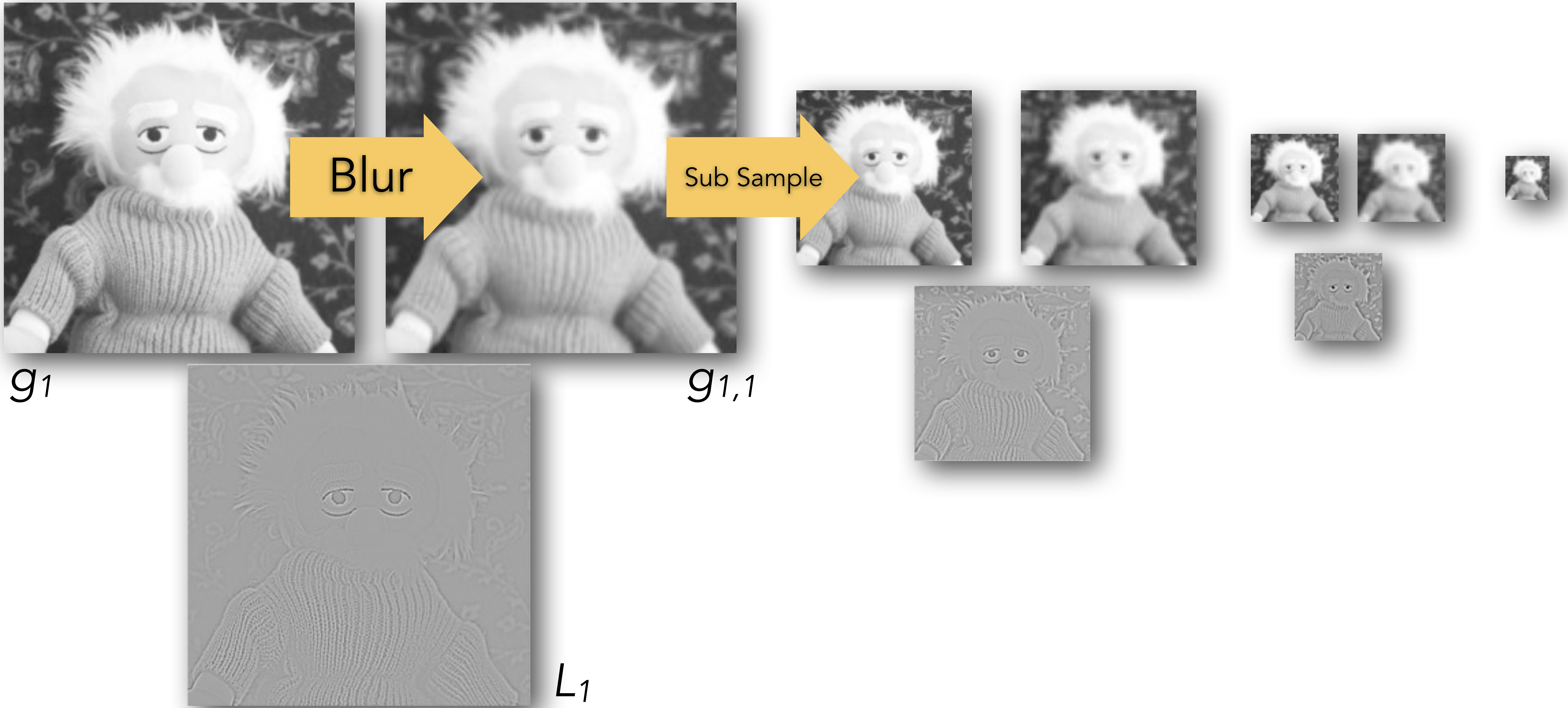
Computing Gaussian and Laplacian Pyramids



Computing Gaussian and Laplacian Pyramids

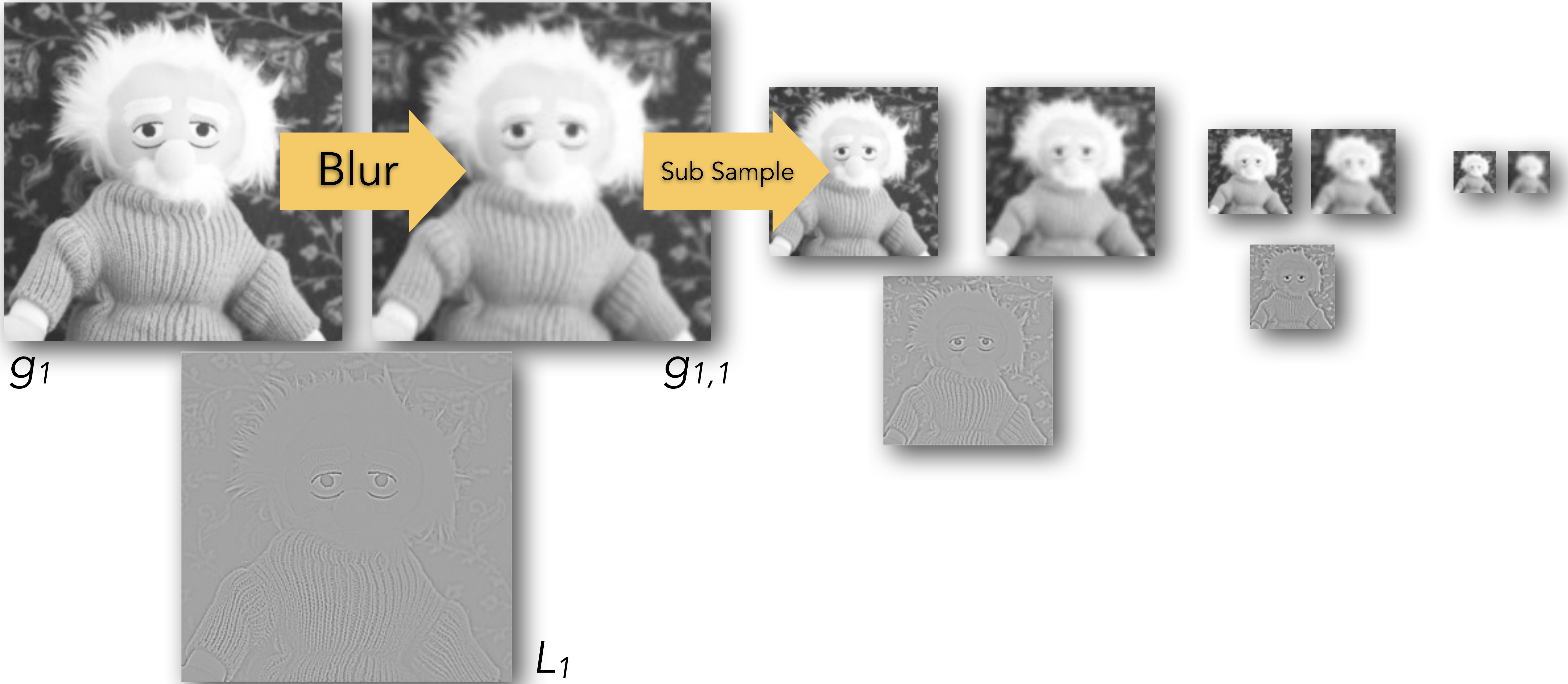


Computing Gaussian and Laplacian Pyramids

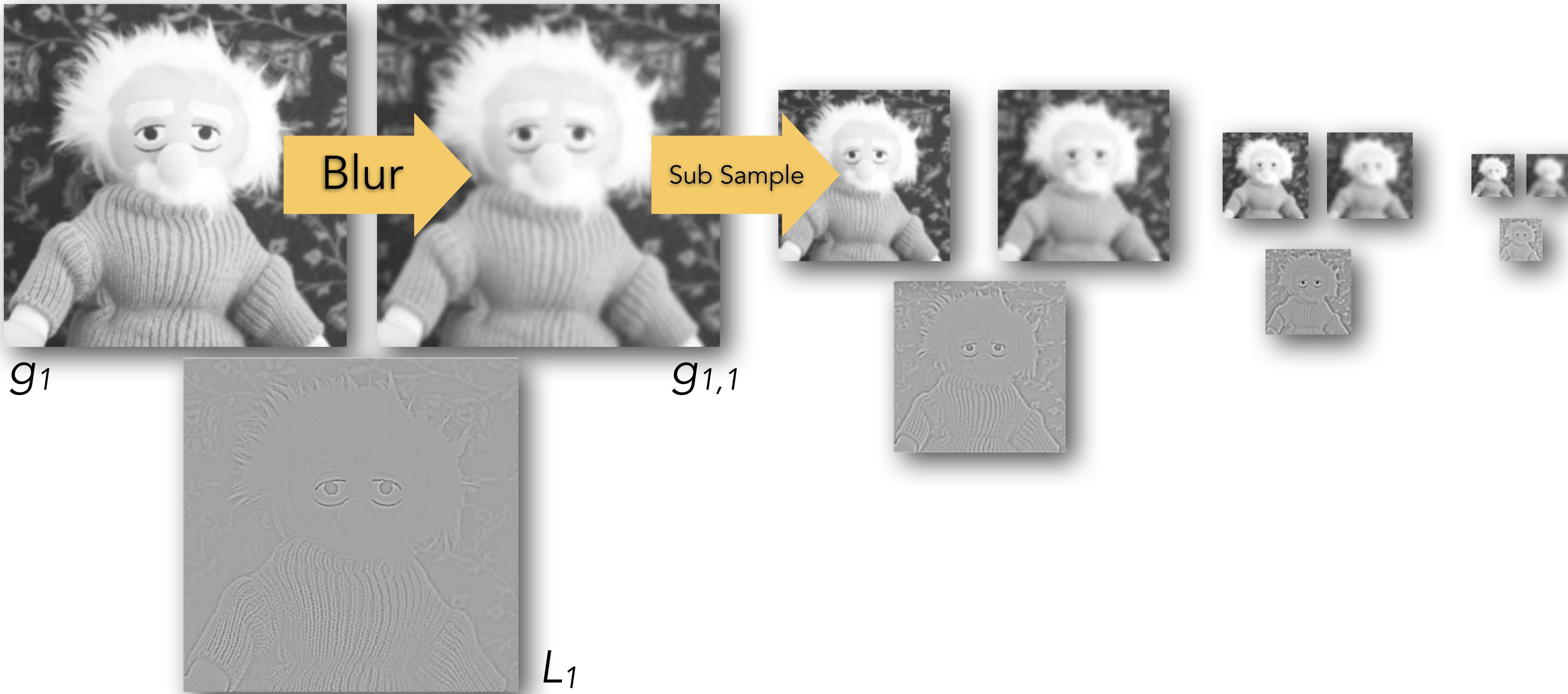


Computing Gaussian and Laplacian Pyramids

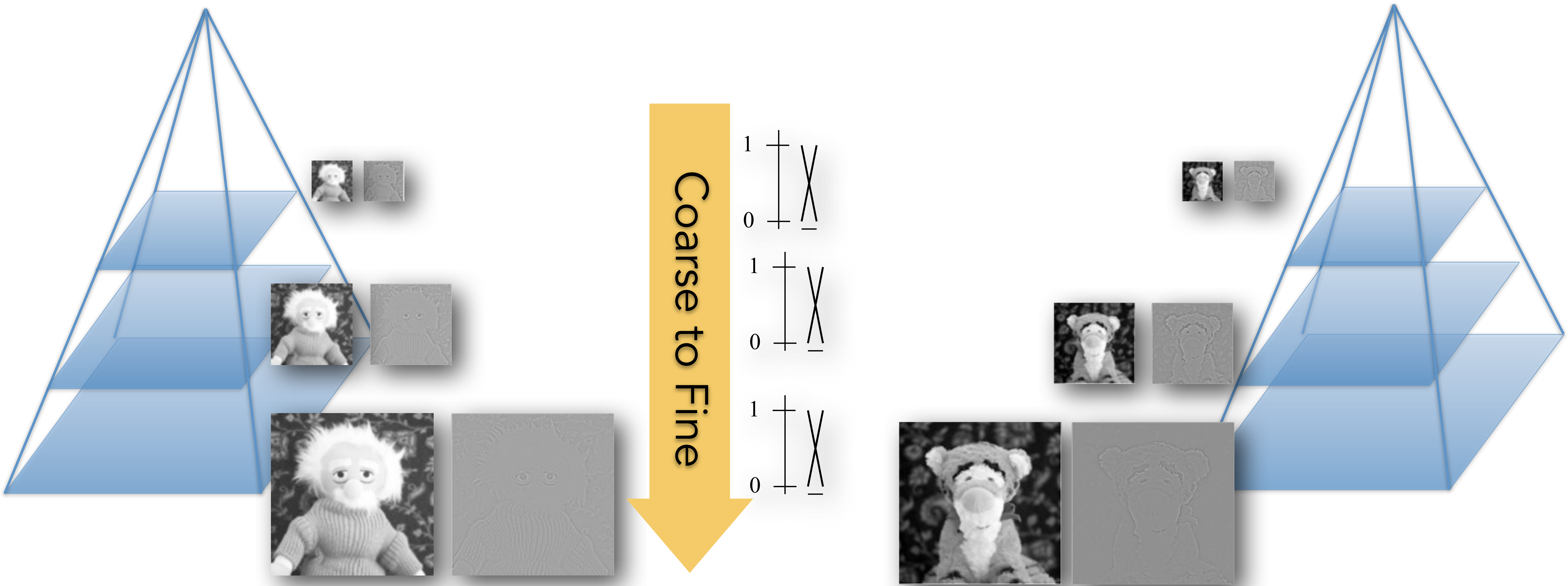
Georgia Institute of Technology



Computing Gaussian and Laplacian Pyramids



Computing Gaussian and Laplacian Pyramids



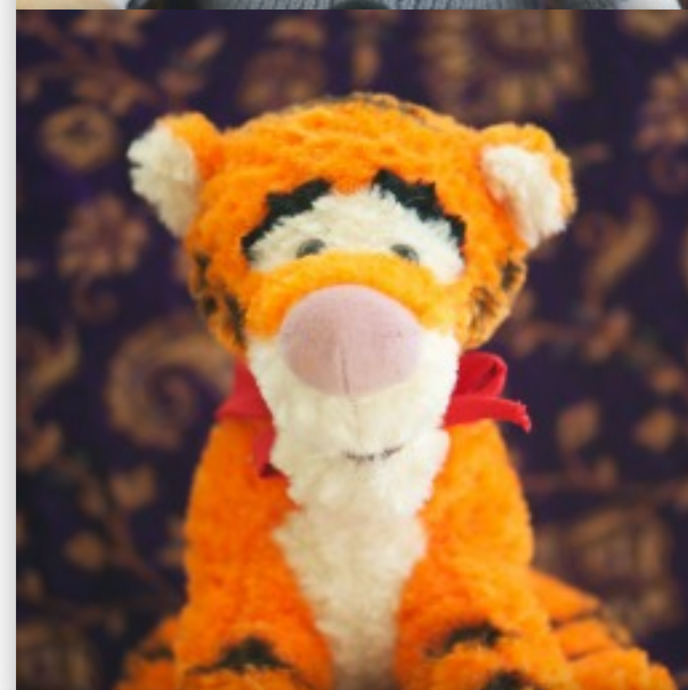
Pyramid Blending

Blend 1

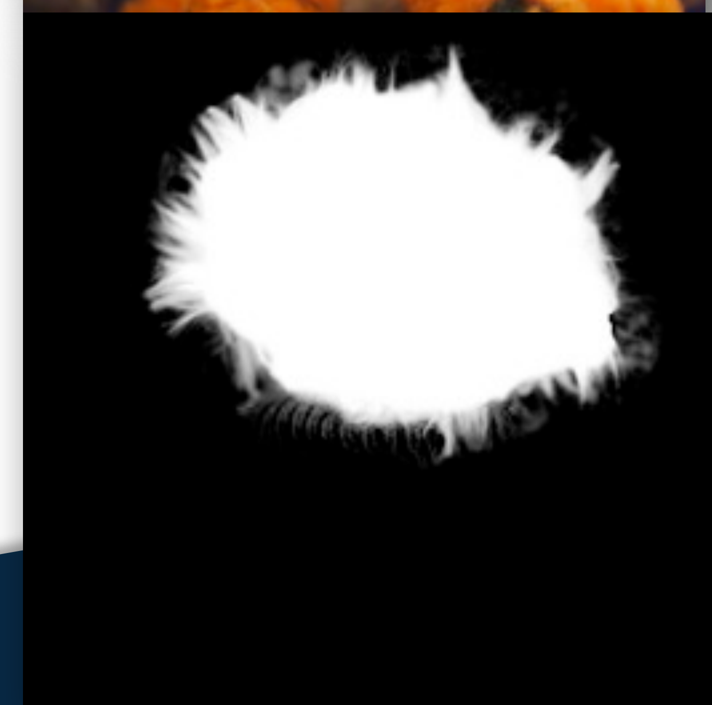
Pyramid Blending Process

A

B



R

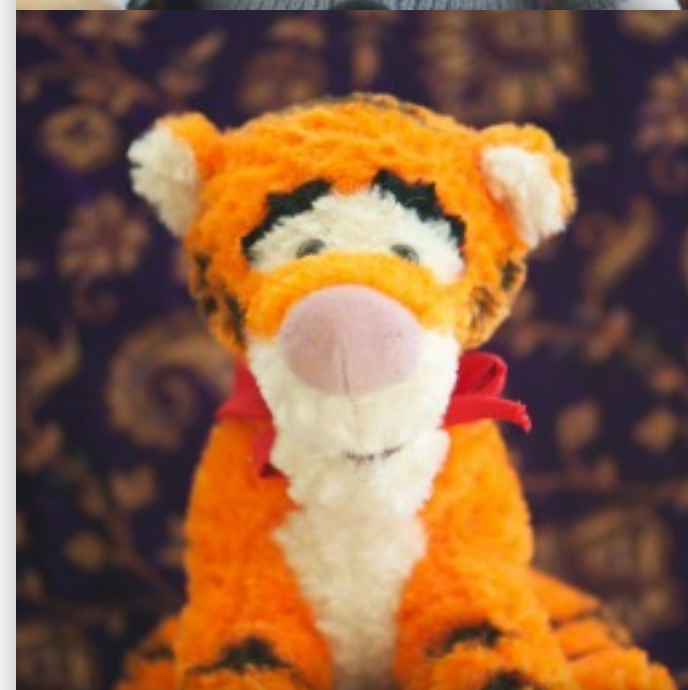


Pyramid Blending Process

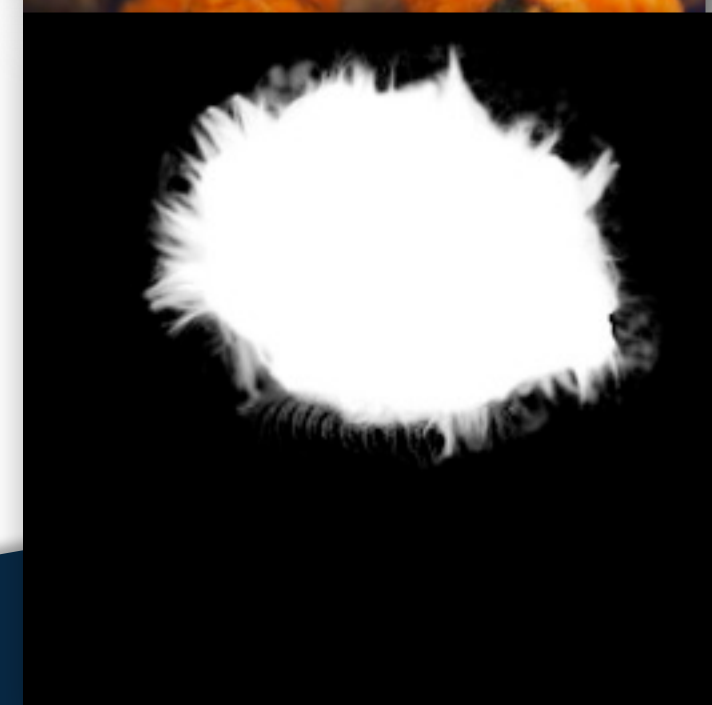
★ Build Laplacian pyramids L_A and L_B from images A and B

A

B



R

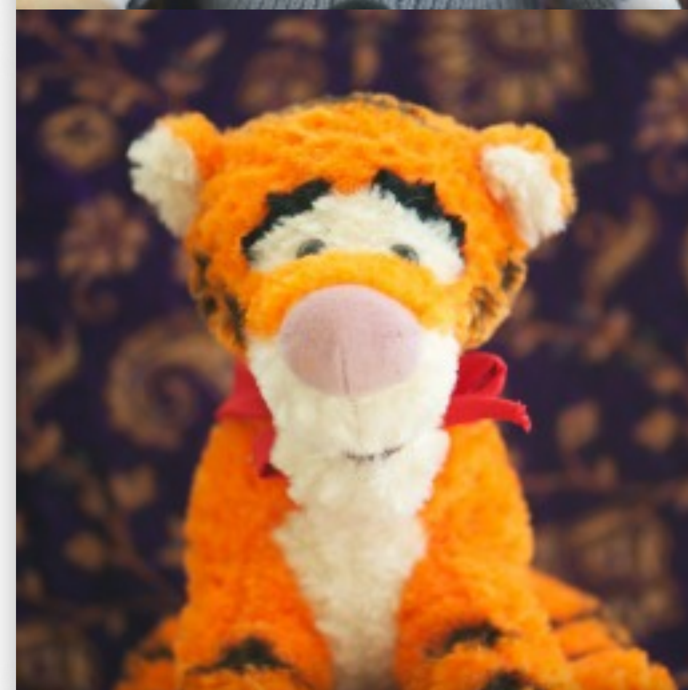


Pyramid Blending Process

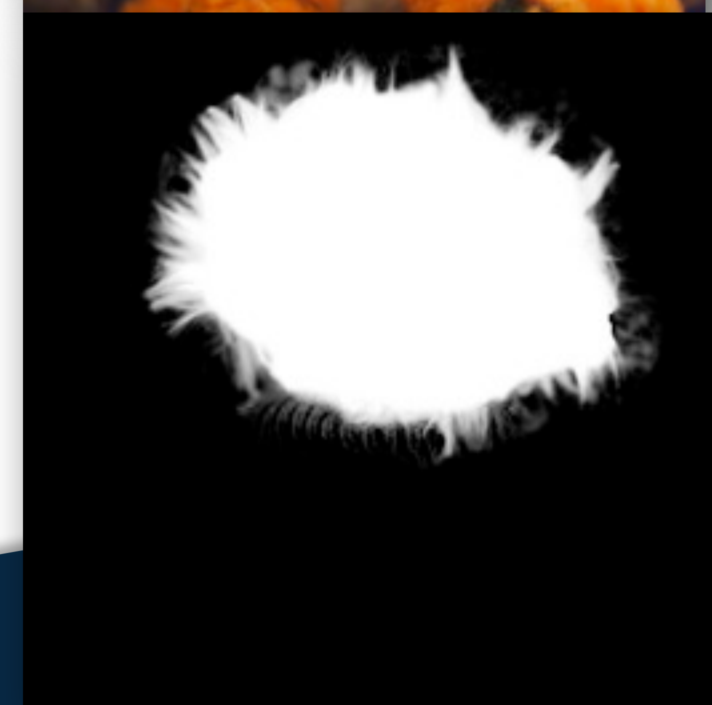
- ★ Build Laplacian pyramids L_A and L_B from images A and B
- ★ Build a Gaussian pyramid G_R from selected region R (aka. the Mask Image!).

A

B



R



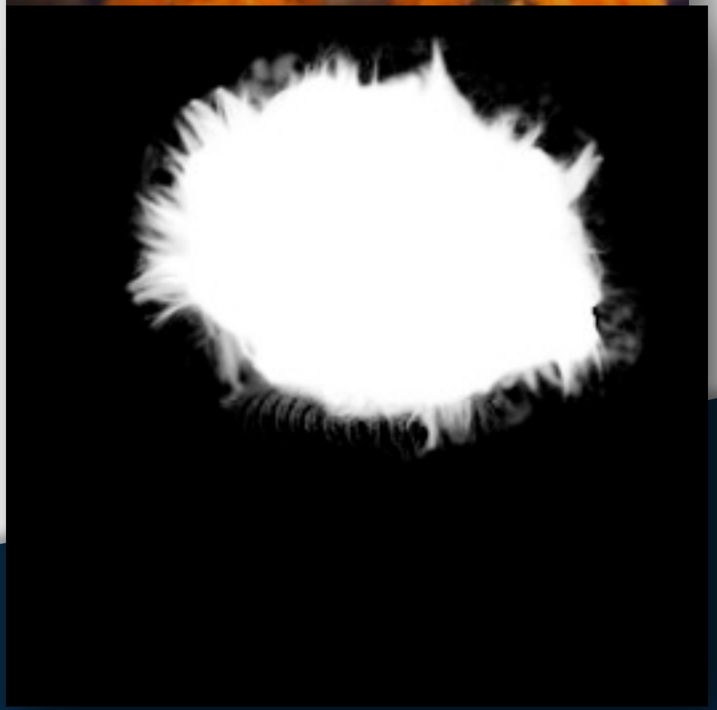
Pyramid Blending Process

- Build Laplacian pyramids L_A and L_B from images A and B
- Build a Gaussian pyramid G_R from selected region R (aka. the Mask Image!).
- Form a combined pyramid L_S from L_A and L_B using nodes of G_R as weights:
  - $L_S(i,j) = G_R(I,j,)^* \, L_A(I,j) + (1-G_R(I,j))^* \, L_B(I,j)$

A

B

R



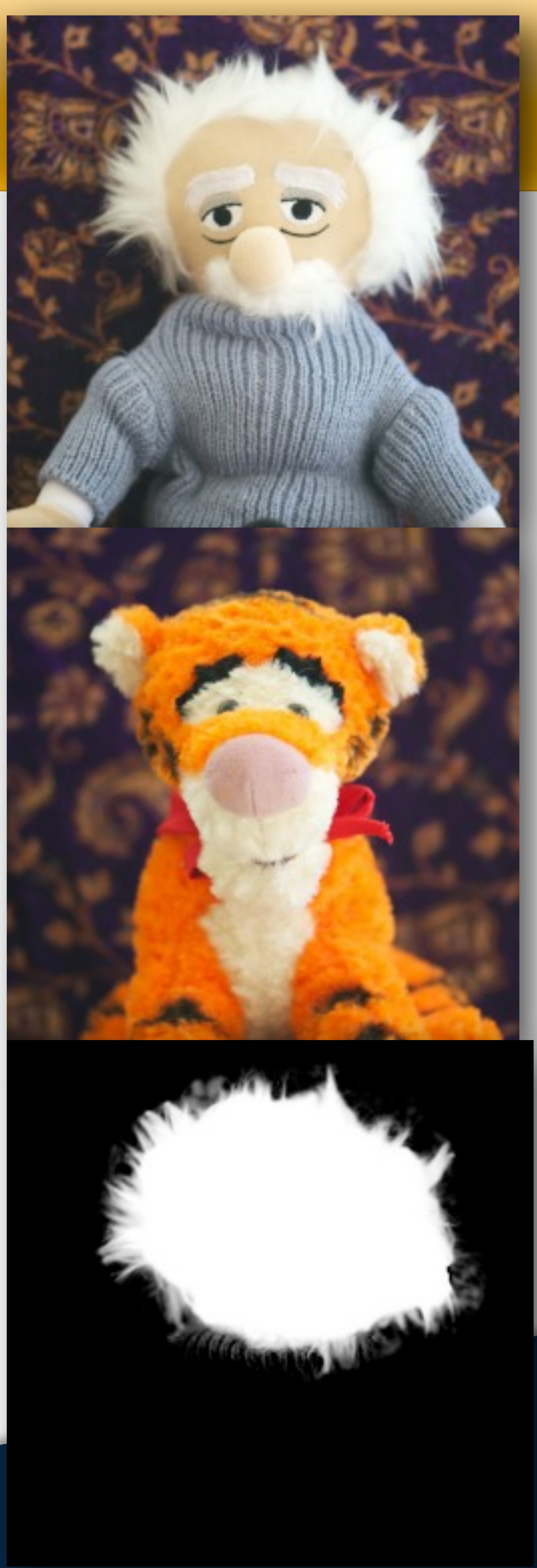
Pyramid Blending Process

- ★ Build Laplacian pyramids L_A and L_B from images A and B
- ★ Build a Gaussian pyramid G_R from selected region R (aka. the Mask Image!).
- ★ Form a combined pyramid L_S from L_A and L_B using nodes of G_R as weights:
  - $L_S(i,j) = G_R(I,j,)^* \; L_A(I,j) + (1-G_R(I,j))^* \; L_B(I,j)$
- ★ Collapse the L_S pyramid to get the final blended image

A

B

R

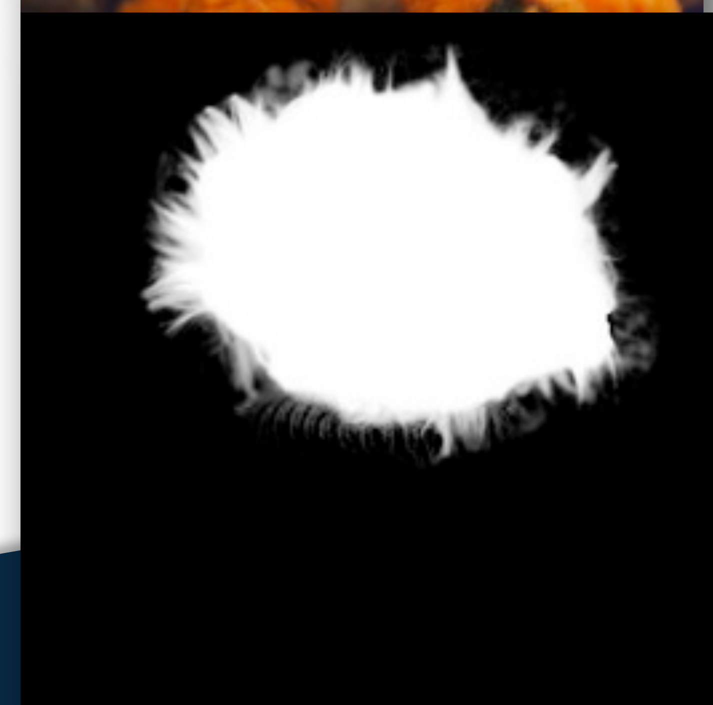


Pyramid Blending Process

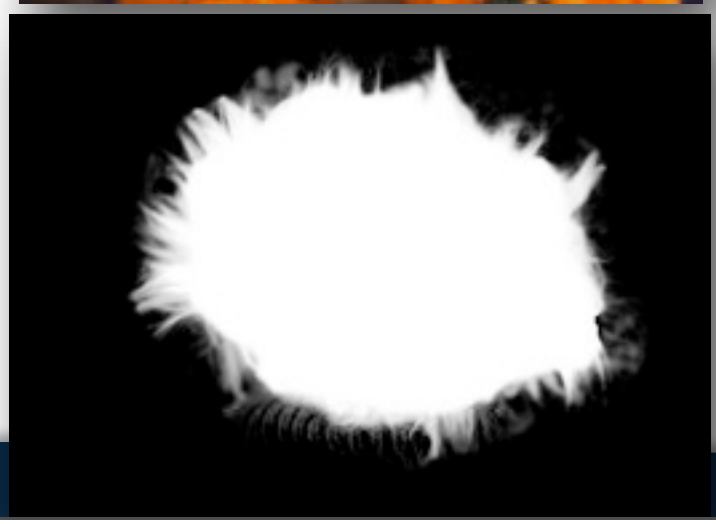
A

B

R



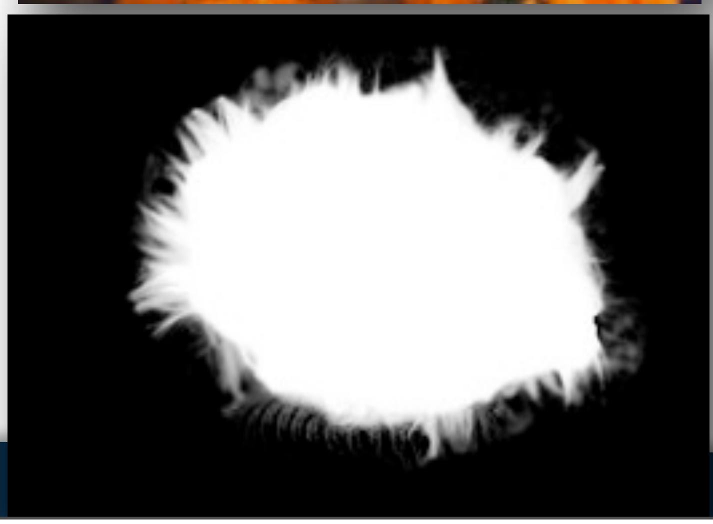
- ★ Build Laplacian pyramids L_A and L_B from images A and B
- ★ Build a Gaussian pyramid G_R from selected region R (aka. the Mask Image!).
- ★ Form a combined pyramid L_S from L_A and L_B using nodes of G_R as weights:
  - $L_S(i,j) = G_R(I,j,)* L_A(I,j) + (1-G_R(I,j))* L_B(I,j)$
- ★ Collapse the L_S pyramid to get the final blended image



Blend 2







Blend 2

Georgia Institute of Technology

Summary

- ★ Introduced the concept of merging two images leveraging the Frequency Domain.
- ★ Introduced the Gaussian and the Laplacian Pyramids.
- ★ Discussed the Mathematical formulation to compute a Laplacian.
- ★ Outlined the specific process to blend two images using Pyramids.

Next Class

- Merging and Blending of Images: Cutting Images

Credits

- For more information, see
  - Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer.
  - Burt and Adelson (1983) “The Laplacian Pyramid as a Compact Image Code”, In IEEE Transactions on Communications, 31 (4). p 532-540. 1983 (DOI)
  - Burt and Adelson (1983) “A multiresolution spline with application to image mosaics”. In ACM Transactions on Graphics, 2 (4). 1983 (DOI)
- Some concepts in slides motivated by similar slides by A. Efros and J. Hays.
- Some images retrieved from
  - http://commons.wikimedia.org/.
  - List will be available on website.
  - by Irfan Essa

www.flickr.com/photos/neneonline/231886965/

18

Computational Photography

Dr. Irfan Essa
Professor
School of Interactive Computing

Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.