THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

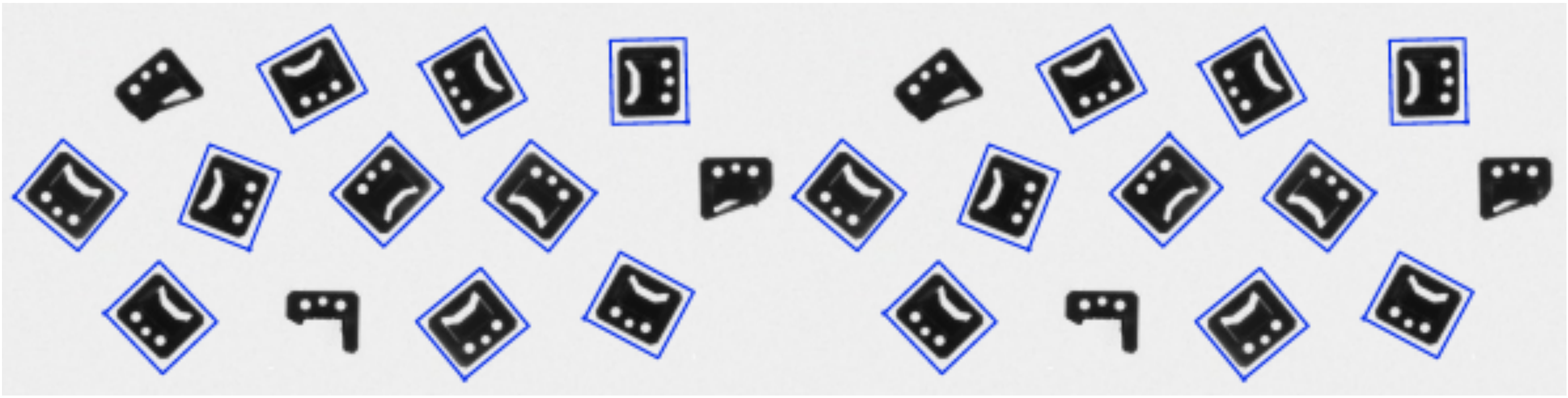


Image Credit: https://docs.adaptive-vision.com/4.7/studio/machine_vision_guide/TemplateMatching.html

Lecture 7: Template Matching, Scaled Representations

(unless otherwise stated slides are taken or adopted from **Bob Woodham, Jim Little** and **Fred Tung**)

Menu for Today (**September 26, 2024**)

Topics:

- **Digital Imaging** Pipeline
- **Scaled** Representations
- Template **Matching**
- Normalised **Correlation**

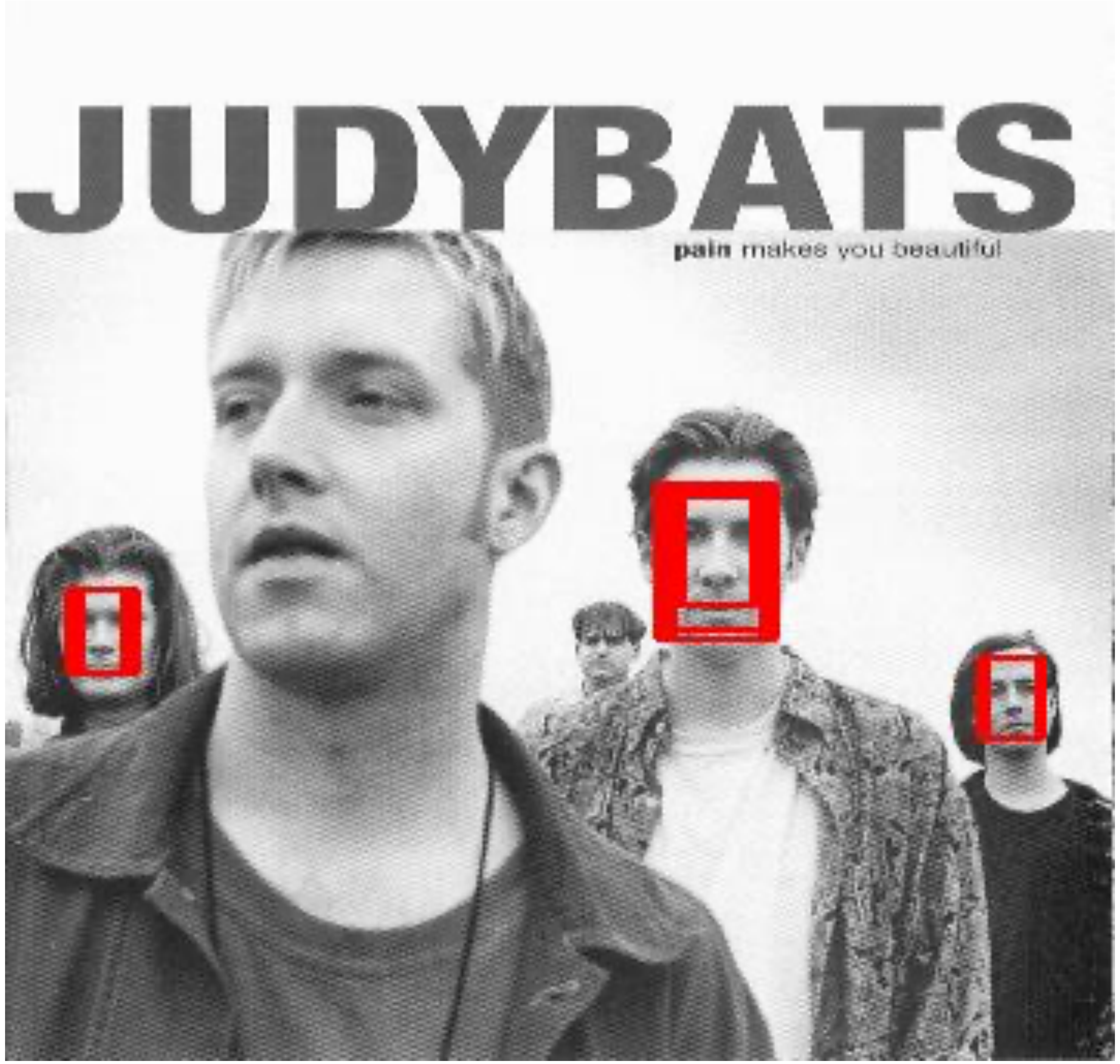
Readings:

- **Today's** Lecture: Szeliski 2.3, 3.5, Forsyth & Ponce (2nd ed.) 4.5 - 4.7

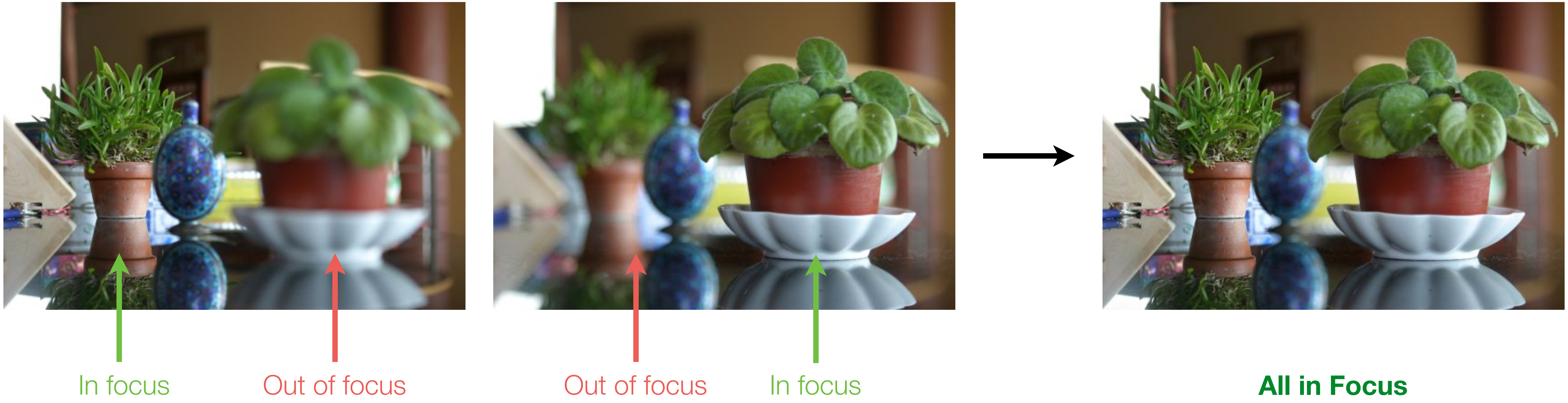
Reminders:

- **Assignment 1:** Image Filtering and Hybrid Images due **today**
- **Assignment 2**: Scaled Representations, Face Detection and Image Blending (will be out **today**)
- **Quiz 1** is out and due today, 11:59pm

Assignment 2: Preview — Part 1: Face Detection



Assignment 2: Preview — Part 2: Image Blending



Assignment 2: Preview — Part 2: Image Blending

(imaging plane closer to f)

(imaging further than f)

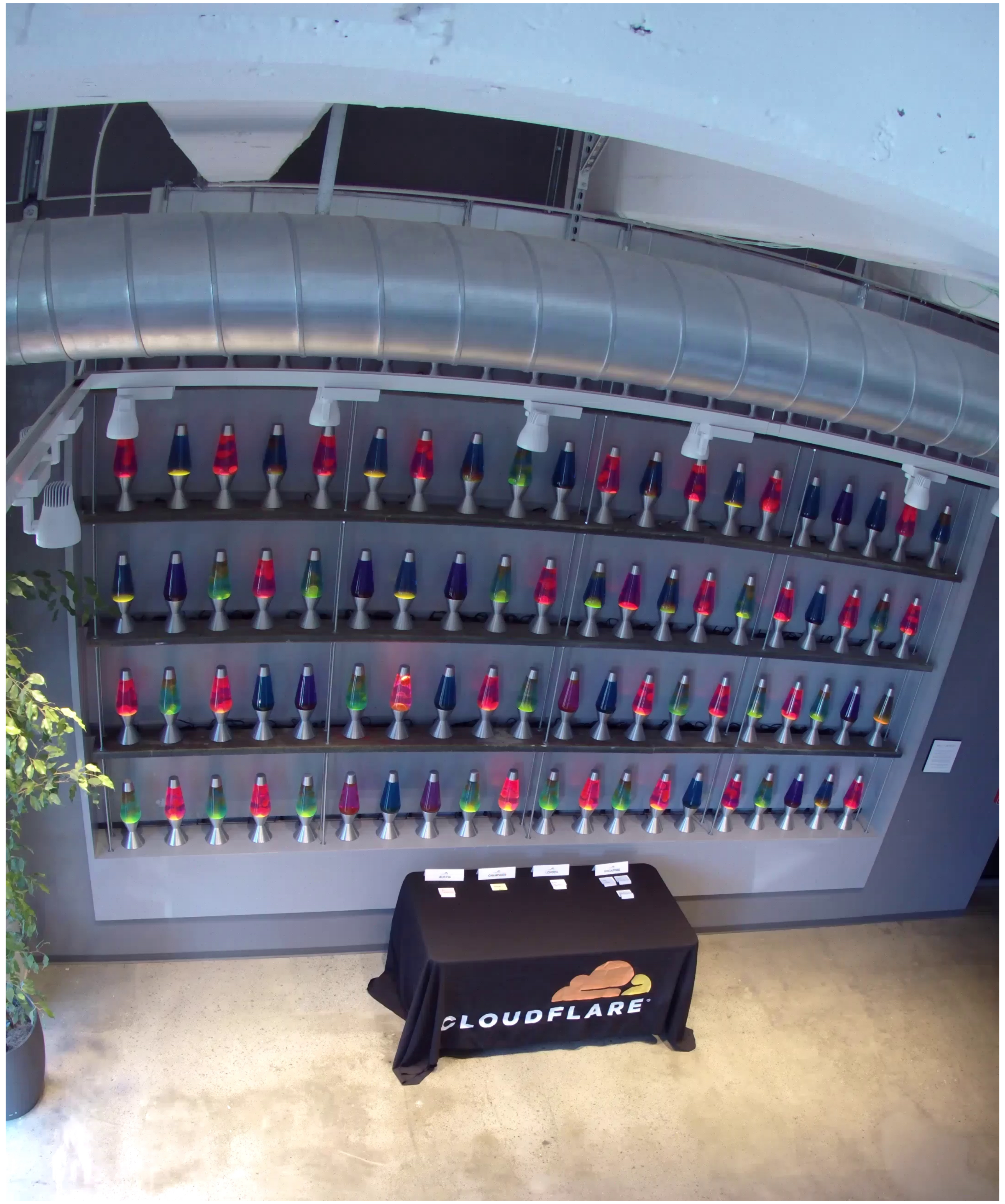
Today’s “**fun**” Example: NCIS

Today's "**fun**" Example: NCIS

Today's "**fun**" Example: NCIS

Today's "**fun**" Example: LavaRAND

Today’s “**fun**” Example: LavaRAND



Lecture 6: Re-cap

In the **continuous** case, images are functions of two spatial variables, x and y.

The **discrete** case is obtained from the continuous case via sampling (i.e. spatial tessellation, grayscale quantization).

If a signal is **bandlimited** then it is possible to design a sampling strategy such that the sampled signal captures the underlying continuous signal exactly.

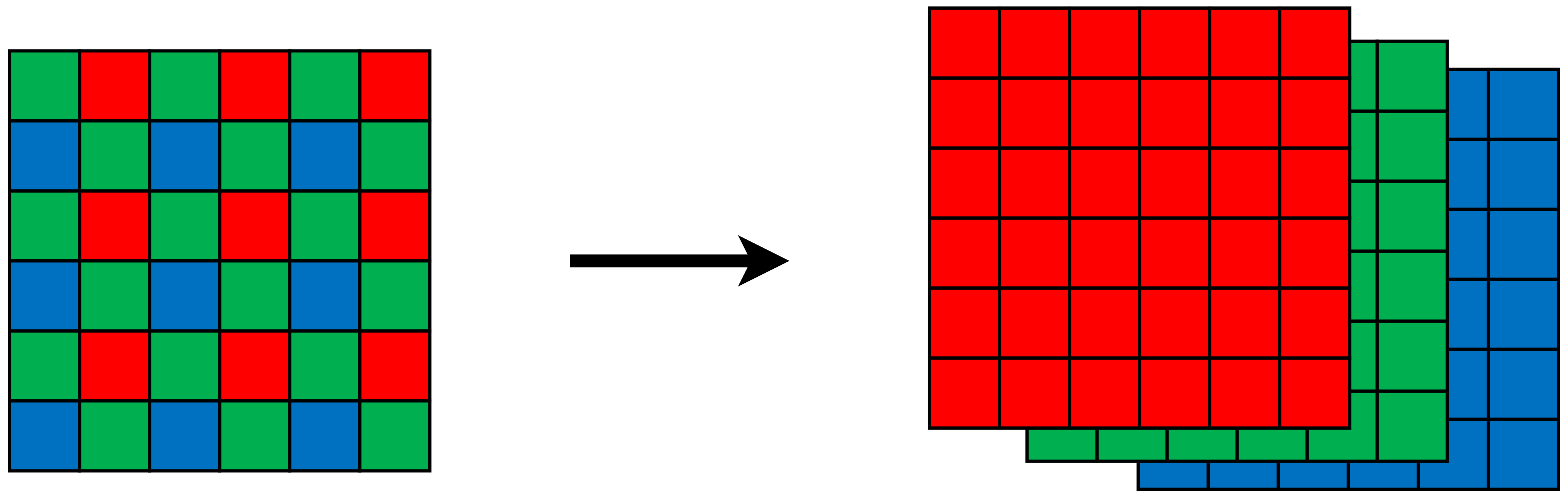
If we know what we imaging (position and texture of objects, etc.) and how (distance of those object to the camera, lens parameters of the camera, etc.) then we can calculate what resolution sensor we may need to “trust” our imaging

Lecture 6: Re-cap

“Color” is **not** an objective physical property of light (electromagnetic radiation). Instead, light is characterized by its wavelength.

Color Filter Arrays (CFAs) allow capturing of mosaiced color information; the layout of the mosaic is called **Bayer** pattern.

Demosaicing is the process of taking the RAW image and interpolating missing color pixels per channel



Goal

1. See how **image filtering** can be used in **practice**

2. Understand the concepts behind **template matching**

Template Matching

How can we find a part of one image that matches another?

or,

How can we find instances of a pattern in an image?

Template Matching

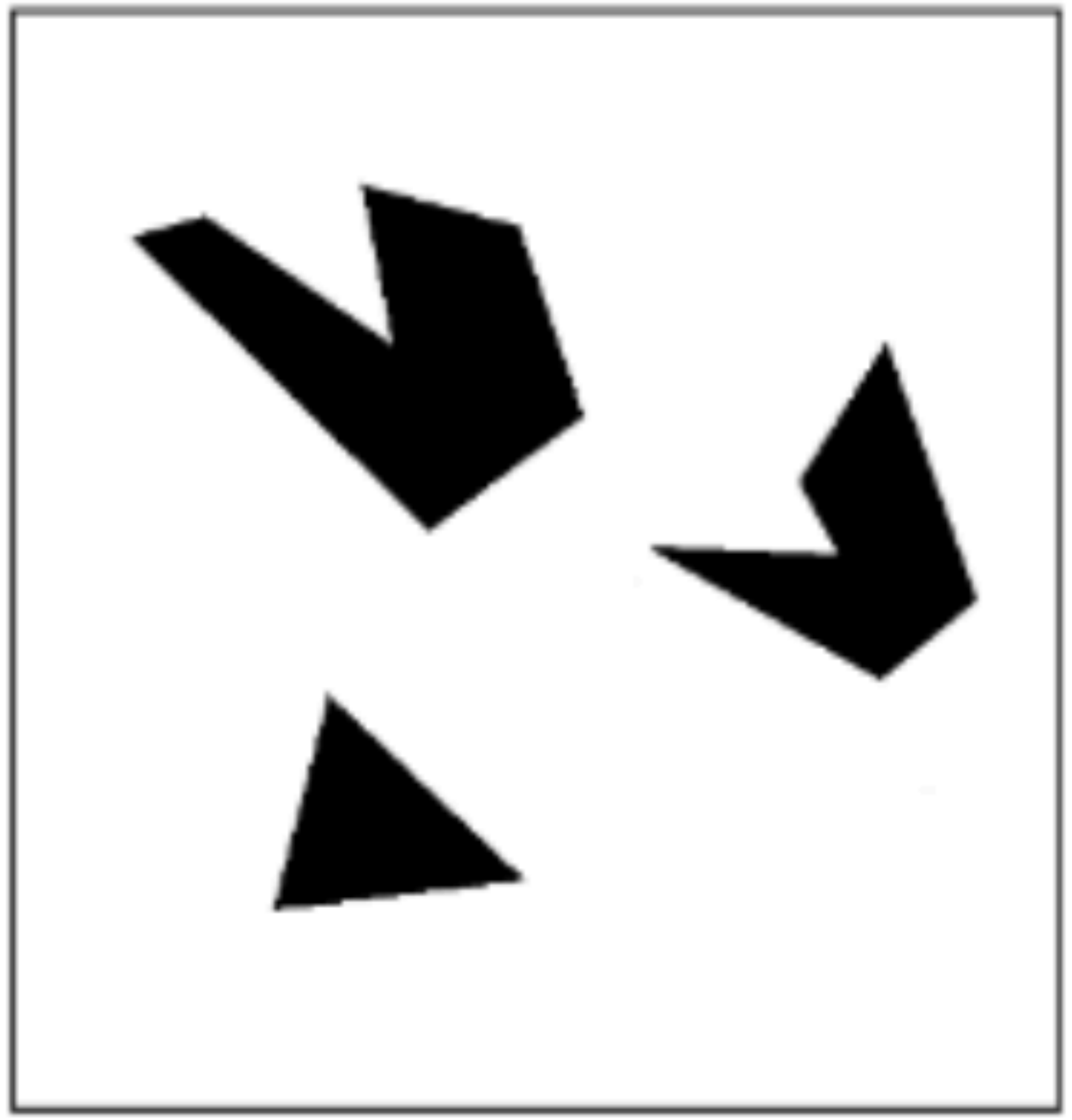
How can we find a part of one image that matches another?

or,

How can we find instances of a pattern in an image?

Key Idea: Use the pattern as a **template**

Template Matching



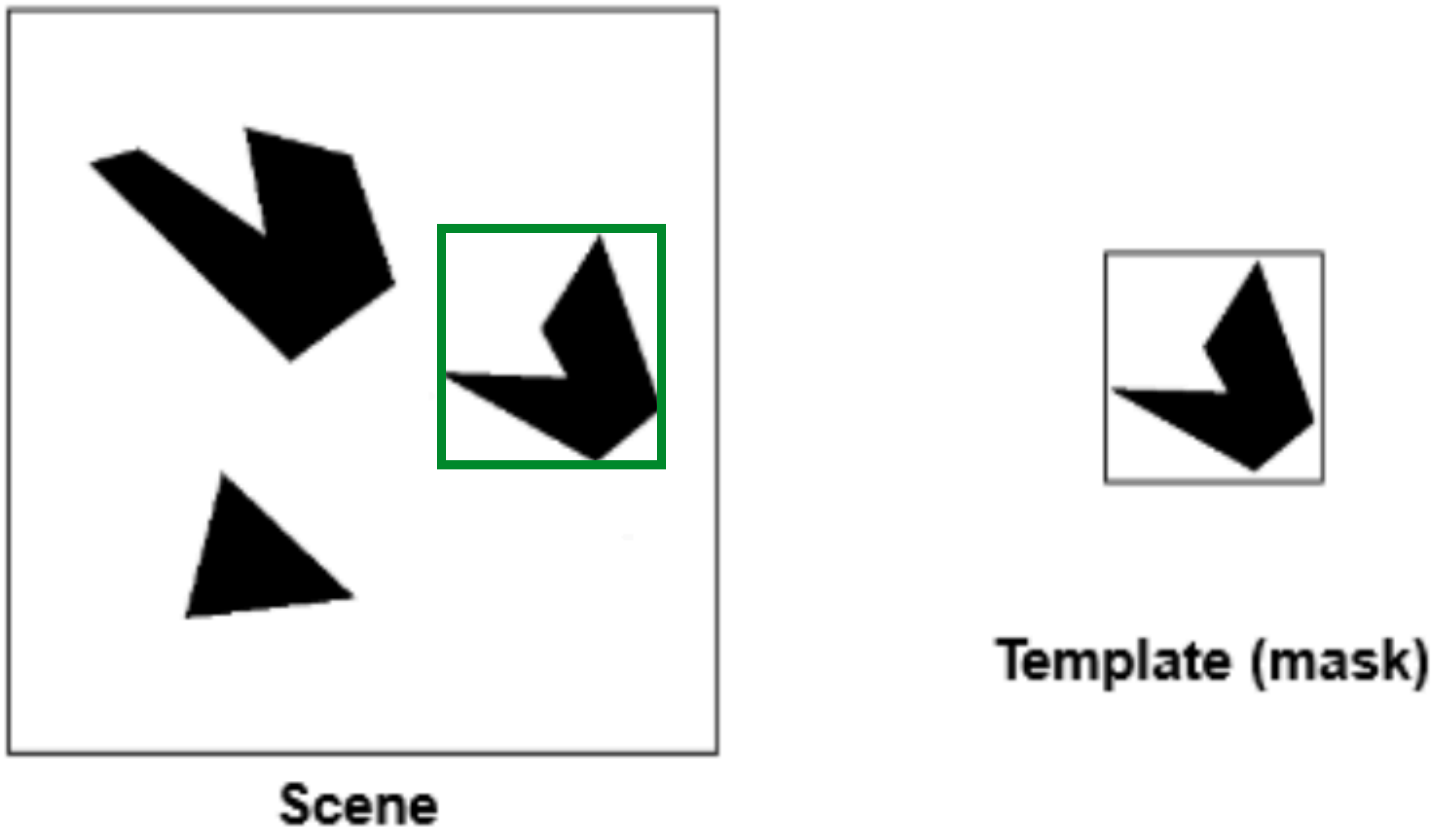
Scene

Template (mask)

A toy example

Slide Credit: Kristen Grauman

Template Matching



A toy example

Slide Credit: Kristen Grauman

Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

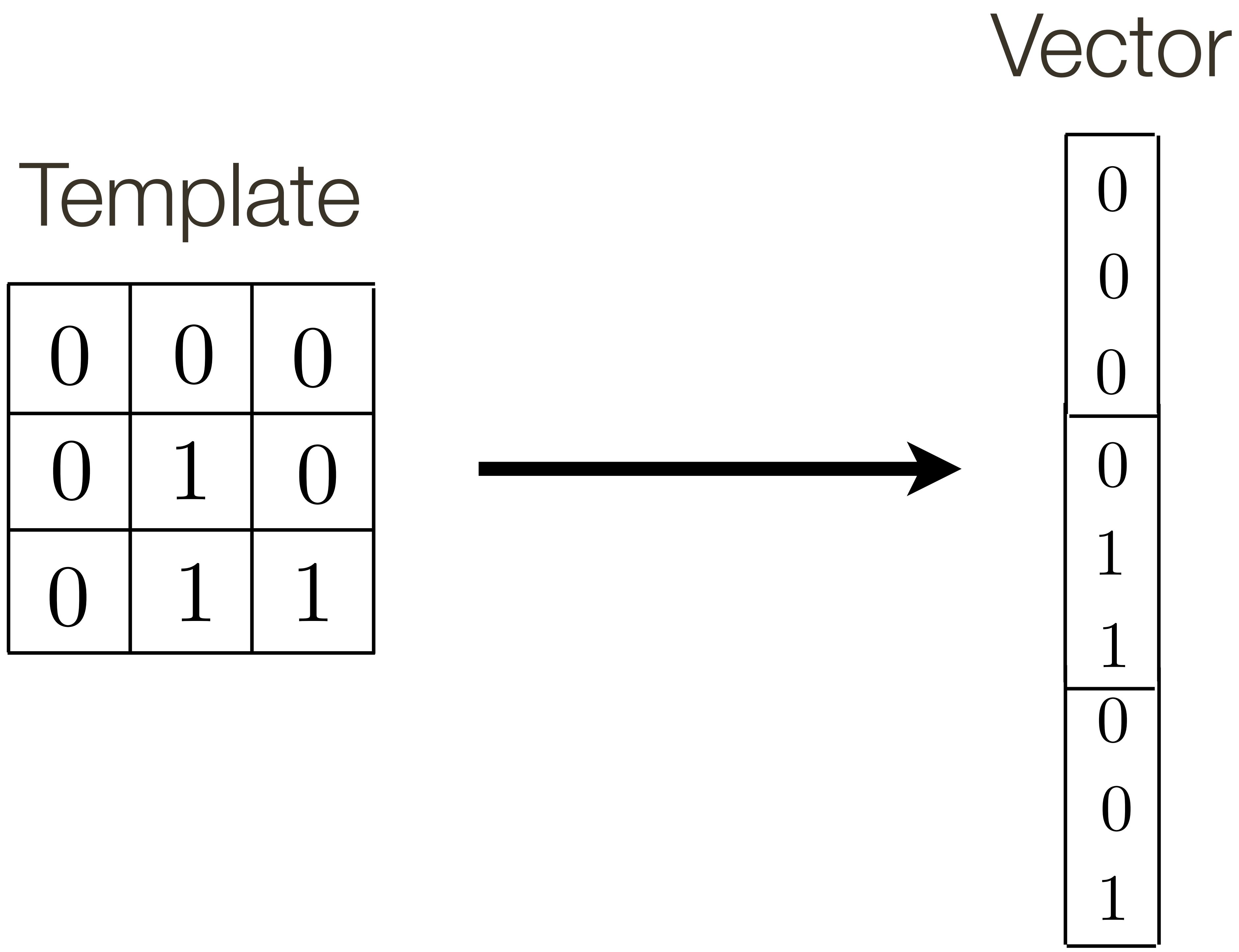
— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.

Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.



Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.

Image Patch 1

0	0	0
0	1	0
0	1	1

Image Patch 2

1	0	1
0	1	0
0	0	0

Template

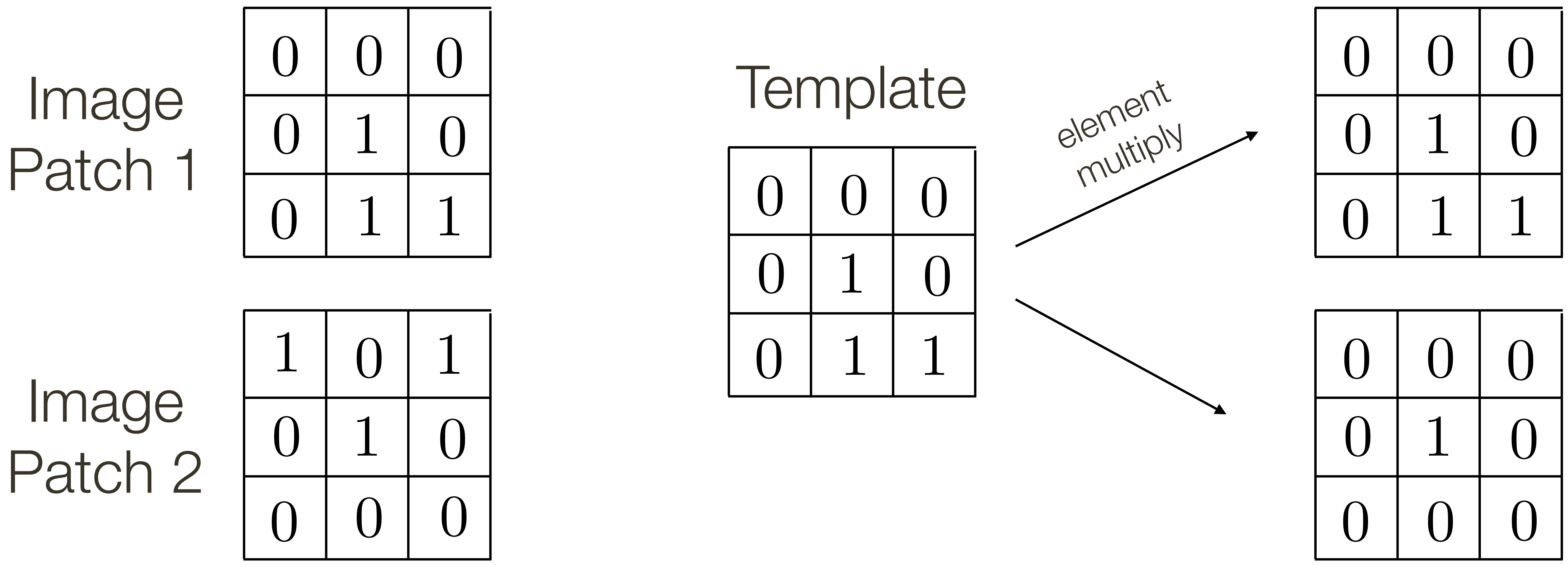
0	0	0
0	1	0
0	1	1

Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.

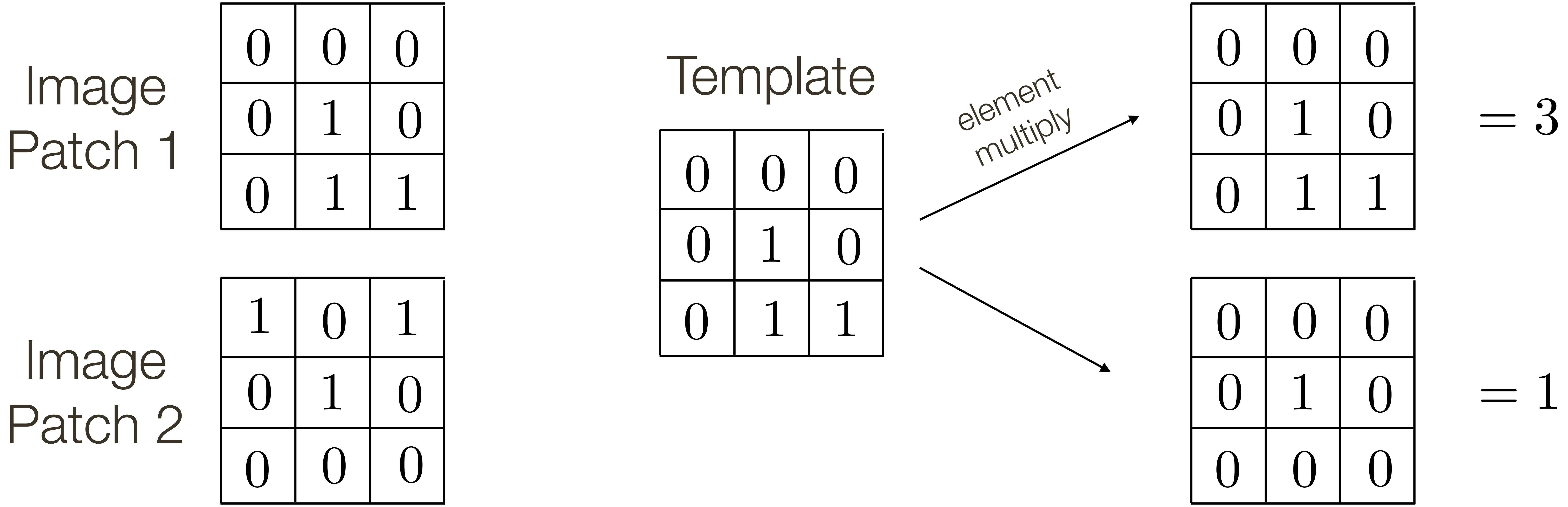


Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.

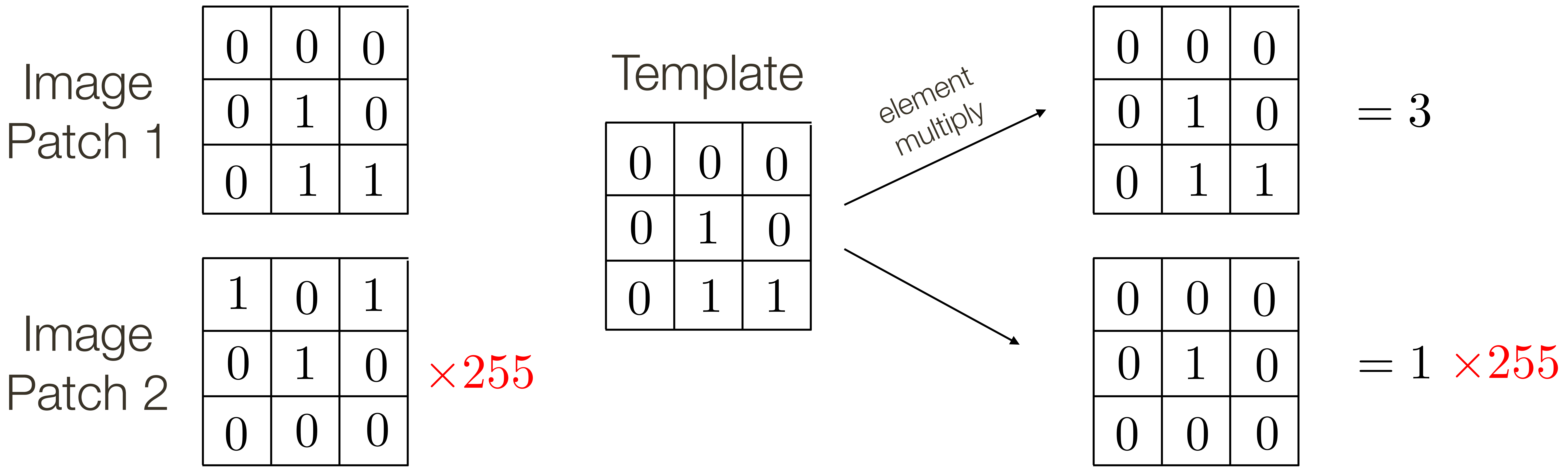


Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.

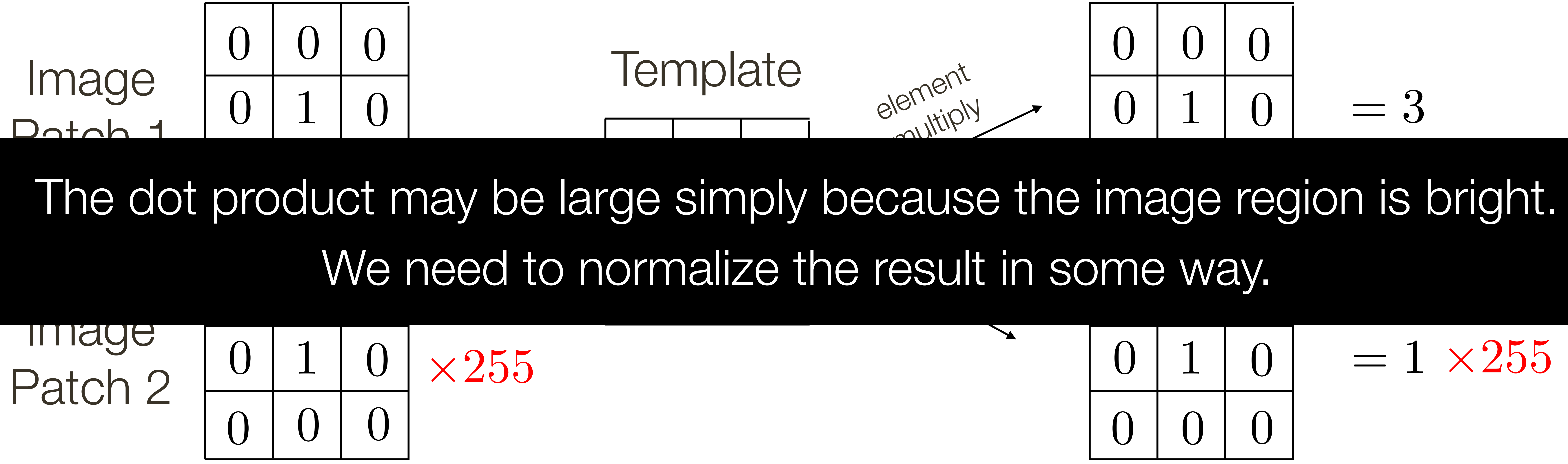


Template Matching

We can think of convolution/**correlation** as comparing a template (the filter) with each local image patch.

— Consider the filter and image patch as vectors.

— Applying a filter at an image location can be interpreted as computing the dot product between the filter and the local image patch.



The dot product may be large simply because the image region is bright.
We need to normalize the result in some way.

Template Matching

Similarity measures between a filter $\mathbf{J}$ local image region $\mathbf{I}$

Correlation, CORR = $\mathbf{I} \cdot \mathbf{J} = \mathbf{I}^T \mathbf{J}$

Normalised Correlation, NCORR = $\dfrac{\mathbf{I}^T \mathbf{J}}{|\mathbf{I}||\mathbf{J}|} = \cos\theta$

Normalized correlation varies between –1 and 1, attains the value 1 when the filter and image region are identical (up to a scale factor)

Because images are positive, the range would actually be 0 to 1

Template Matching

Similarity measures between a filter $\mathbf{J}$ local image region $\mathbf{I}$

Correlation, CORR = $\mathbf{I} \cdot \mathbf{J} = \mathbf{I}^T \mathbf{J}$

Normalised Correlation, NCORR = $\dfrac{\mathbf{I}^T \mathbf{J}}{|\mathbf{I}||\mathbf{J}|} = \cos\theta$

Normalized correlation varies between –1 and 1, attains the value 1 when the filter and image region are identical (up to a scale factor)

Because images are positive, the range would actually be 0 to 1

Template Matching

Similarity measures between a filter $\mathbf{J}$ local image region $\mathbf{I}$

Correlation, CORR = $\mathbf{I} \cdot \mathbf{J} = \mathbf{I}^T \mathbf{J}$

Normalised Correlation, NCORR = $\dfrac{\mathbf{I}^T \mathbf{J}}{|\mathbf{I}||\mathbf{J}|} = \cos\theta$

Normalized correlation varies between –1 and 1, attains the value 1 when the filter and image region are identical (up to a scale factor)

Because images are positive, the range would actually be 0 to 1

Template Matching

Similarity measures between a filter $\mathbf{J}$ local image region $\mathbf{I}$

Correlation, CORR = $\mathbf{I} \cdot \mathbf{J} = \mathbf{I}^T \mathbf{J}$

Normalised Correlation, NCORR = $\dfrac{\mathbf{I}^T \mathbf{J}}{|\mathbf{I}||\mathbf{J}|} = \cos\theta$

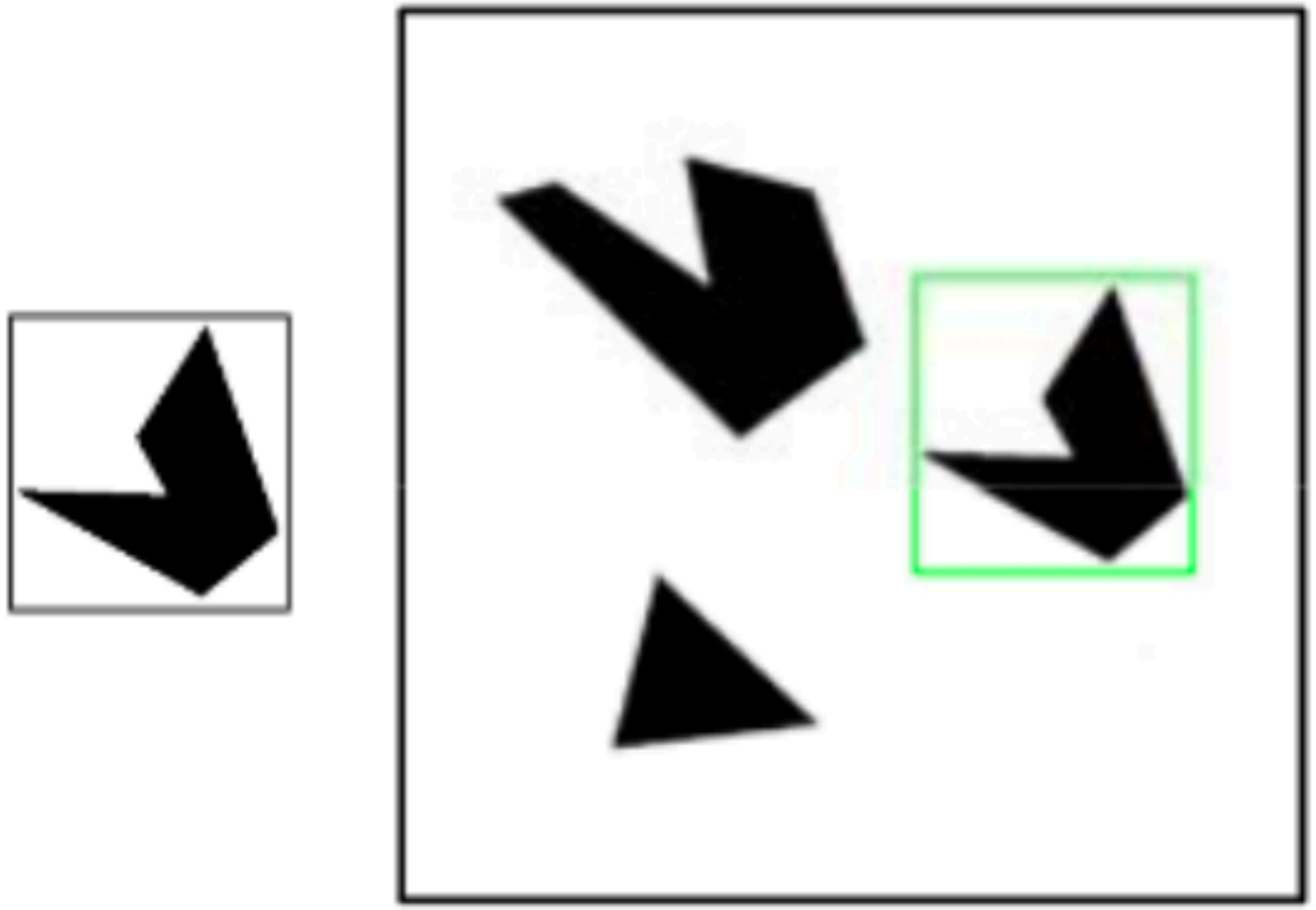
Sum Squared Difference, SSD = $|\mathbf{I} - \mathbf{J}|^2$

Normalized correlation varies between –1 and 1, attains the value 1 when the filter and image region are identical (up to a scale factor)

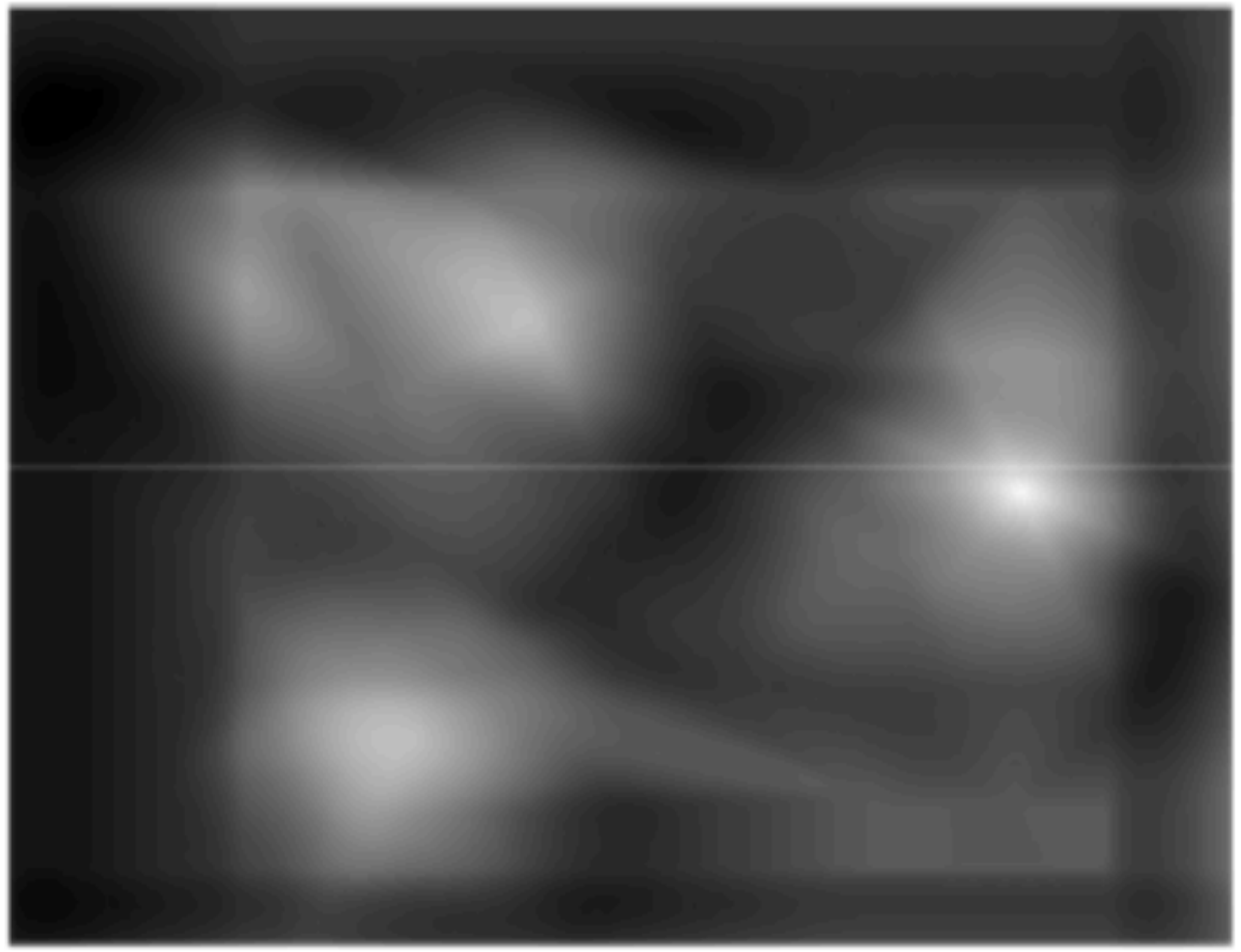
Minimising SSD and maximizing Normalized Correlation are equivalent if $|\mathbf{I}| = |\mathbf{J}| = 1$

Template Matching

Assuming template is all positive, what does this tell us about correlation map?



Detected template

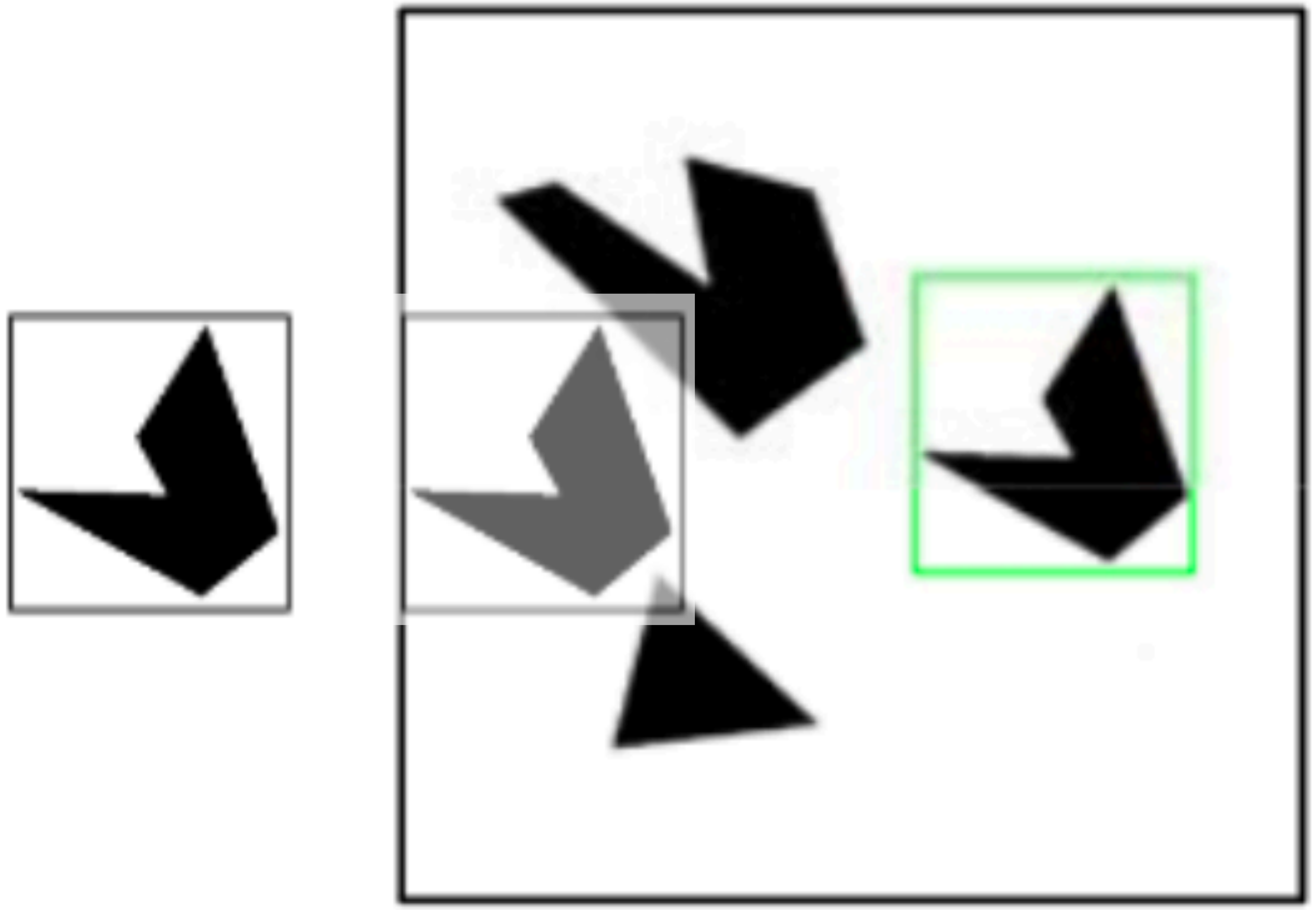


Correlation map

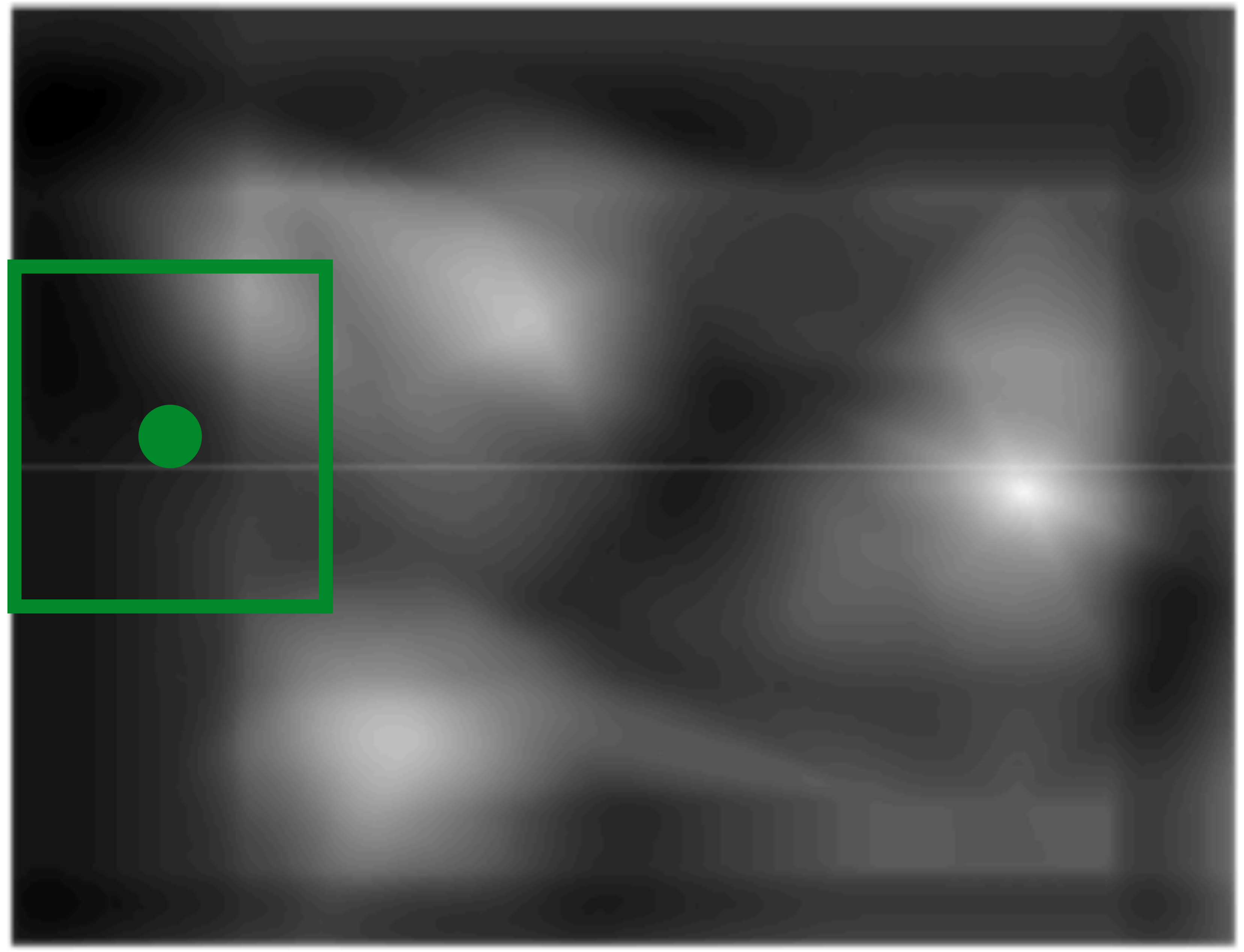
Slide Credit: Kristen Grauman

Template Matching

Assuming template is all positive, what does this tell us about correlation map?



Detected template



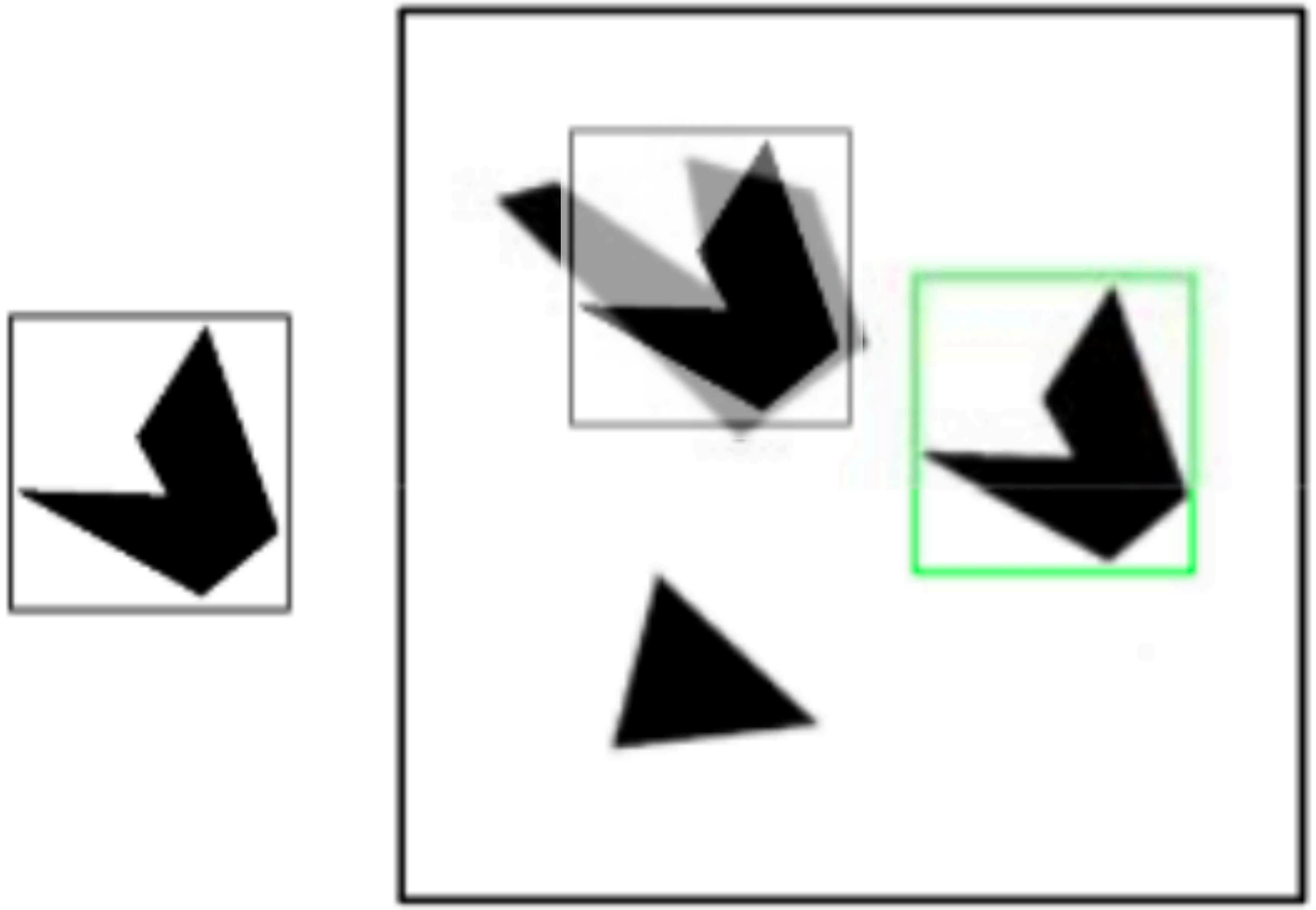
Correlation map

$$\frac{a}{|a|}\frac{b}{|b|} = ?$$

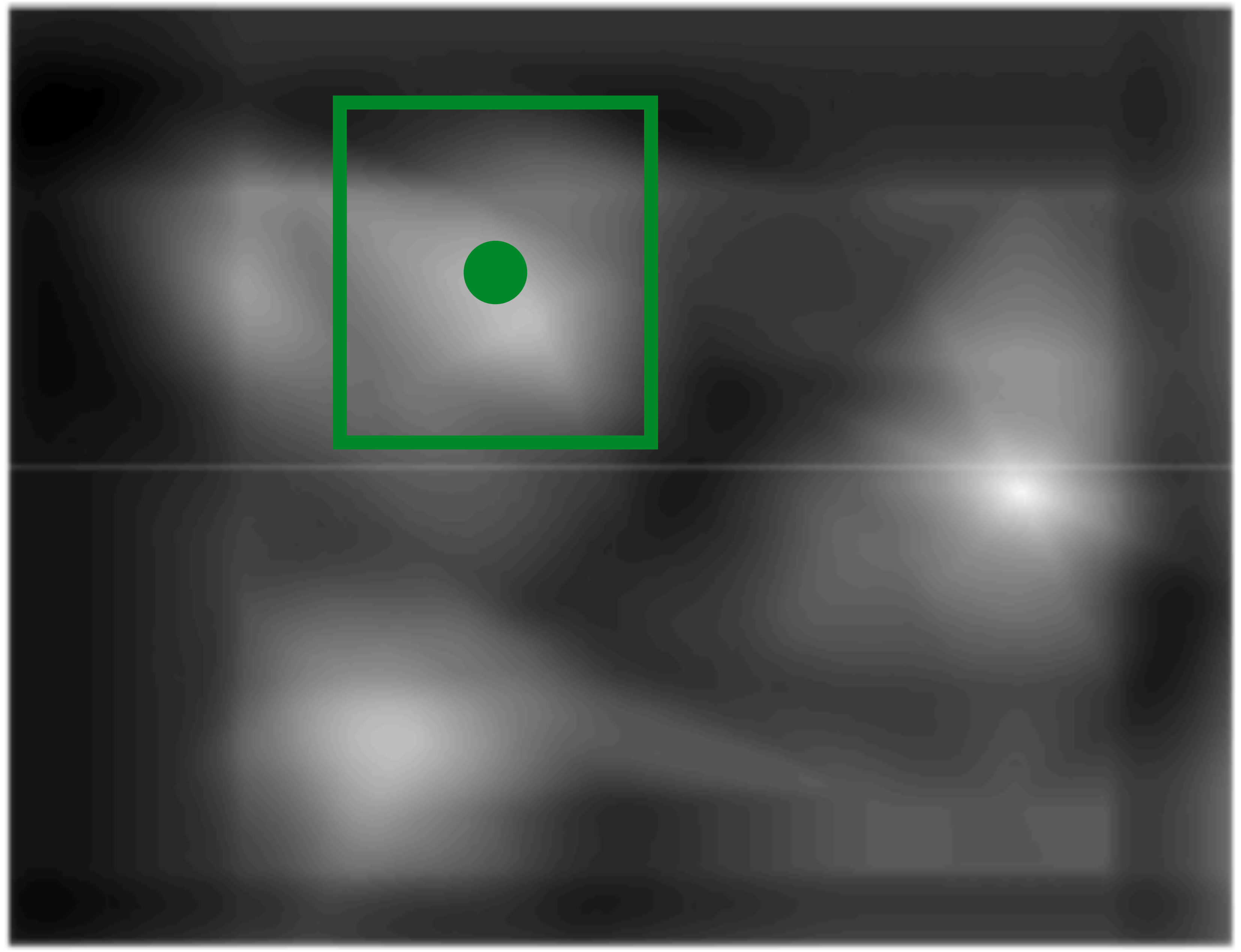
Slide Credit: Kristen Grauman

Template Matching

Assuming template is all positive, what does this tell us about correlation map?



Detected template



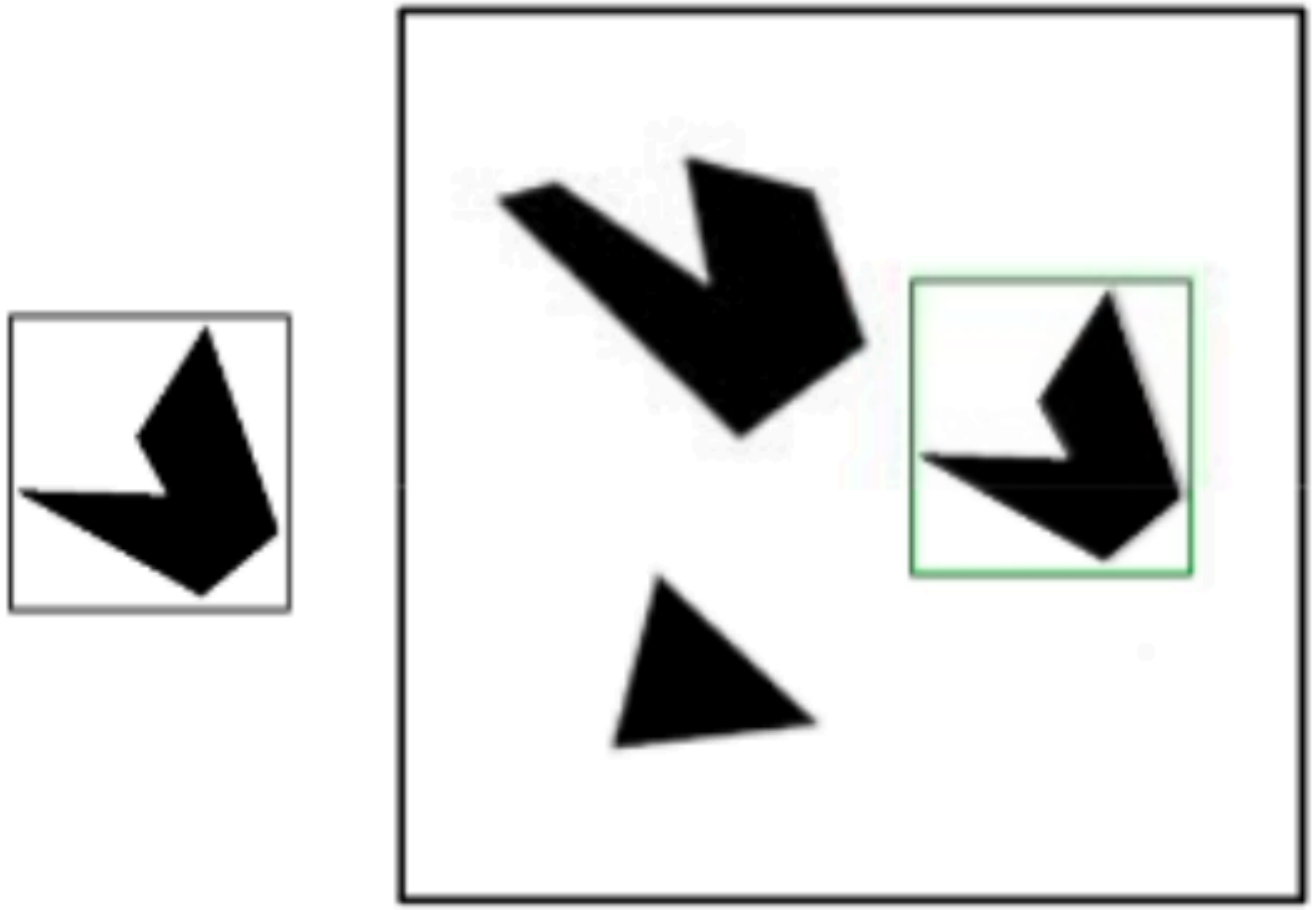
Correlation map

$$\frac{a}{|a|}\frac{b}{|b|} = ?$$

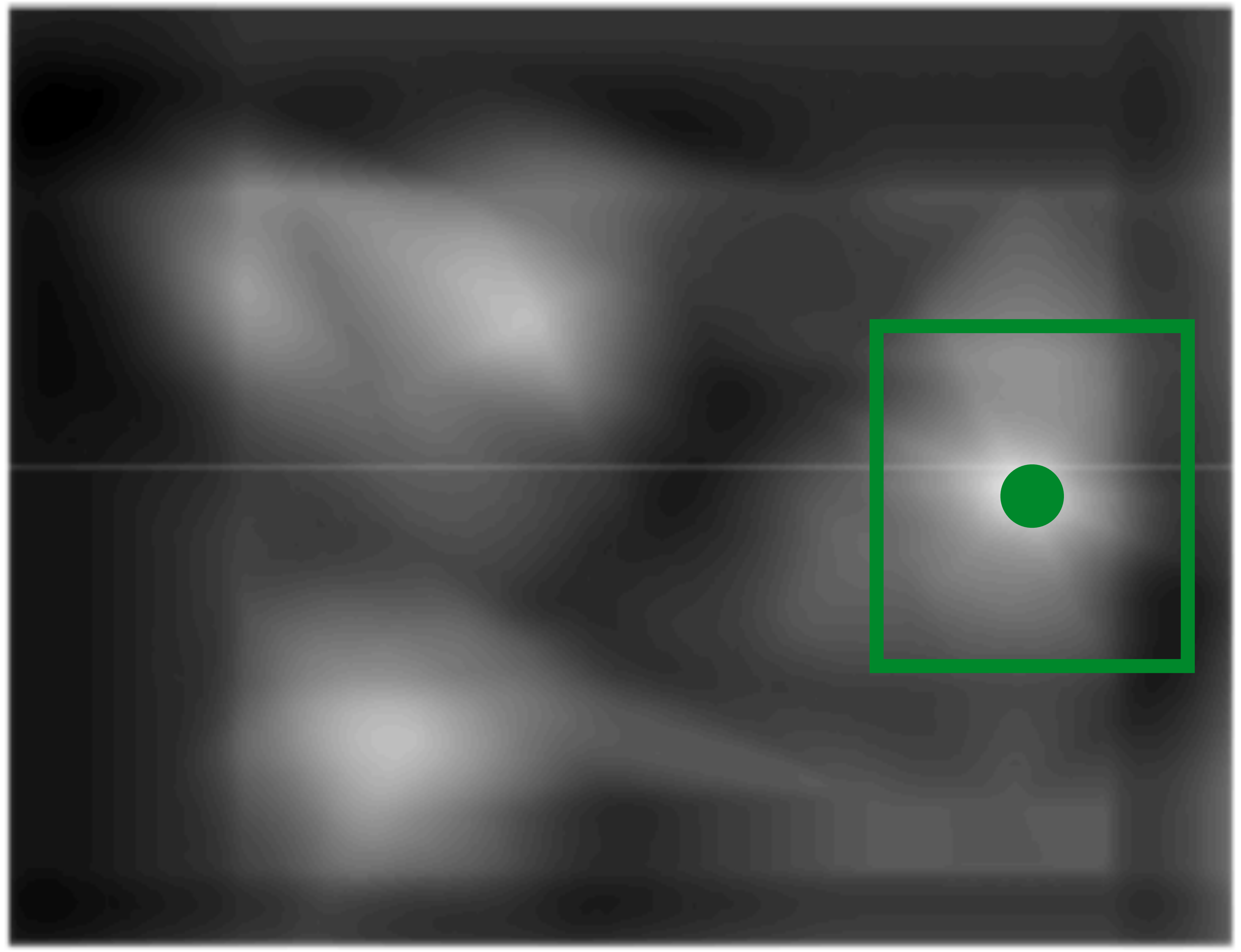
Slide Credit: Kristen Grauman

Template Matching

Assuming template is all positive, what does this tell us about correlation map?



Detected template



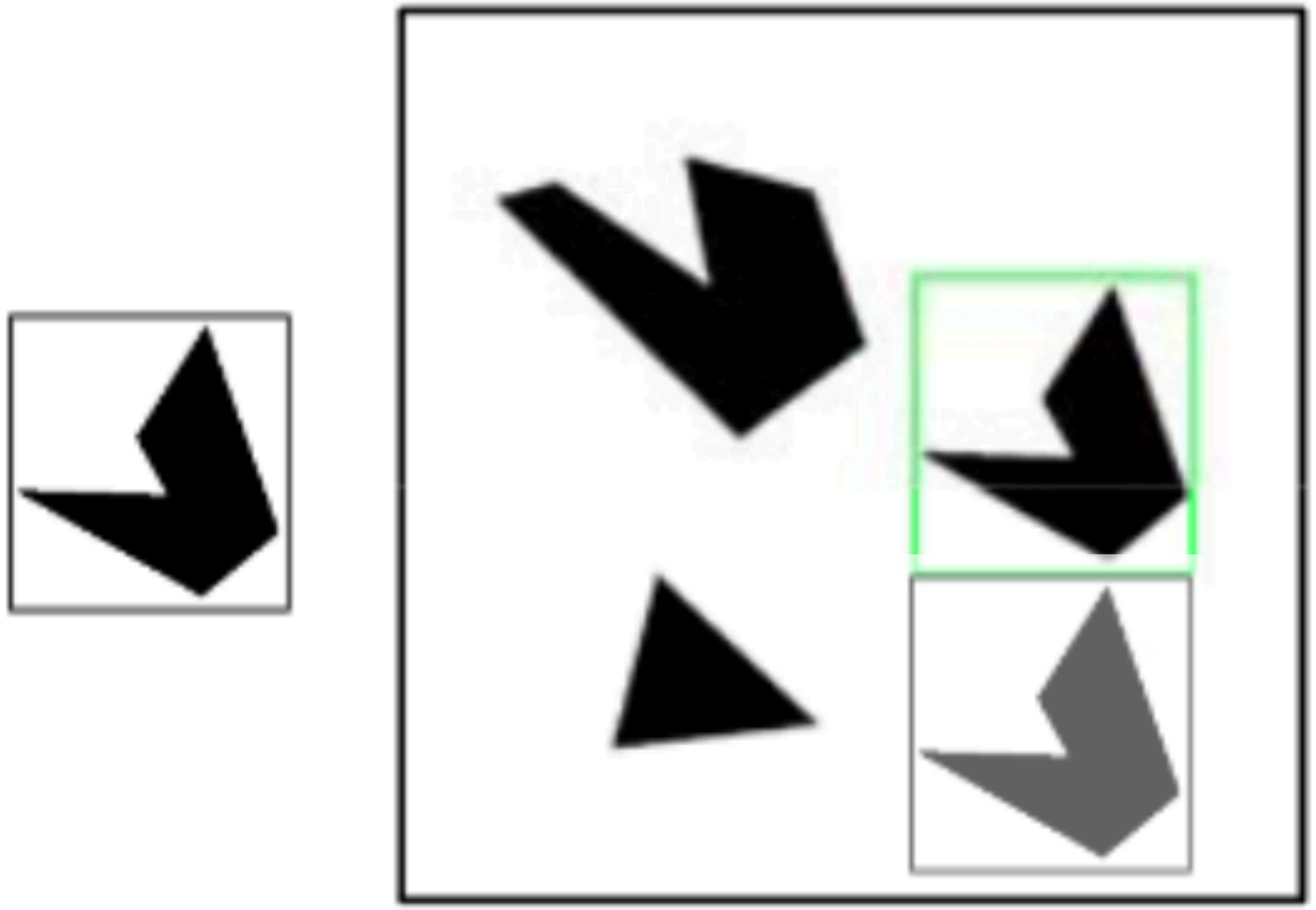
Correlation map

$$\frac{a}{|a|}\frac{b}{|b|} = ?$$

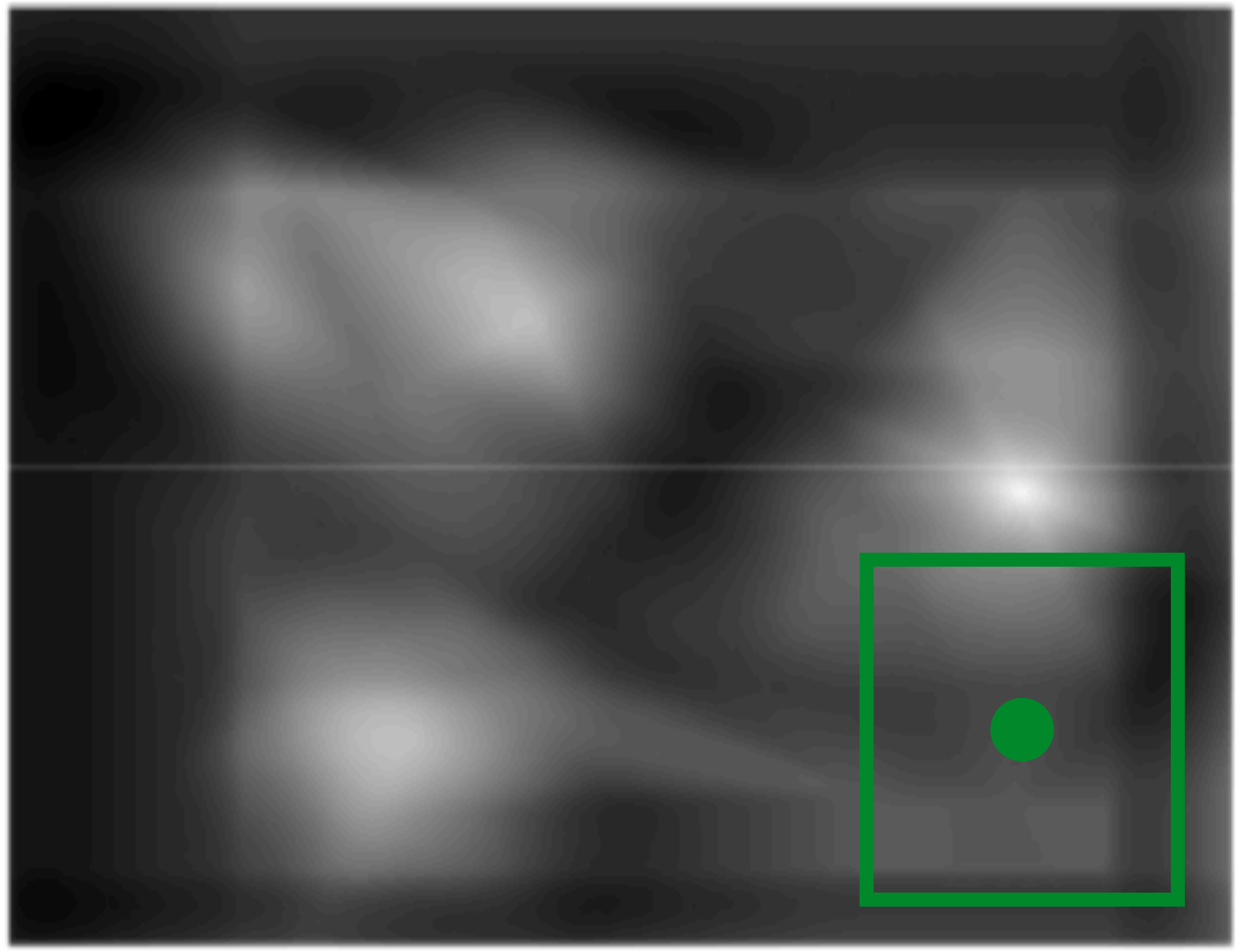
Slide Credit: Kristen Grauman

Template Matching

Assuming template is all positive, what does this tell us about correlation map?



Detected template



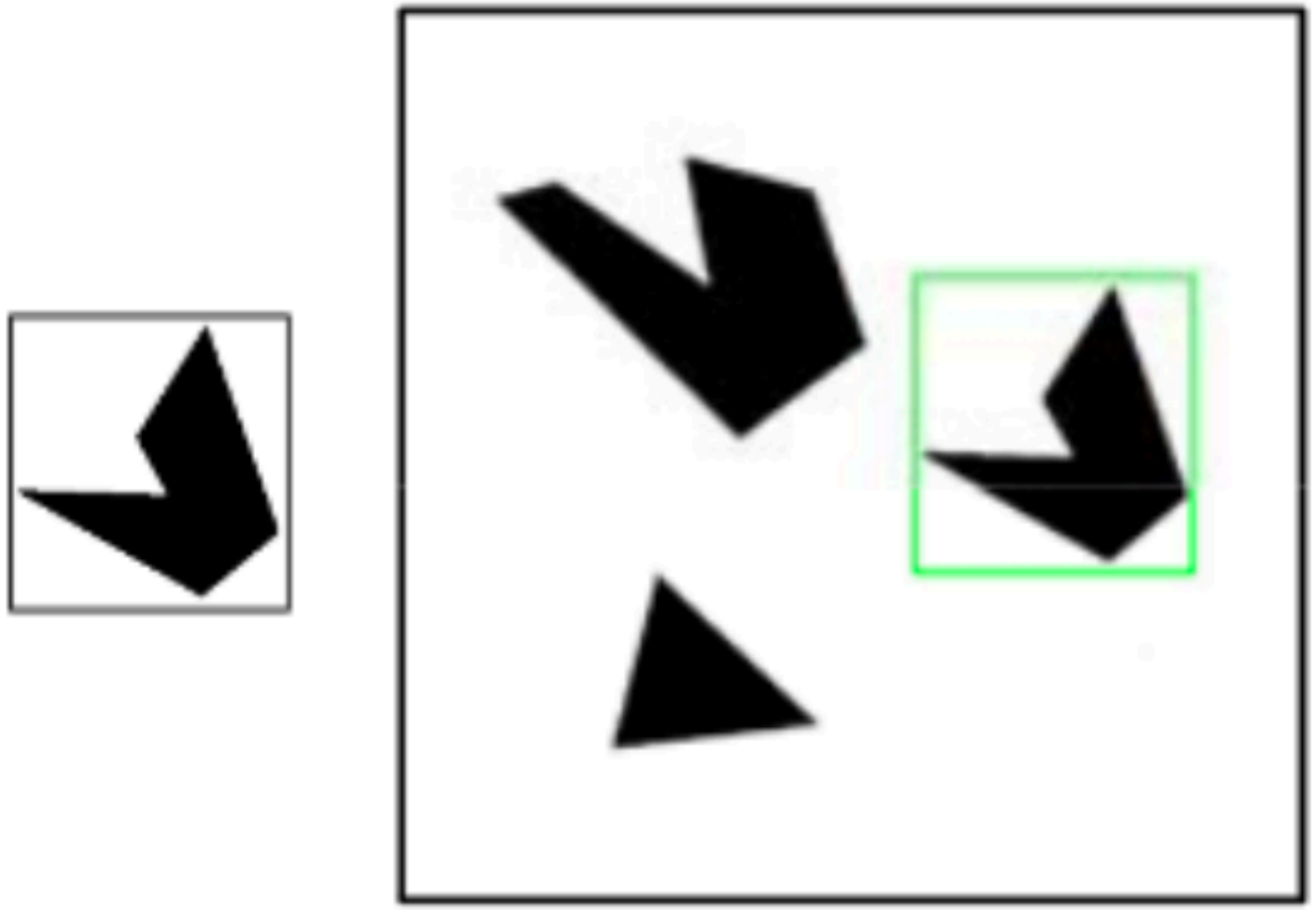
Correlation map

$$\frac{a}{|a|}\frac{b}{|b|} = ?$$

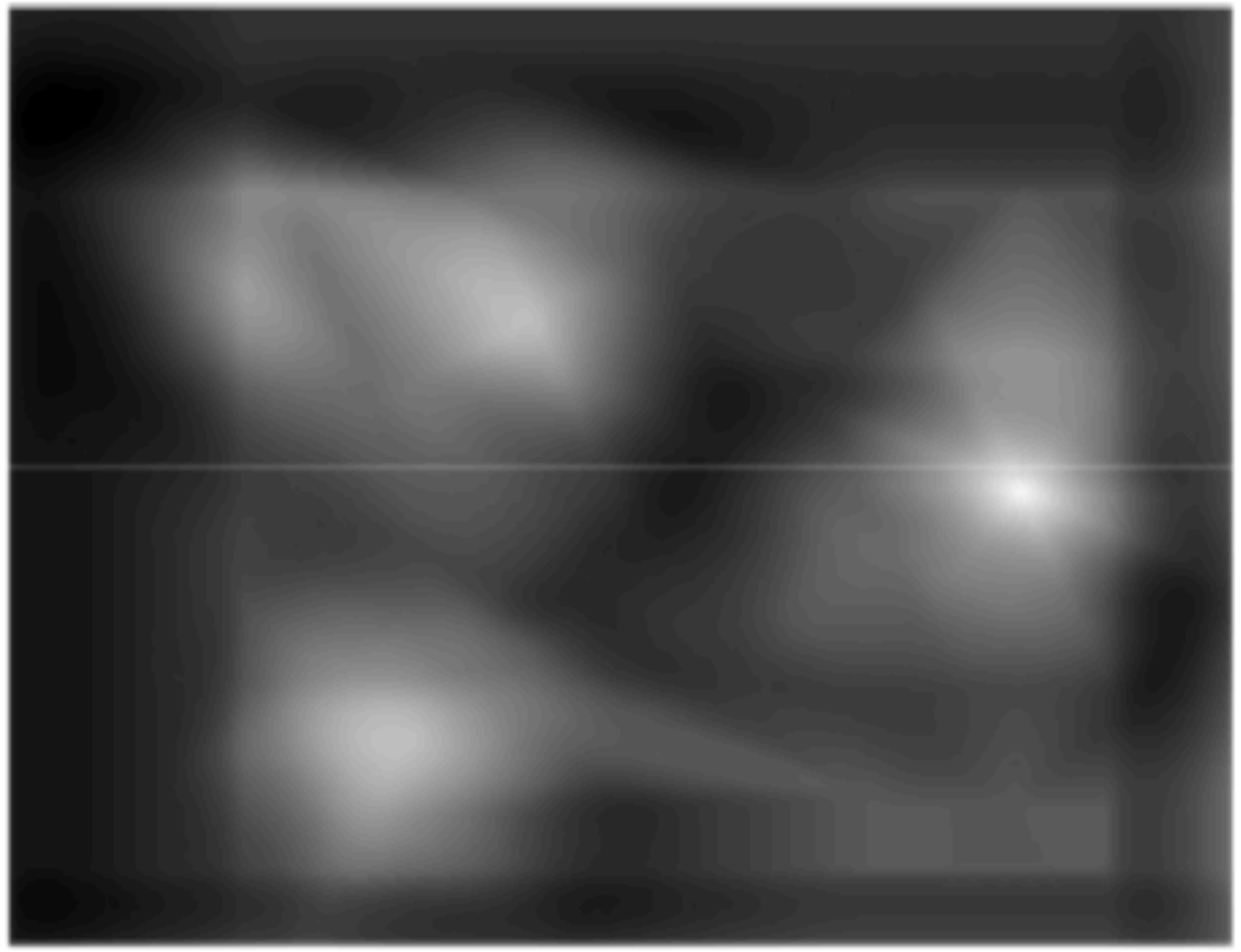
Slide Credit: Kristen Grauman

Template Matching

Detection can be done by comparing correlation map score to a threshold



Detected template



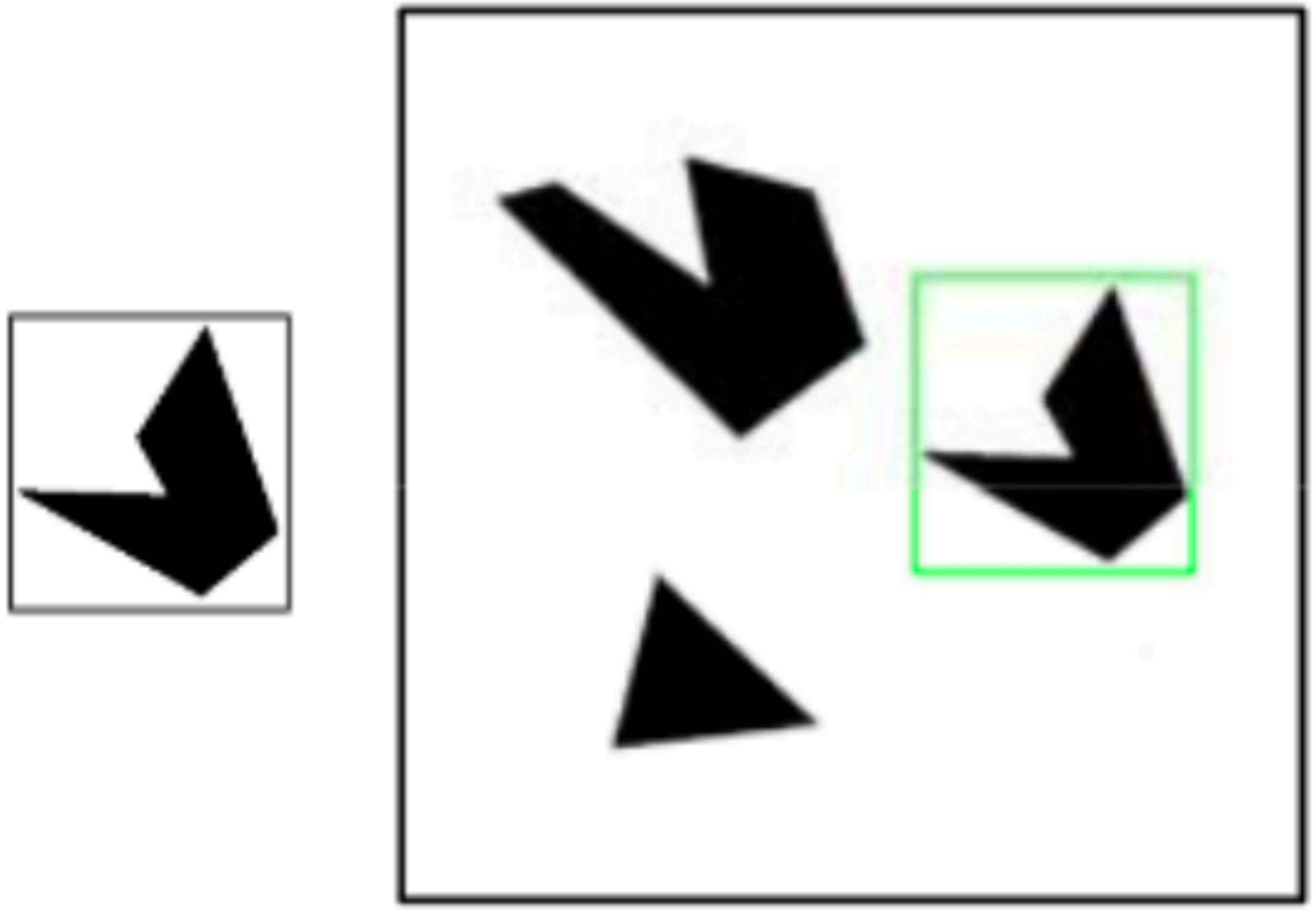
Correlation map

What happens if the threshold is relatively low?

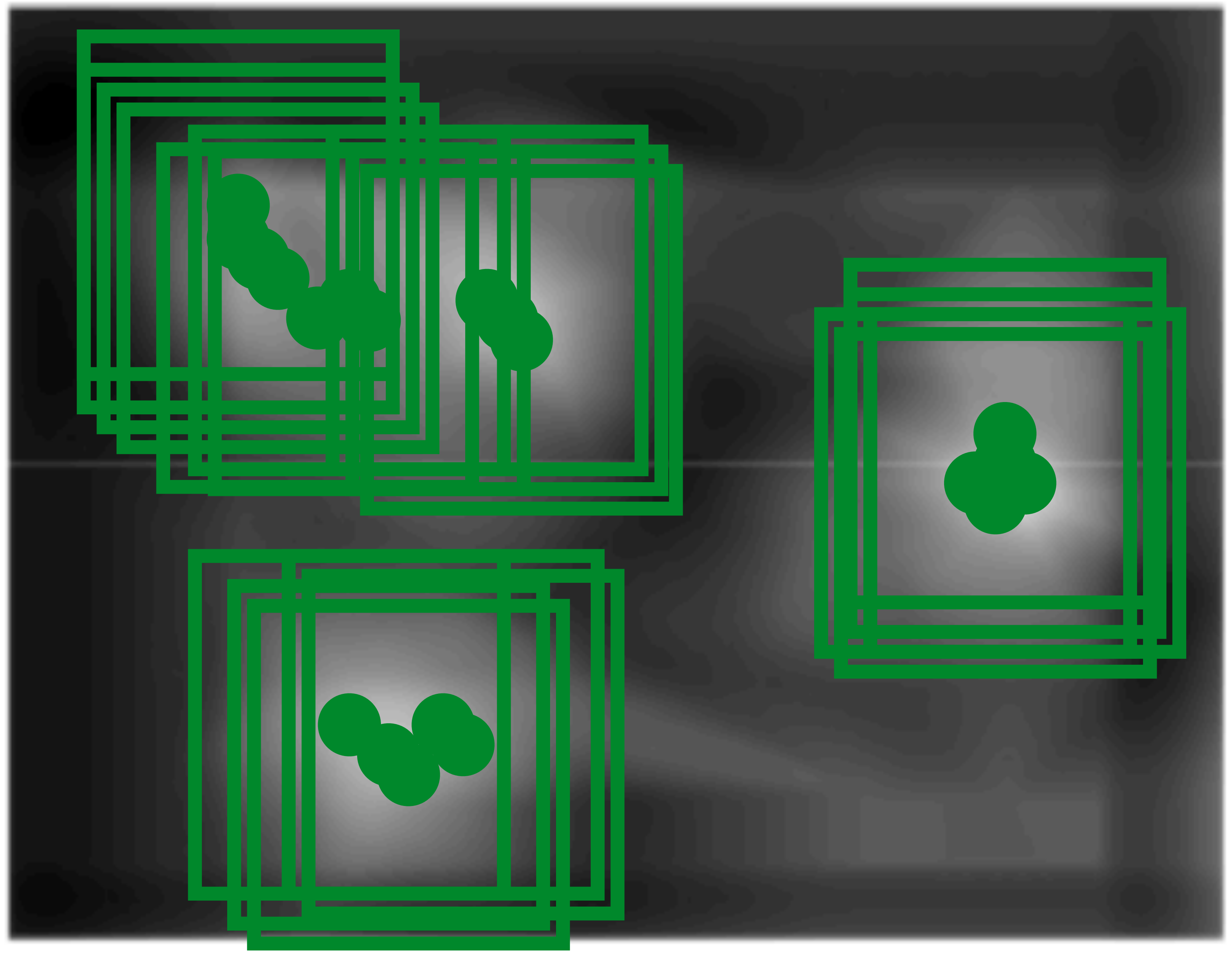
Slide Credit: Kristen Grauman

Template Matching

Detection can be done by comparing correlation map score to a threshold



Detected template



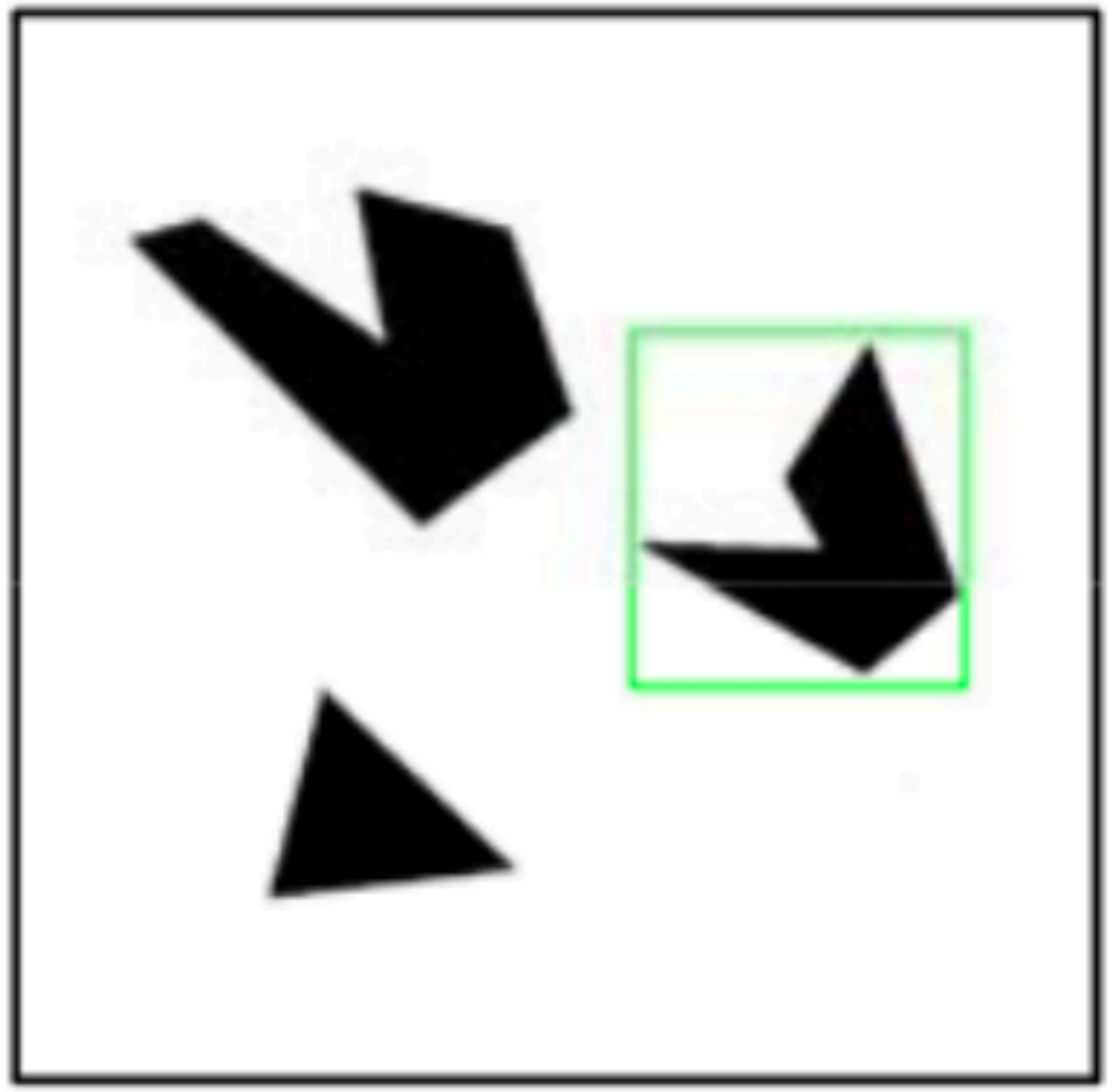
Correlation map

What happens if the threshold is relatively low?

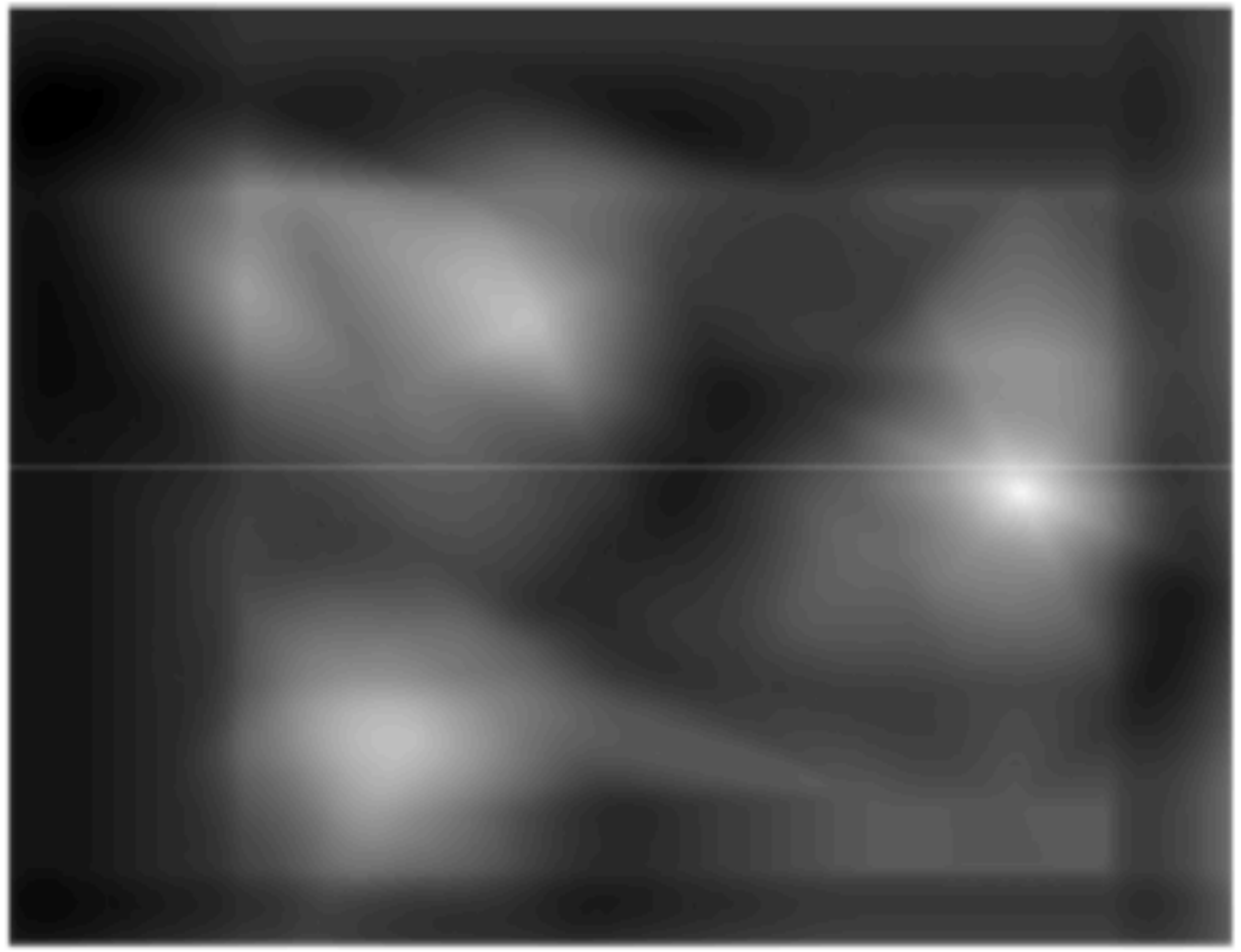
Slide Credit: Kristen Grauman

Template Matching

Detection can be done by comparing correlation map score to a threshold



Detected template



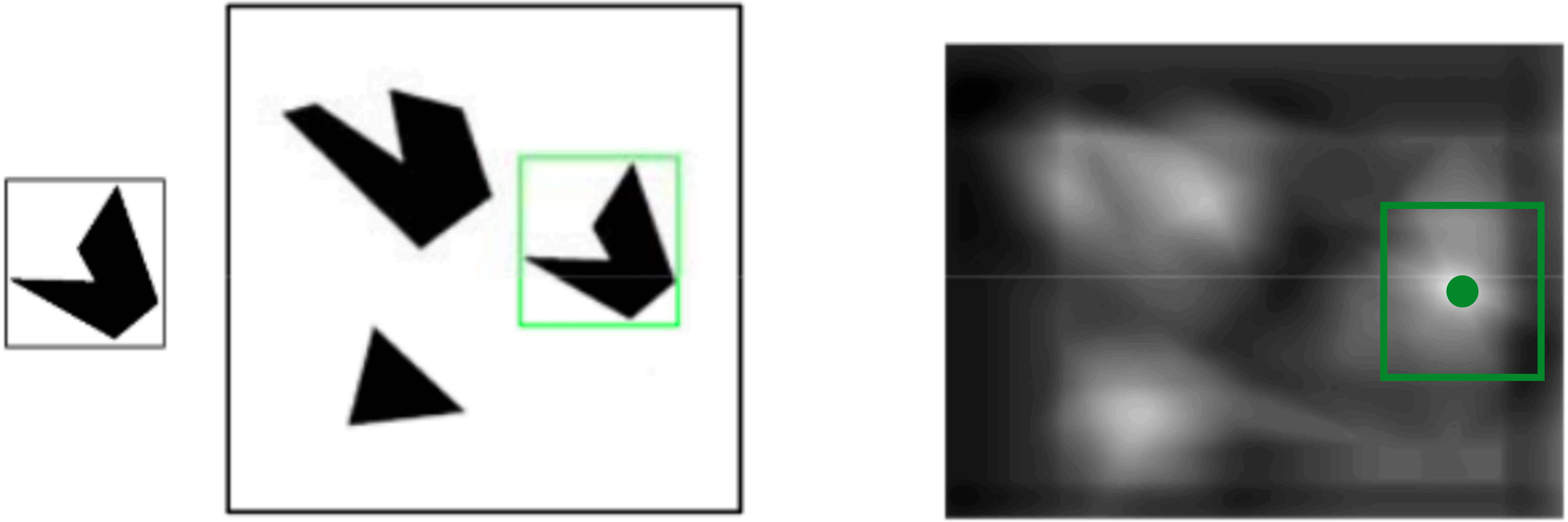
Correlation map

What happens if the threshold is very high (e.g., 0.99)?

Slide Credit: Kristen Grauman

Template Matching

Detection can be done by comparing correlation map score to a threshold



Detected template | **Correlation map**

What happens if the threshold is very high (e.g., 0.99)?

Slide Credit: Kristen Grauman

Template Matching

Linear filtering the entire image computes the entire set of dot products, one for each possible alignment of filter and image

Important Insight:

— filters look like the pattern they are intended to find

— filters find patterns they look like

Linear filtering is sometimes referred to as **template matching**

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\boxed{\frac{b}{|b|}}$$

where · is dot product and | | is vector magnitude

1. Normalize the template / filter (b) in the beginning

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

Template (*b*)

5	7	98
14	8	32
24	9	63

where · is dot product and | | is vector magnitude

1. Normalize the template / filter (b) in the beginning

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

1. Normalize the template / filter (b) in the beginning

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a\cdot b}{|a||b|} = \frac{a\cdot b}{\sqrt{(a\cdot a)(b\cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Image (*a*)

1	17	3	5
43	24	1	11
13	24	8	15
6	17	9	19

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Image (*a*)

1	17	3	5
43	24	1	11
13	24	8	15
6	17	9	19

square →

1	289	9	25
1849	576	1	121
169	576	64	225
36	289	81	361

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Image (*a*)

1	17	3	5
43	24	1	11
13	24	8	15
6	17	9	19

square →

1	289	9	25
1849	576	1	121
169	576	64	225
36	289	81	361

convolve →

1	1	1
1	1	1
1	1	1

0	0	0	0
0	3378	1886	0
0	3485	2294	0
0	0	0	0

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Image (*a*)

1	17	3	5
43	24	1	11
13	24	8	15
6	17	9	19

square →

1	289	9	25
1849	576	1	121
169	576	64	225
36	289	81	361

convolve →

1	1	1
1	1	1
1	1	1

0	0	0	0
0	3378	1886	0
0	3485	2294	0
0	0	0	0

sqrt →

0	0	0	0
0	58.12	43.43	0
0	59.03	47.90	0
0	0	0	0

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Image (*a*)

1	17	3	5
43	24	1	11
13	24	8	15
6	17	9	19

square →

1	289	9	25
1849	576	1	121
169	576	64	225
36	289	81	361

convolve →

1	1	1
1	1	1
1	1	1

0	0	0	0
0	3378	1886	0
0	3485	2294	0
0	0	0	0

sqrt →

0	0	0	0
0	58.12	43.43	0
0	59.03	47.90	0
0	0	0	0

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|}\frac{b}{|b|}$$

where · is dot product and | | is vector magnitude

Template (*b*)

$\frac{1}{156.70}$

5	7	98
14	8	32
24	9	63

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response

Image (*a*)

1	17	3	5
43	24	1	11
13	24	8	15
6	17	9	19

square →

1	289	9	25
1849	576	1	121
169	576	64	225
36	289	81	361

convolve →

1	1	1
1	1	1
1	1	1

0	0	0	0
0	3378	1886	0
0	3485	2294	0
0	0	0	0

sqrt →

0	0	0	0
0	58.12	43.43	0
0	59.03	47.90	0
0	0	0	0

Template Matching

Let a and b be vectors. Let θ be the angle between them. We know

$$\cos\theta = \frac{a \cdot b}{|a||b|} = \frac{a \cdot b}{\sqrt{(a \cdot a)(b \cdot b)}} = \frac{a}{|a|} \frac{b}{|b|}$$

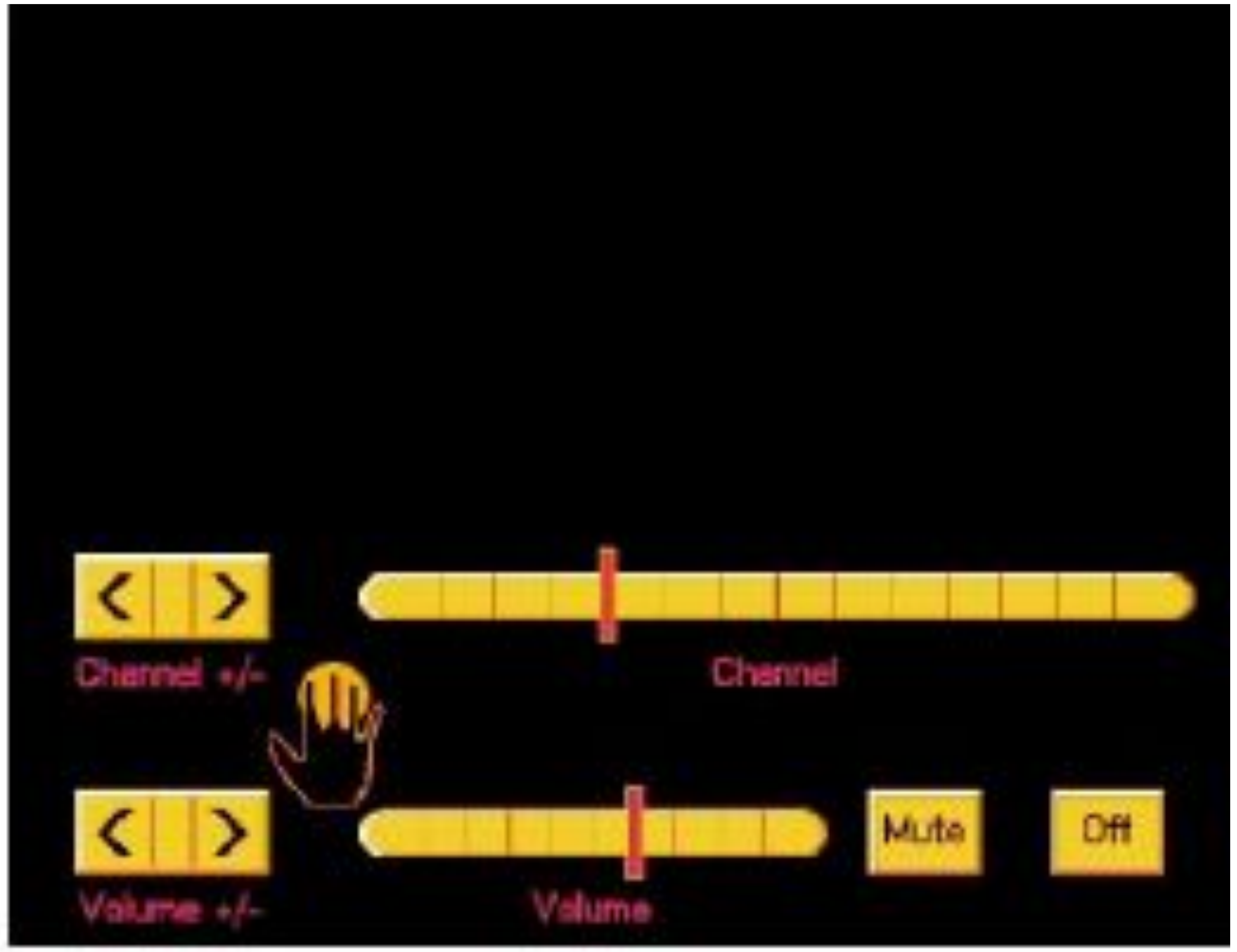
where · is dot product and | | is vector magnitude

1. Normalize the template / filter (b) in the beginning
2. Compute norm of $|a|$ by convolving squared image with a filter of all 1's of equal size to the the template and square-rooting the response
3. We can compute the dot product by correlation of image (a) with normalized filter (b)
4. We can finally compute the normalized correlation by dividing element-wise result in Step 3 by result in Step 2

Example 1:

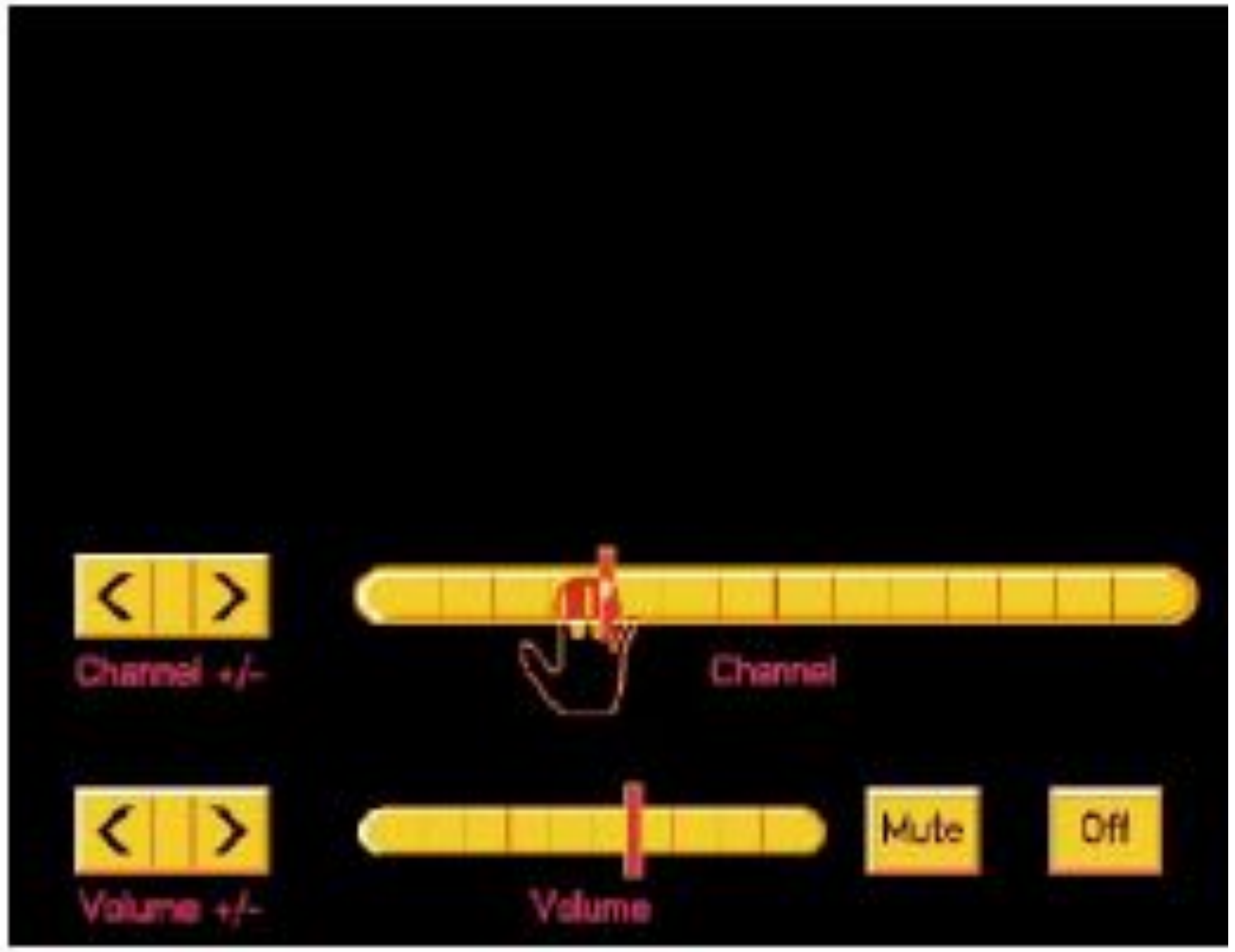
Credit: W. Freeman et al., “Computer Vision for Interactive Computer Graphics,” IEEE Computer Graphics and Applications, 1998

Example 1:



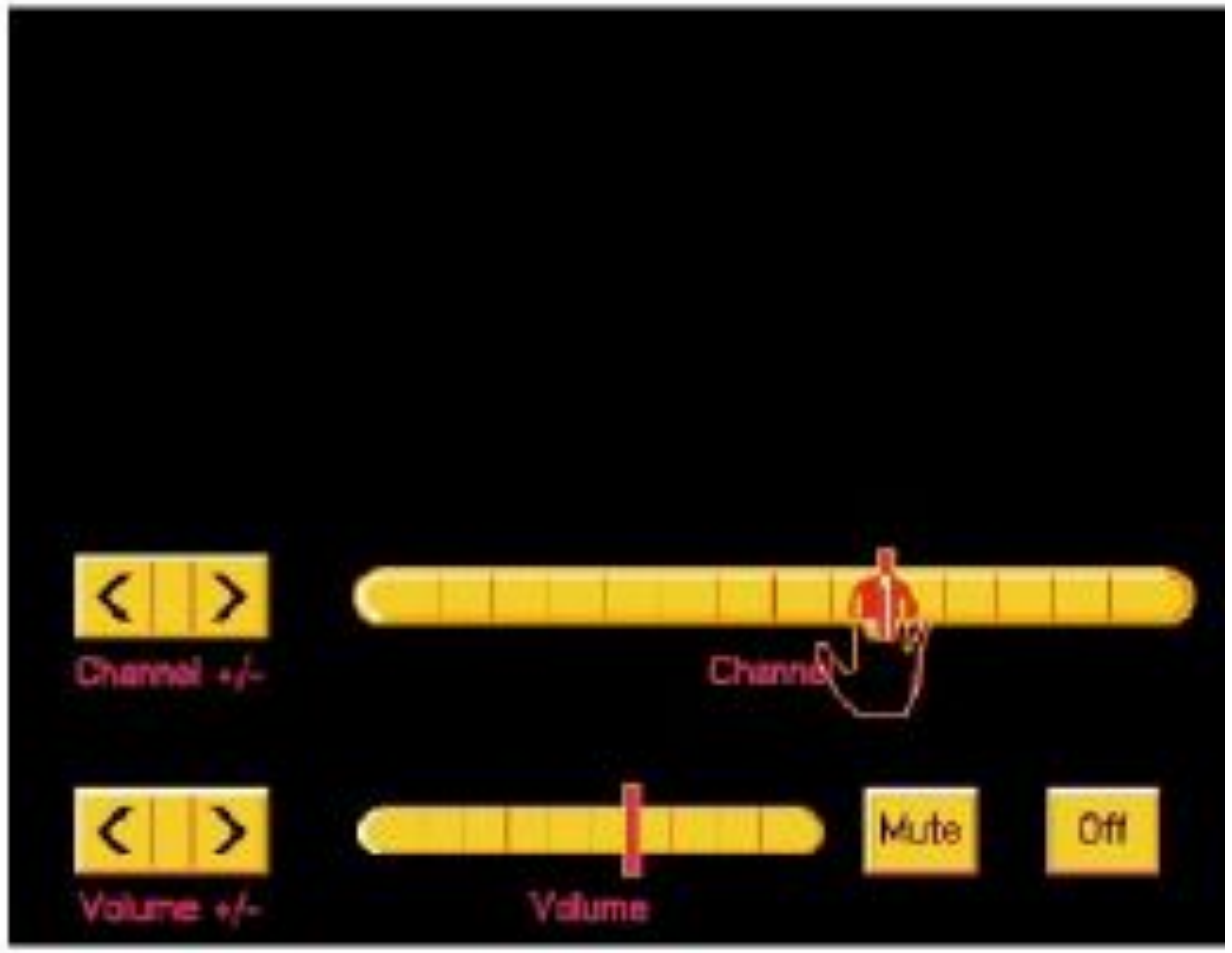
Credit: W. Freeman et al., "Computer Vision for Interactive Computer Graphics," IEEE Computer Graphics and Applications, 1998

Example 1:



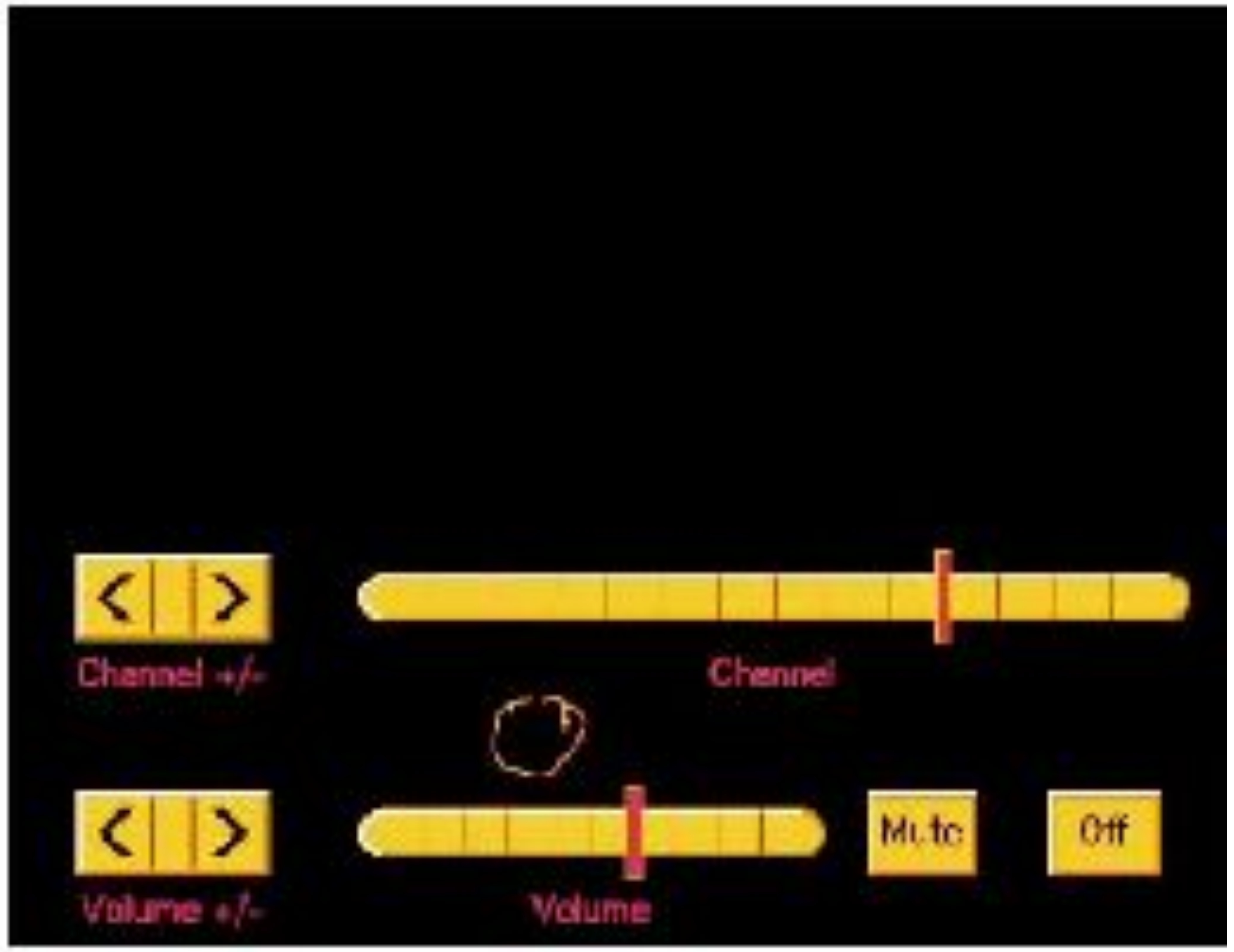
Credit: W. Freeman et al., "Computer Vision for Interactive Computer Graphics," IEEE Computer Graphics and Applications, 1998

Example 1:



Credit: W. Freeman et al., “Computer Vision for Interactive Computer Graphics,” IEEE Computer Graphics and Applications, 1998

Example 1:

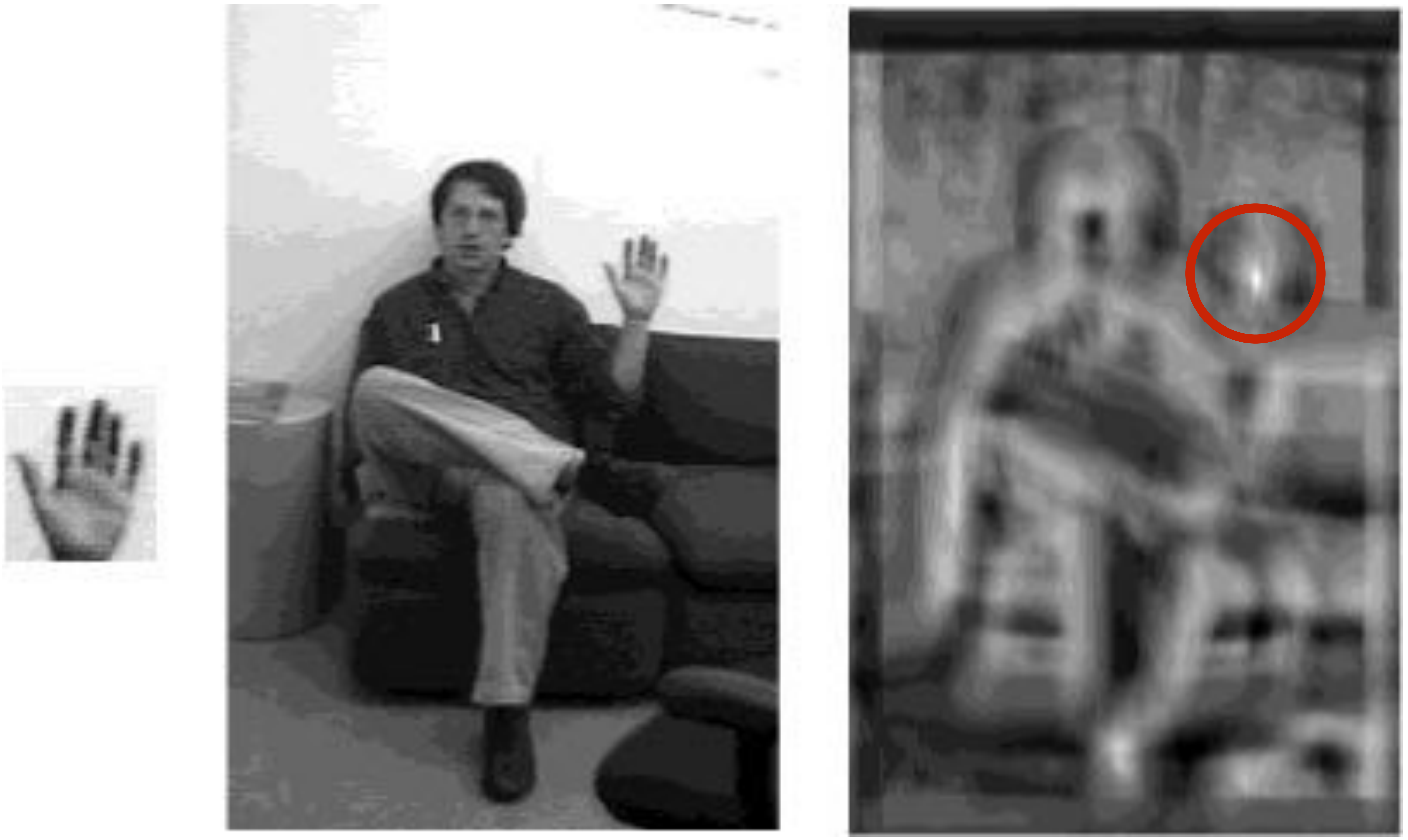


Credit: W. Freeman et al., "Computer Vision for Interactive Computer Graphics," IEEE Computer Graphics and Applications, 1998

Example 1:

Template (left), image (middle), normalized correlation (right)

Note peak value at the true position of the hand



Credit: W. Freeman et al., “Computer Vision for Interactive Computer Graphics,” IEEE Computer Graphics and Applications, 1998

Template Matching

When might **template matching fail**?

Template Matching

When might **template matching fail**?

— Different scales  

Template Matching

When might **template matching fail**?

— Different scales  

— Different orientation 

Template Matching

When might **template matching fail**?

— Different scales  

— Different orientation 

— Lighting conditions 

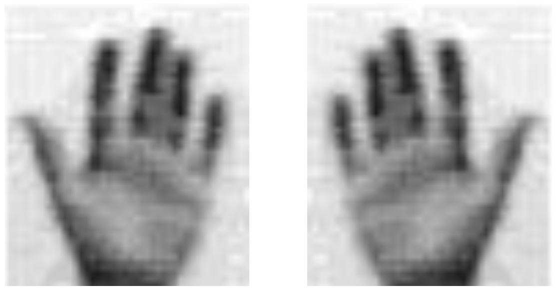
Template Matching

When might **template matching fail**?

— Different scales 

— Different orientation 

— Lighting conditions 

— Left vs. Right hand 

Template Matching

When might **template matching fail**?

— Different scales 

— Different orientation 

— Lighting conditions 

— Left vs. Right hand 

— Partial Occlusions 

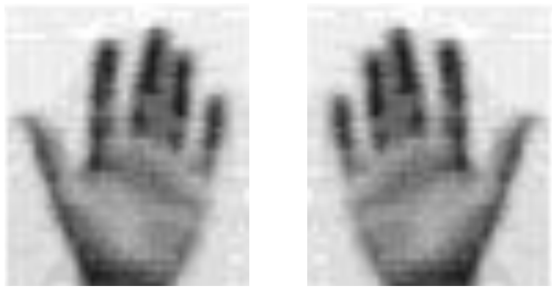
Template Matching

When might **template matching fail**?

— Different scales  

— Different orientation 

— Lighting conditions 

— Left vs. Right hand 

— Partial Occlusions 

— Different Perspective

— Motion / blur

Template Matching Summary

Good News:

— works well in presence of noise

— relatively easy to compute

Bad News:

— sensitive to (spatial) scale change

— sensitive to 2D rotation

More Bad News:

When imaging 3D worlds:

— sensitive to viewing direction and pose

— sensitive to conditions of illumination

Template Matching

When might **template matching fail**?

— Different scales 

Scaled Representations

N

N

M  = Template

M

Scaled Representations

N

N

M  = Template

M

$2M$ 

$2M$

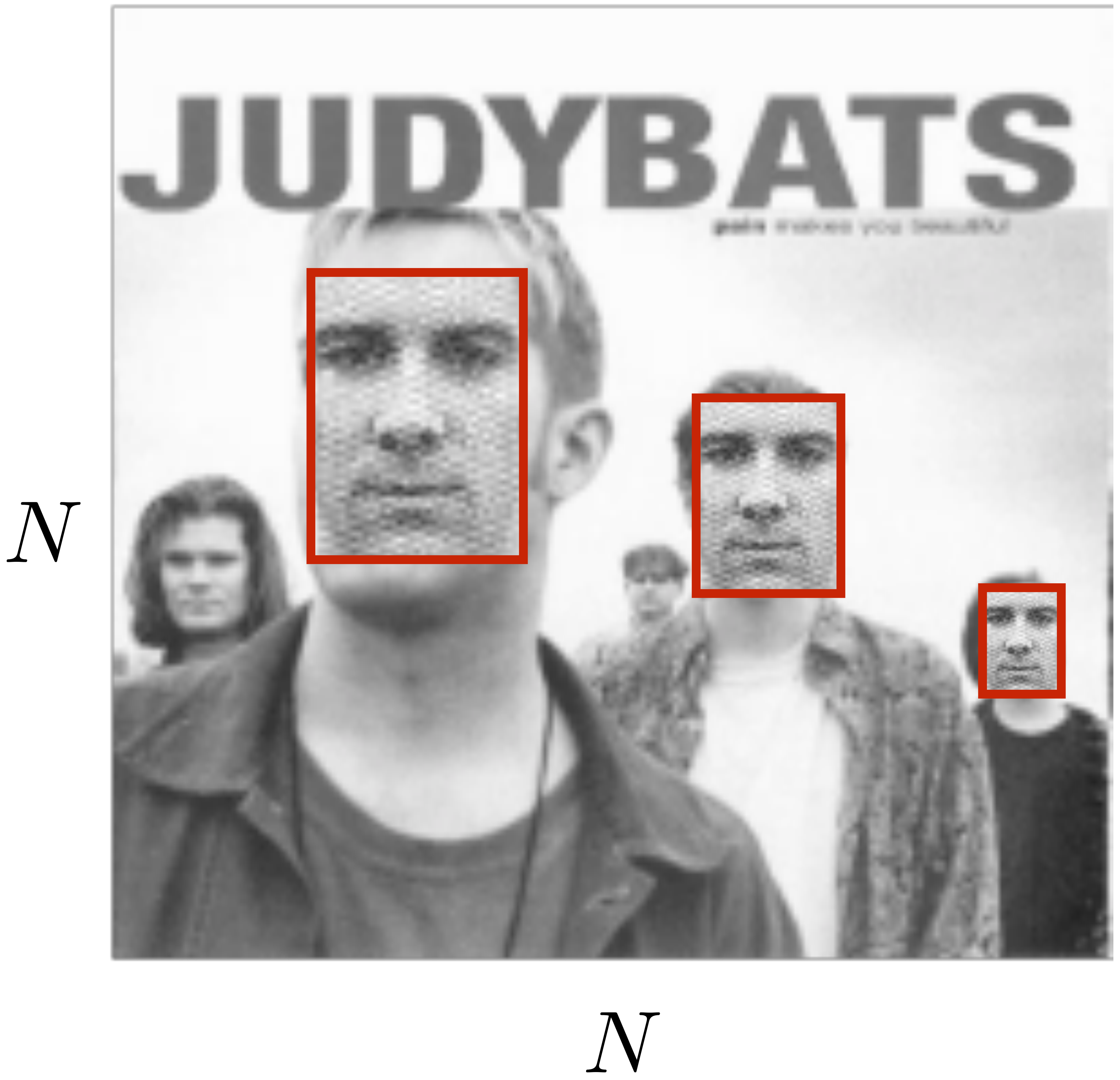
$4M$ 

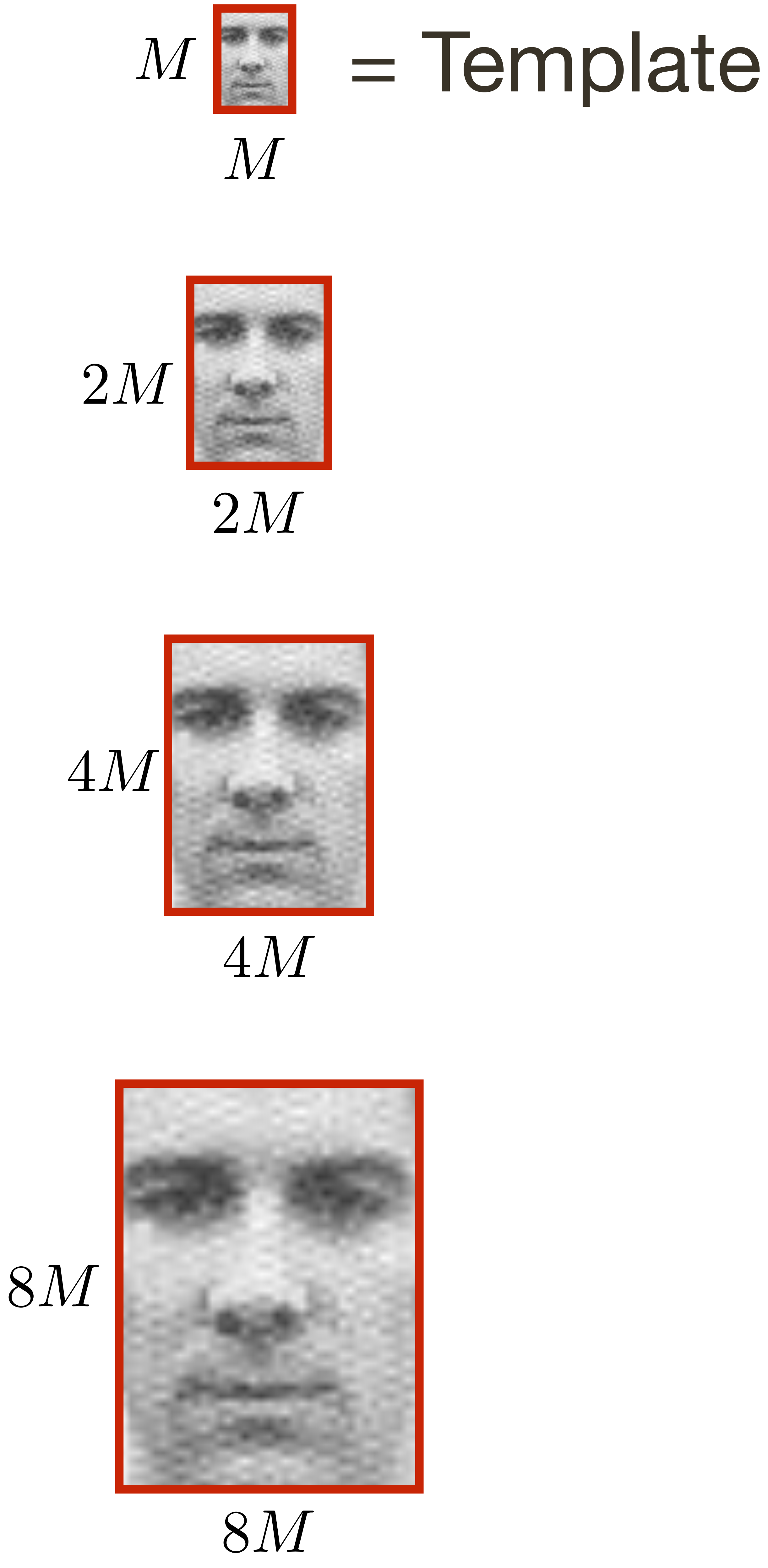
$4M$

$8M$ 

$8M$

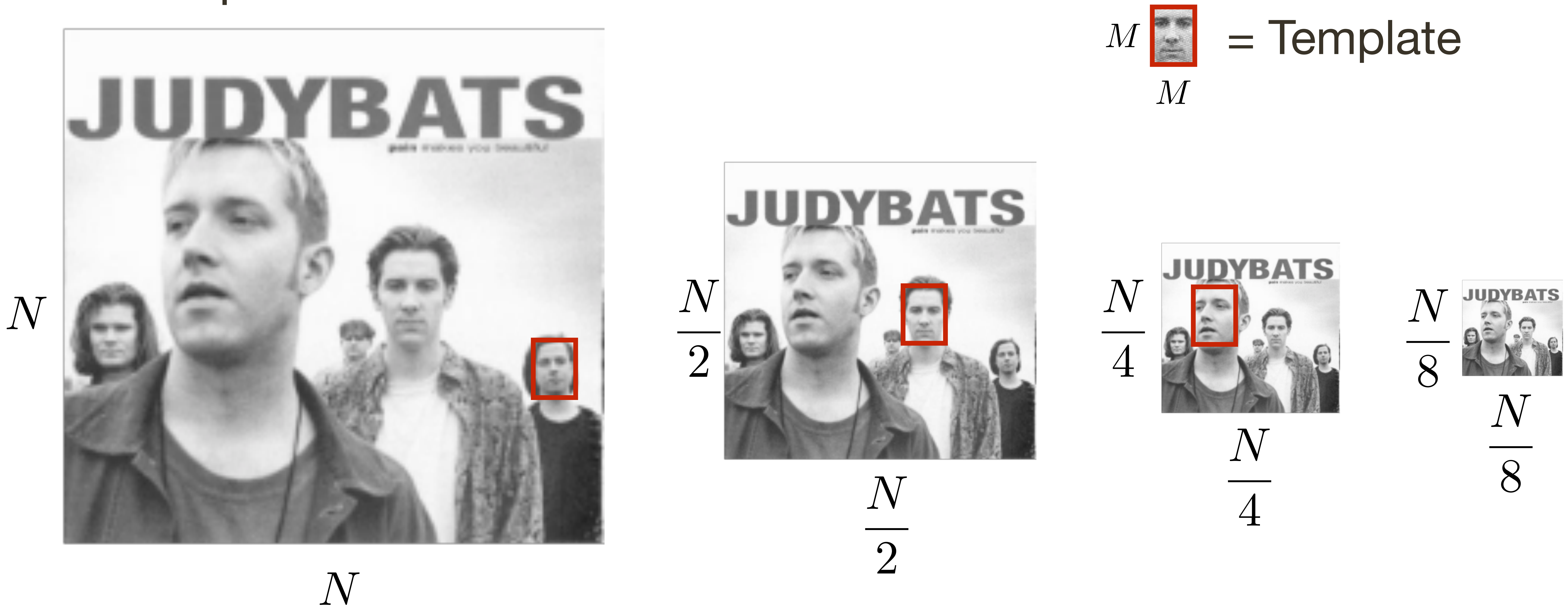
Scaled Representations



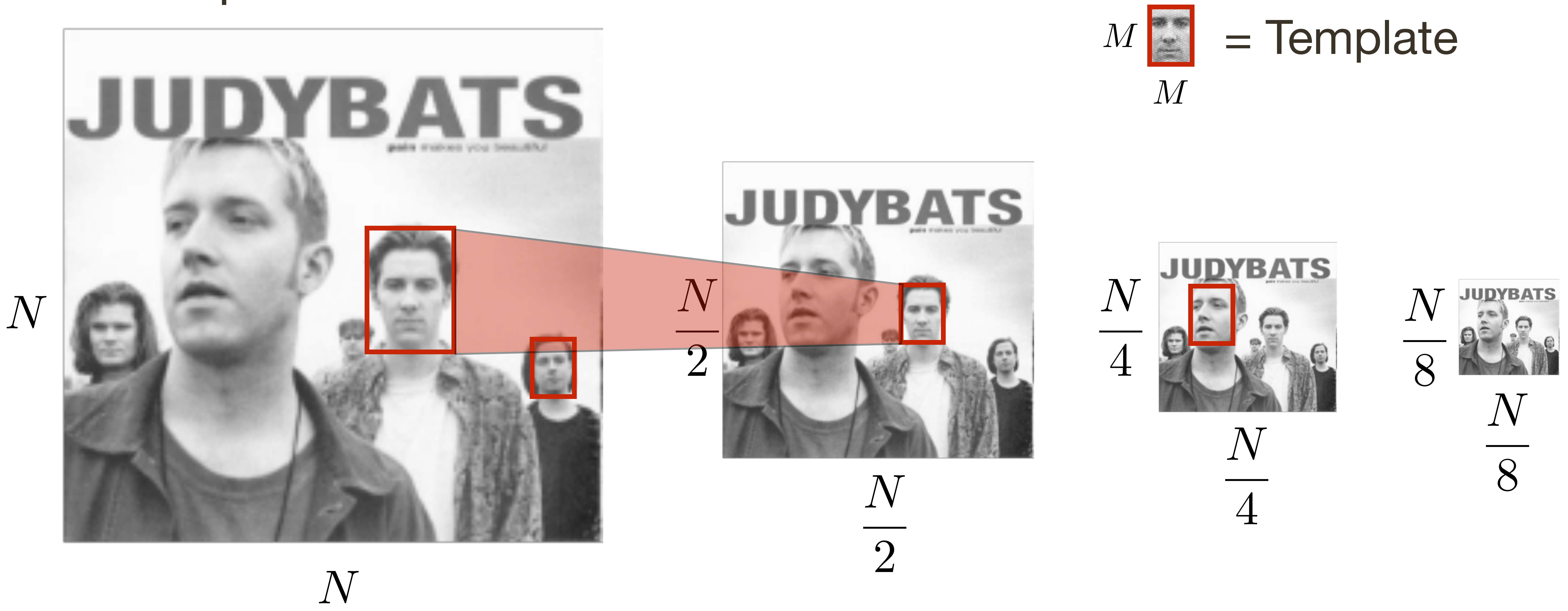


Scaled Representations

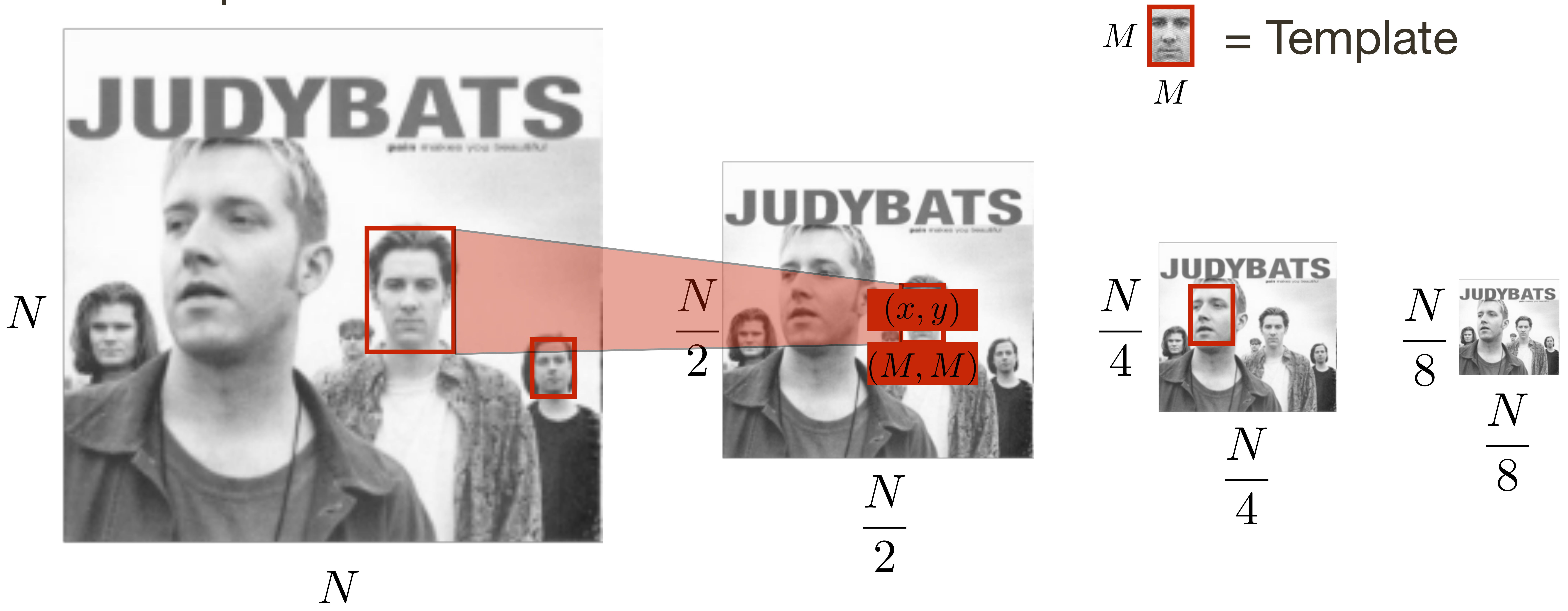
Scaled Representations



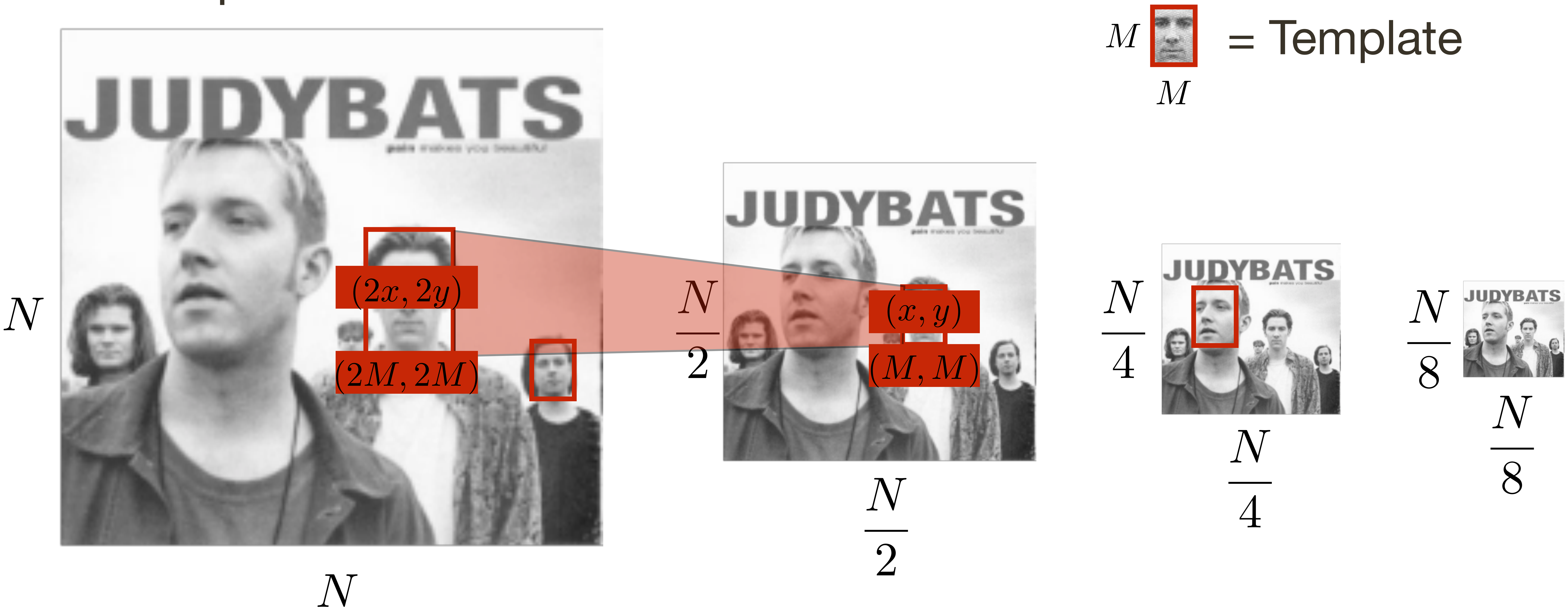
Scaled Representations



Scaled Representations



Scaled Representations



Scaled Representations

Why build a scaled representation of the **image** instead of scaled representation of the **template**?

Scaled Representations

N

N

$$M^2N^2 + 4M^2N^2 + 16M^2N^2 + \\ + 64M^2N^2 = 85M^2N^2$$

M  = Template

M

$2M$ 

$2M$

$4M$ 

$4M$

$8M$ 

$8M$

Scaled Representations

M  = Template

M

N

N

$\frac{N}{2}$

$\frac{N}{2}$

$\frac{N}{4}$

$\frac{N}{4}$

$\frac{N}{8}$

$\frac{N}{8}$

$$M^2N^2 + M^2\left(\frac{N}{2}\right)^2 + M^2\left(\frac{N}{4}\right)^2 + M^2\left(\frac{N}{8}\right)^2 \approx 1\frac{1}{2}M^2N^2$$

Scaled Representations: Goals

to find **template matches** at all scales

— template size constant, image scale varies

— finding hands or faces when we don't know what size they are in the image

Scaled Representations: Goals

to find **template matches** at all scales

— template size constant, image scale varies

— finding hands or faces when we don't know what size they are in the image

efficient search for image–to–image correspondences

— look first at coarse scales, refine at finer scales

— much less cost (but may miss best match)

Scaled Representations: Goals

to find **template matches** at all scales

— template size constant, image scale varies

— finding hands or faces when we don't know what size they are in the image

efficient search for image–to–image correspondences

— look first at coarse scales, refine at finer scales

— much less cost (but may miss best match)

Scaled Representations: Goals

to find **template matches** at all scales

— template size constant, image scale varies

— finding hands or faces when we don't know what size they are in the image

efficient search for image–to–image correspondences

— look first at coarse scales, refine at finer scales

— much less cost (but may miss best match)

to examine all **levels of detail**

— find edges with different amounts of blur

— find textures with different spatial frequencies (i.e., different levels of detail)

Shrinking the Image

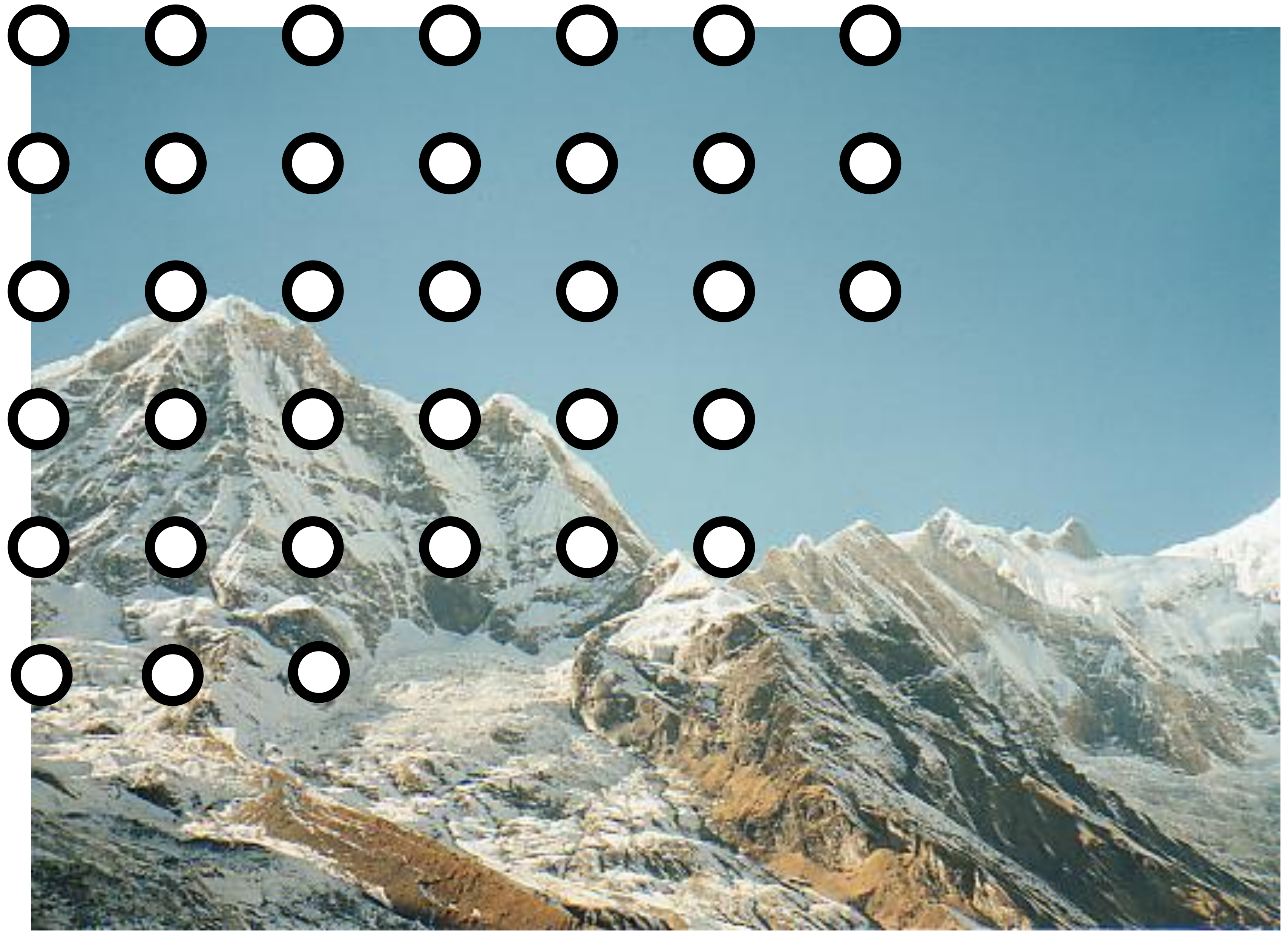
We can’t shrink an image simply by taking every second pixel

Shrinking the Image

We can’t shrink an image simply by taking every second pixel

Why?

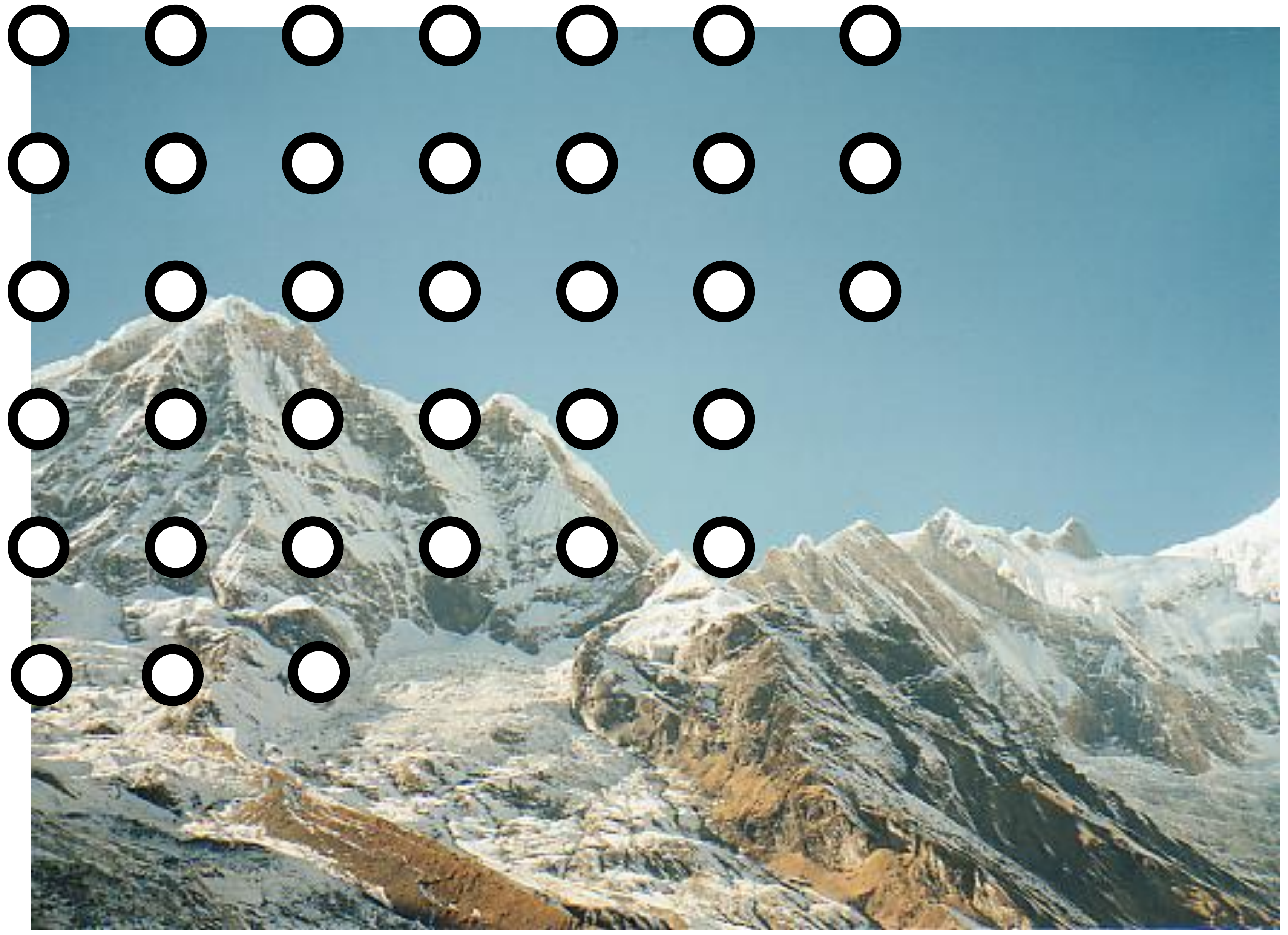
Aliasing Example



No filtering

Gaussian Blur $\sigma = 3.0$

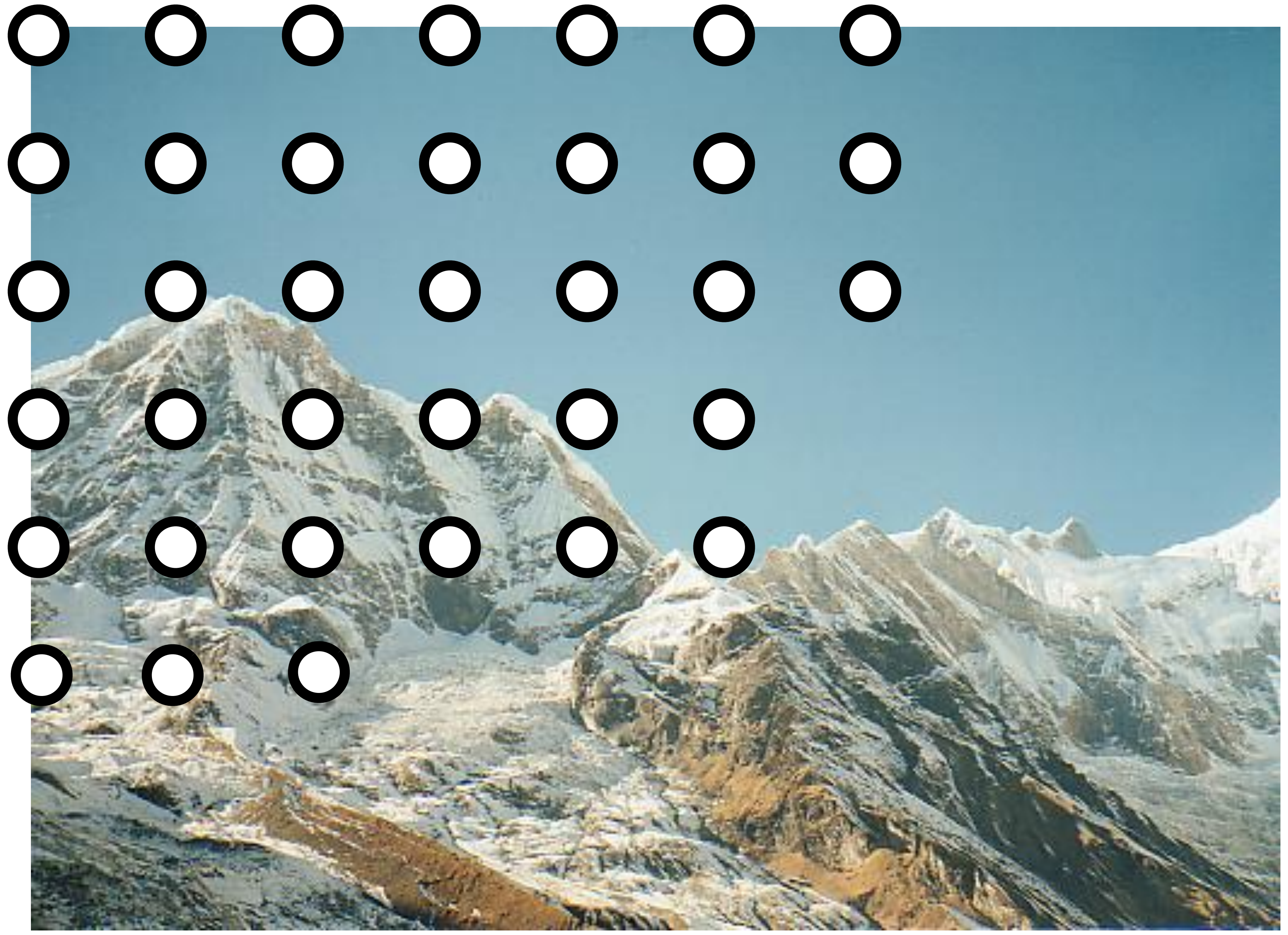
Aliasing Example



No filtering

Gaussian Blur $\sigma = 3.0$

Aliasing Example



No filtering

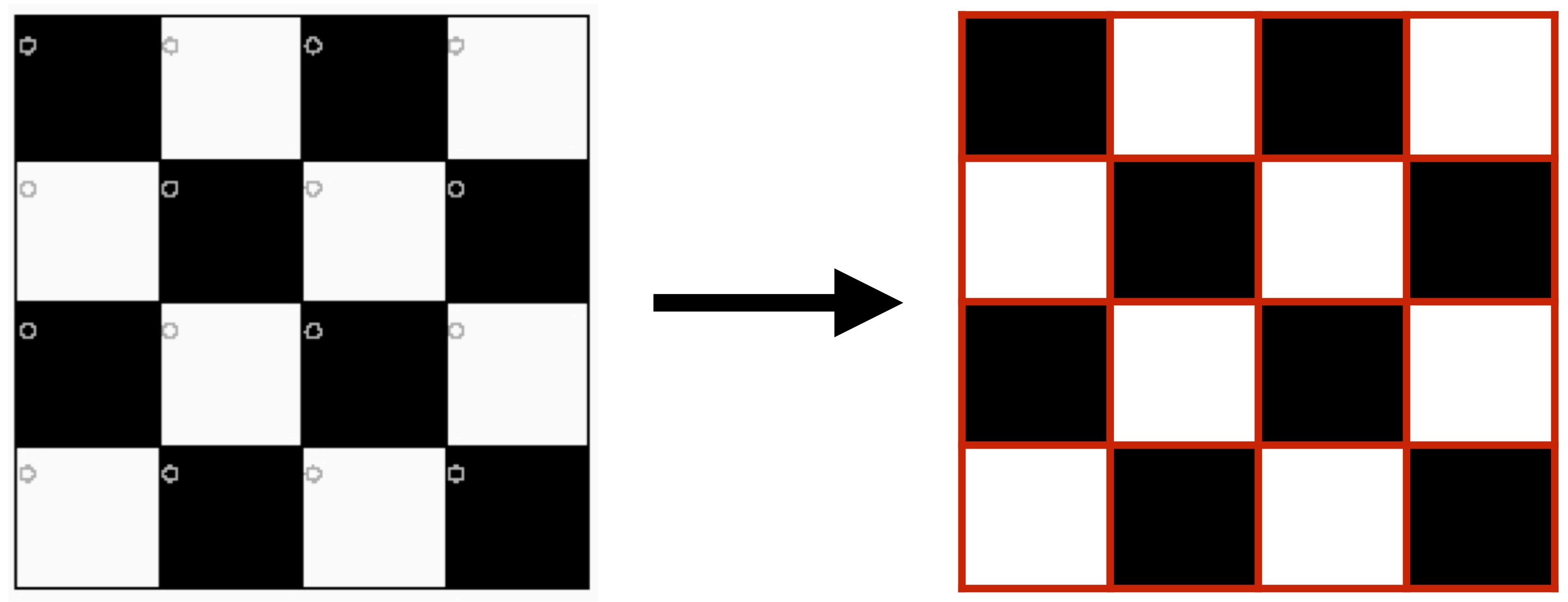
Gaussian Blur $\sigma = 3.0$

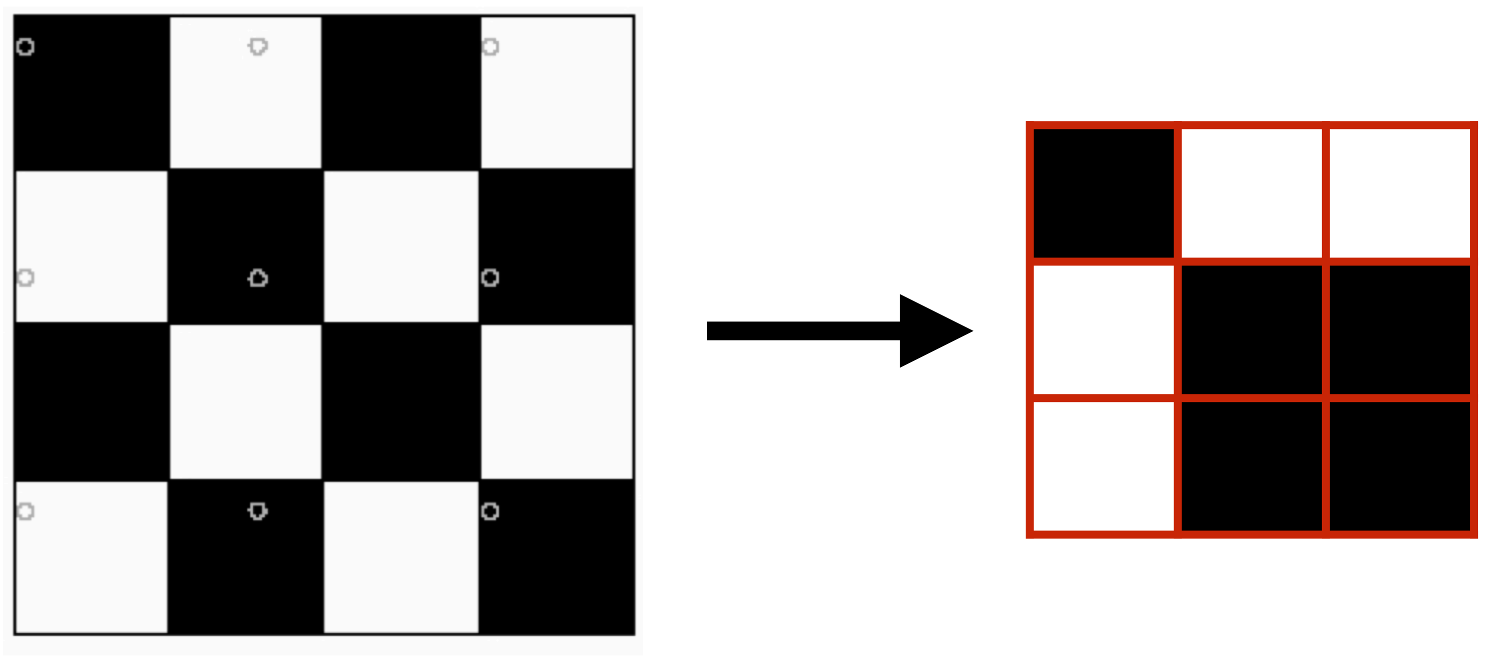
Nyquist Sampling

To avoid aliasing a signal must be sampled at twice the maximum frequency:

$$f_s > 2 \times f_{max}$$

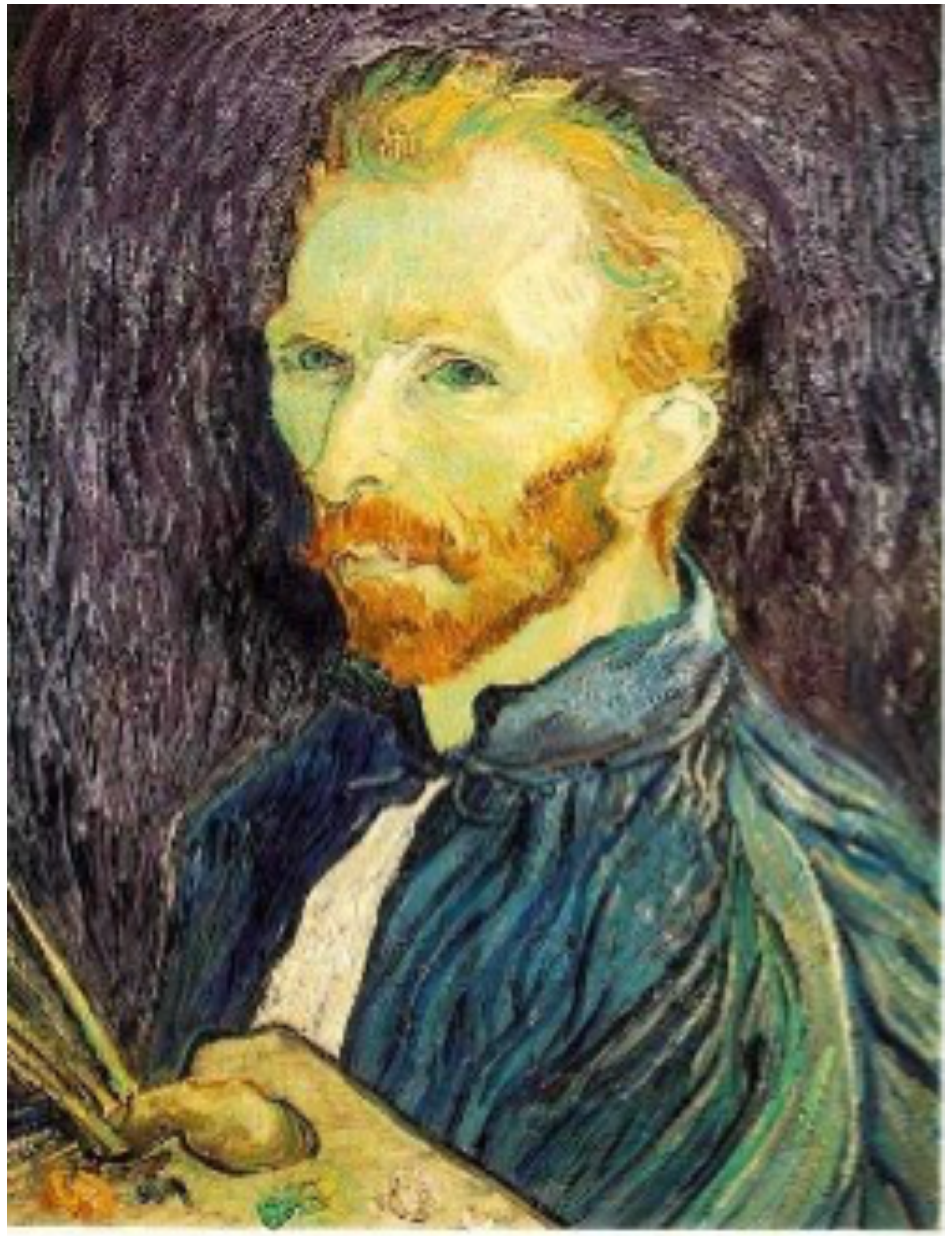
For Images: We need to sample the underlying continuous signal **at least once per pixel** to avoid aliasing (assuming a correctly sampled image)





undersampling = aliasing

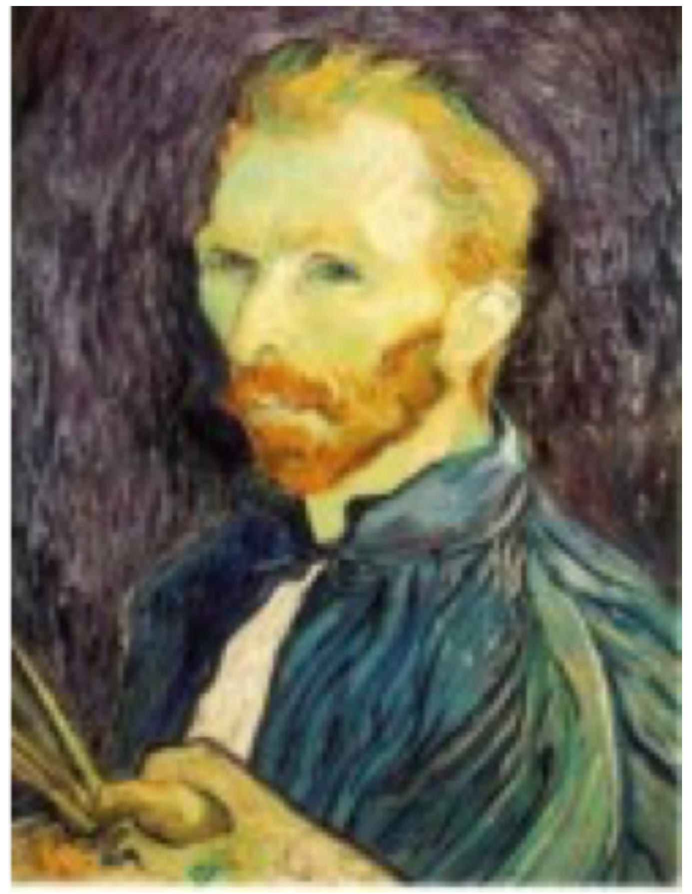
Template Matching: Sub-sample with Gaussian Pre-filtering



1/2

Apply a smoothing filter first, then throw away half the rows and columns

Gaussian filter
delete even rows
delete even columns



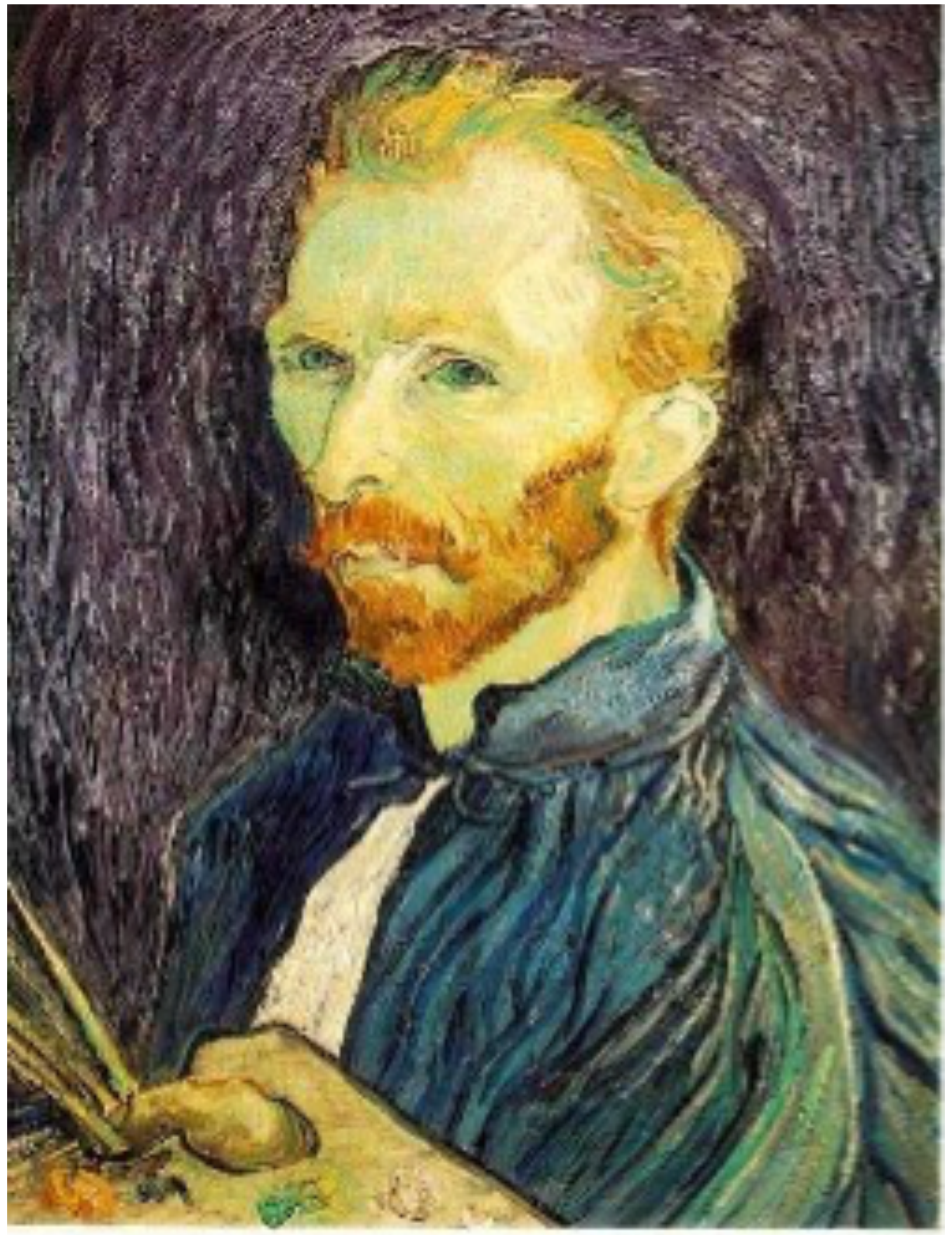
1/4

Gaussian filter
delete even rows
delete even columns

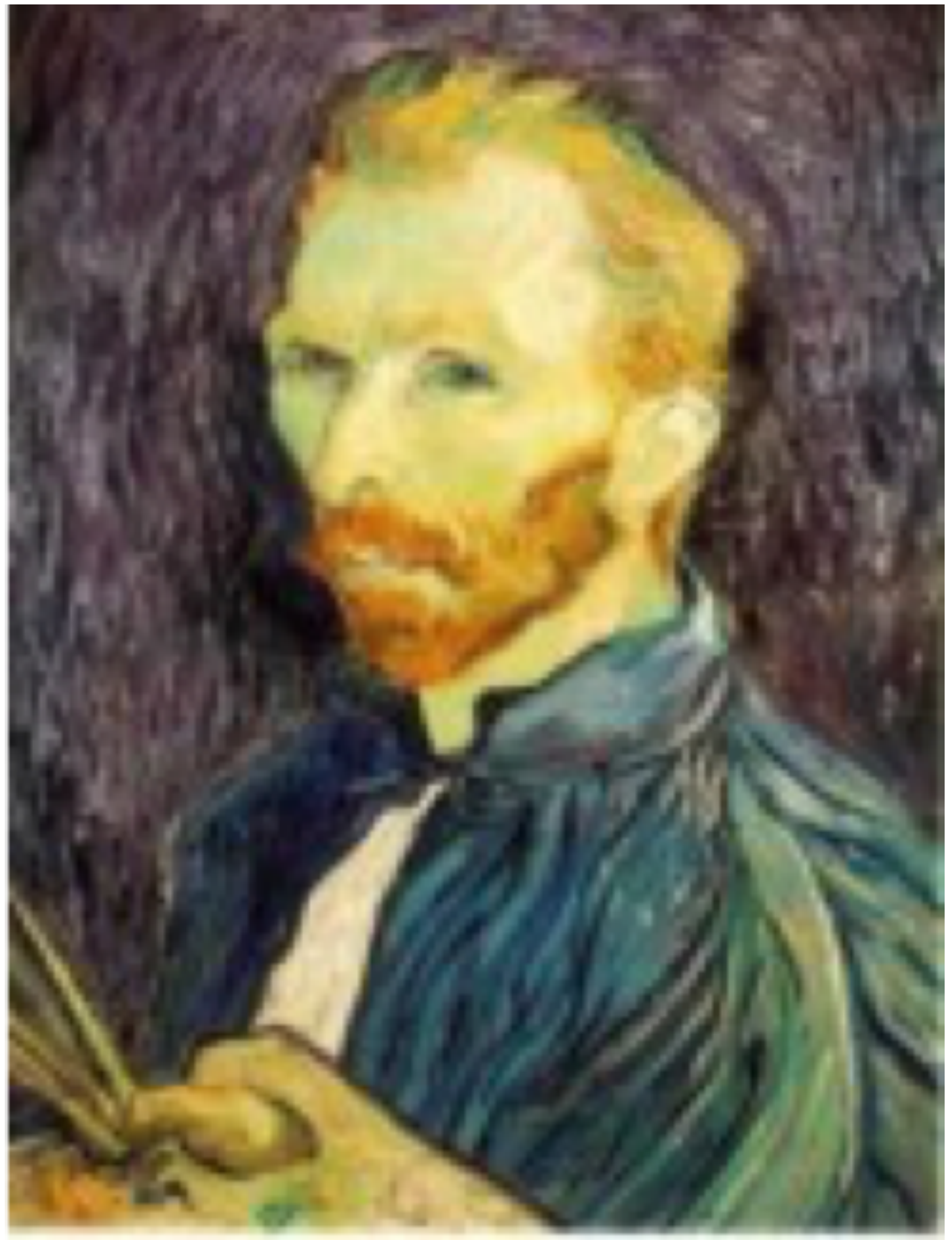
1/8

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Template Matching: Sub-sample with Gaussian Pre-filtering



1/2

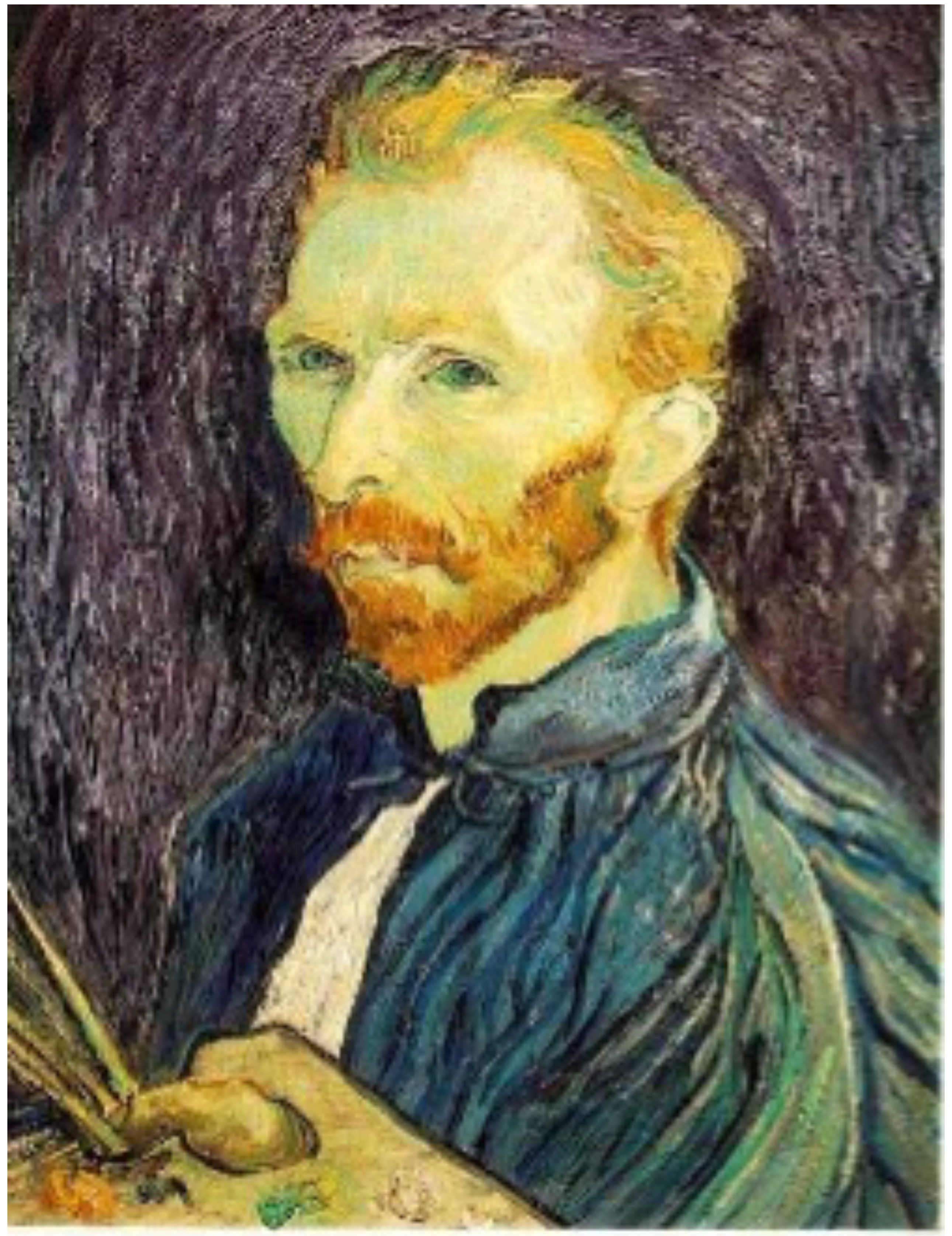


1/4 (2x zoom)

1/8 (4x zoom)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Template Matching: Sub-sample with NO Pre-filtering



1/2

1/4 (2x zoom)

1/8 (4x zoom)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Gaussian Pre-filtering

Question: How much smoothing is needed to avoid aliasing?

Gaussian Pre-filtering

Question: How much smoothing is needed to avoid aliasing?

Answer: Smoothing should be sufficient to ensure that the resulting image is band limited "enough" to ensure we can sample every other pixel.

Practically: For every image reduction of 0.5, smooth by $\sigma = 1$

Image **Pyramid**

An **image pyramid** is an efficient way to represent an image at multiple scales

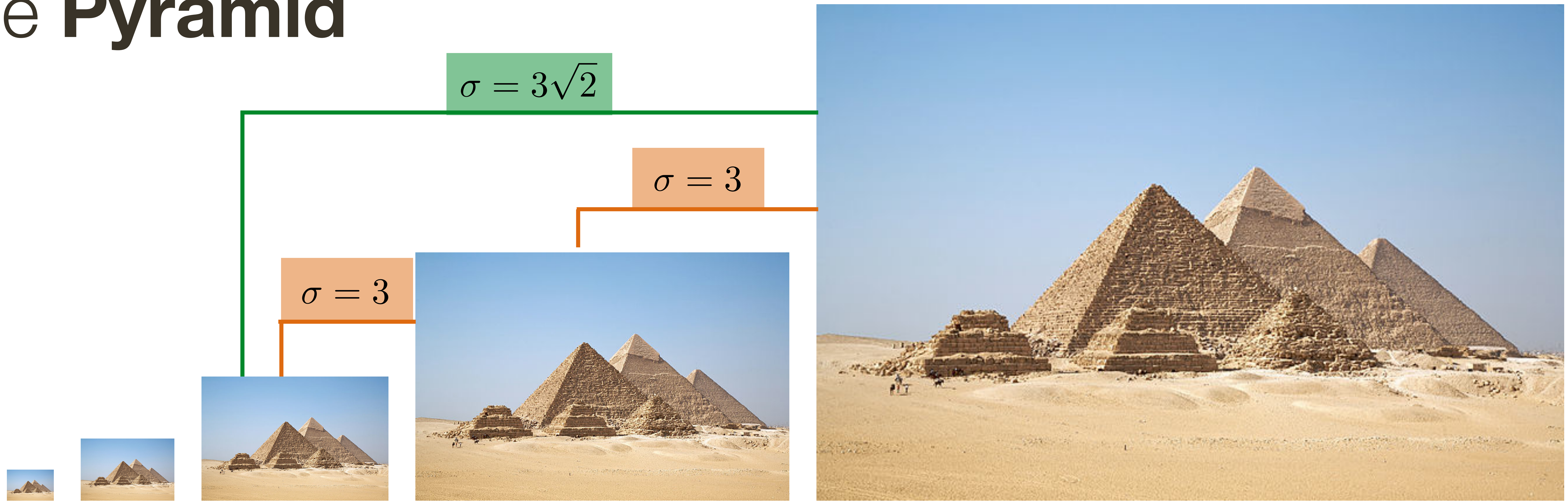
Image **Pyramid**

An **image pyramid** is an efficient way to represent an image at multiple scales

In a **Gaussian pyramid**, each layer is smoothed by a Gaussian filter and resampled to get the next layer, taking advantage of the fact that

$$G_{\sigma_1}(x) \otimes G_{\sigma_2}(x) = G_{\sqrt{\sigma_1^2+\sigma_2^2}}(x)$$

Image **Pyramid**

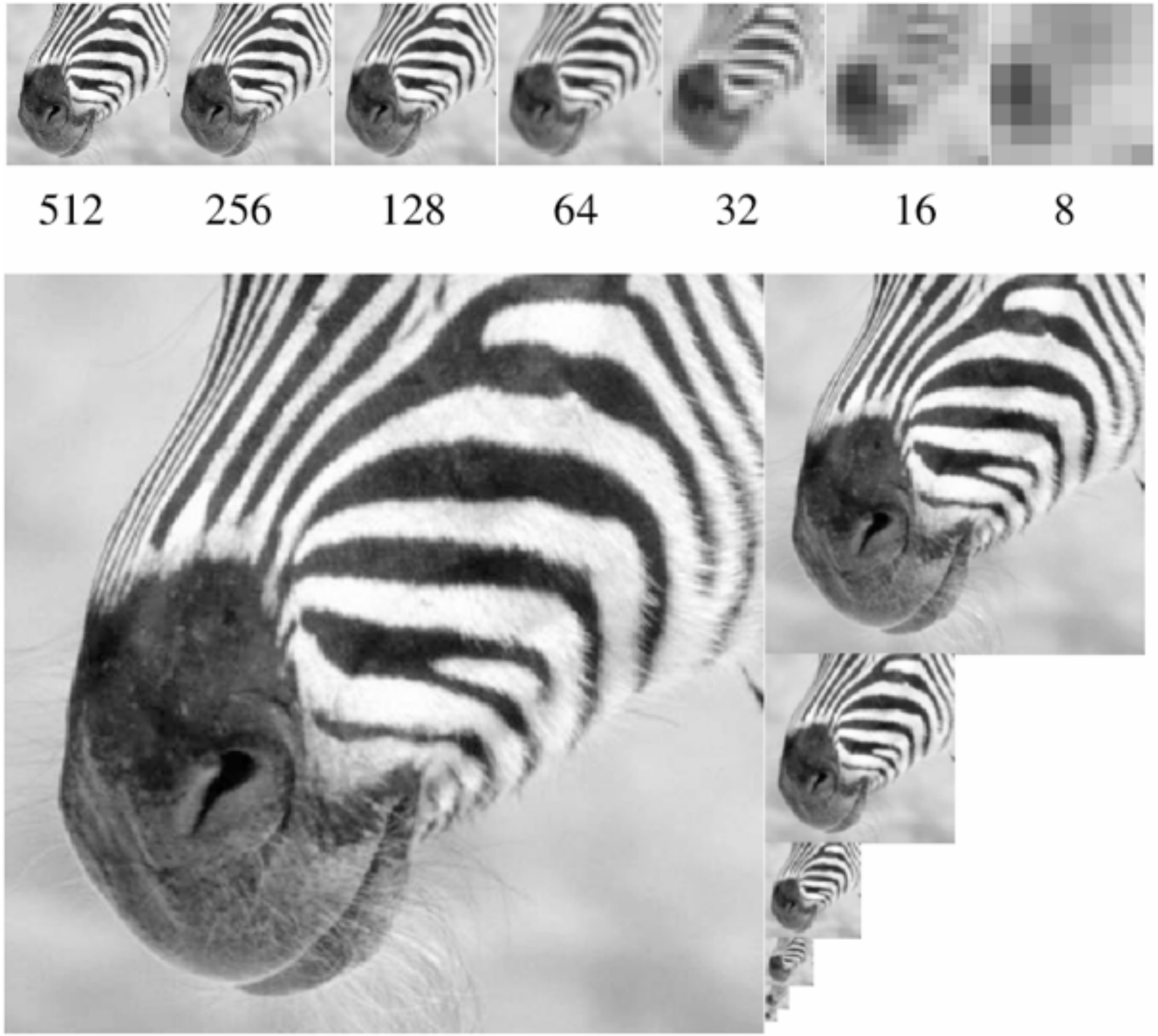


An **image pyramid** is an efficient way to represent an image at multiple scales

In a **Gaussian pyramid**, each layer is smoothed by a Gaussian filter and resampled to get the next layer, taking advantage of the fact that

$$G_{\sigma_1}(x) \otimes G_{\sigma_2}(x) = G_{\sqrt{\sigma_1^2+\sigma_2^2}}(x)$$

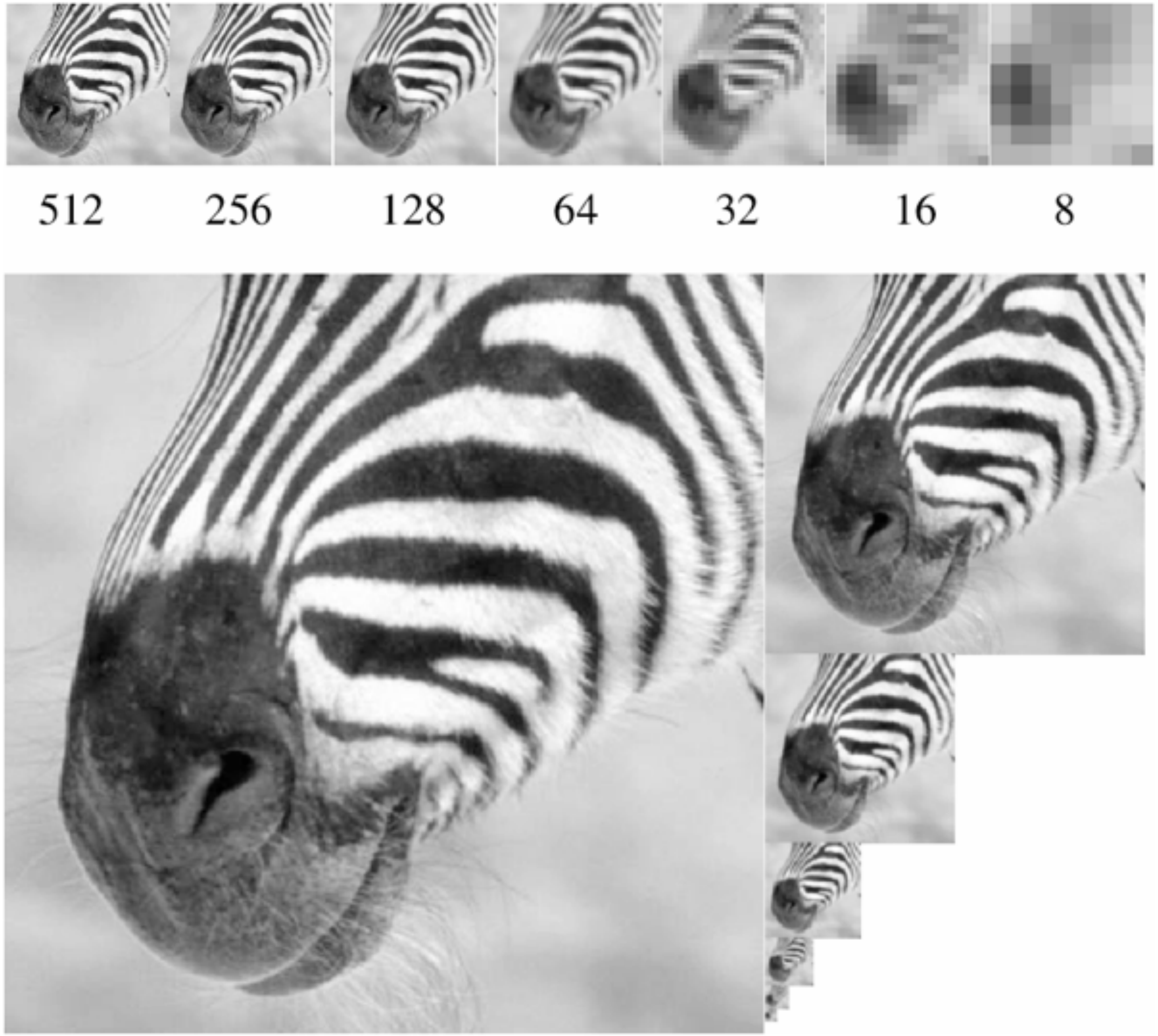
Example 2: Gaussian Pyramid



Forsyth & Ponce (2nd ed.) Figure 4.17

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example 2: Gaussian Pyramid

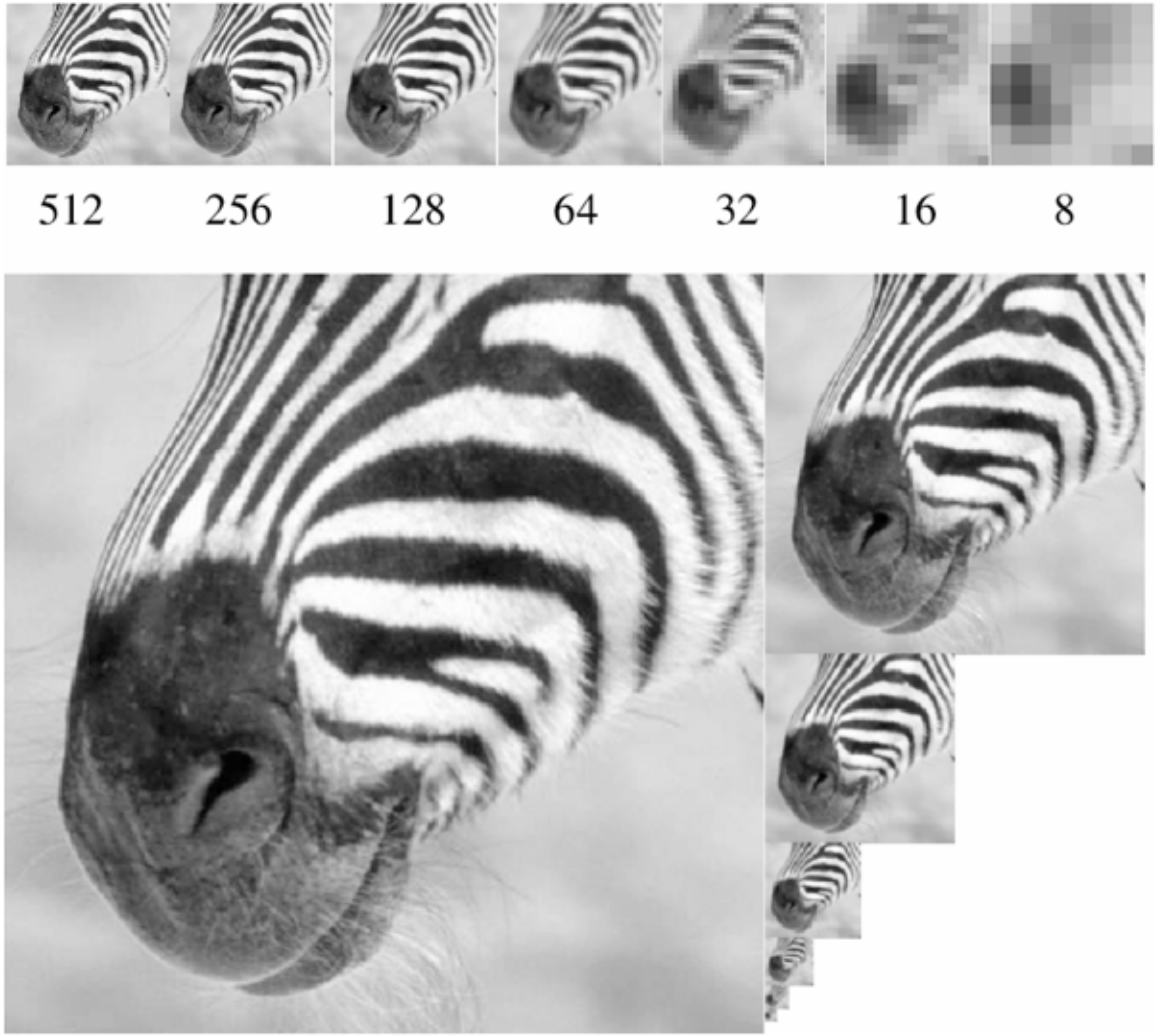


Forsyth & Ponce (2nd ed.) Figure 4.17

What happens to the details?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example 2: Gaussian Pyramid



Forsyth & Ponce (2nd ed.) Figure 4.17

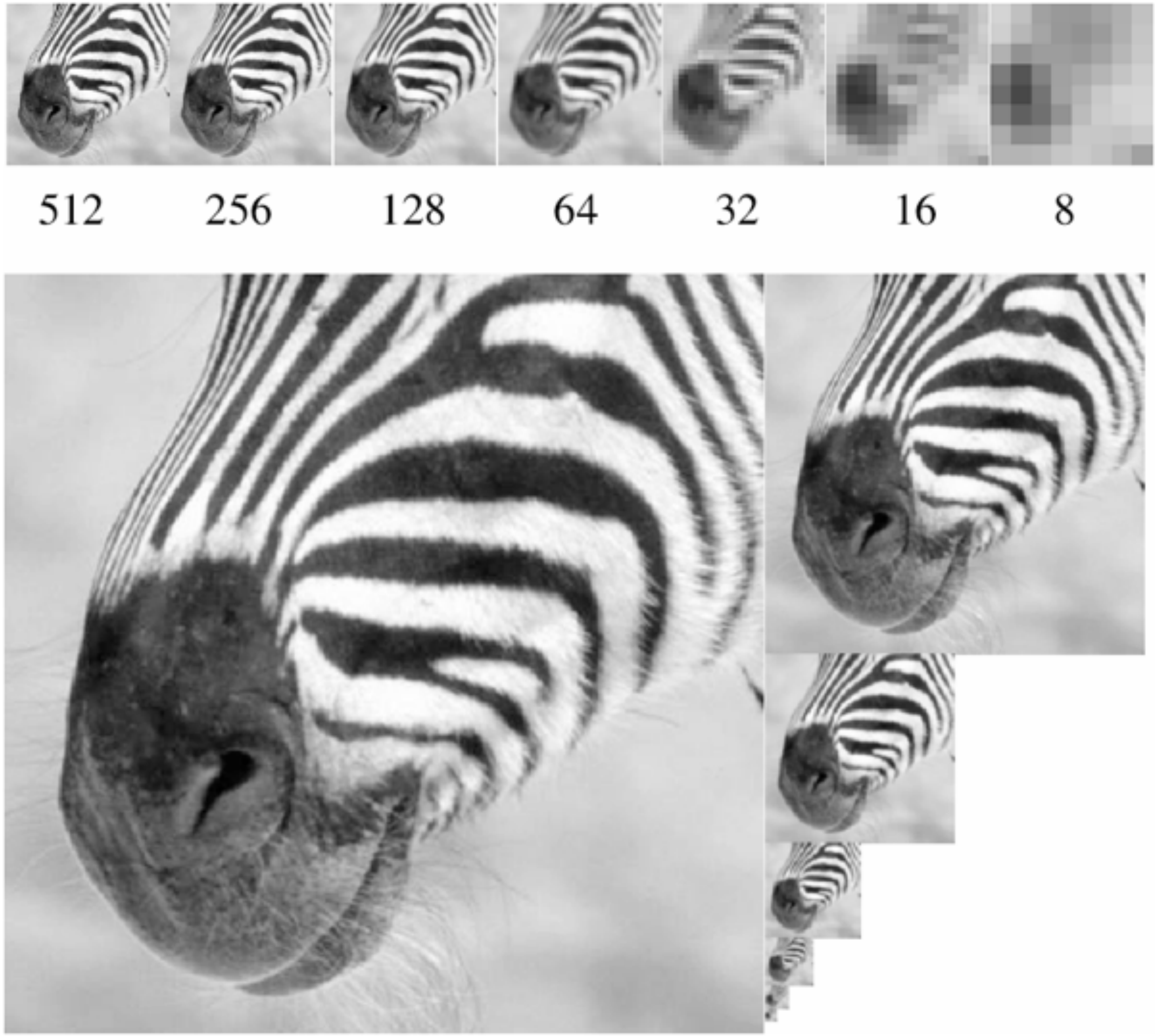
What happens to the details?

– They get smoothed out as we move to higher levels

What is preserved at the higher levels?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example 2: Gaussian Pyramid



Forsyth & Ponce (2nd ed.) Figure 4.17

What happens to the details?

— They get smoothed out as we move to higher levels

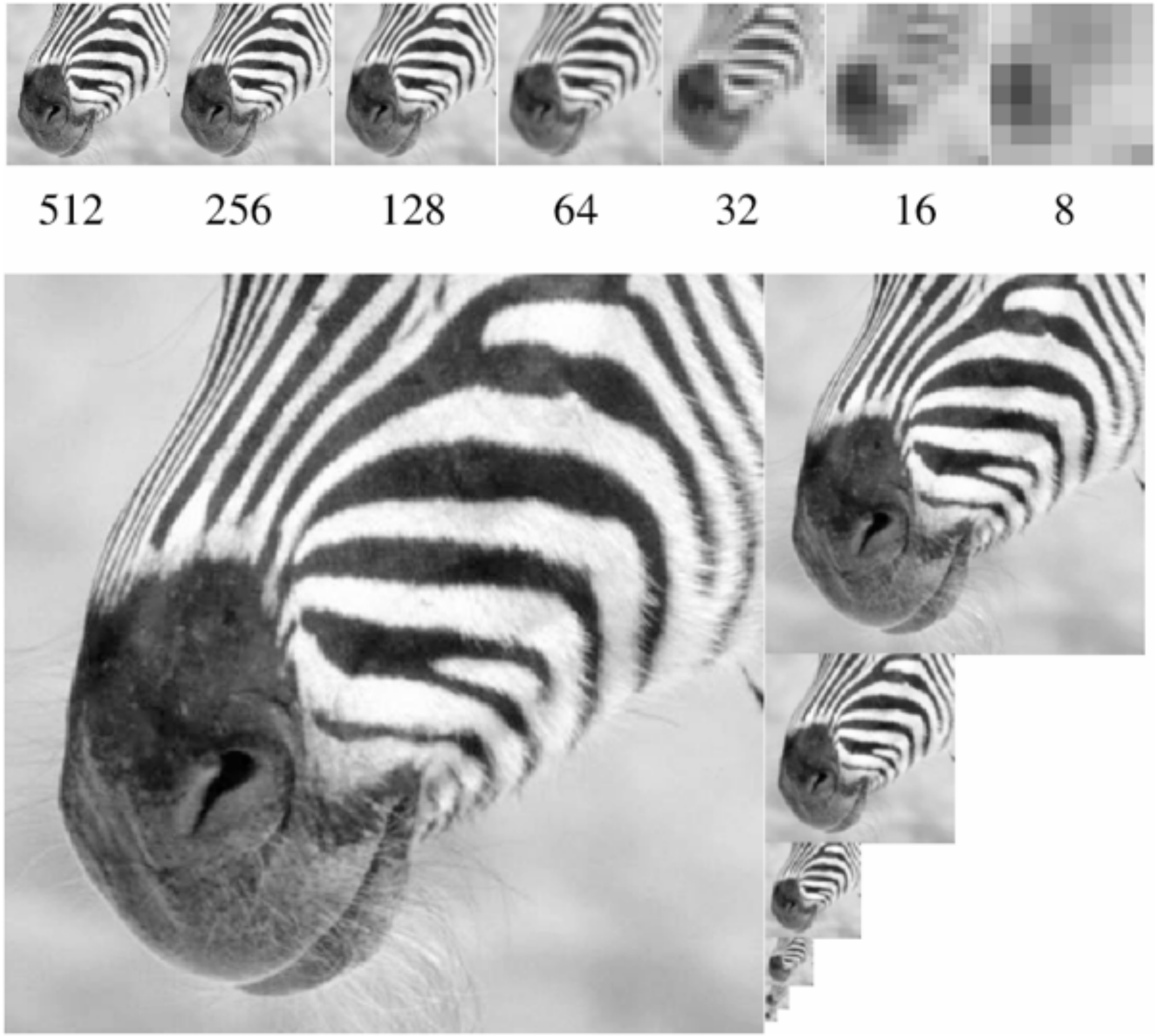
What is preserved at the higher levels?

— Mostly large uniform regions in the original image

How would you reconstruct the original image from the image at the upper level?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example 2: Gaussian Pyramid



Forsyth & Ponce (2nd ed.) Figure 4.17

What happens to the details?

- They get smoothed out as we move to higher levels

What is preserved at the higher levels?

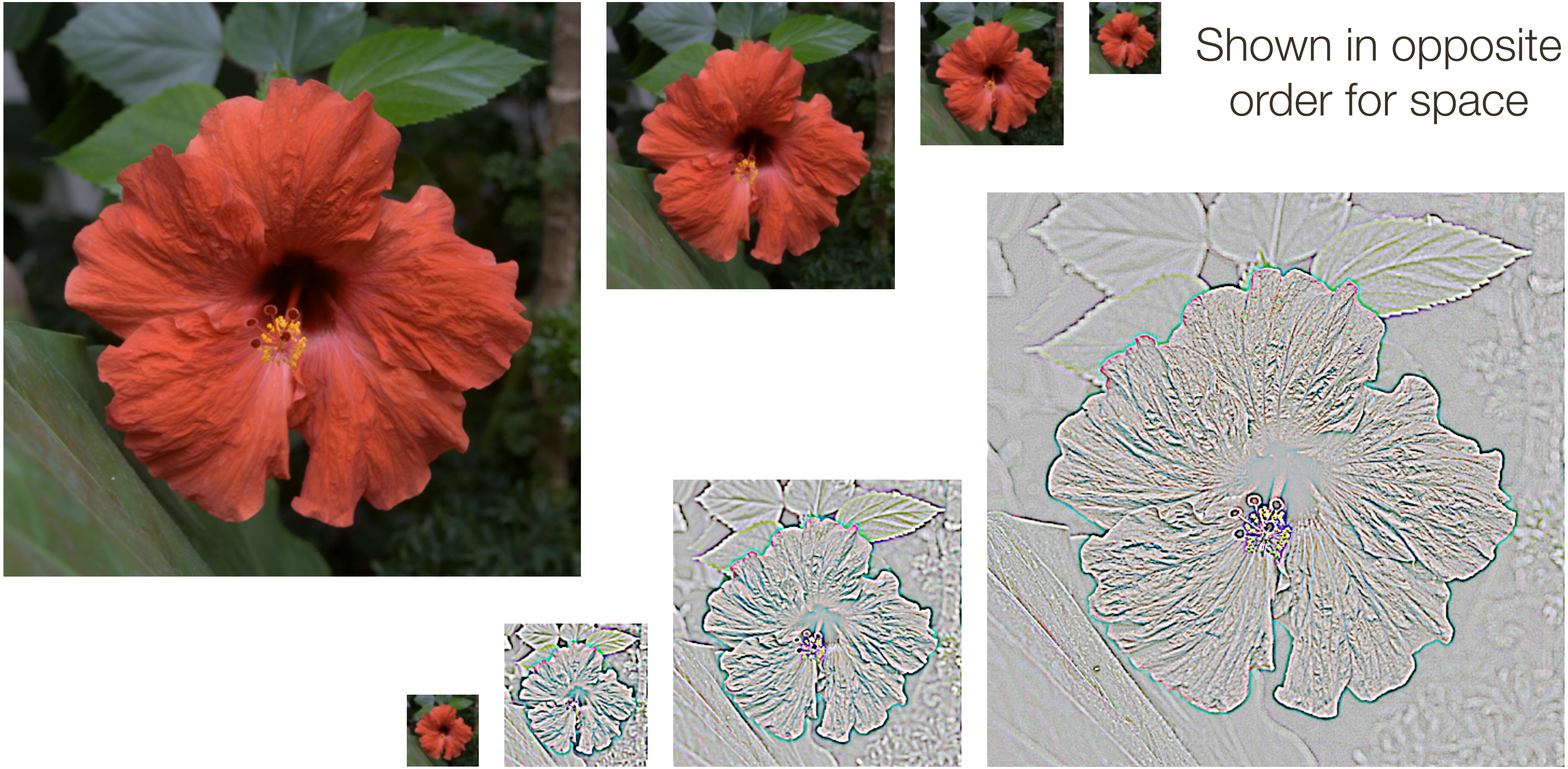
- Mostly large uniform regions in the original image

How would you reconstruct the original image from the image at the upper level?

- That's not possible

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Gaussian vs **Laplacian** Pyramid



Shown in opposite order for space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

$G1$

Blur with a Gaussian kernel, then select every 2nd pixel

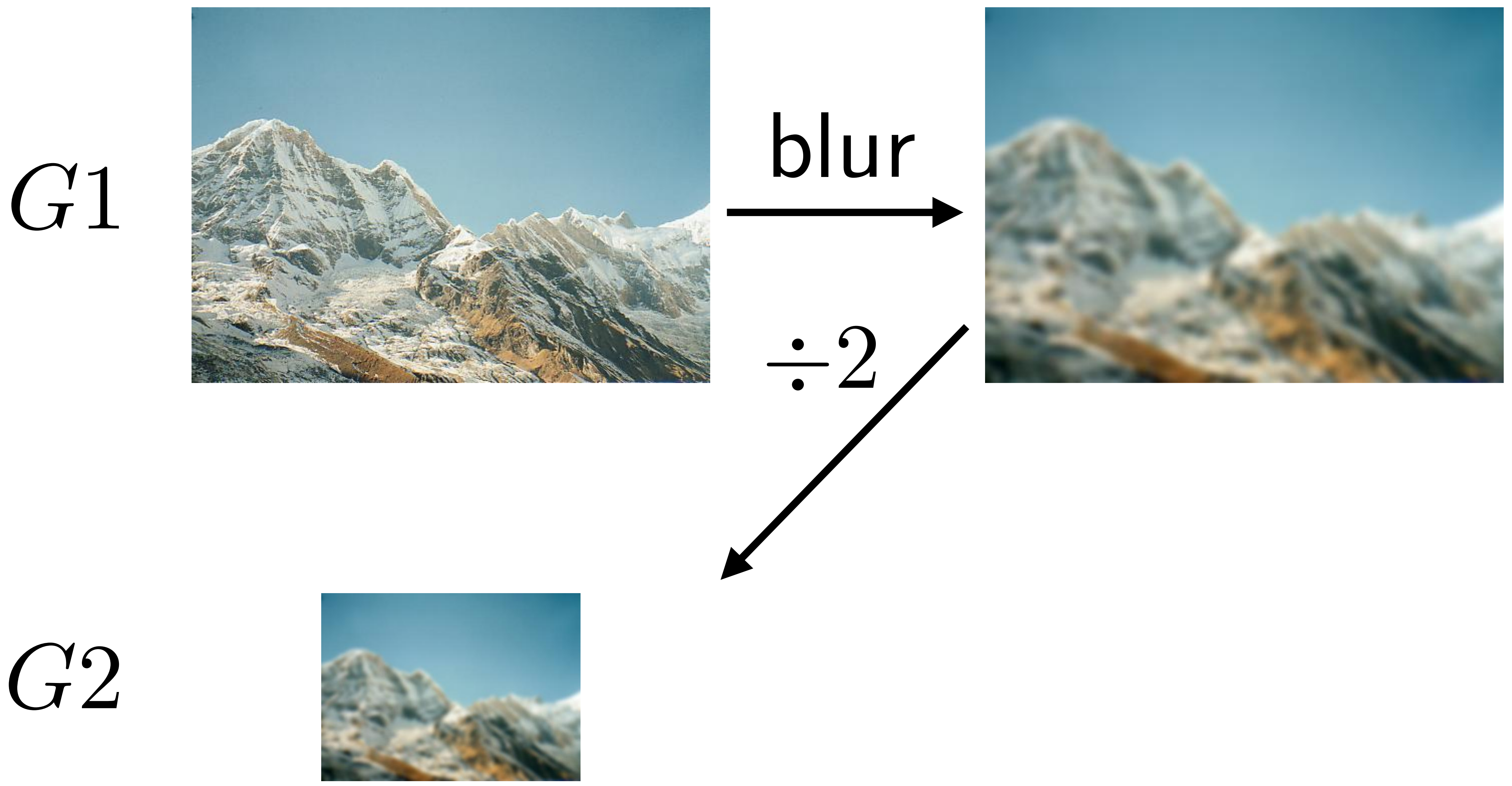
$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$

$G1$

blur →

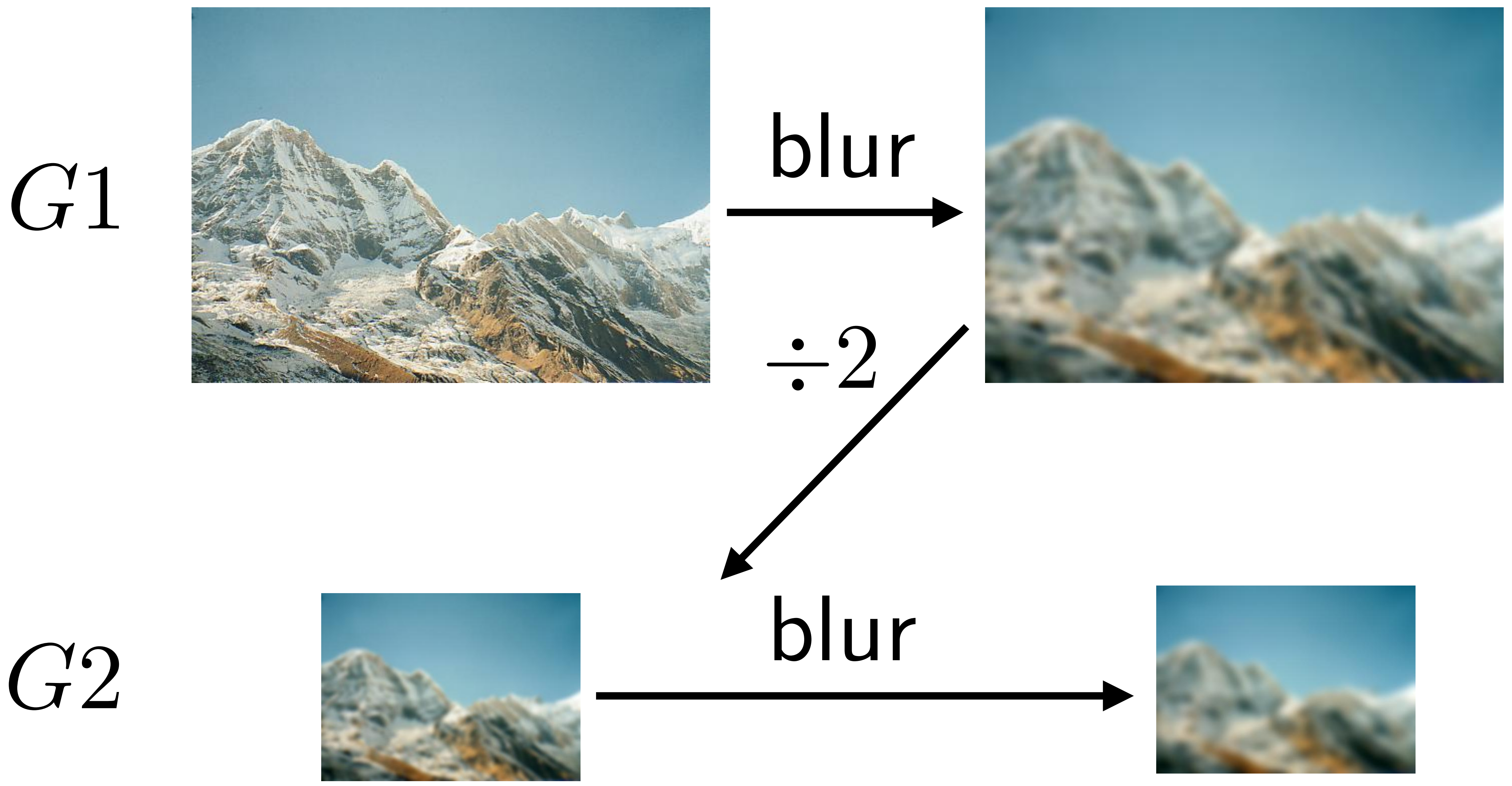
Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$



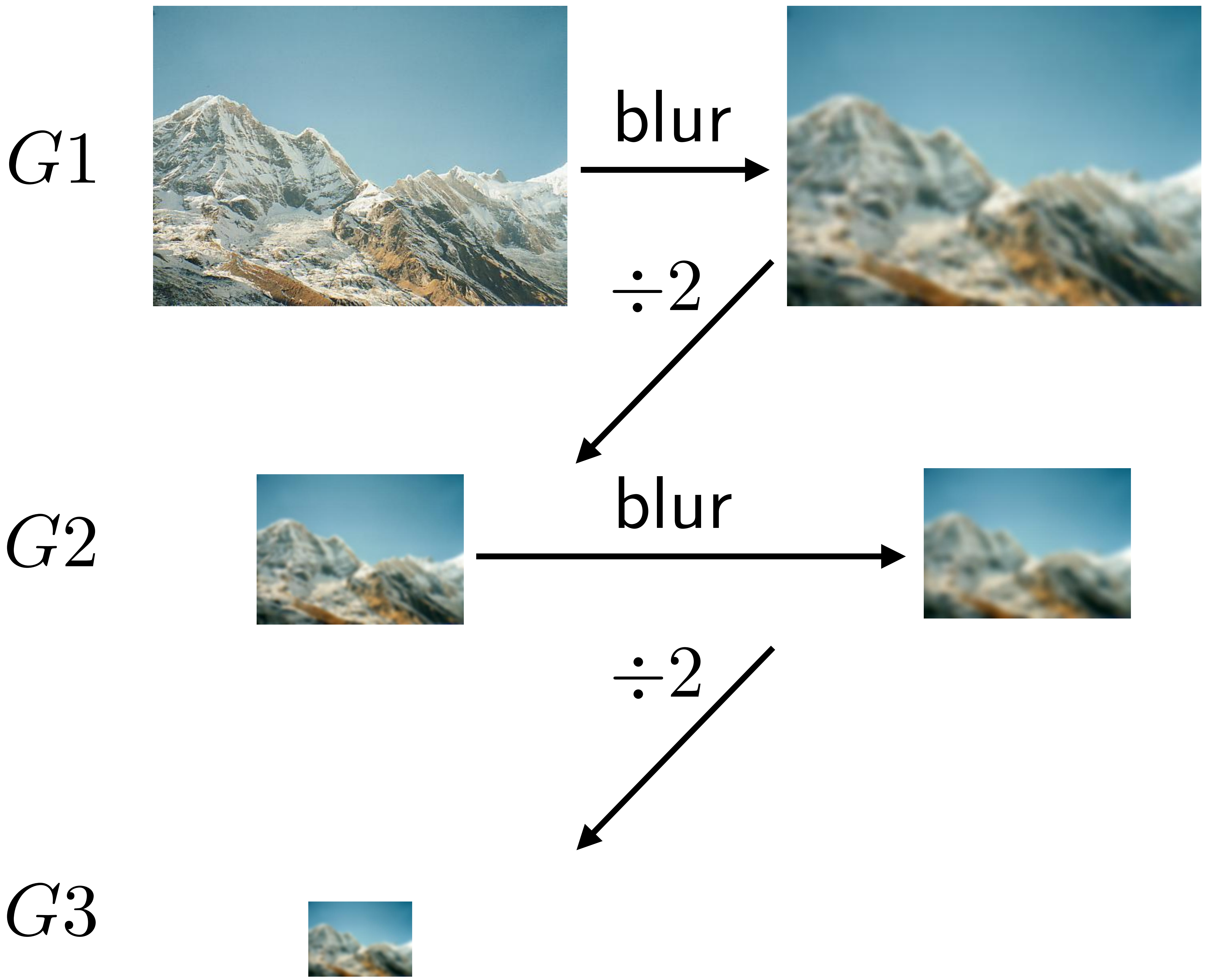
Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$



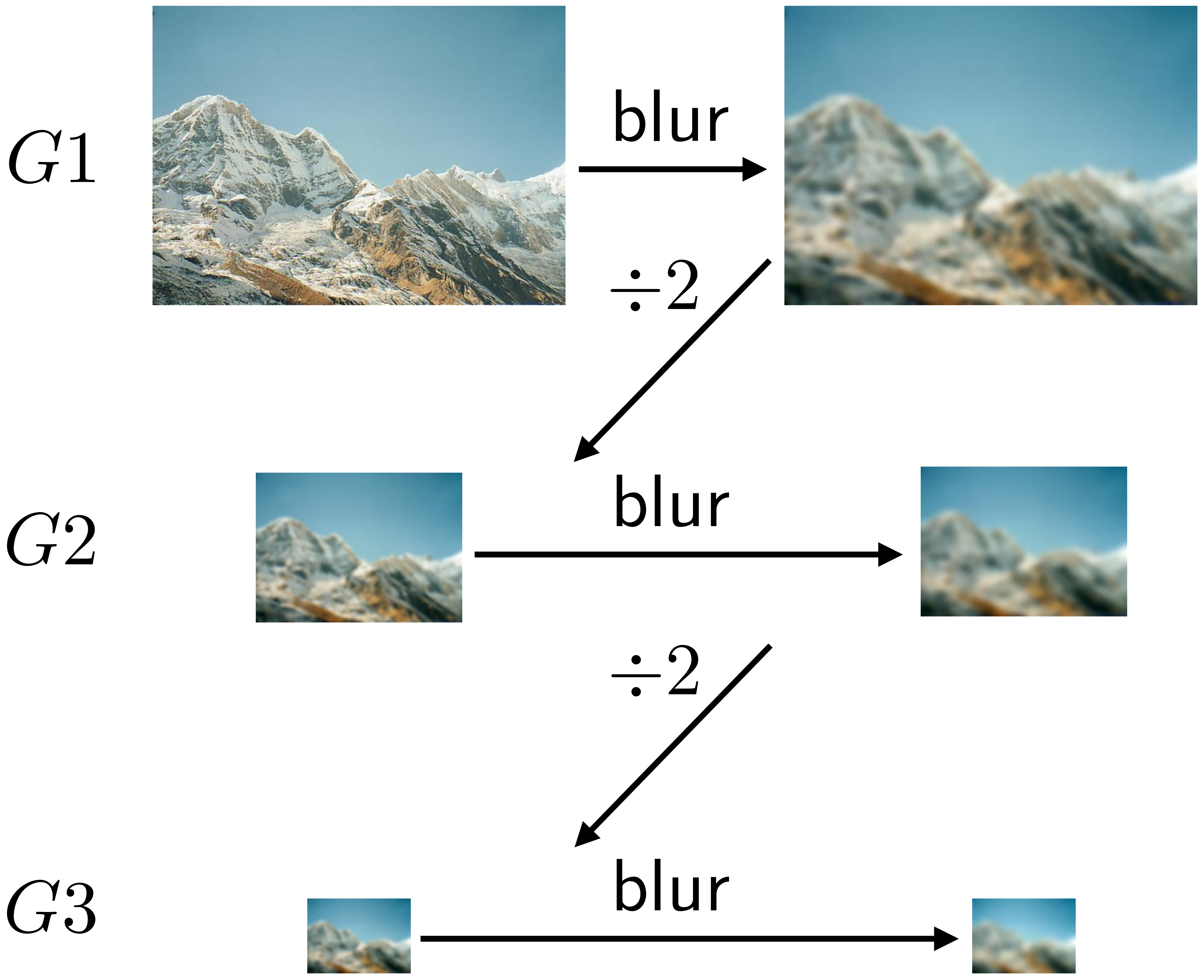
Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$



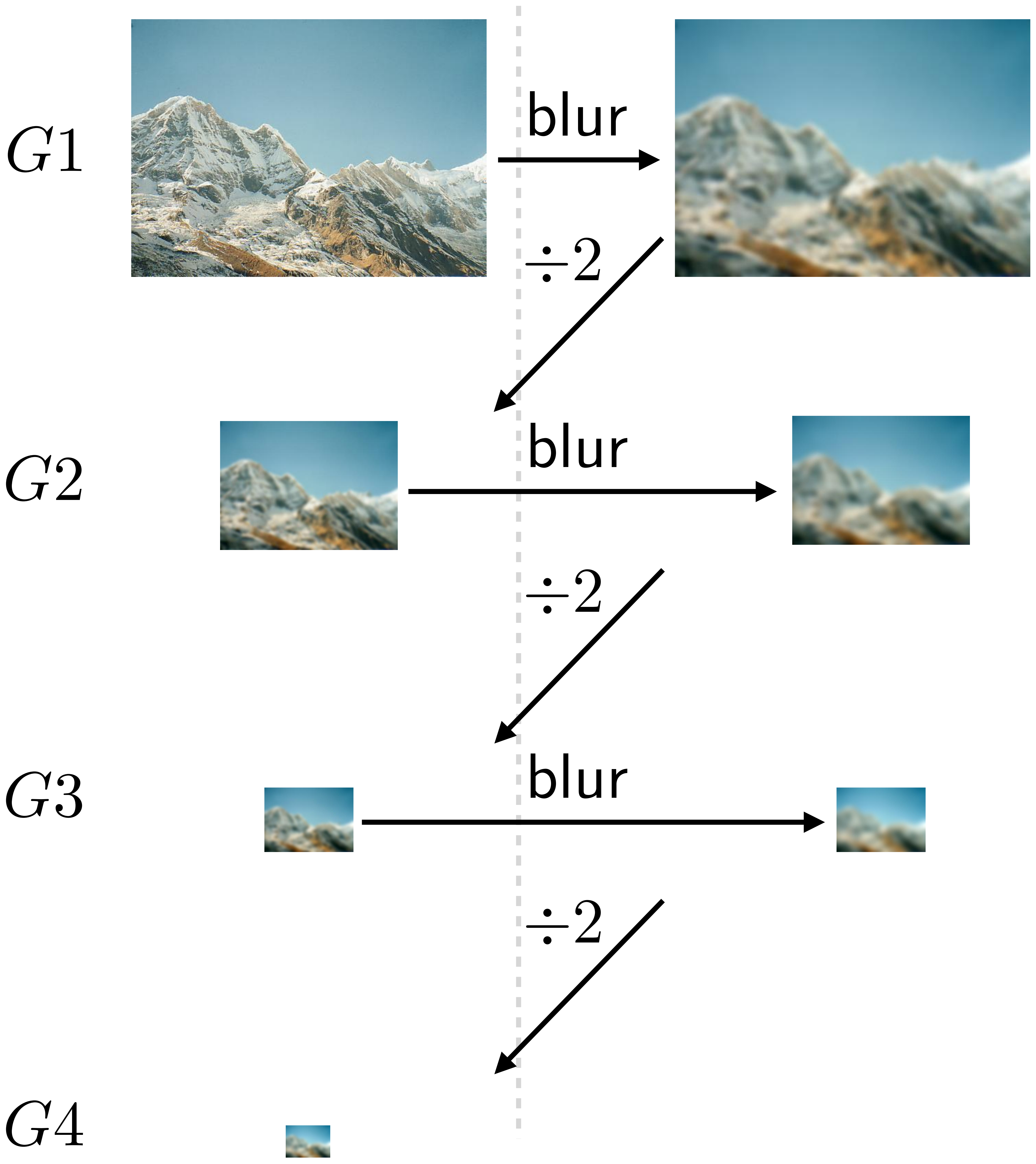
Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$



Blur with a Gaussian kernel, then select every 2nd pixel

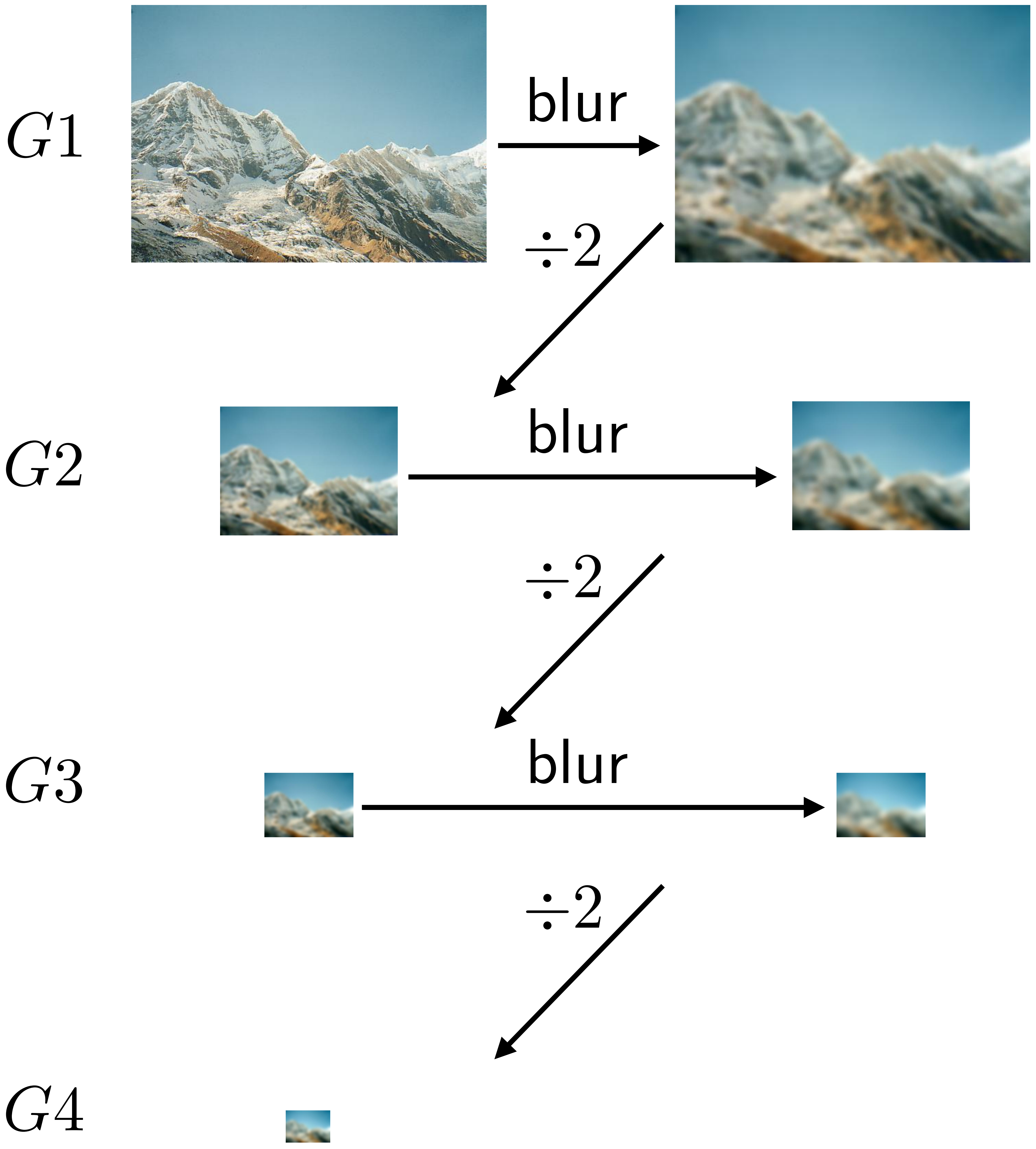
$$I_s(x,y) = I(x,y) * g_\sigma(x,y)$$



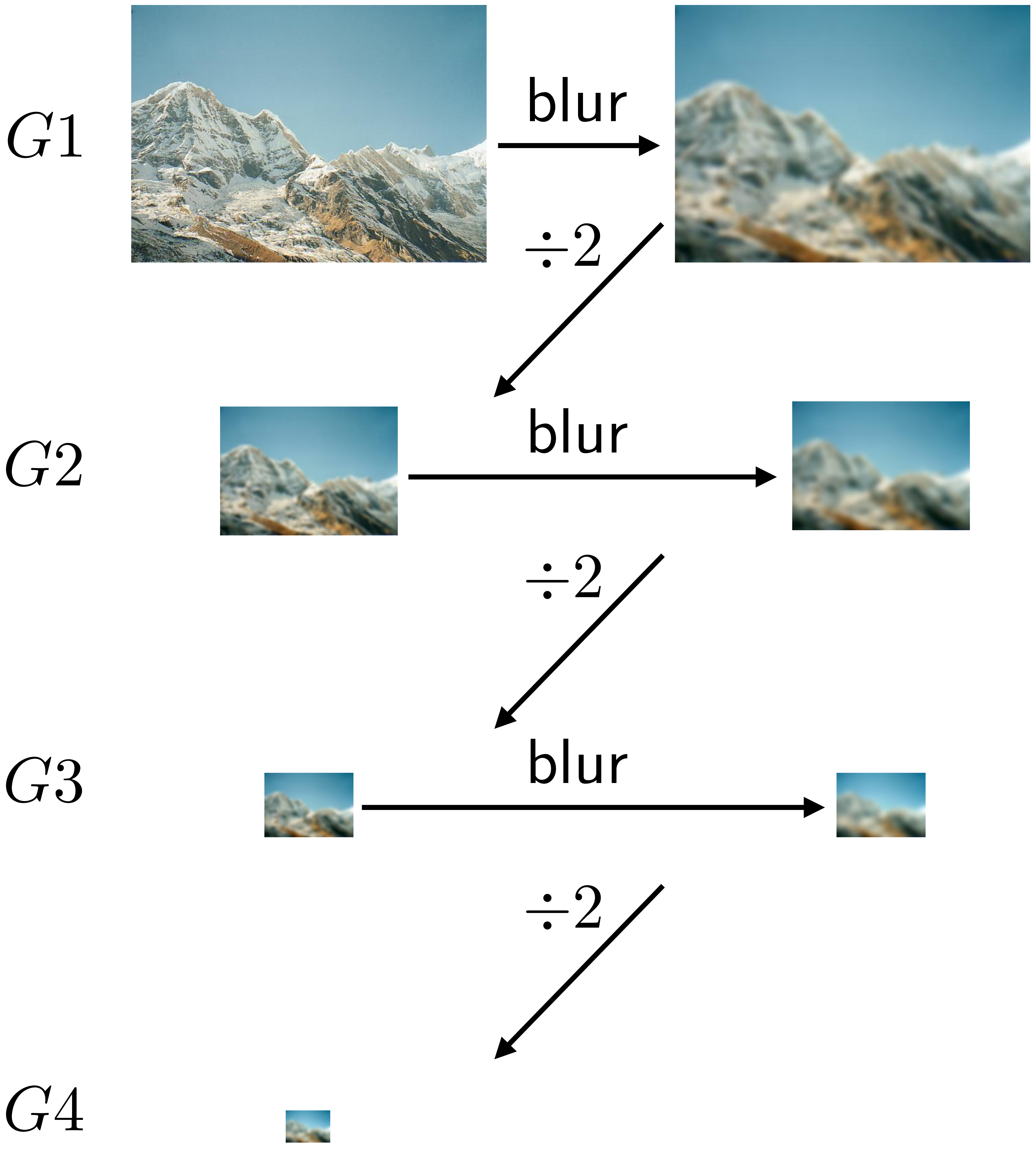
Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$

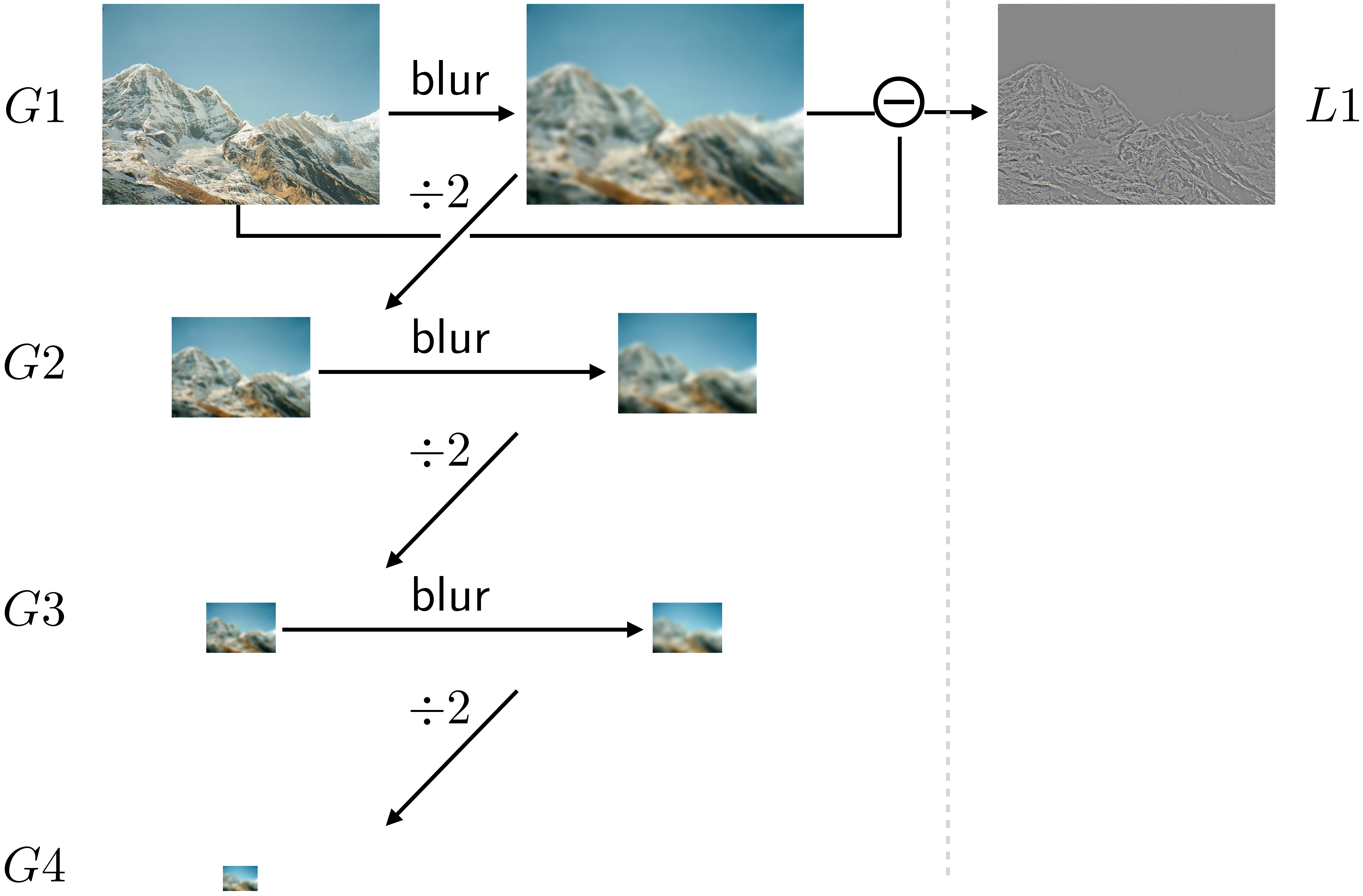
Gaussian Pyramid



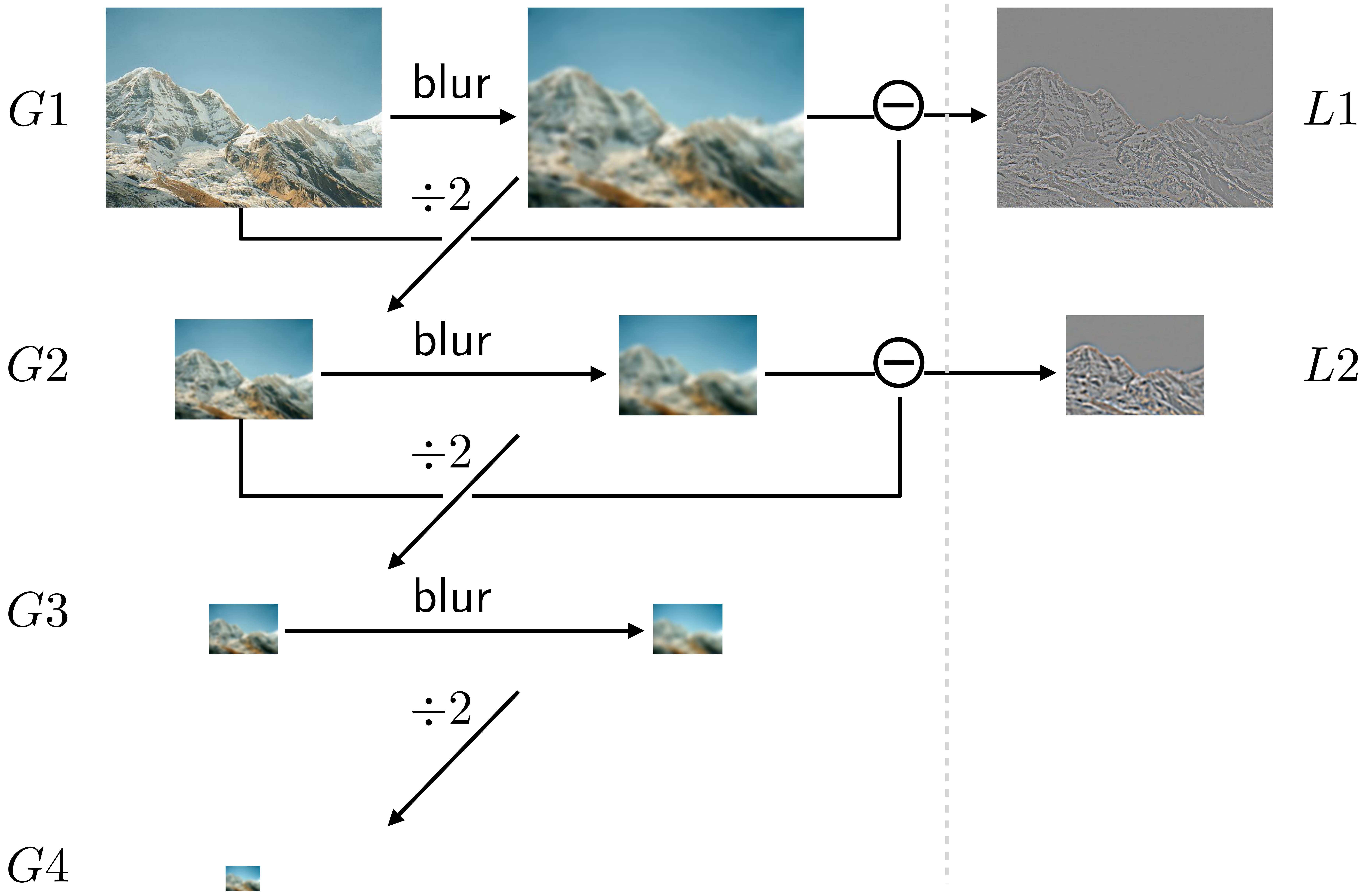
Gaussian Pyramid



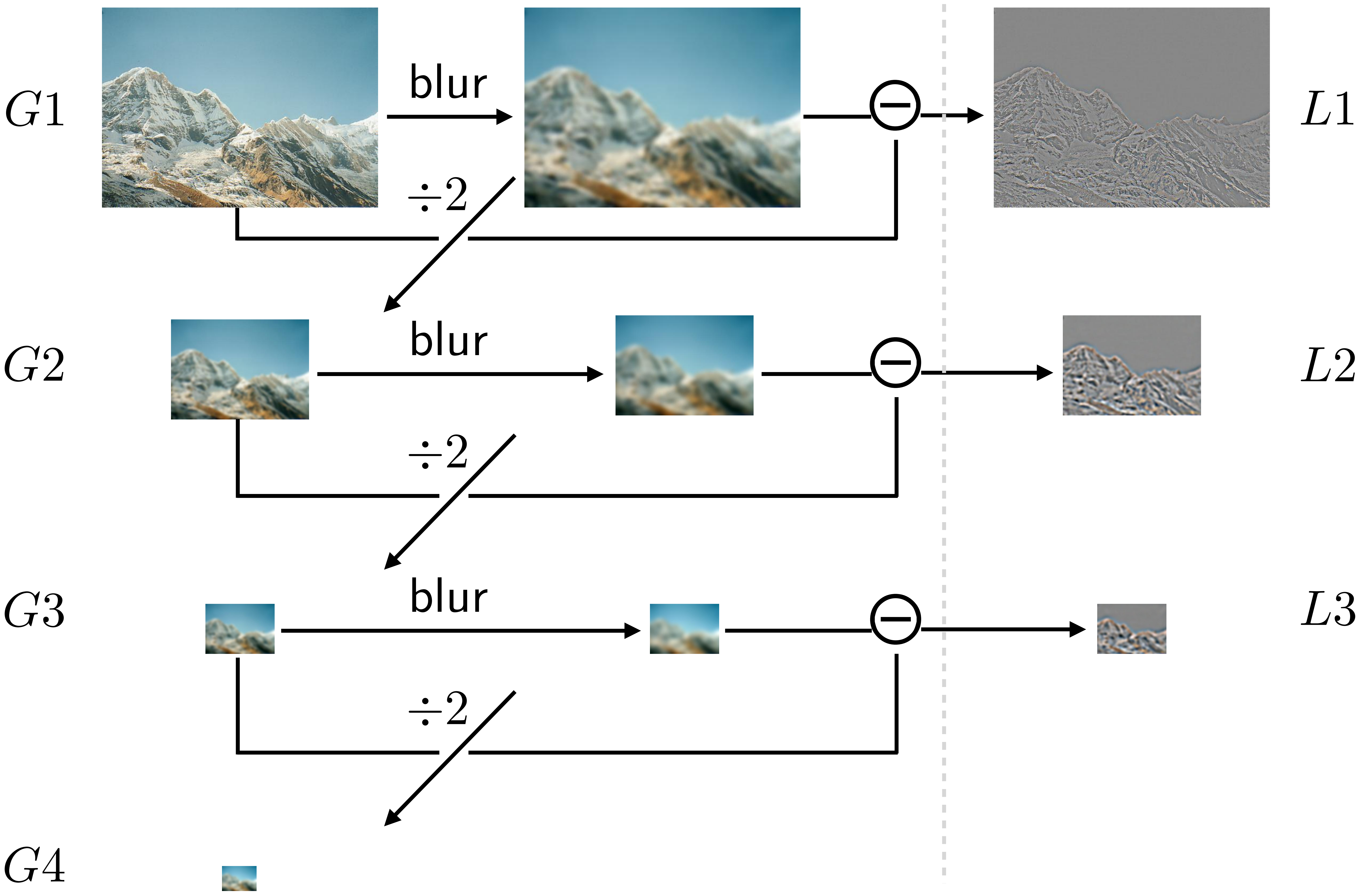
Gaussian Pyramid



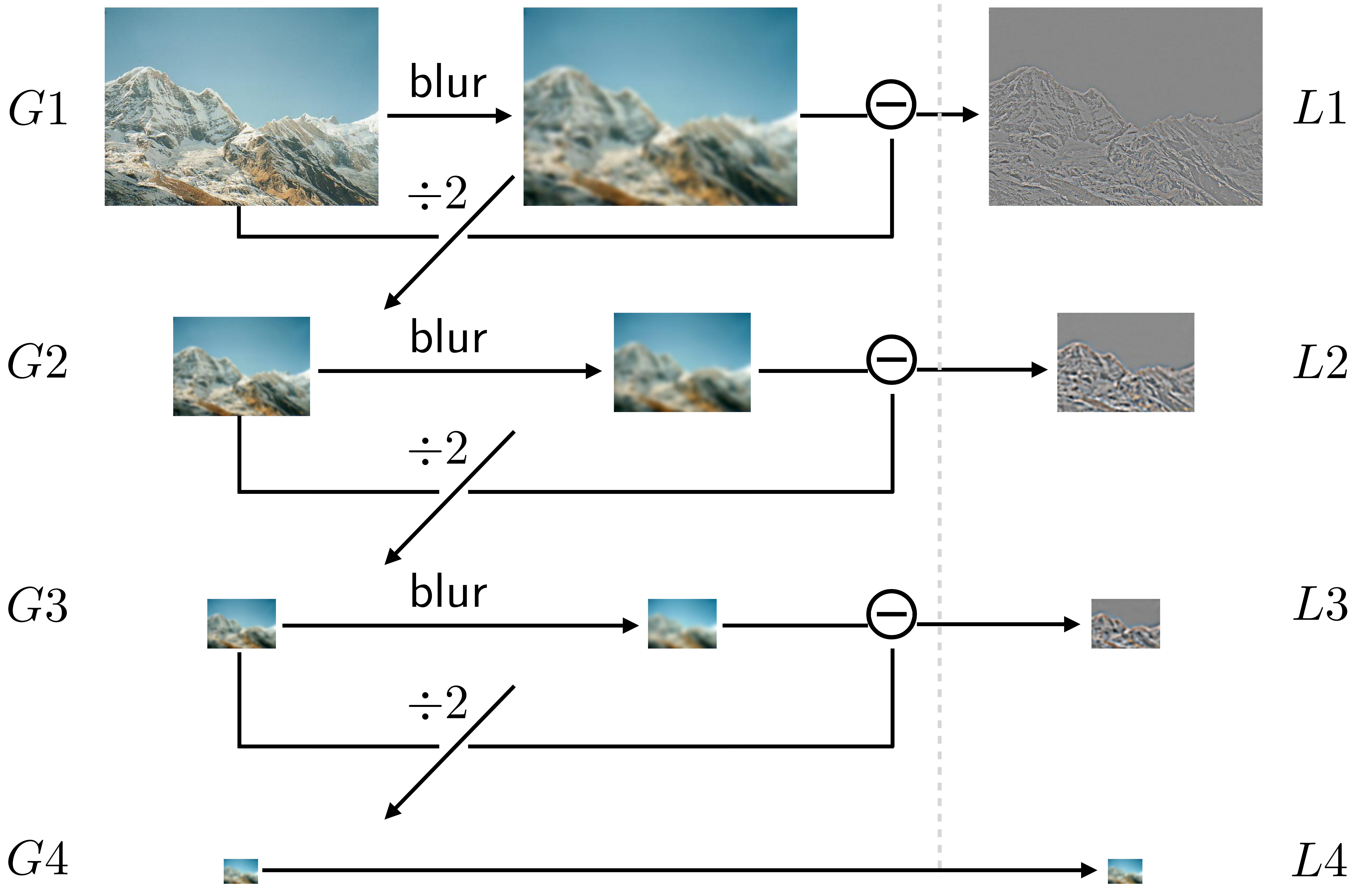
Gaussian Pyramid



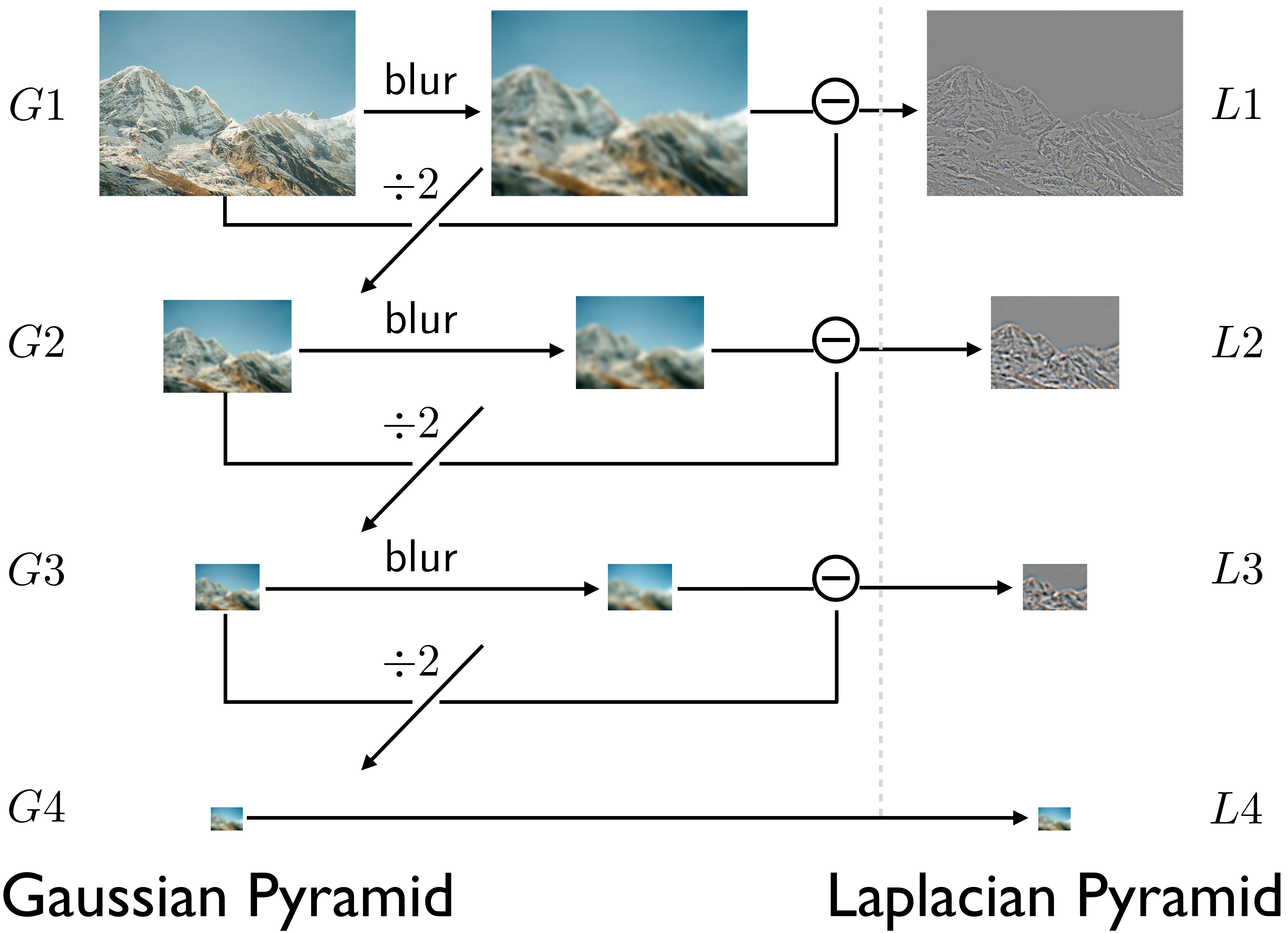
Gaussian Pyramid



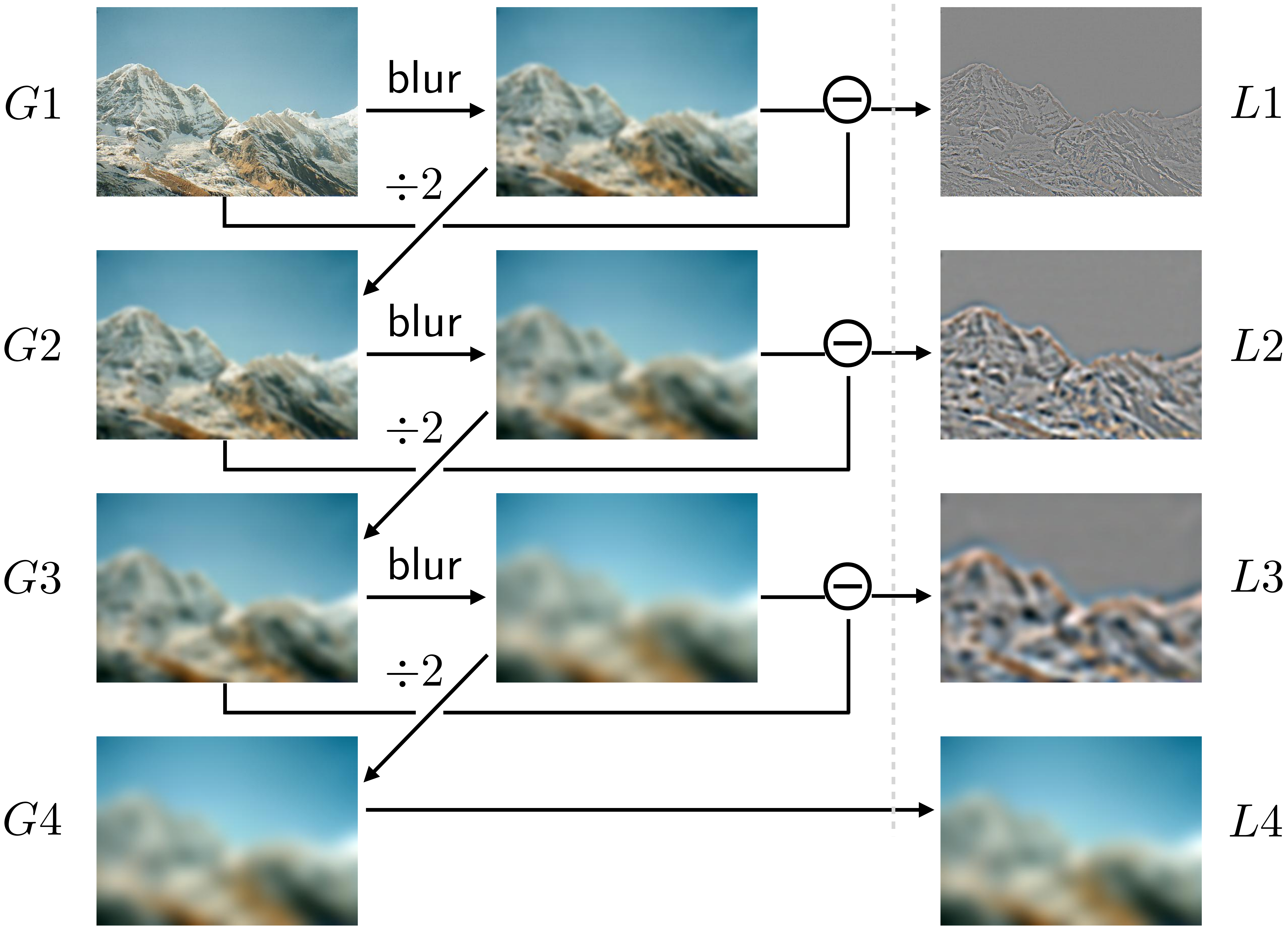
Gaussian Pyramid



Gaussian Pyramid



G1
blur
÷2
L1
G2
blur
÷2
L2
G3
blur
÷2
L3
G4
L4
Gaussian Pyramid
Laplacian Pyramid



$G1$
blur
$\div 2$
$L1$
$G2$
blur
$\div 2$
$L2$
$G3$
blur
$\div 2$
$L3$
$G4$
$L4$

Laplacian Pyramid

Building a **Laplacian** pyramid:

— Create a Gaussian pyramid

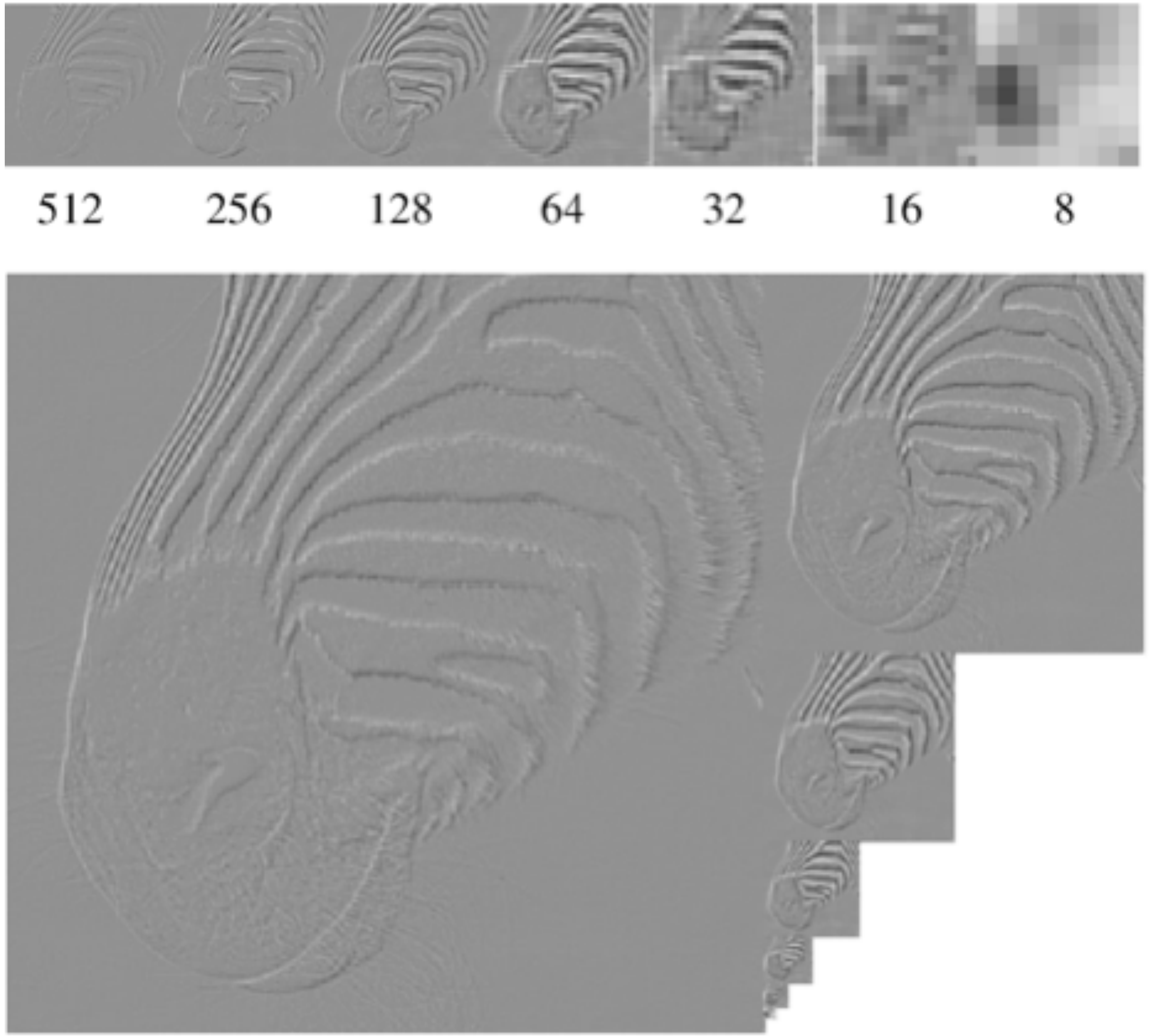
— Take the difference between one Gaussian pyramid level and the next

Properties

— Computes a Laplacian / Difference-of-Gaussian (DoG) function of the image at multiple scales

— It is a band pass filter – each level represents a different band of spatial frequencies

Laplacian Pyramid

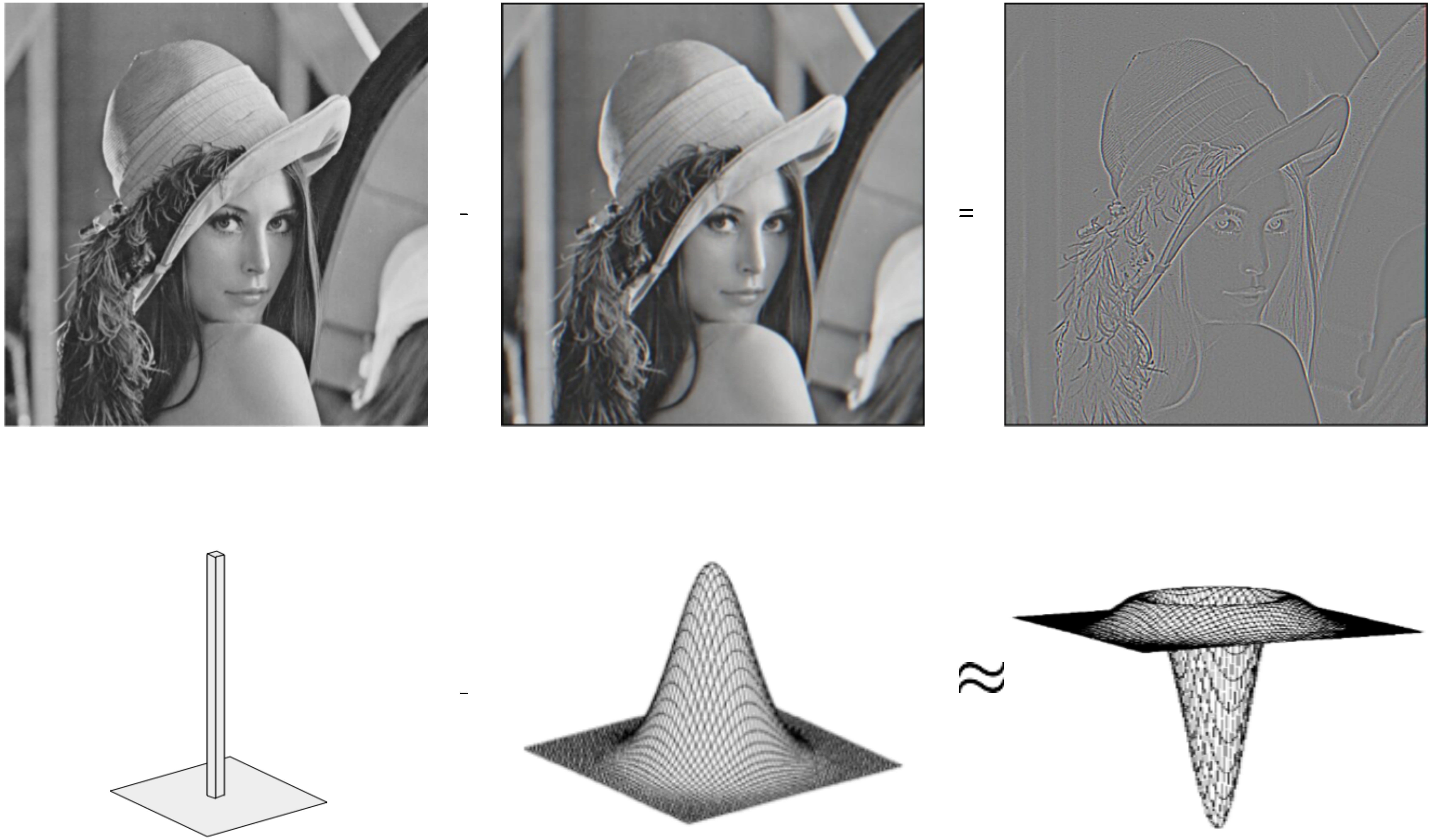


At each level, retain the residuals instead of the blurred images themselves.

Why is it called Laplacian Pyramid?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Why **Laplacian** Pyramid?



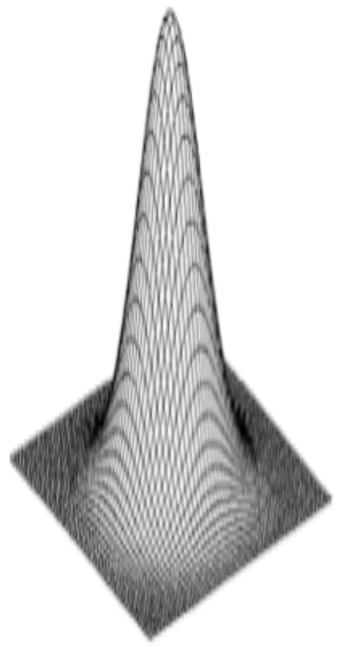
unit Gaussian Laplacian

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

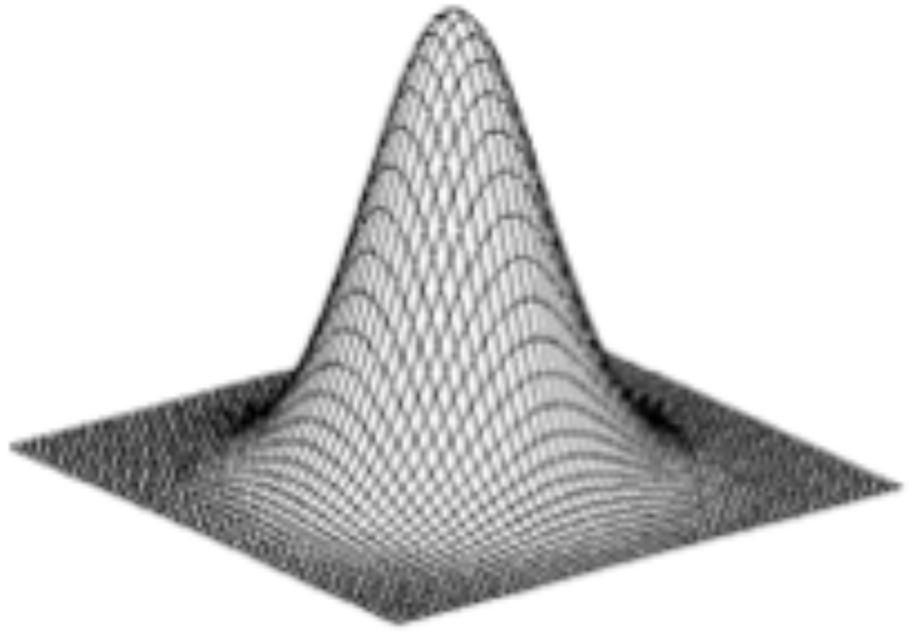
Why **Laplacian** Pyramid?

-

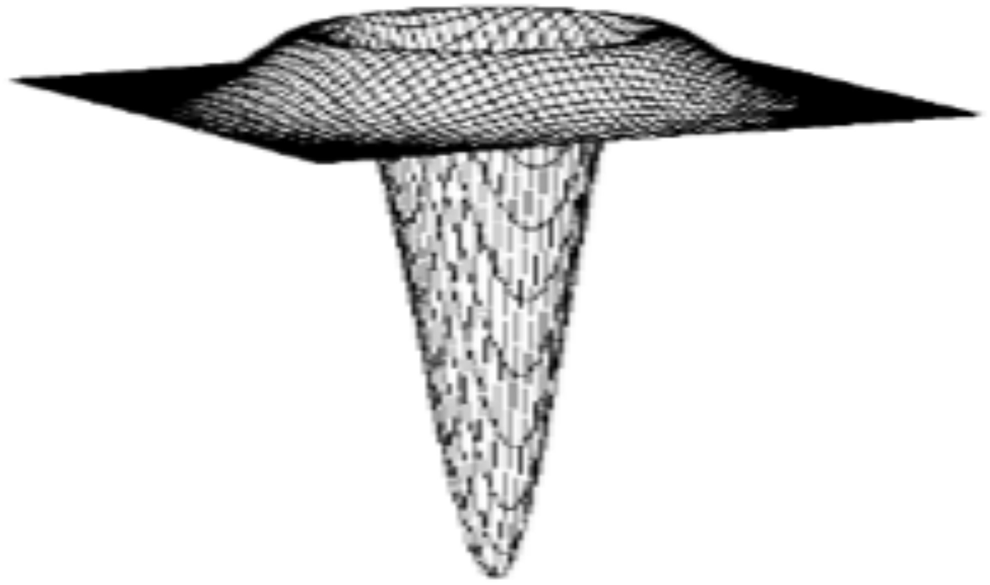
=



-



≈



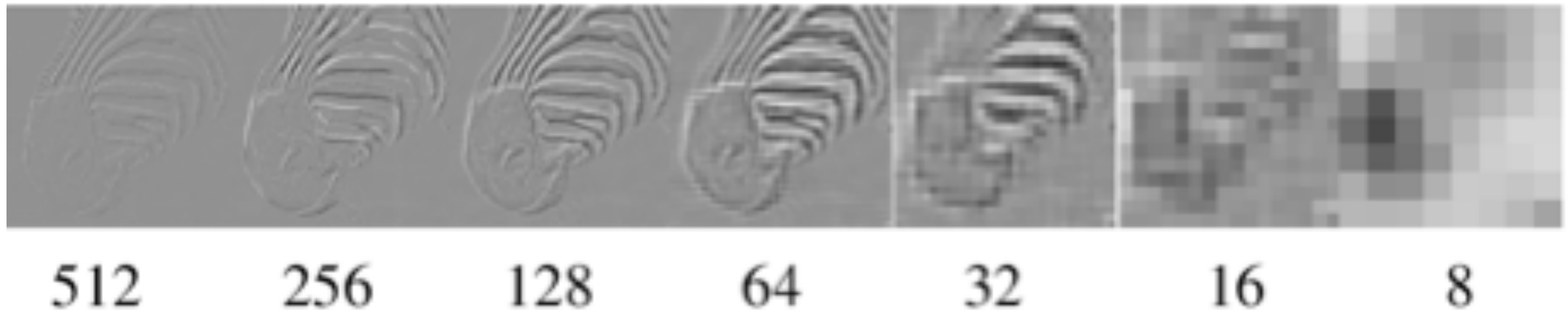
unit

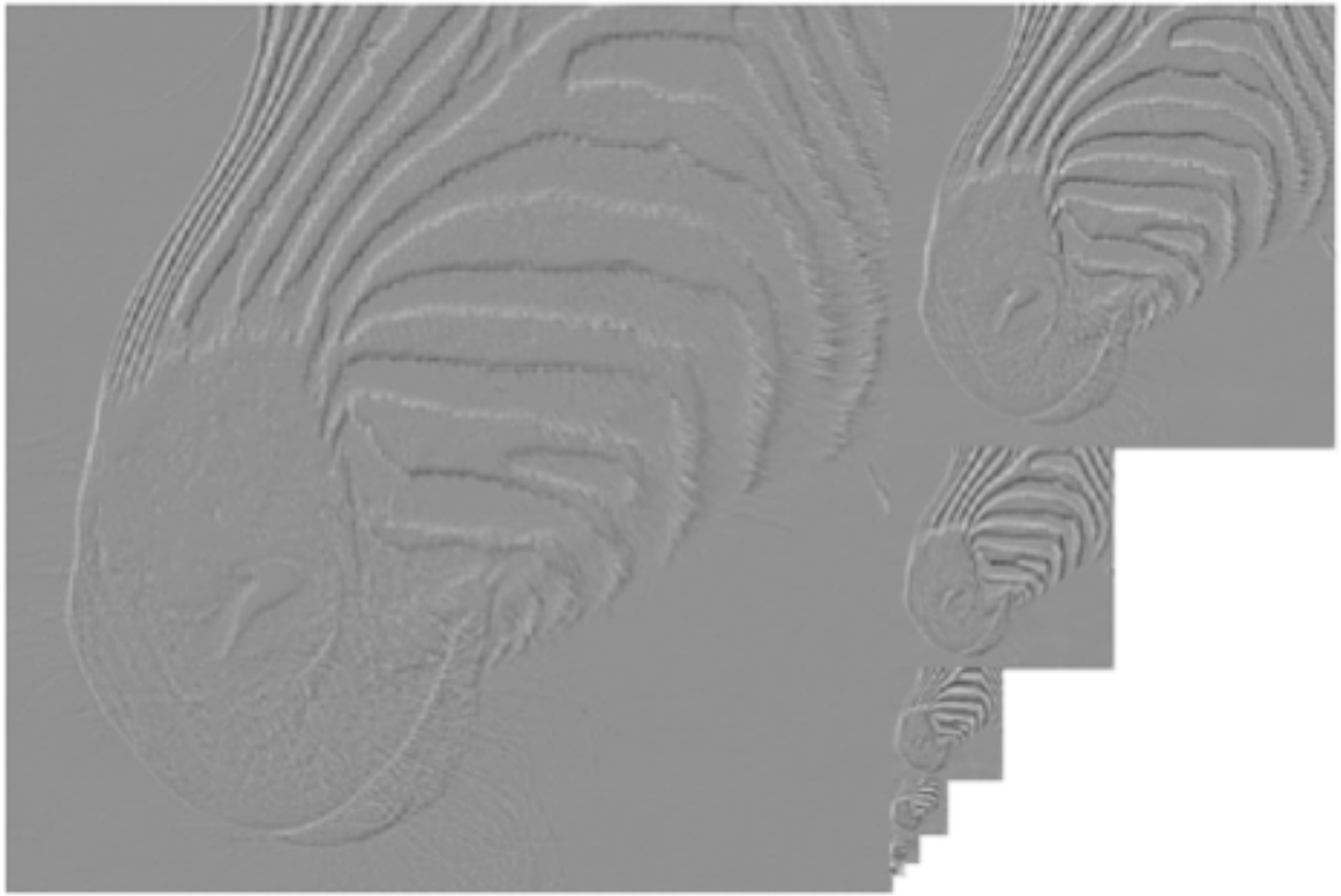
Gaussian

Laplacian

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Laplacian Pyramid





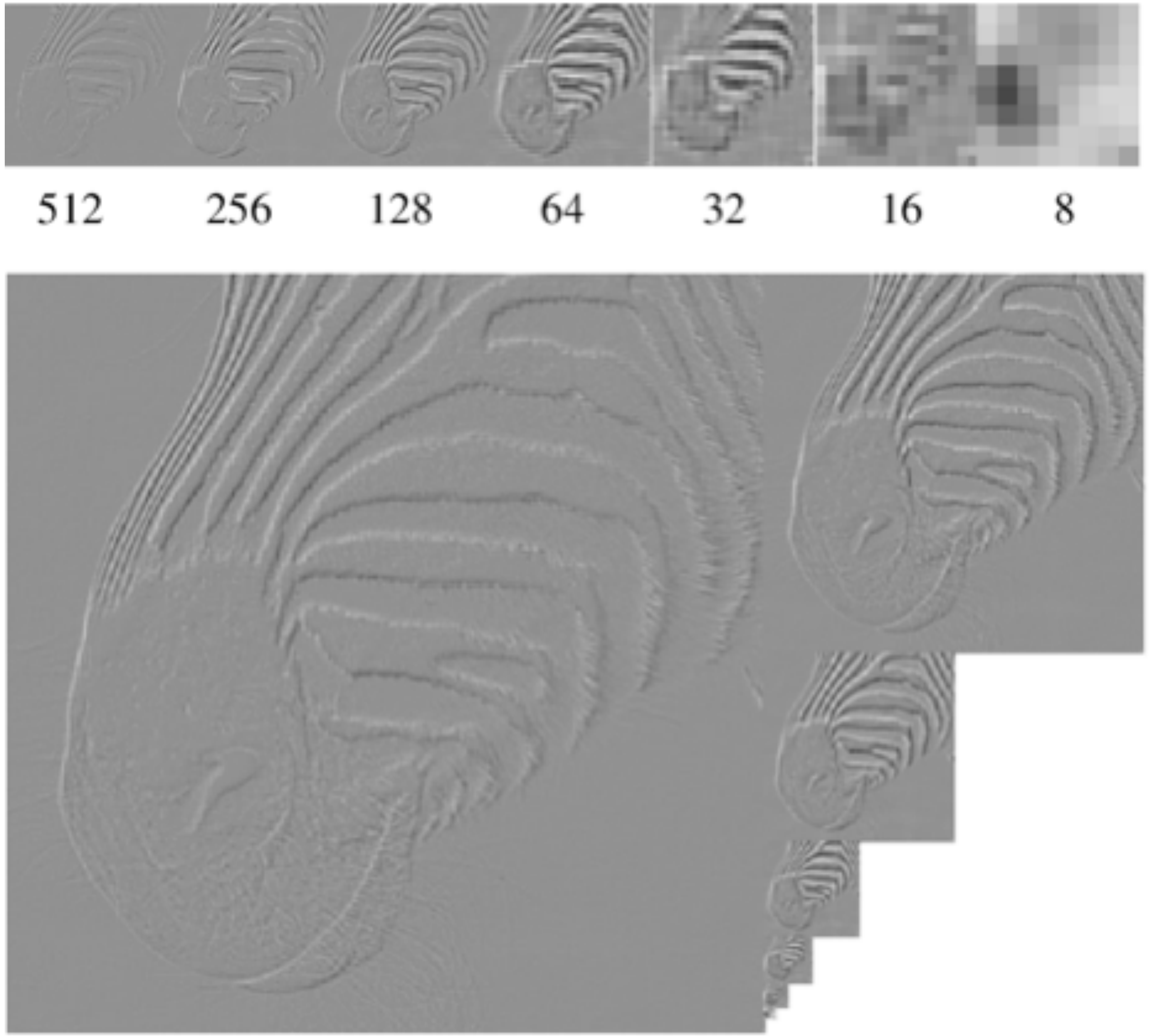
At each level, retain the residuals instead of the blurred images themselves.

Why is it called Laplacian Pyramid?

Can we reconstruct the original image using the pyramid?
— Yes we can!

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Laplacian Pyramid



At each level, retain the residuals instead of the blurred images themselves.

Why is it called Laplacian Pyramid?

Can we reconstruct the original image using the pyramid?
— Yes we can!

What do we need to store to be able to reconstruct the original image?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Let's start by just looking at **one level**

=

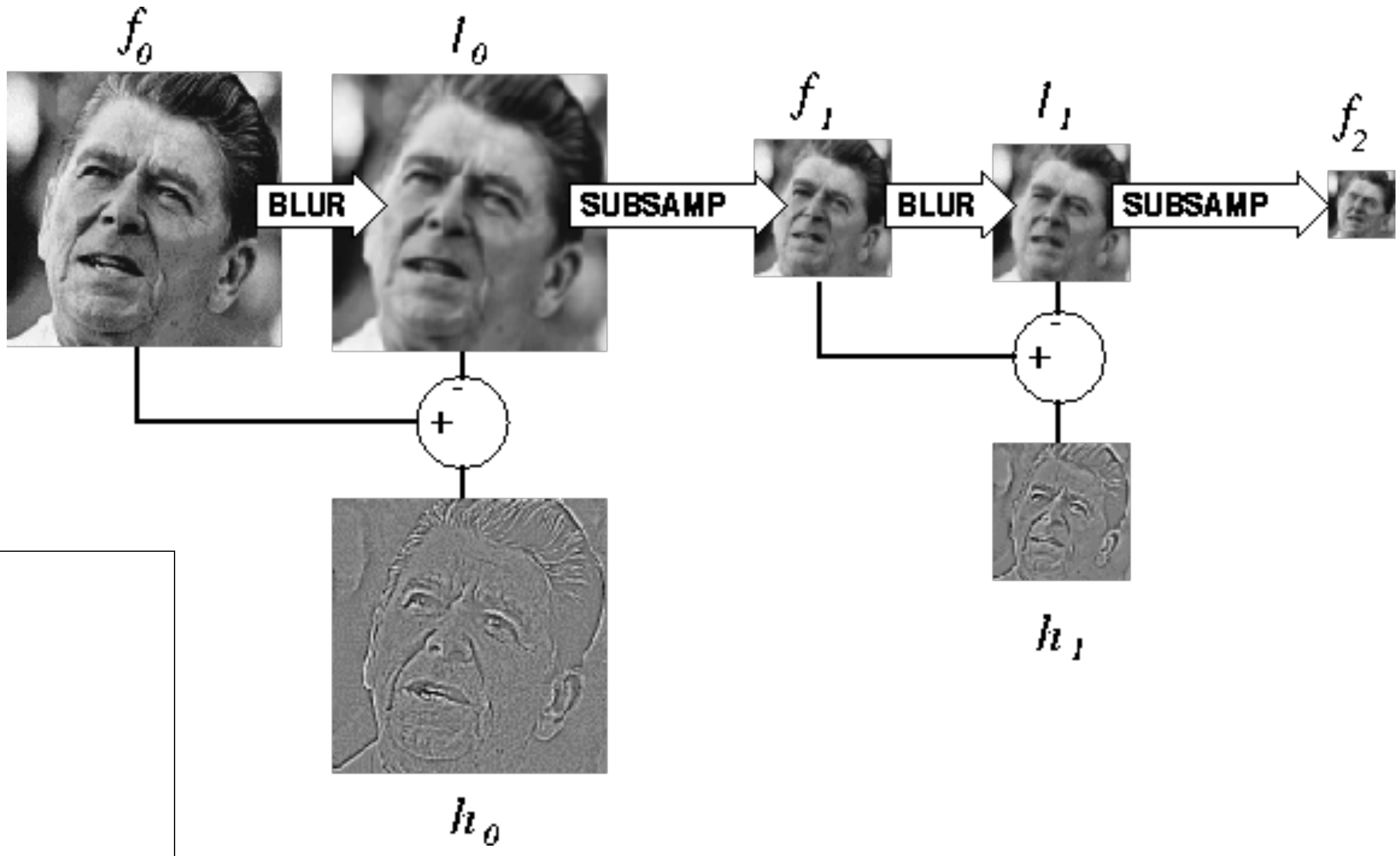
+

level 0 | level 1 (upsampled) | residual

Does this mean we need to store both residuals and the blurred copies of the original?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Constructing a **Laplacian** Pyramid

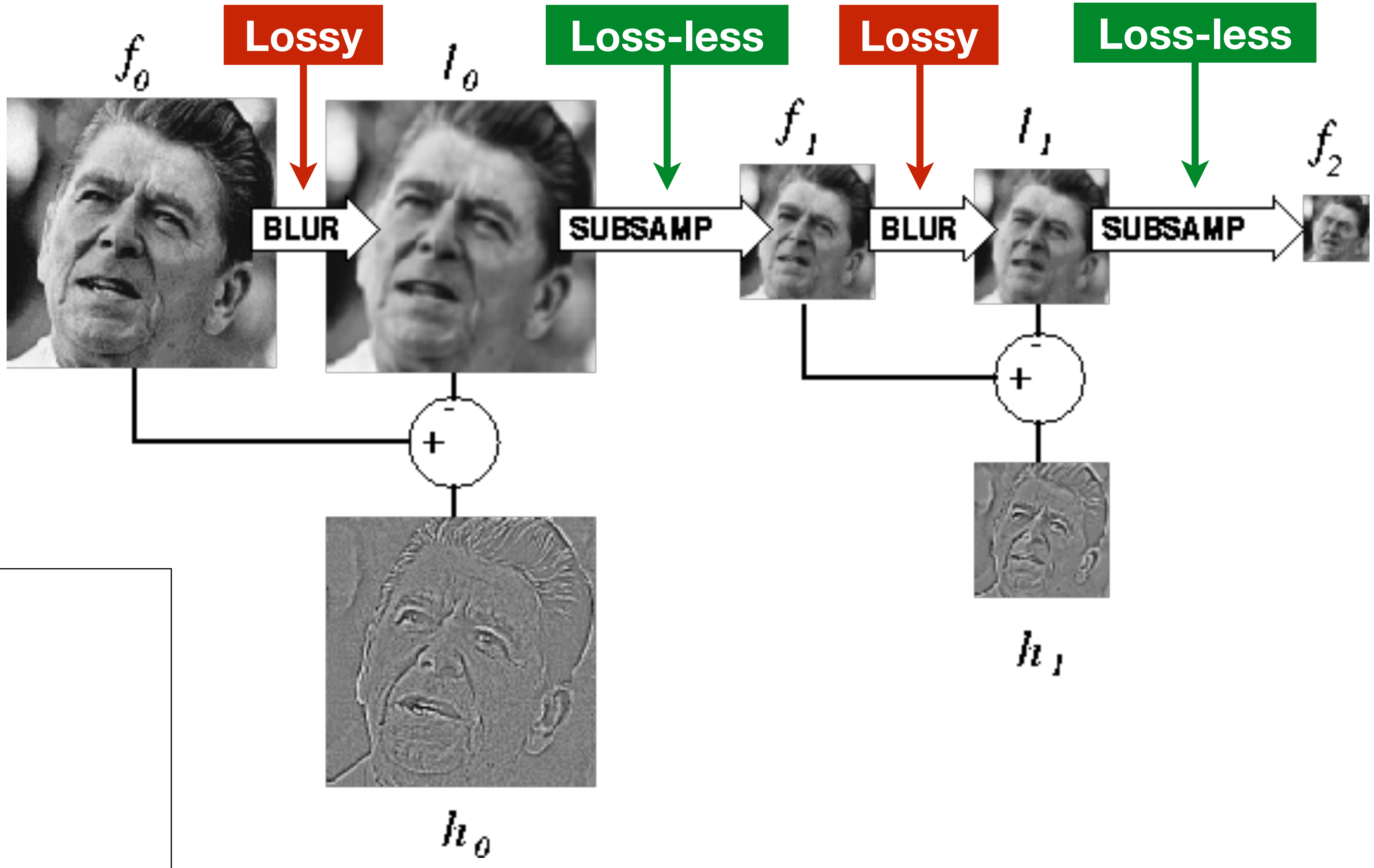


Algorithm

repeat:
  filter
  compute residual
  subsample
until min resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Constructing a **Laplacian** Pyramid

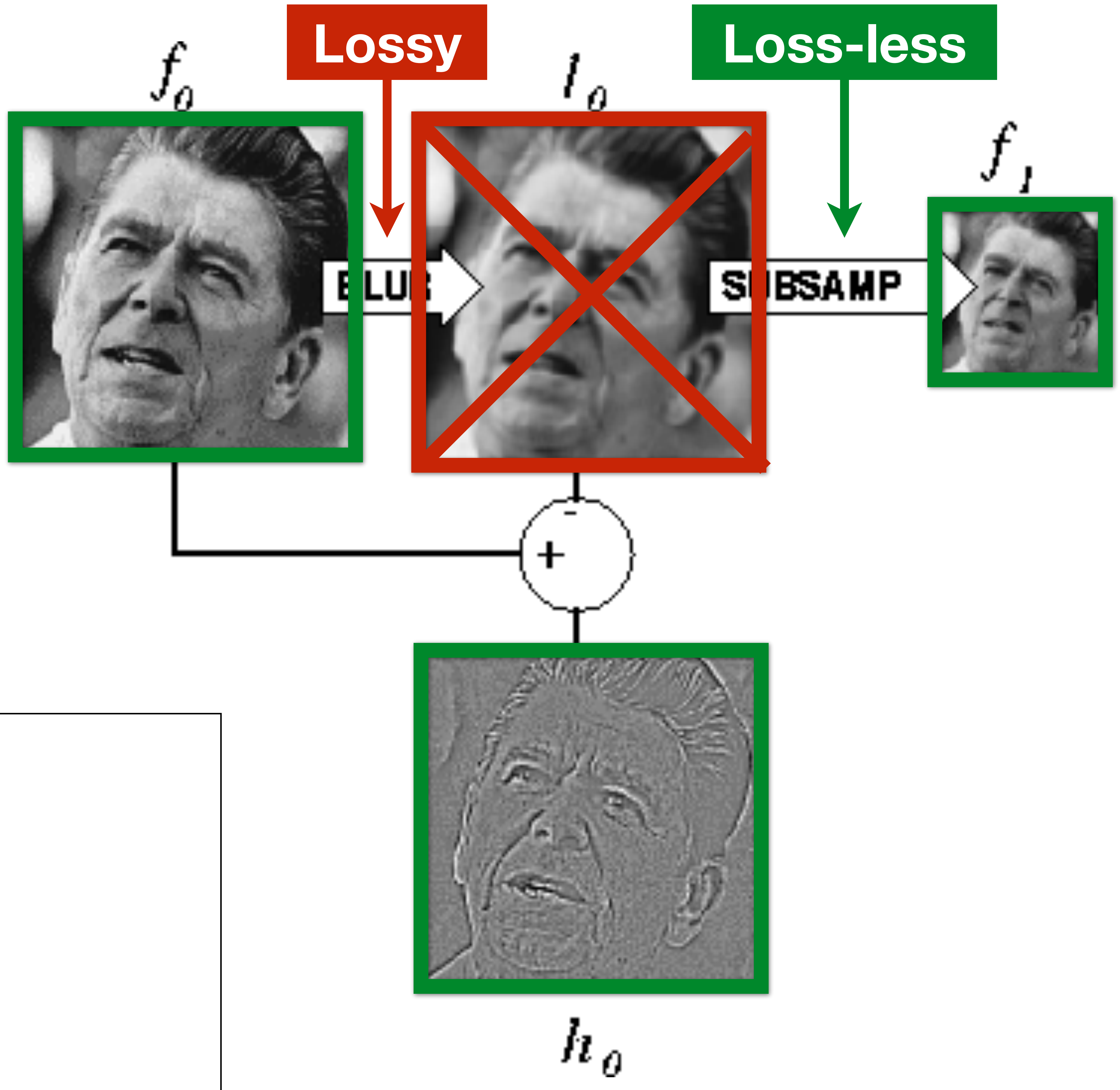


Algorithm

repeat:
filter
compute residual
subsample
until min resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Constructing a **Laplacian** Pyramid



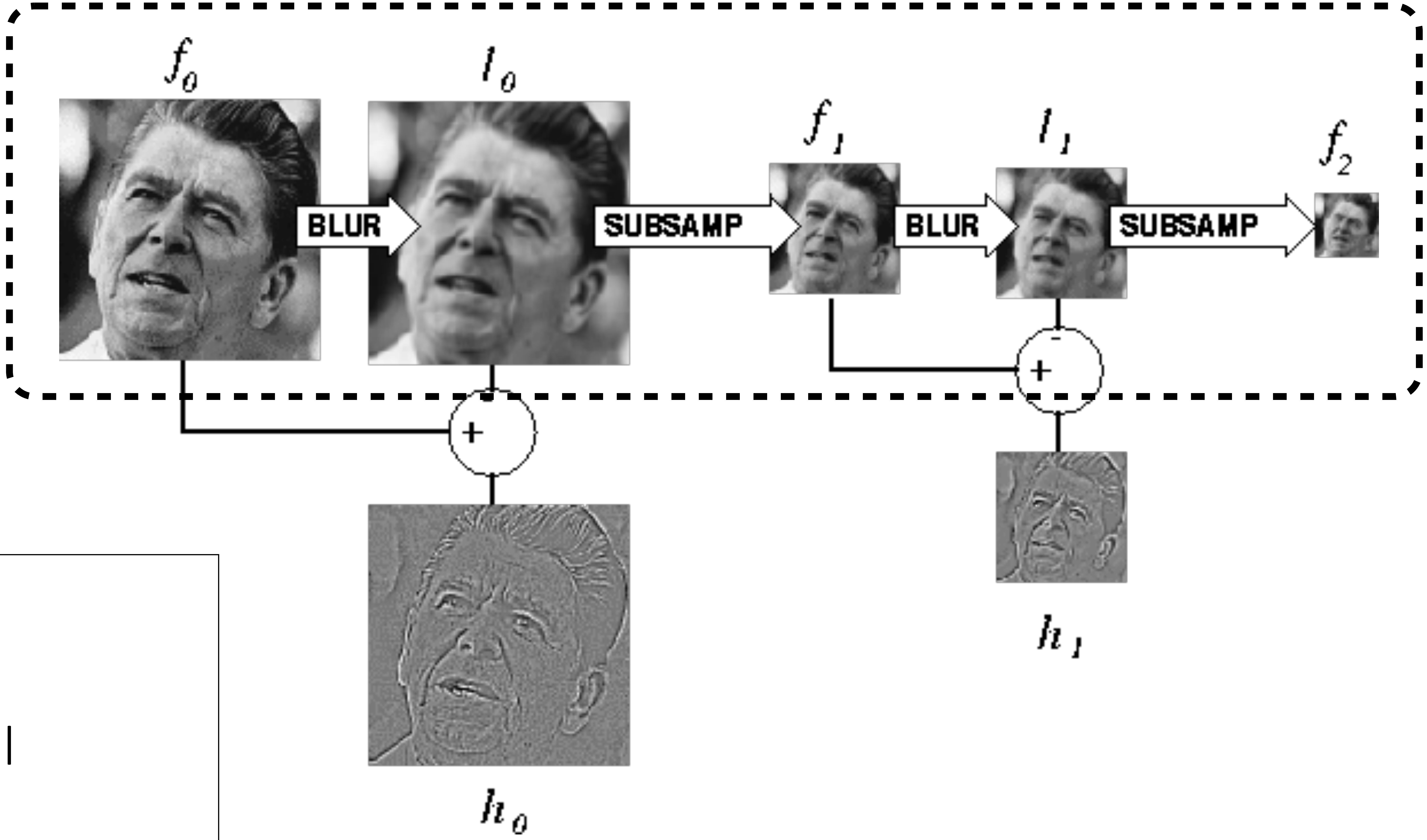
Algorithm

repeat:
filter
compute residual
subsample
until min resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Constructing a **Laplacian** Pyramid

What is this part?



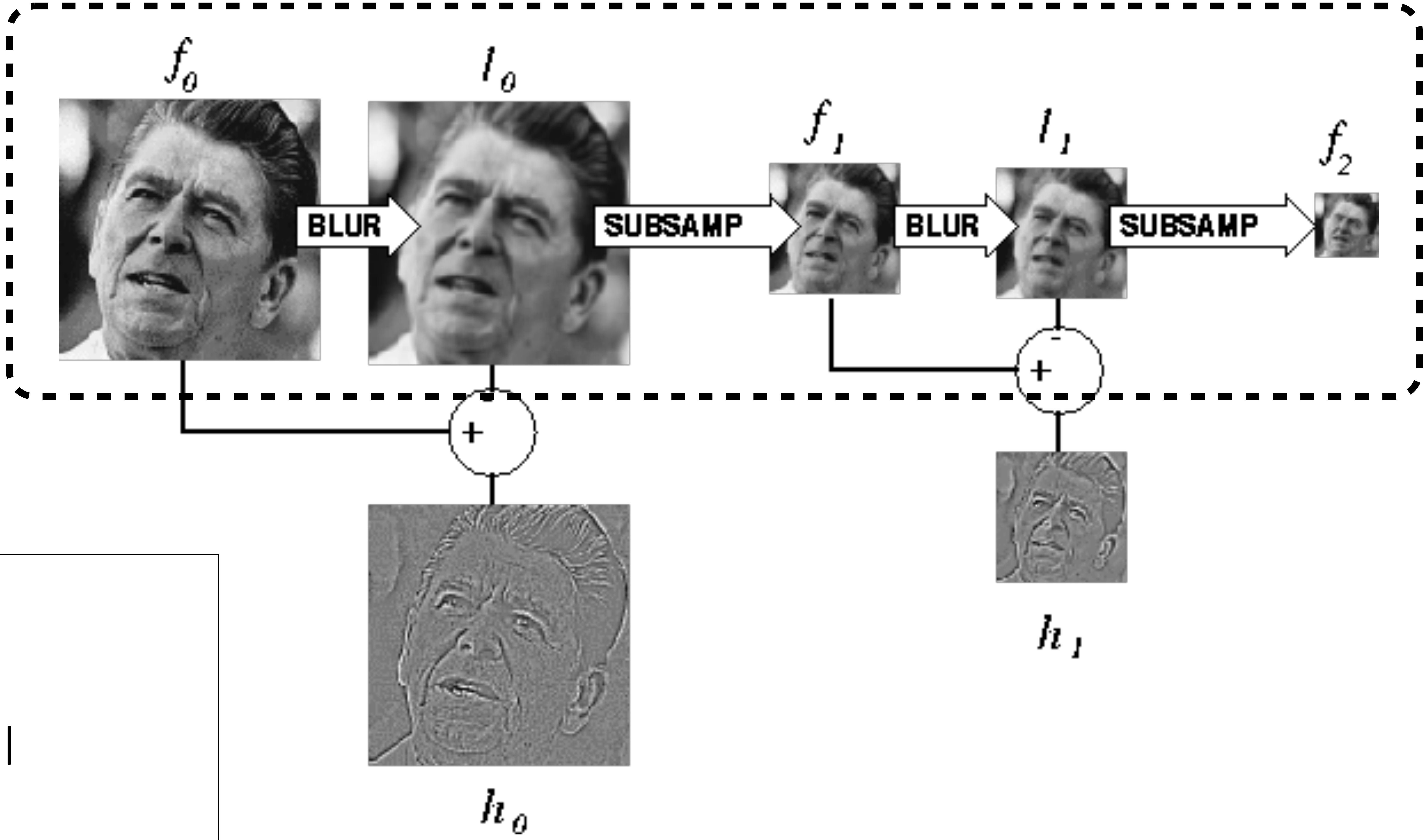
Algorithm

repeat:
  filter
  compute residual
  subsample
until min resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Constructing a **Laplacian** Pyramid

It's a Gaussian Pyramid



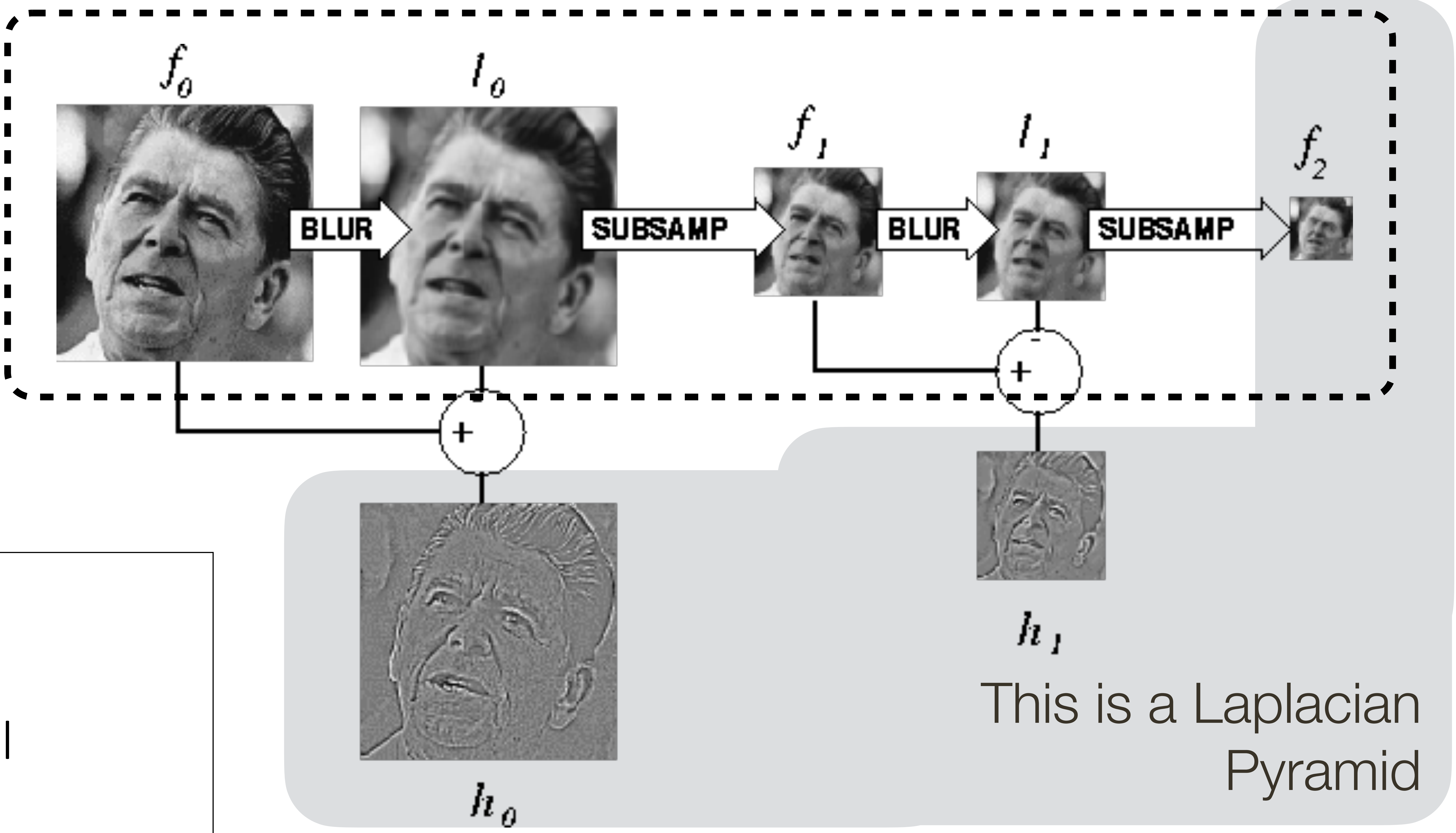
Algorithm

repeat:
filter
compute residual
subsample
until min resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Constructing a **Laplacian** Pyramid

It's a Gaussian Pyramid



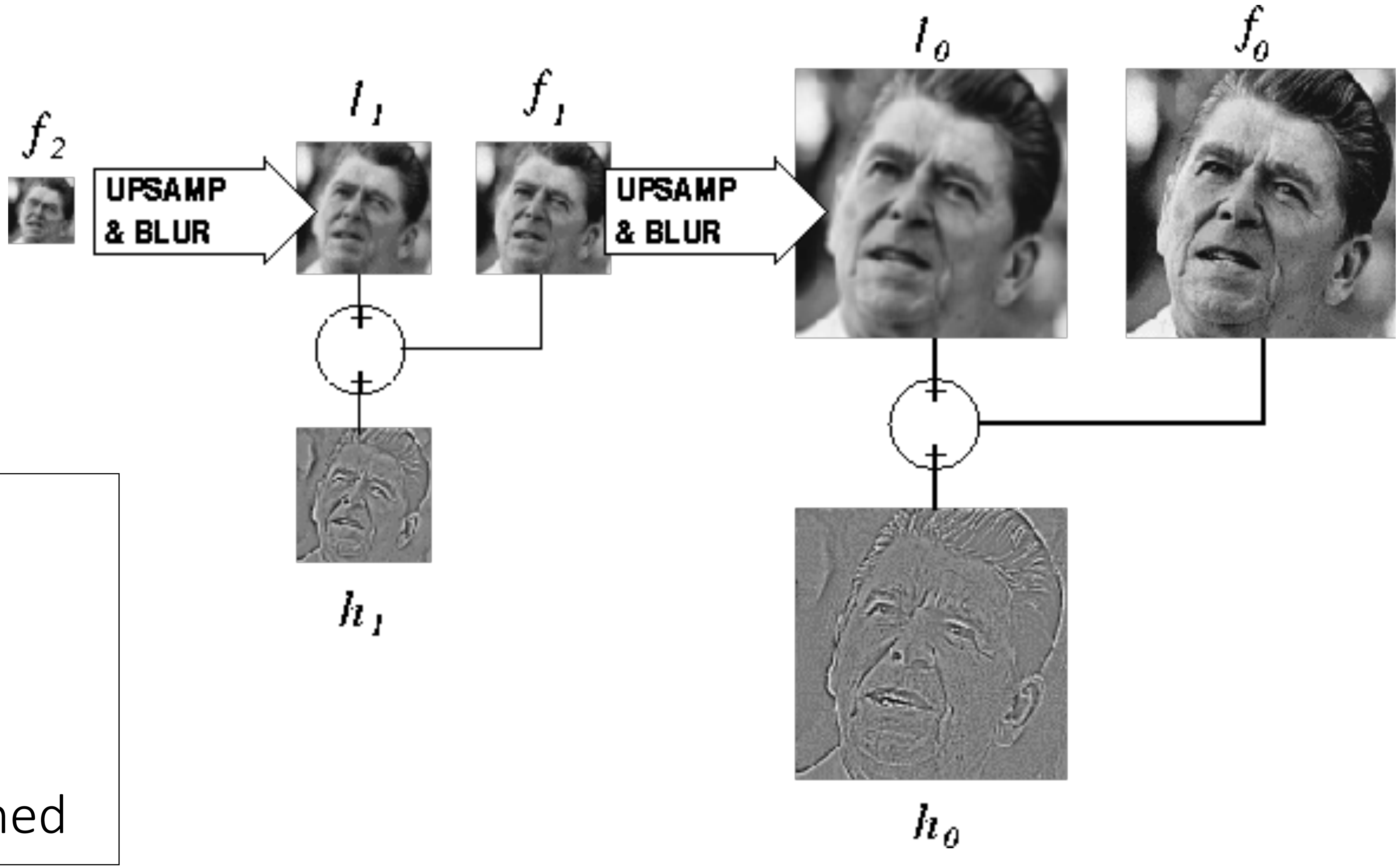
This is a Laplacian Pyramid

Algorithm

repeat:
filter
compute residual
subsample
until min resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Reconstructing the Original Image



Algorithm

repeat:

upsample

sum with residual

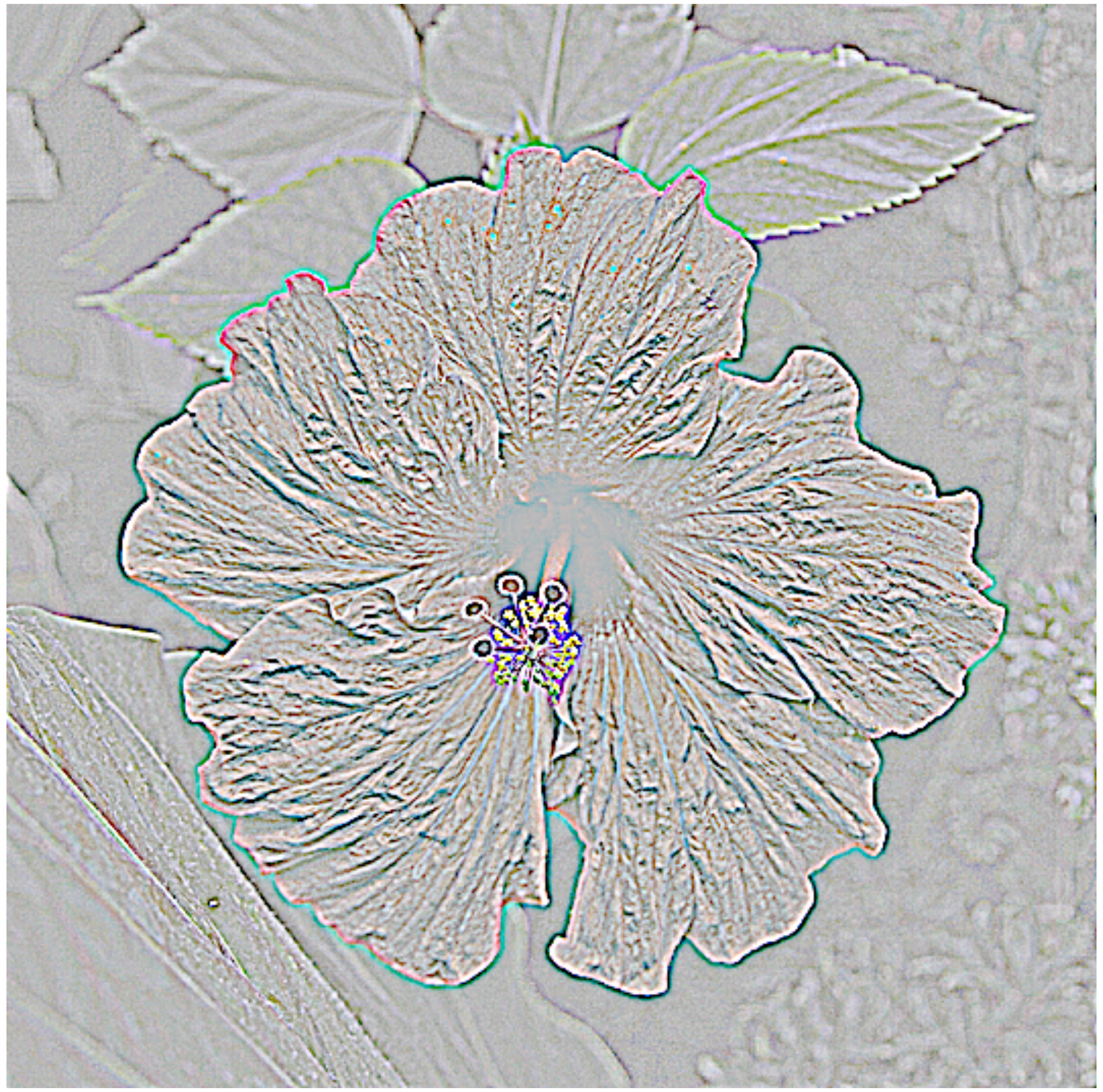
until orig resolution reached

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Gaussian vs **Laplacian** Pyramid

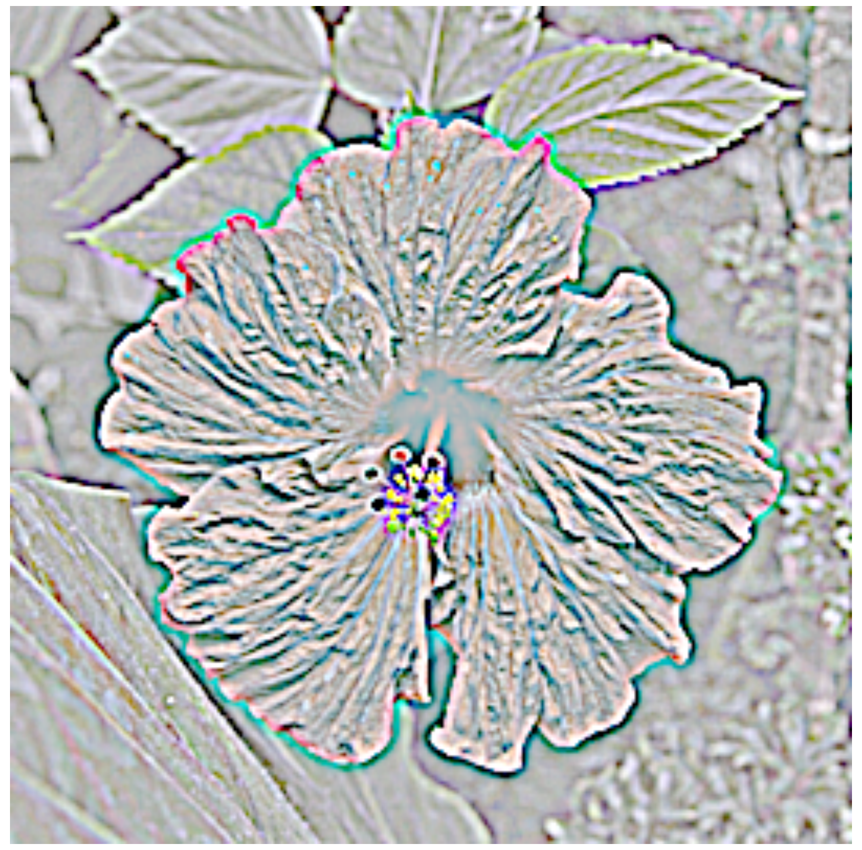



Shown in opposite order for space



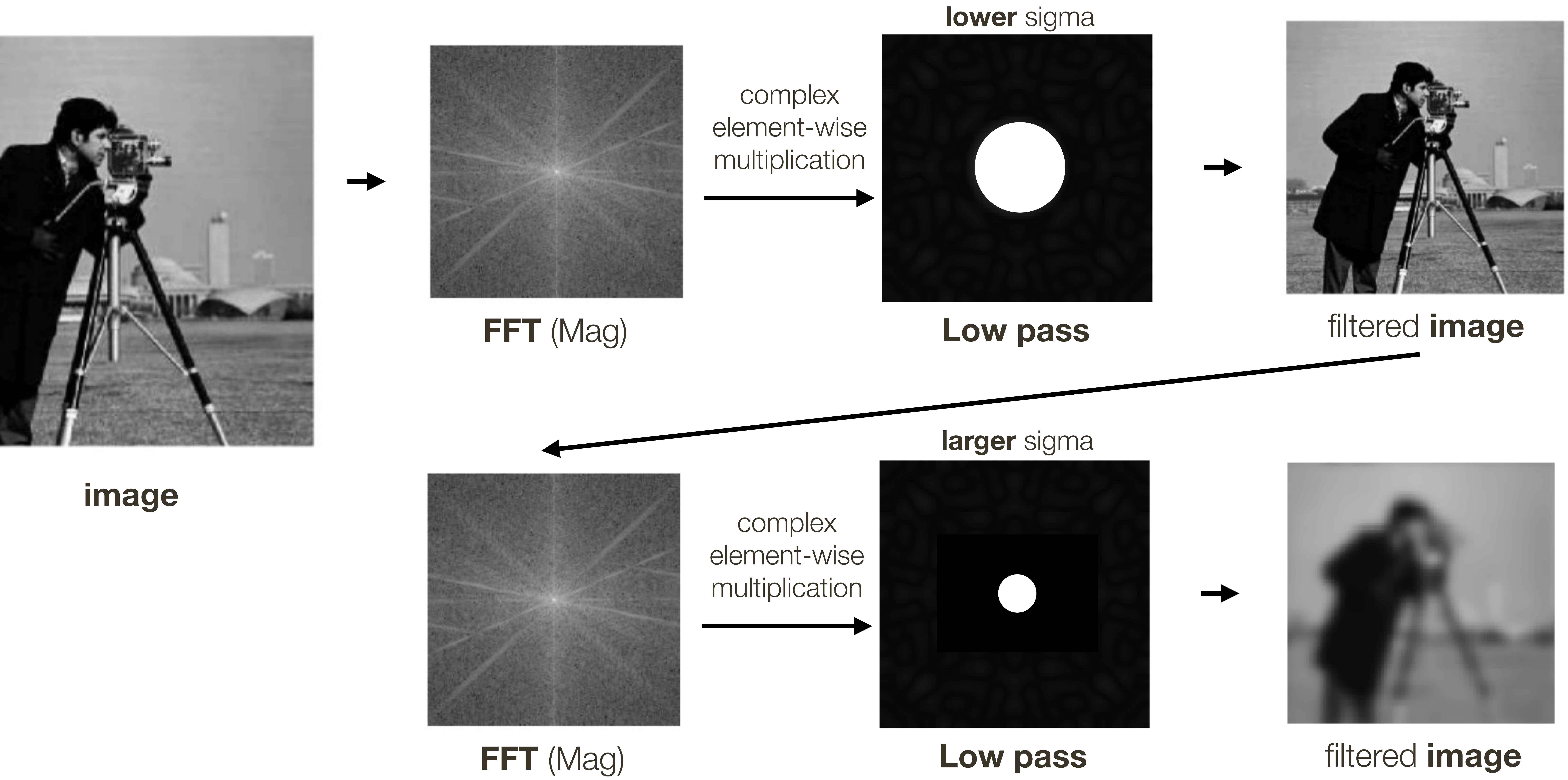
Which one takes more space to store?



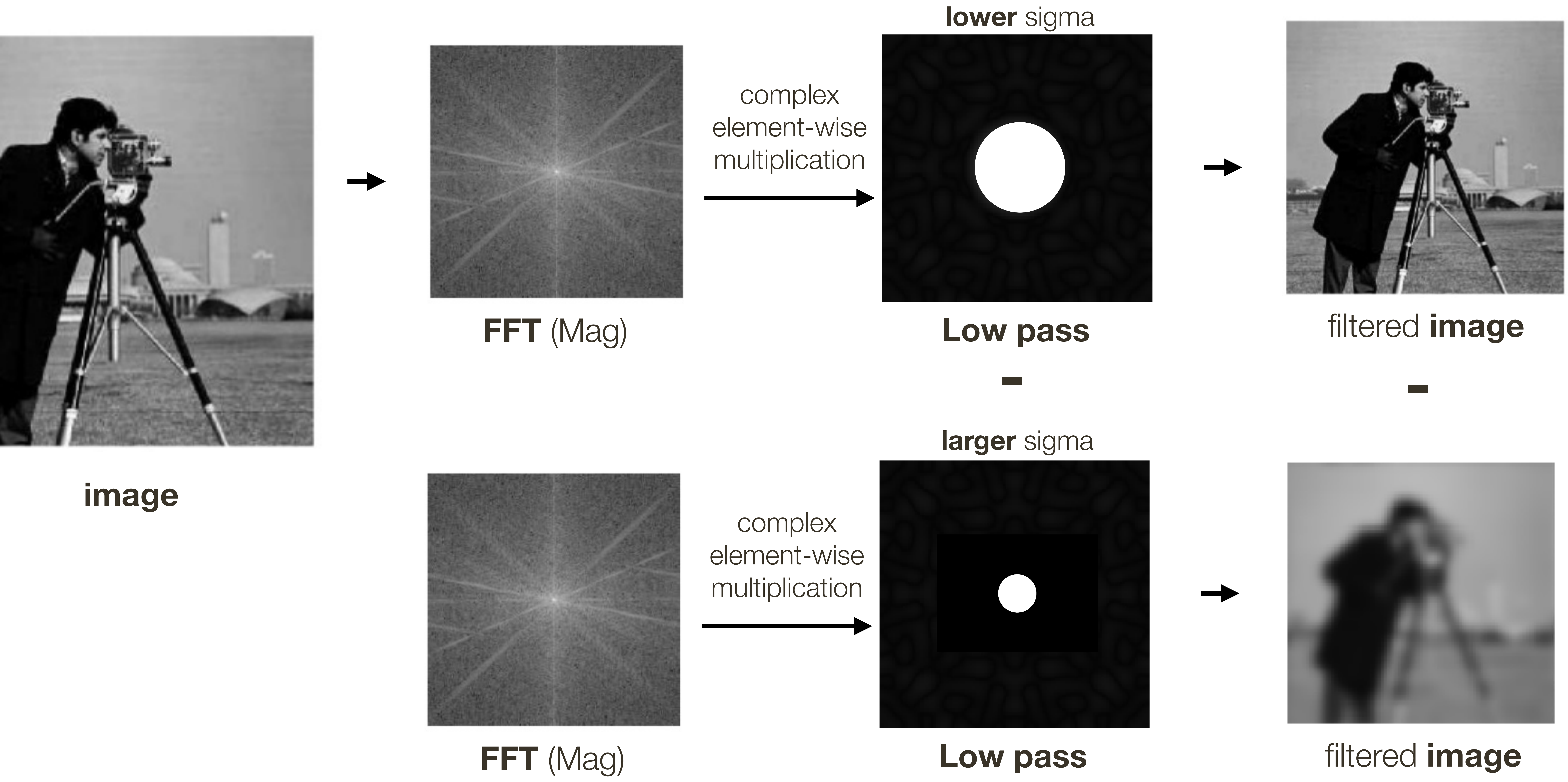


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Laplacian is a Bandpass Filter



Laplacian is a Bandpass Filter



Laplacian is a Bandpass Filter