THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

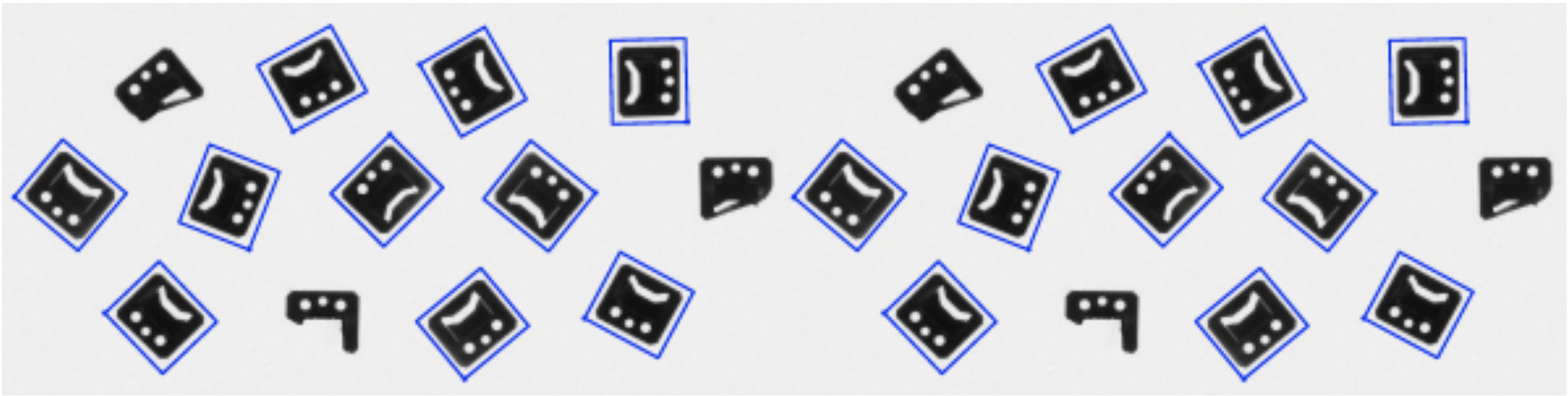


Image Credit: https://docs.adaptive-vision.com/4.7/studio/machine_vision_guide/TemplateMatching.html

Lecture 8: Scaled Representations

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

Menu for Today

Topics:

- Imaging **Blending**
- **Scaled** Representations
- Edge **Detection**
- **Quiz 1**

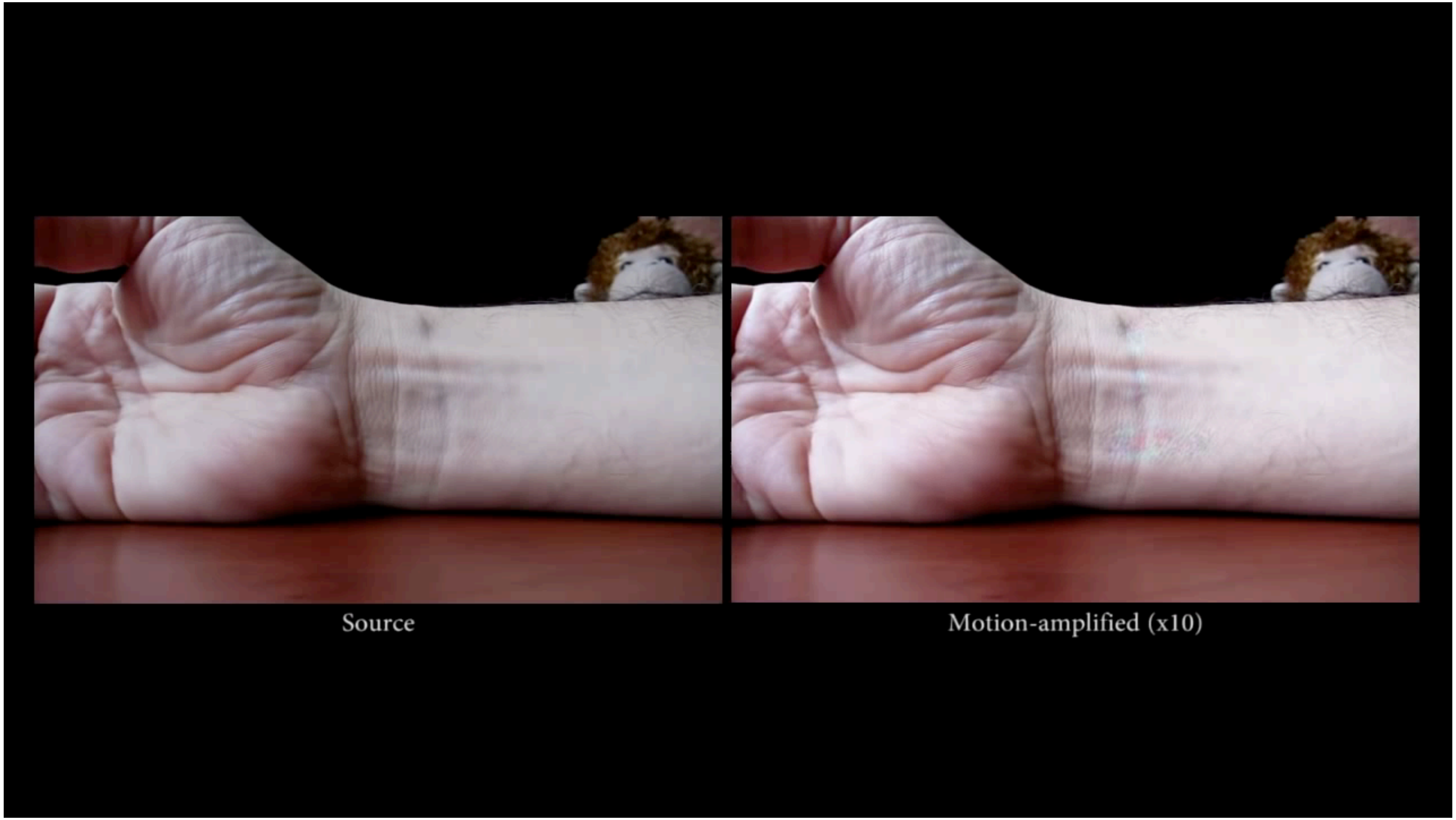
Readings:

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

Reminders:

- **Assignment 2**: Scaled Representations, Face Detection and Image Blending

Today's "**fun**" Example: Eulerian Video Magnification



Video From: Wu at al., Siggraph 2012

Today's "**fun**" Example: Eulerian Video Magnification

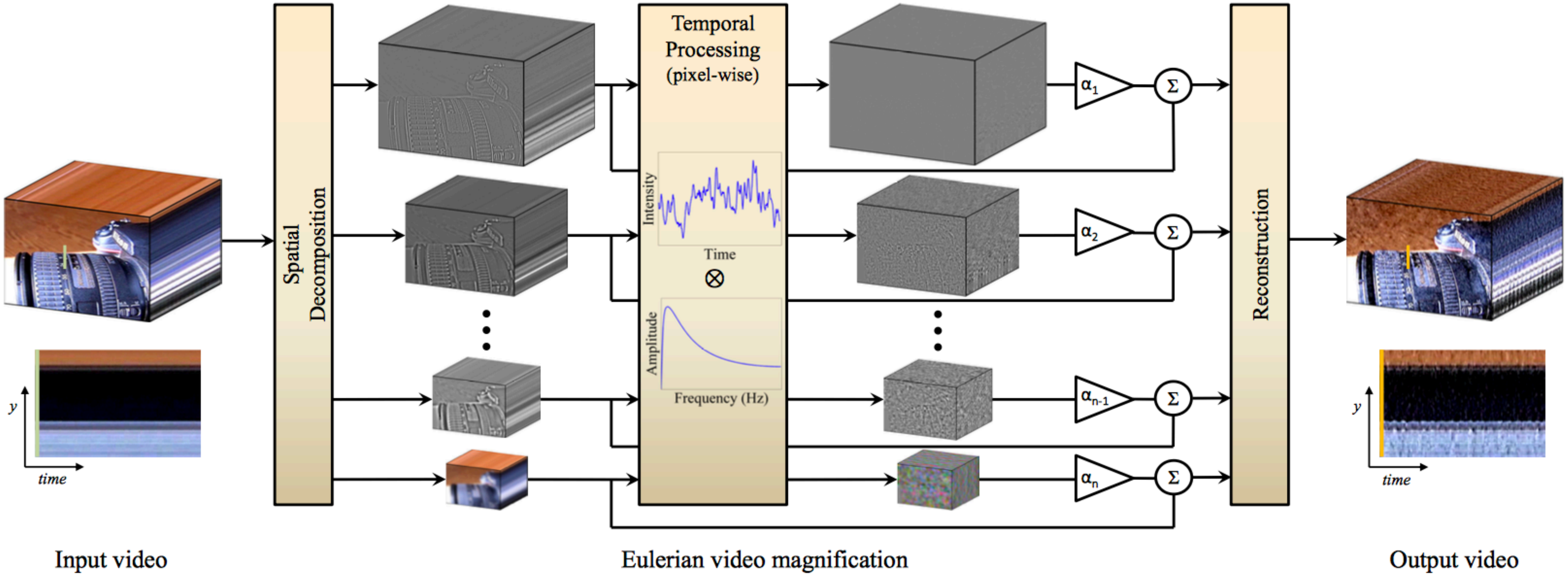
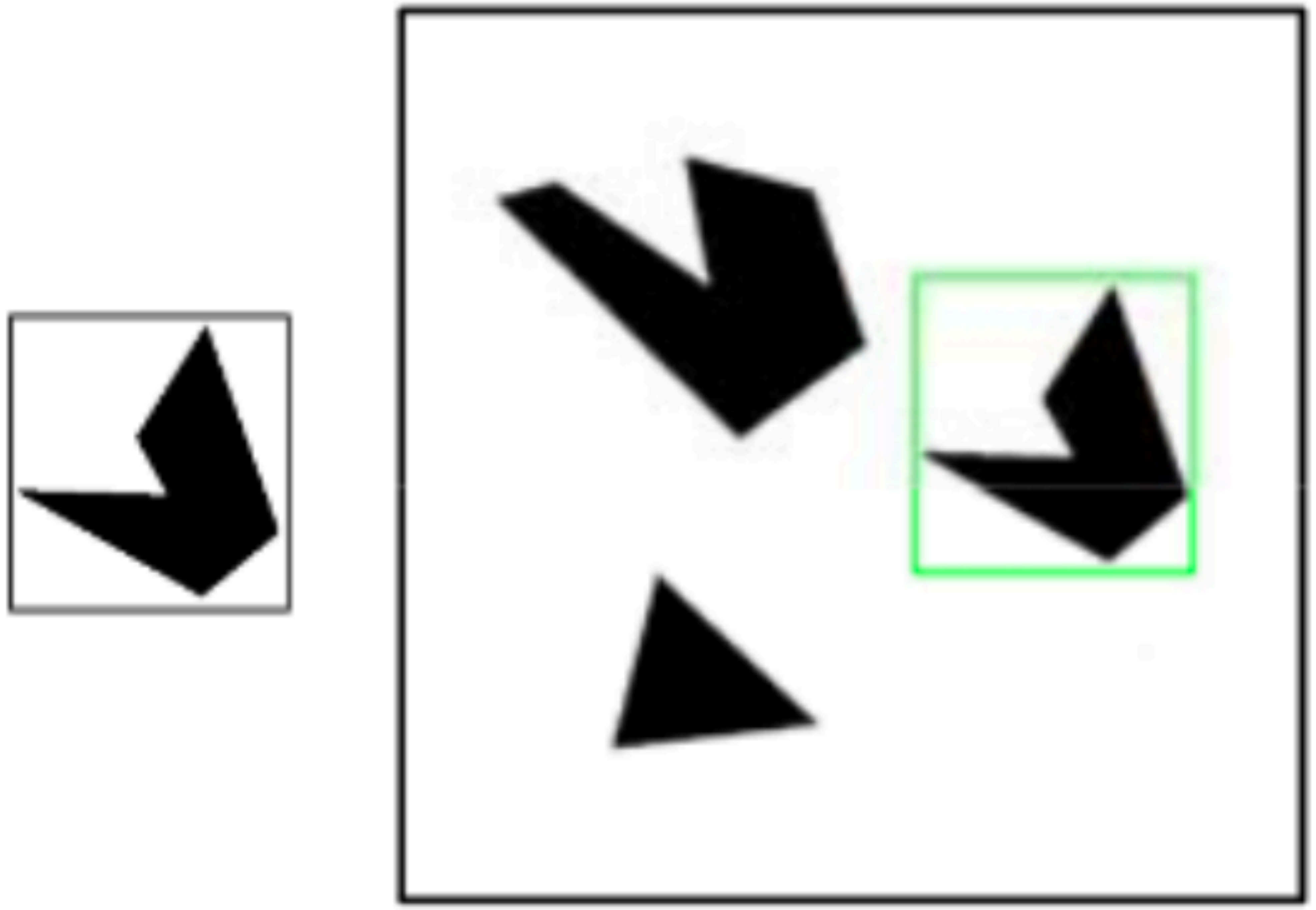
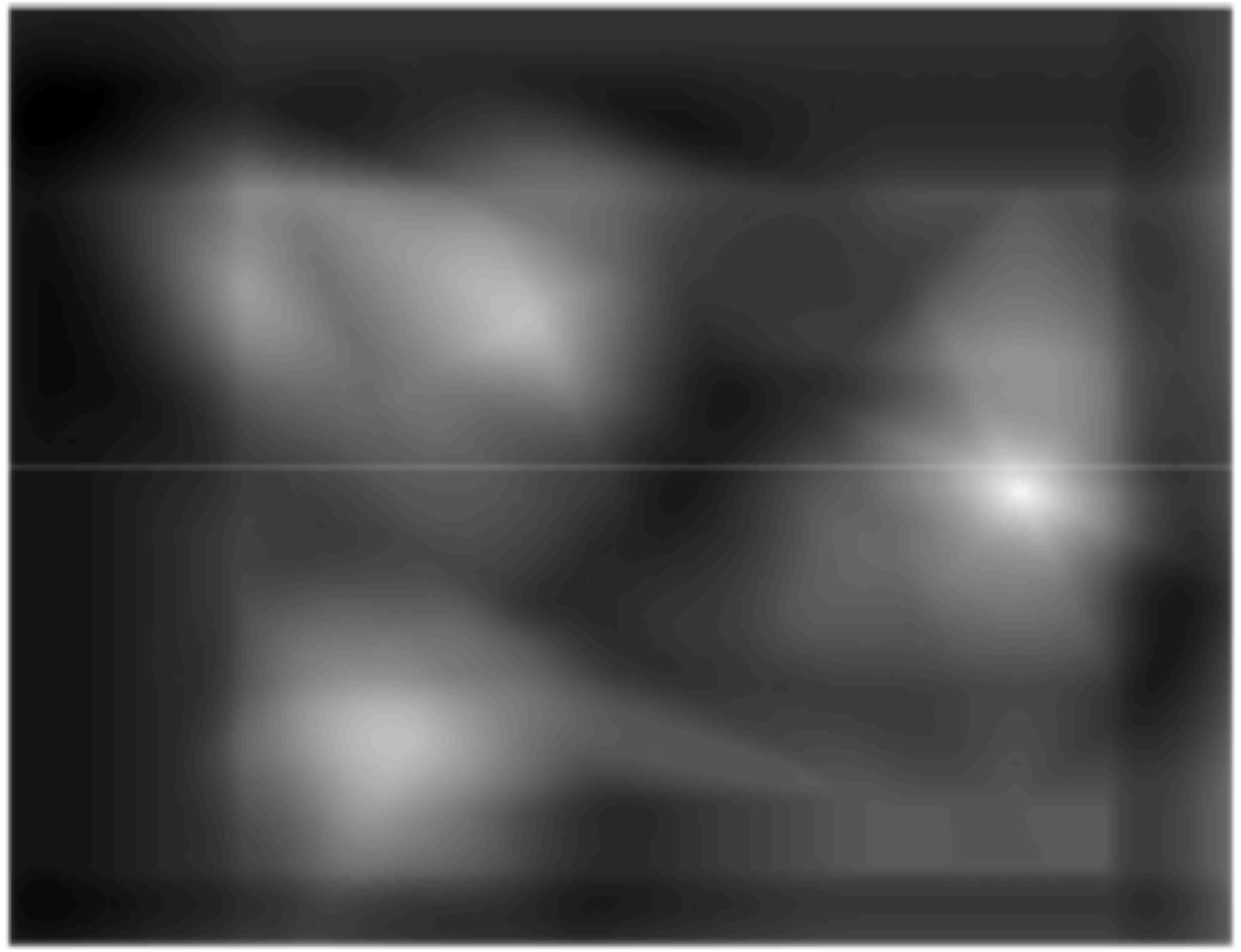


Figure From: Wu at al., Siggraph 2012

Lecture 7: Re-cap **Template** Matching



Detected template



Correlation map

Slide Credit: Kristen Grauman

Lecture 7: Re-cap **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**

Lecture 7: Re-cap **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 = $|\mathbf{I} - \mathbf{J}|^2$

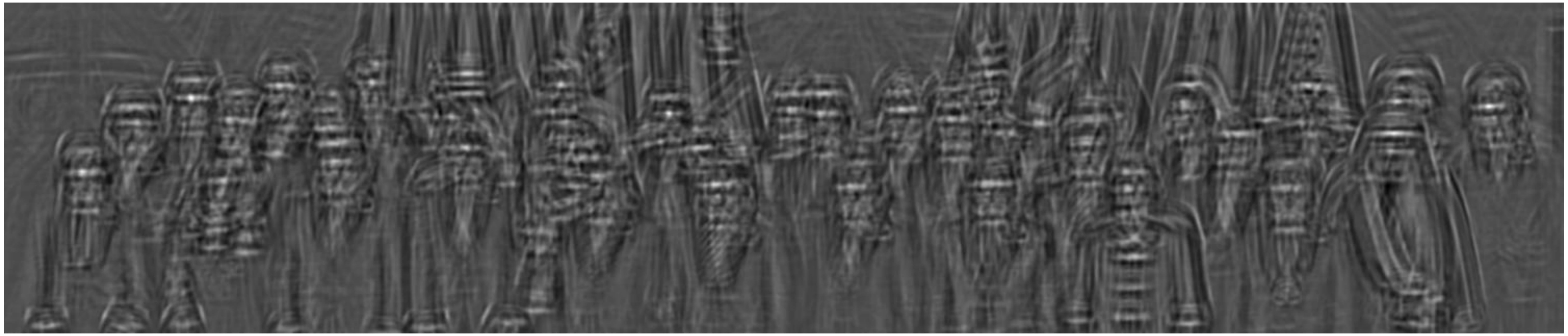
Normalized correlation varies between –1 and 1, attains the value 1 when the filter and image region are identical (up to a scale factor)

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

Template Matching

Correlate image with a template

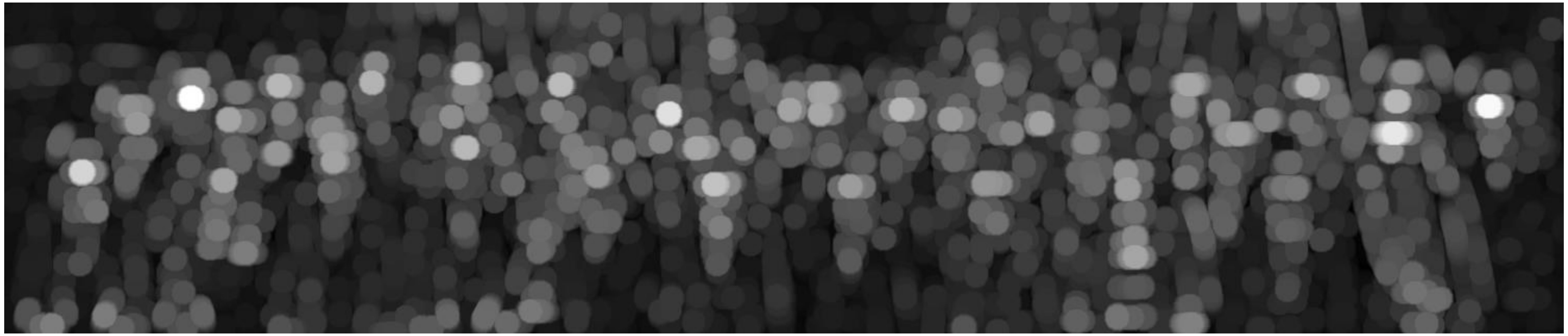
$*$ 



Template Matching

Correlate image with a template

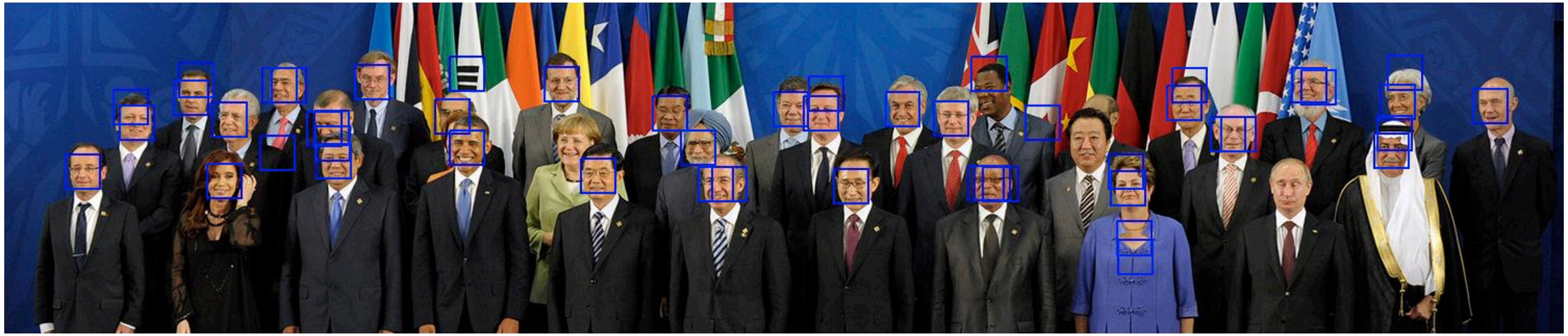
$*$  $\longrightarrow$ Non-max suppress $\longrightarrow$



Template Matching

Correlate image with a template

*  → Non-max suppress
+ threshold →



Detection Performance

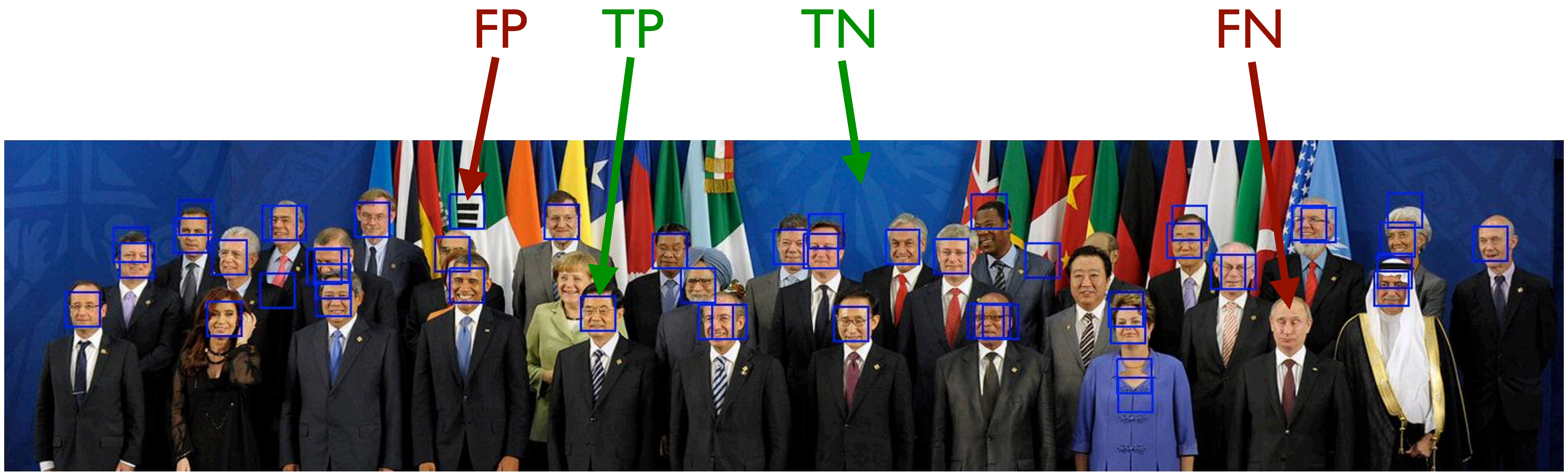
Types of errors in detection:

TP = True positive (true face and detected)

FP = False positive (not face and detected)

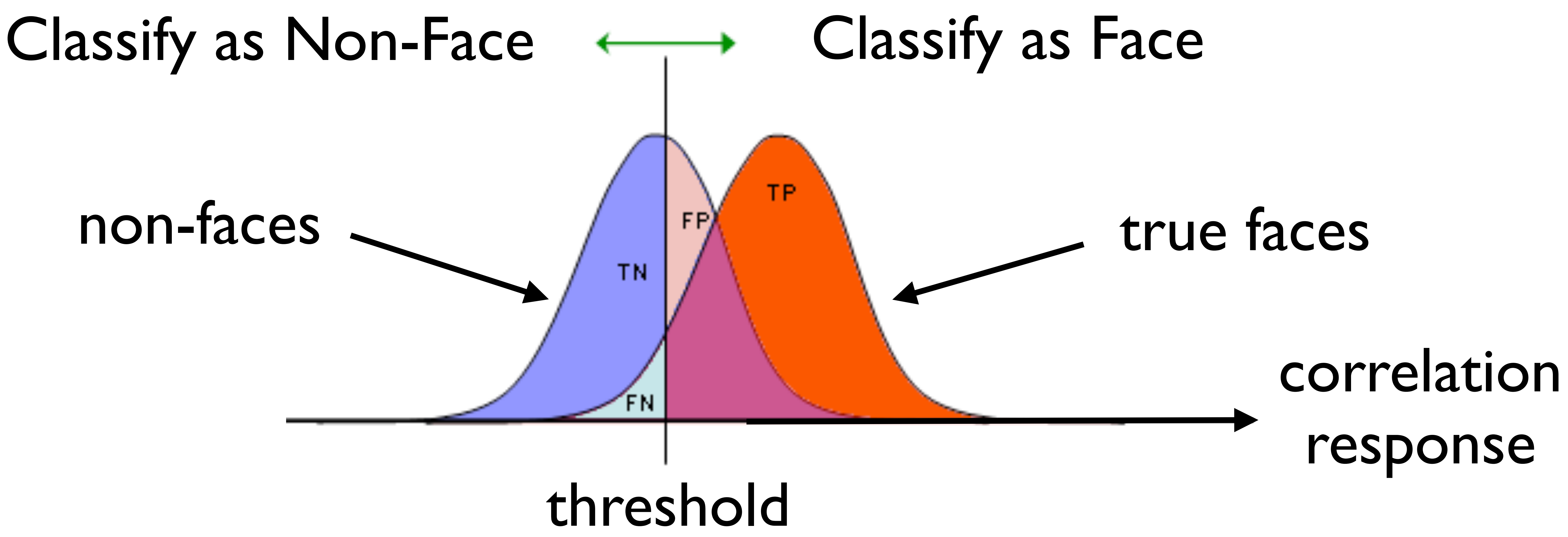
TN = True negative (not face and no detection)

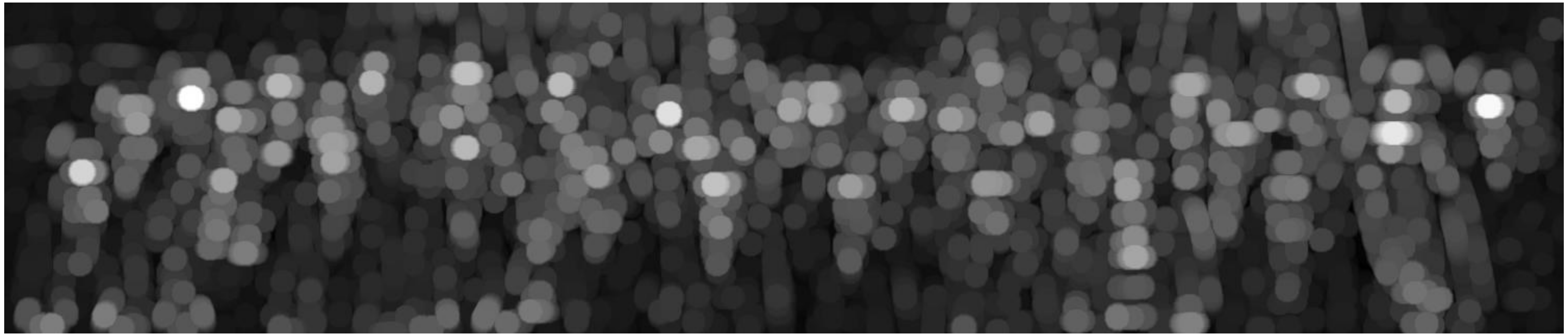
FN = False negative (true face and not detected)



Detection Performance

Depending on where we set the threshold, we can tradeoff between true positives and false positives:





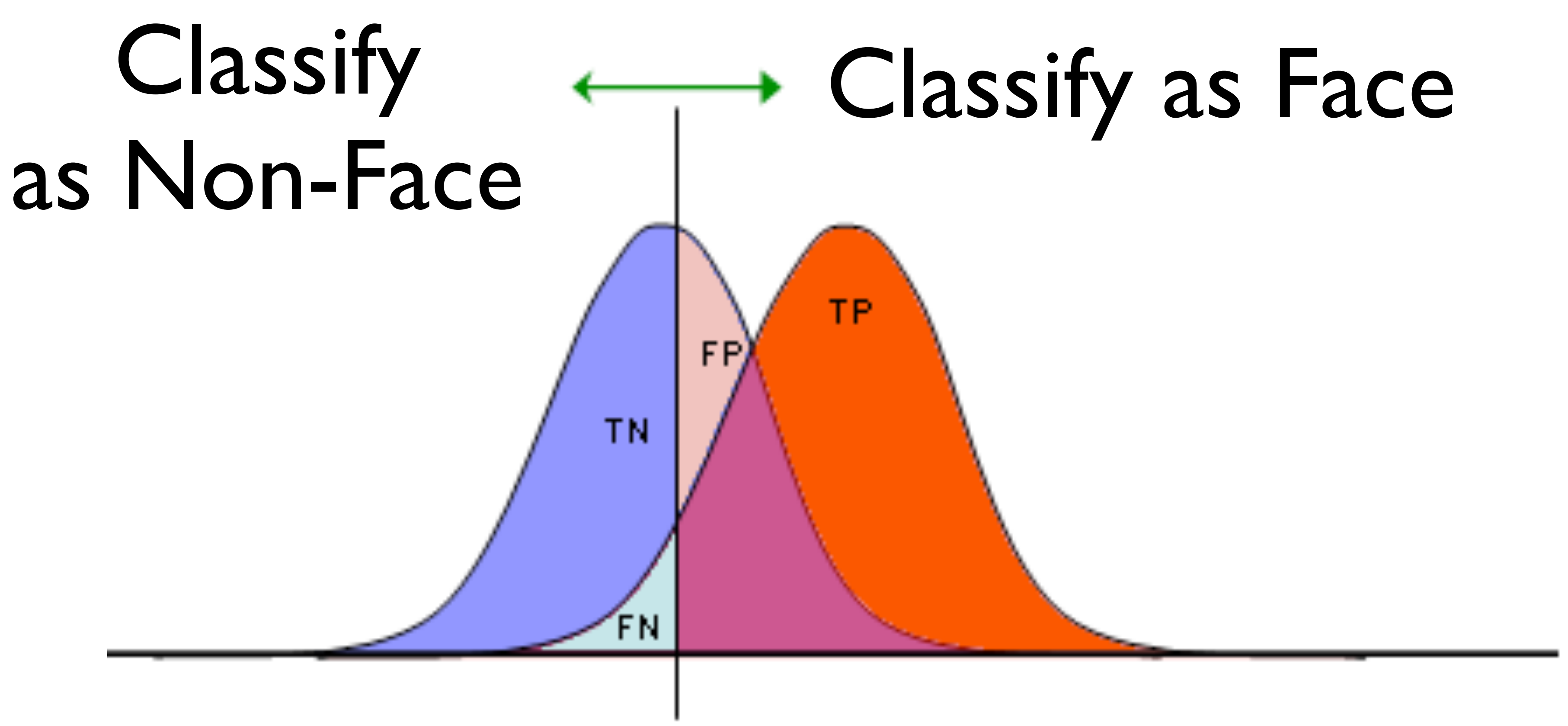
ROC Curves

Note that we can easily get 100% true positives (if we are prepared to get 100% false positives as well!)

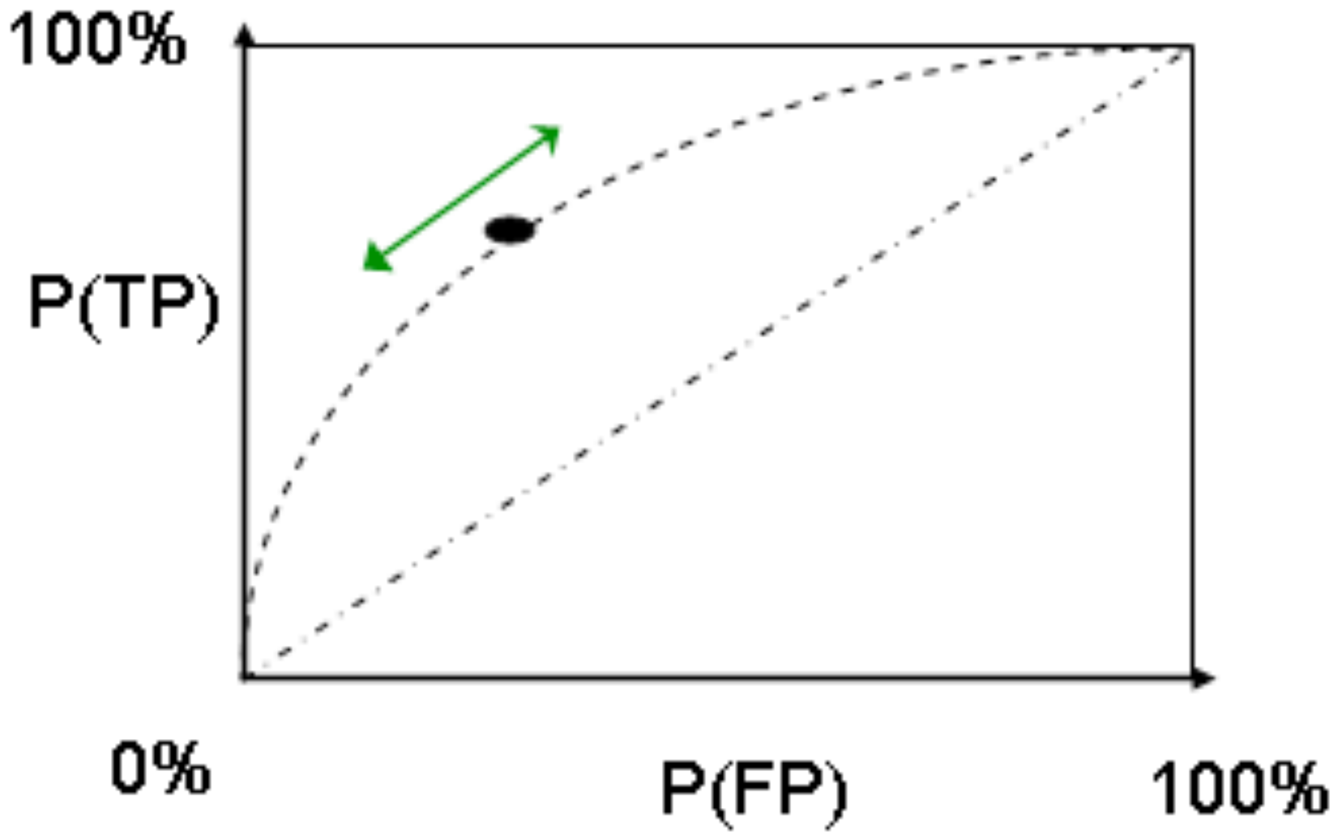
It is a tradeoff between **true positive rate (TP)** and **false positive rate (FP)**

We can plot a curve of all TP rates vs FP rates by varying the classifier threshold

This is a **Receiver Operating Characteristic (ROC)** curve



red = actual faces, blue = actual non-faces



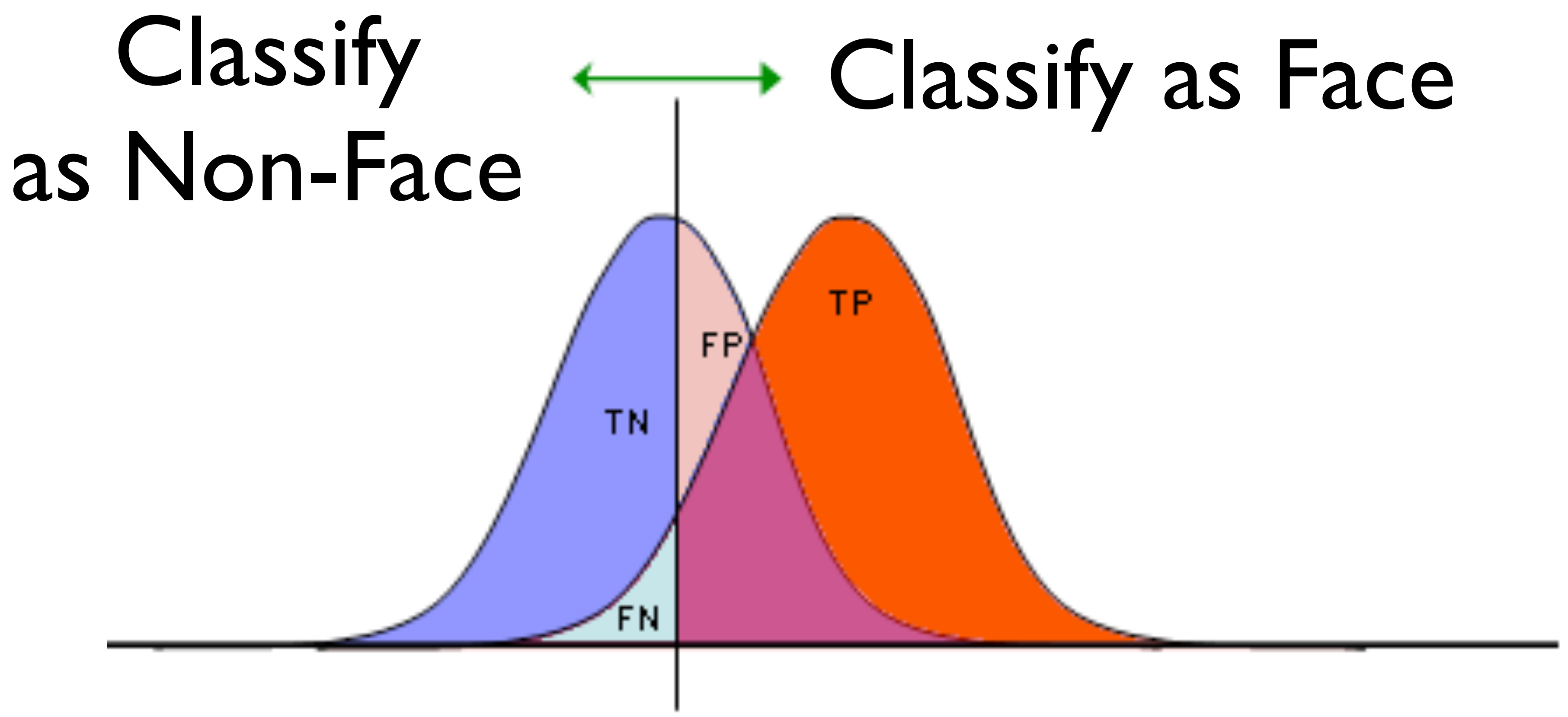
ROC Curves

Note that we can easily get 100% true positives (if we are prepared to get 100% false positives as well!)

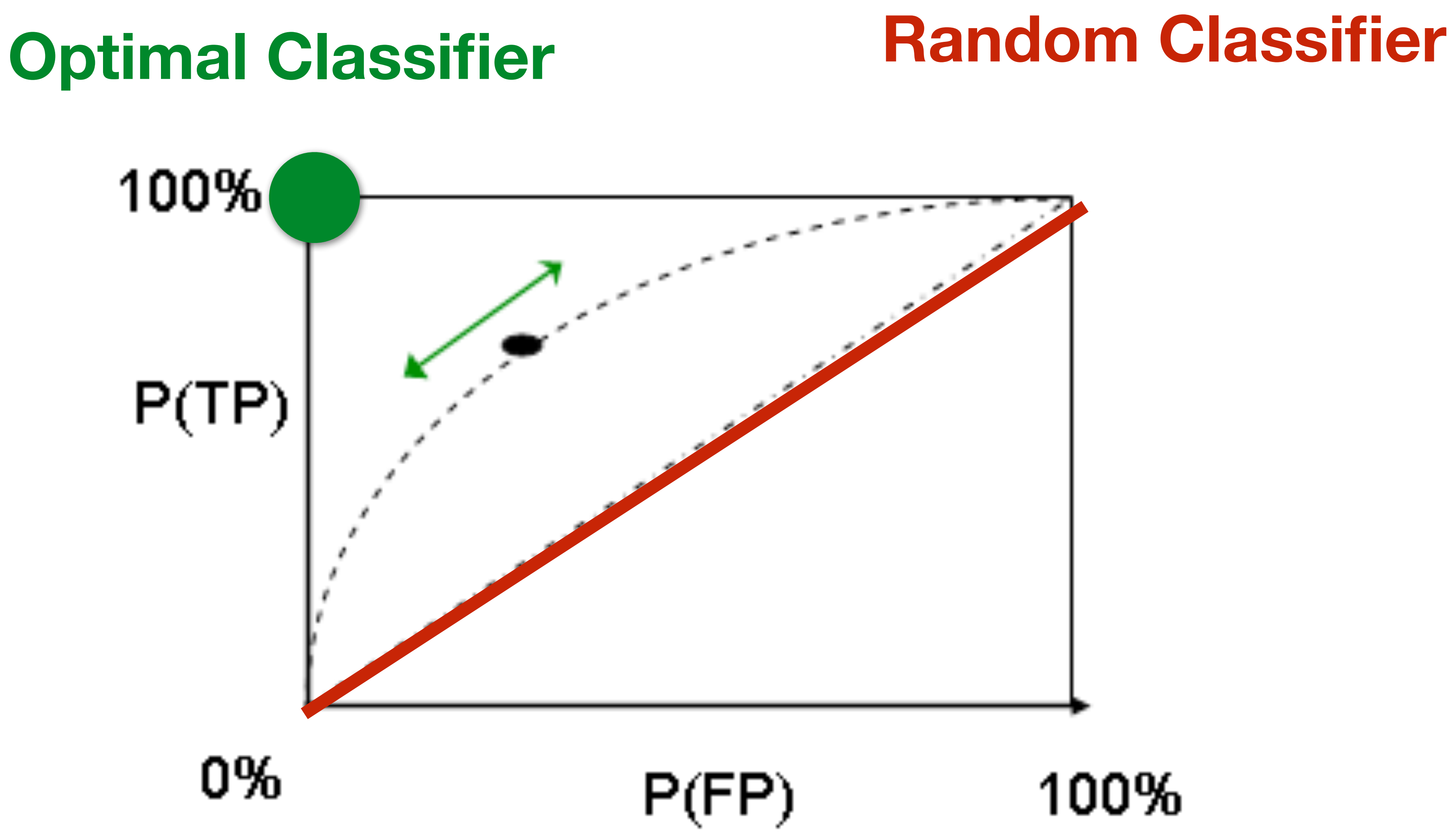
It is a tradeoff between **true positive rate (TP)** and **false positive rate (FP)**

We can plot a curve of all TP rates vs FP rates by varying the classifier threshold

This is a **Receiver Operating Characteristic (ROC)** curve



red = actual faces, blue = actual non-faces



Template Matching

Correlation with a **fixed-sized template** only detects faces at **specific scales**

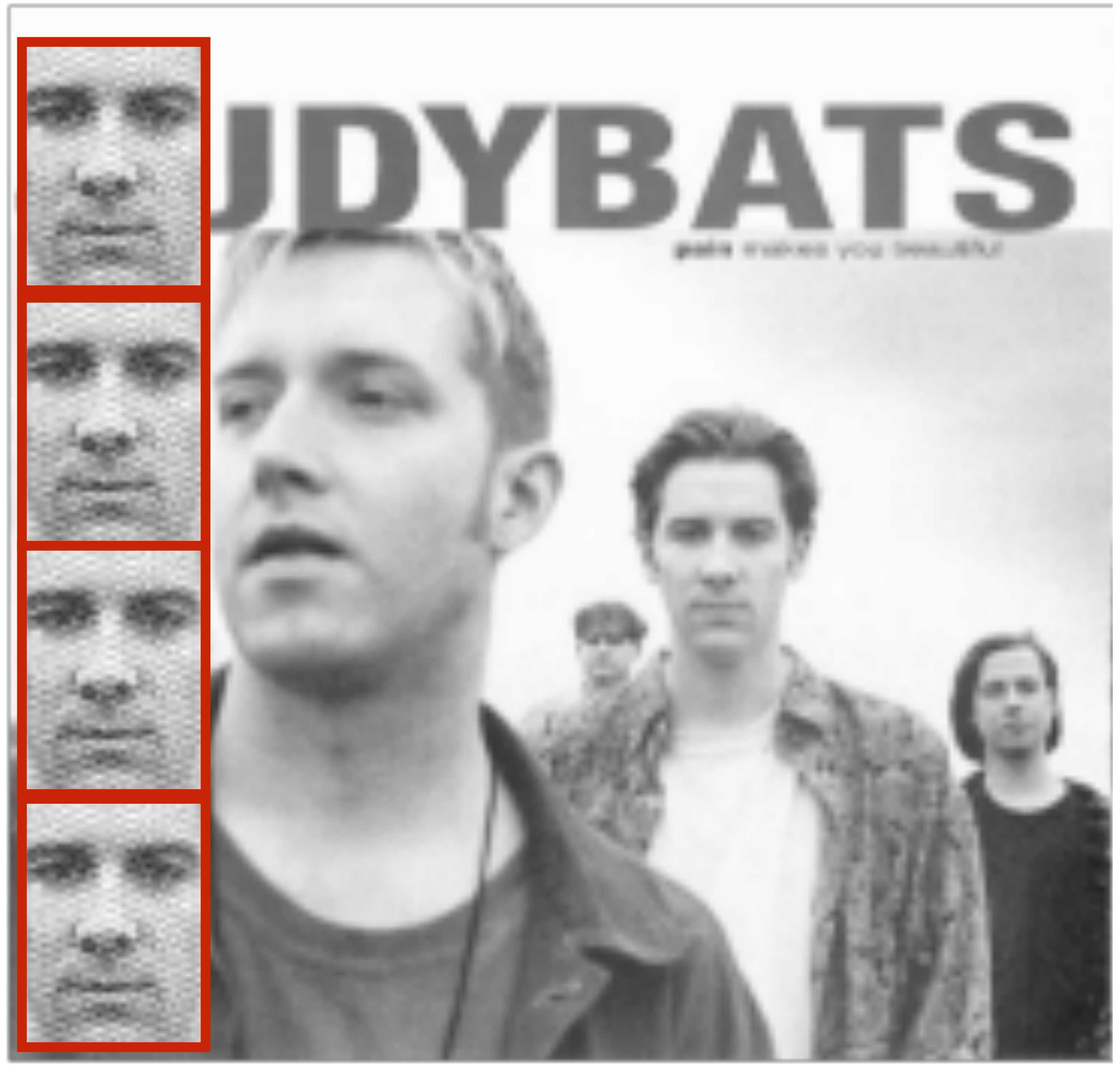
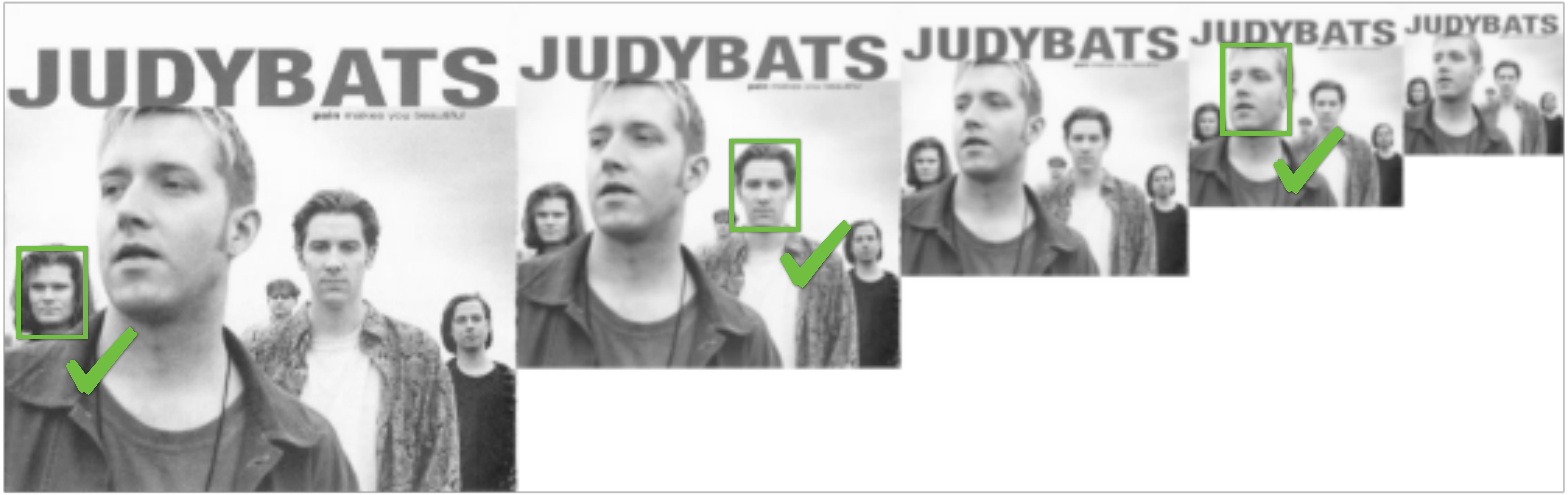


Image **Pyramid**

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

Multi-Scale Template Matching

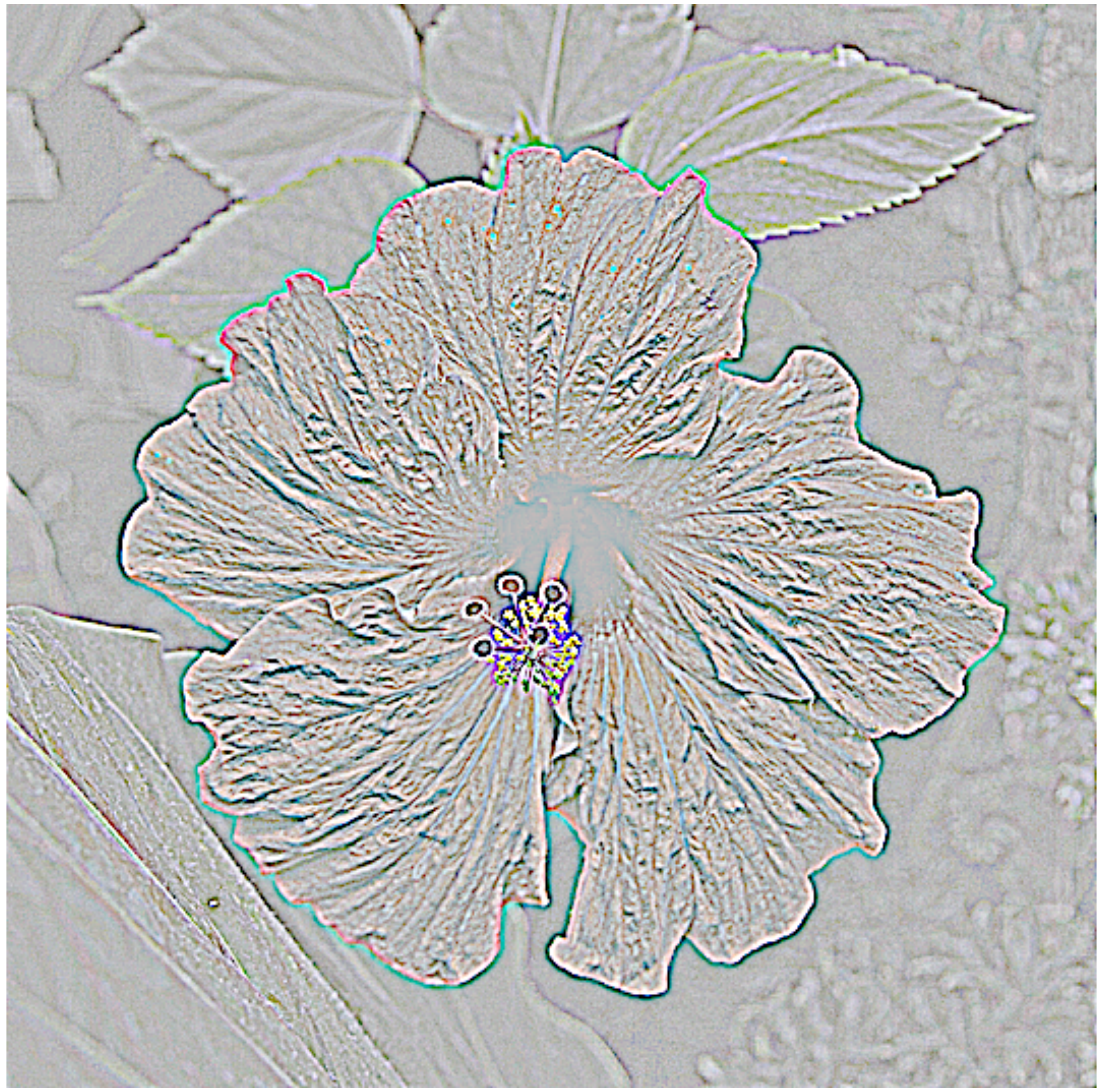
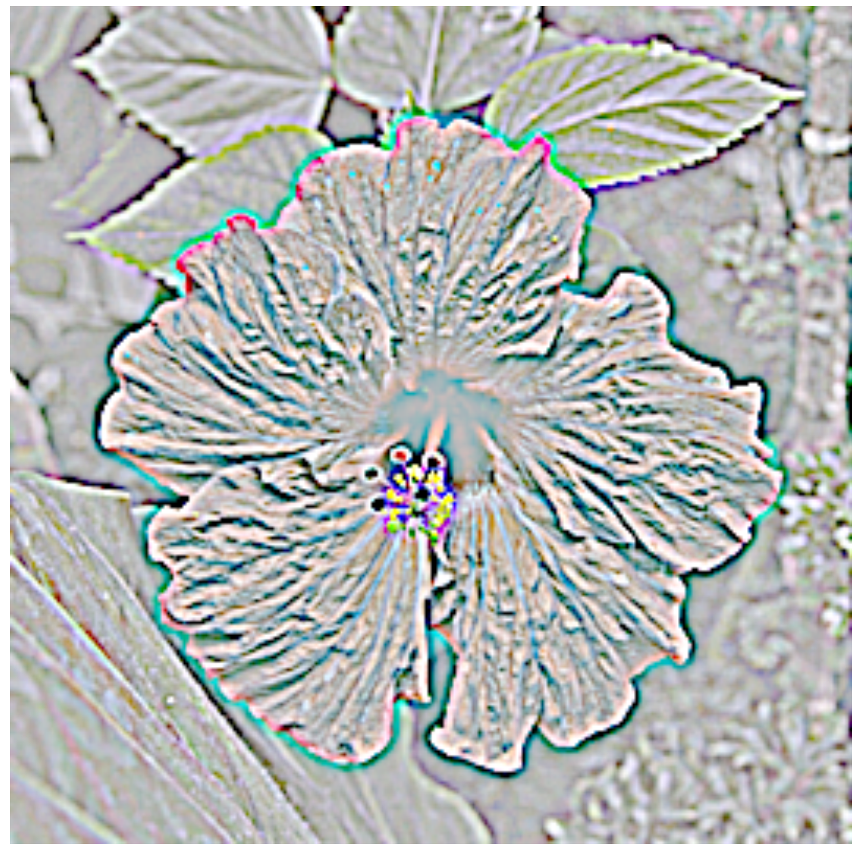
Solution: form a Gaussian Pyramid and convolve with the template at each scale



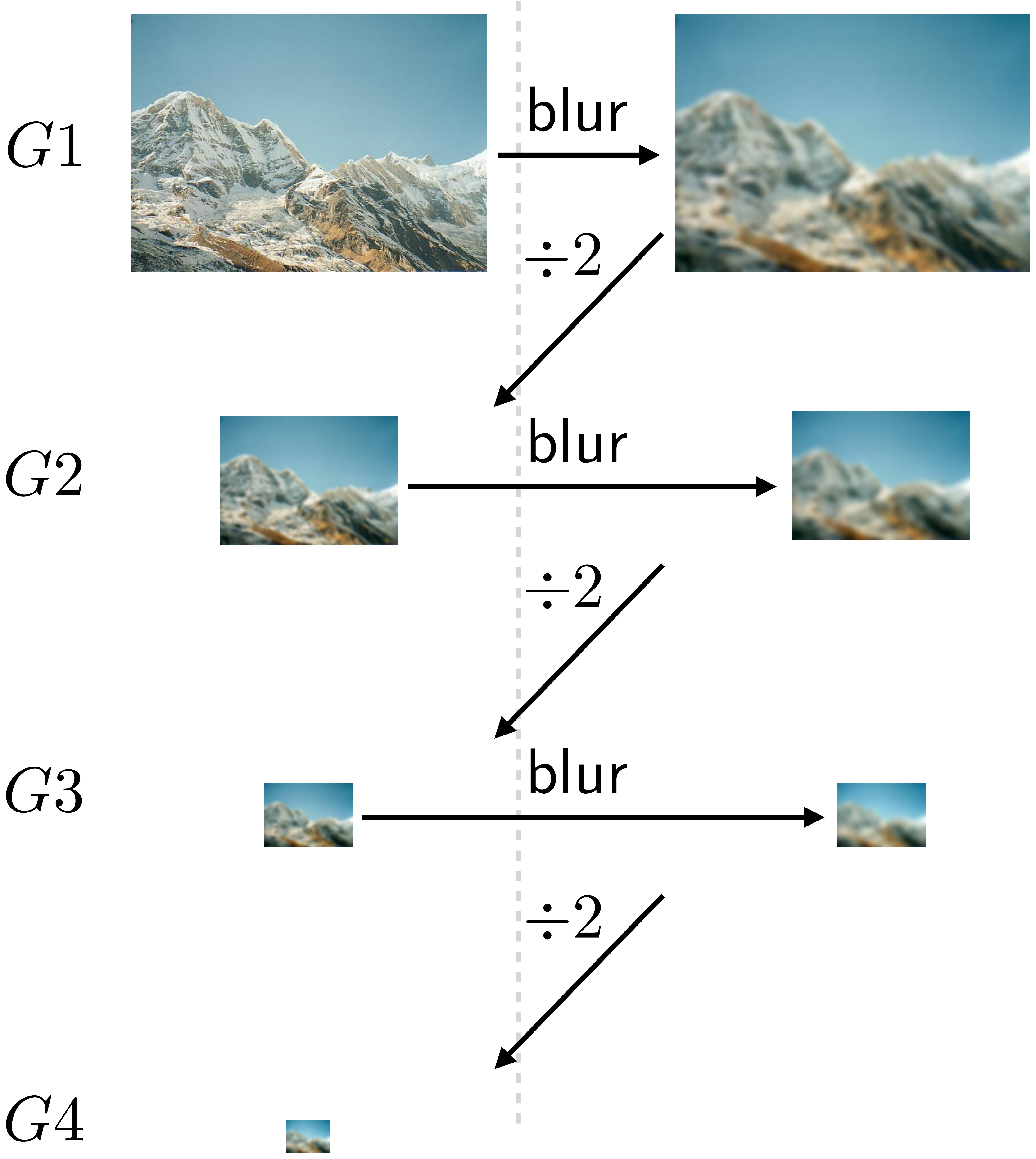
Gaussian vs **Laplacian** Pyramid




Shown in opposite order for space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)



Gaussian Pyramid

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

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

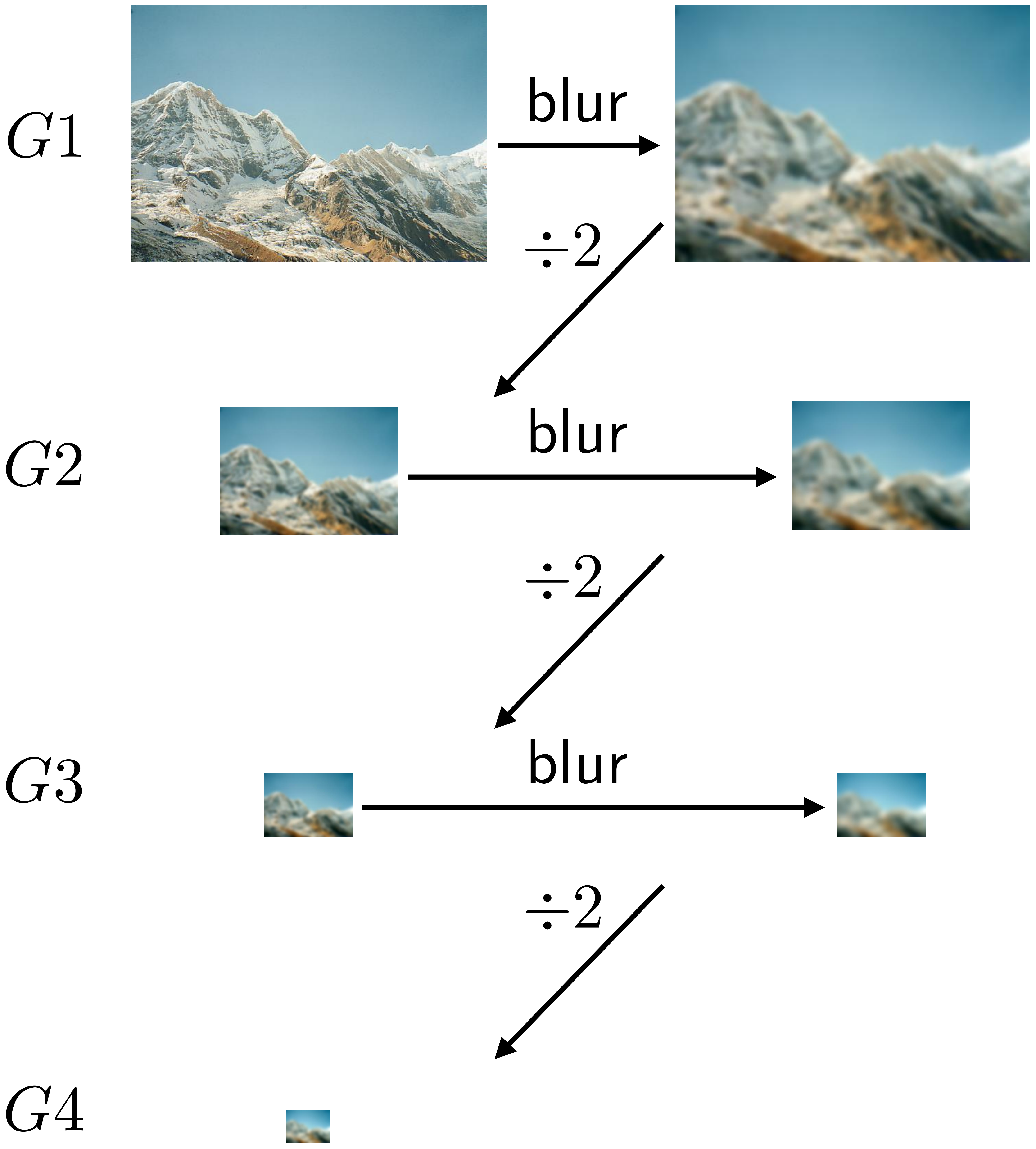
$G1$

$G2$

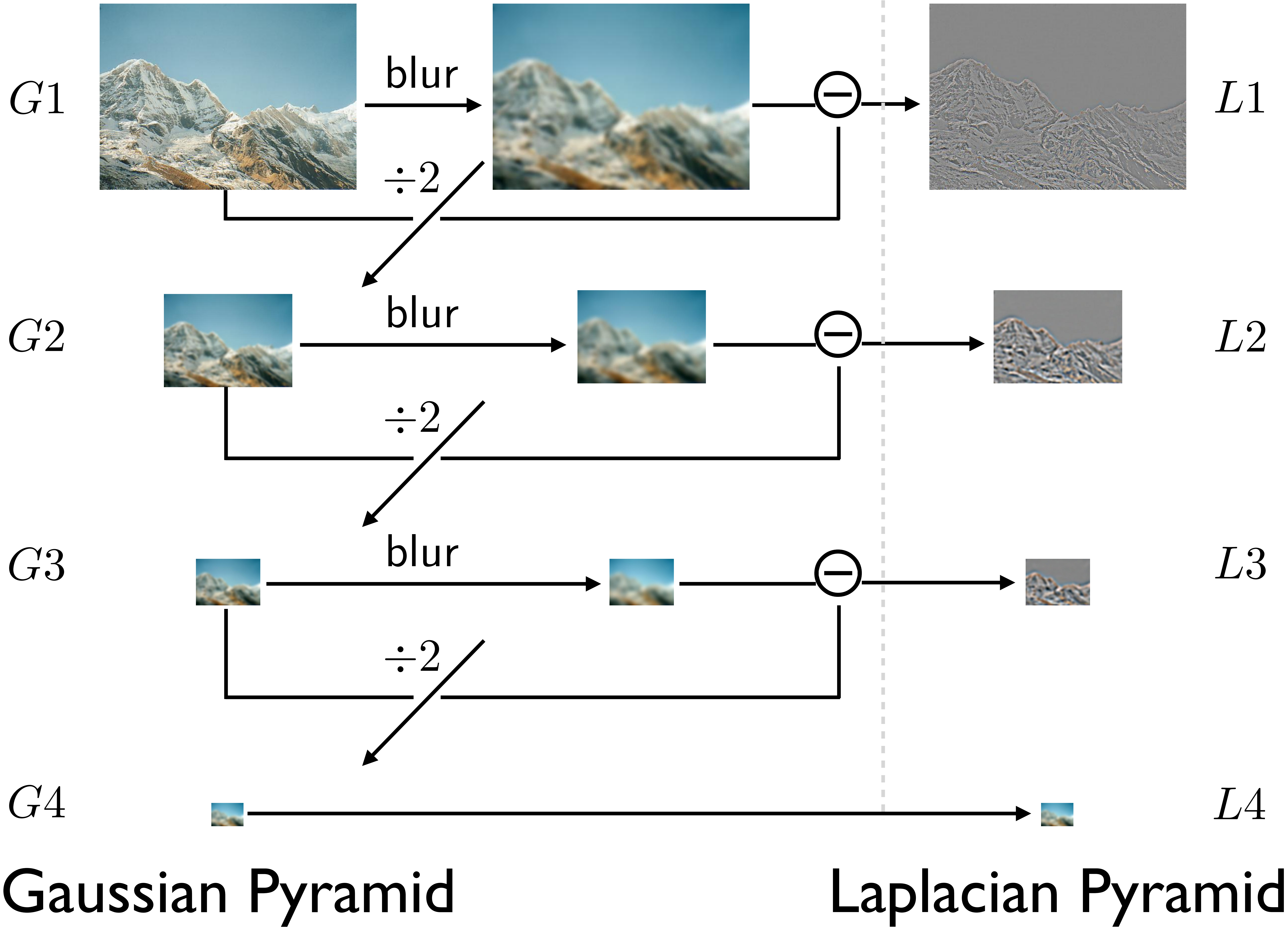
$G3$

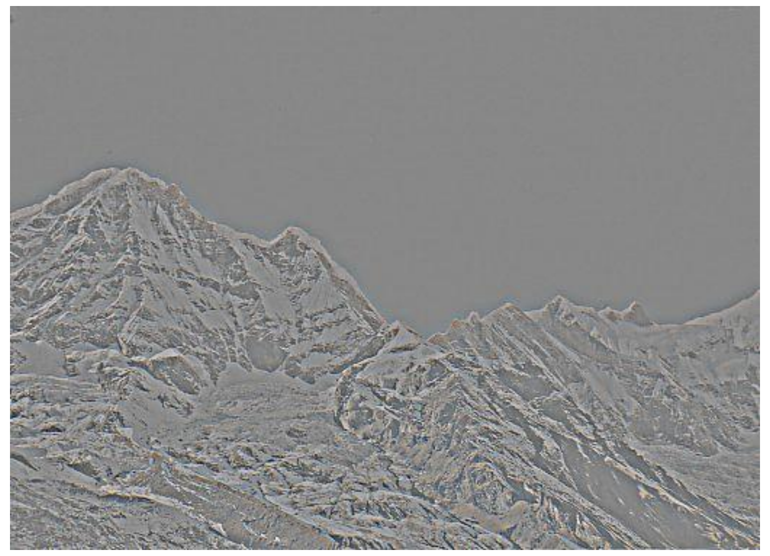
$G4$

Gaussian Pyramid

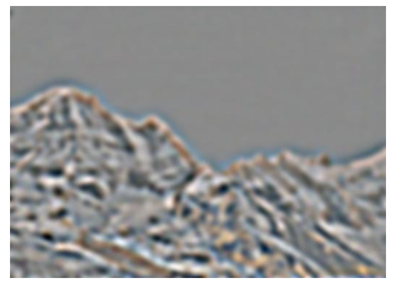


Gaussian Pyramid

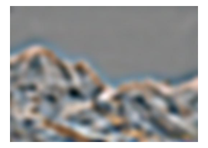

$G1$
blur
$\div 2$
$L1$
$G2$
blur
$\div 2$
$L2$
$G3$
blur
$\div 2$
$L3$
$G4$
$L4$
Gaussian Pyramid
Laplacian Pyramid



$L1$



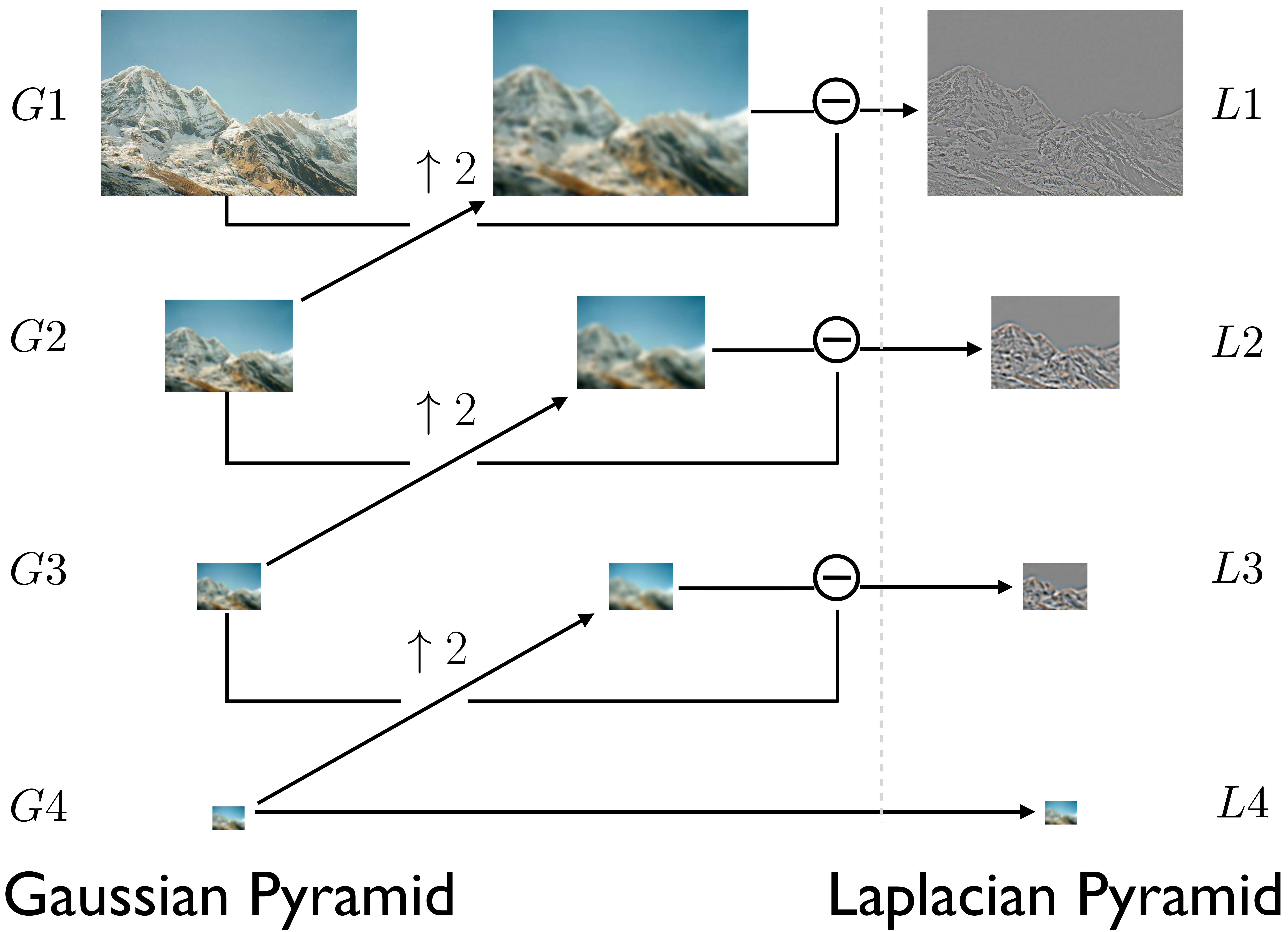
$L2$



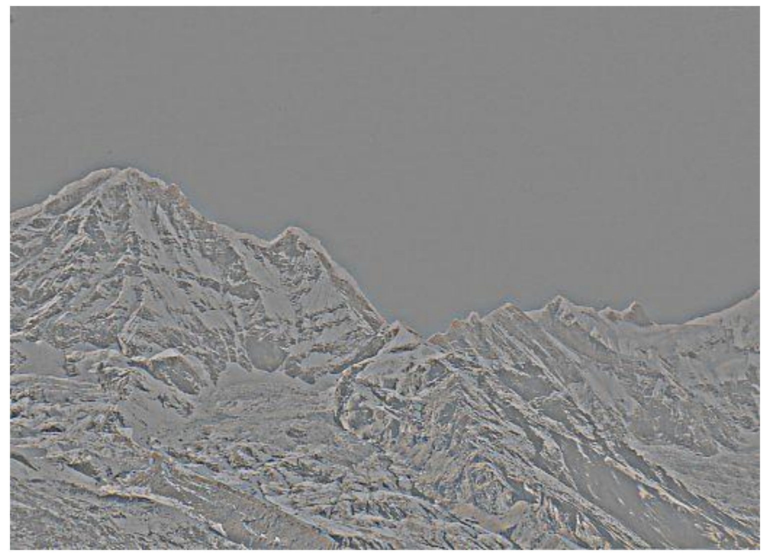
$L3$

$L4$

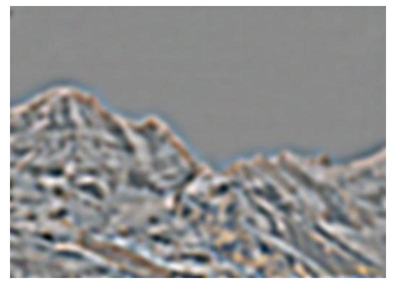
Laplacian Pyramid



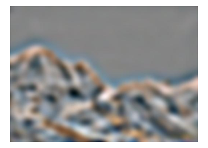
G1
G2
G3
G4
↑ 2
↑ 2
↑ 2
L1
L2
L3
L4
Gaussian Pyramid
Laplacian Pyramid



$L1$



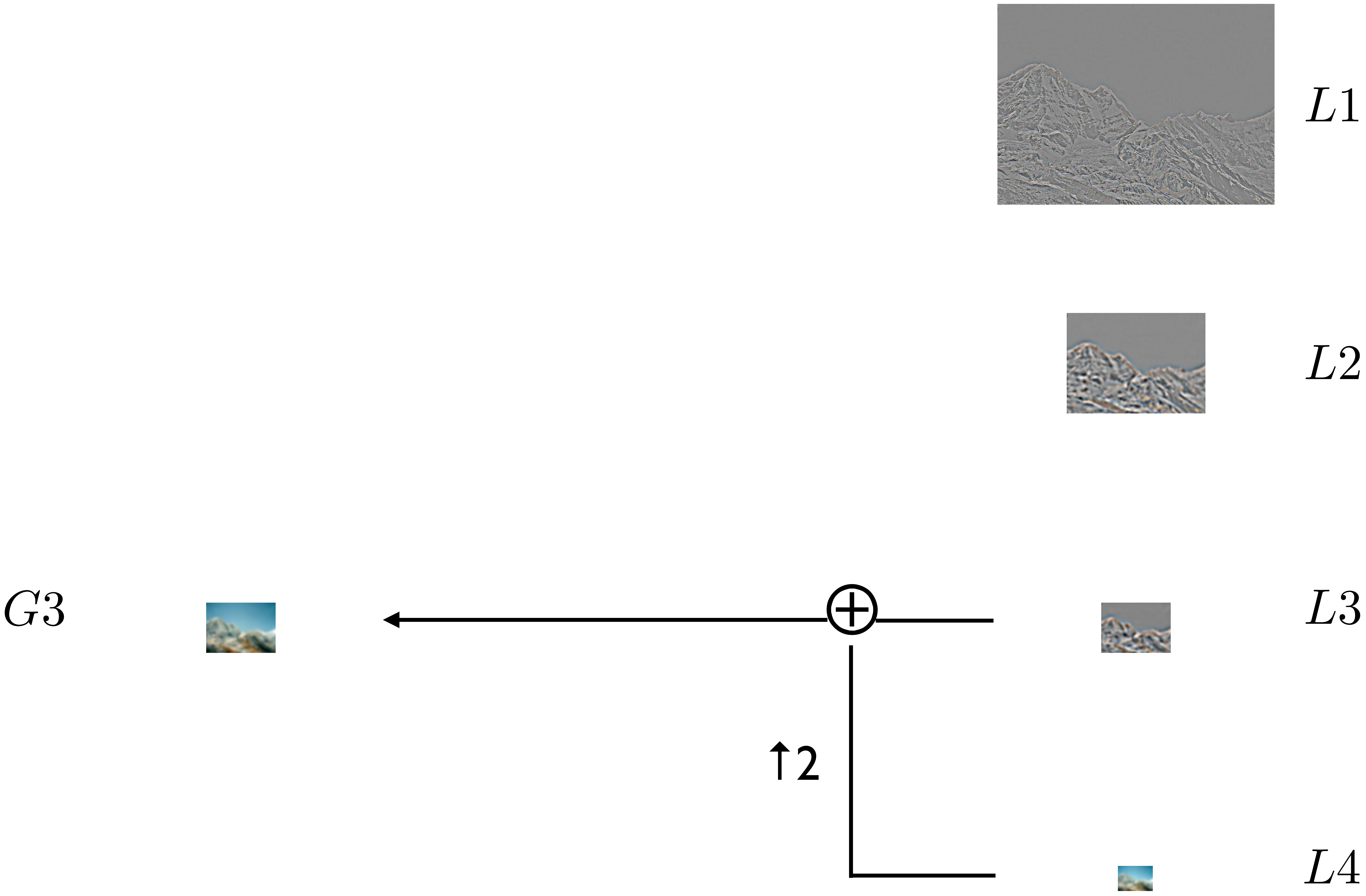
$L2$



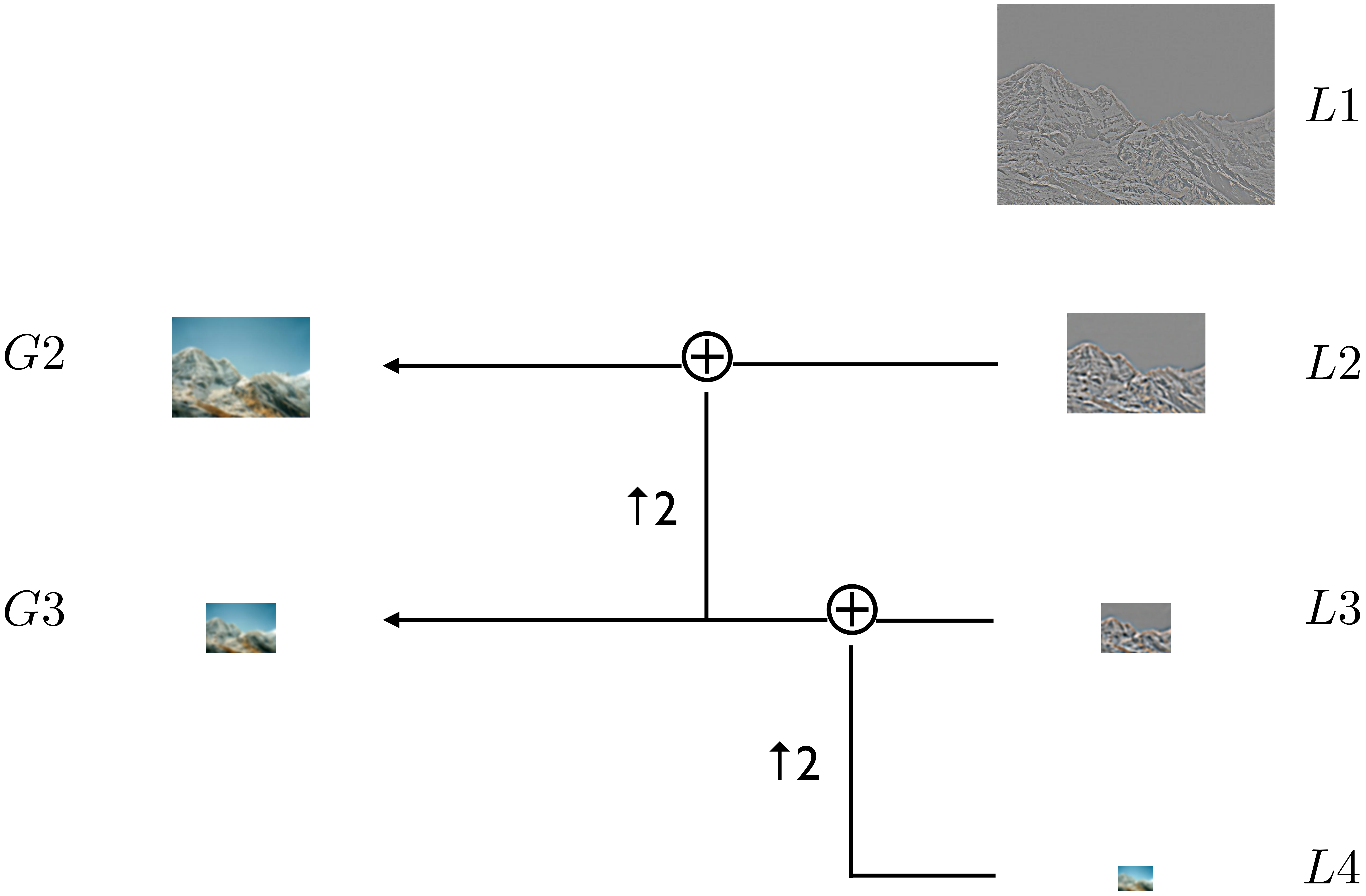
$L3$

$L4$

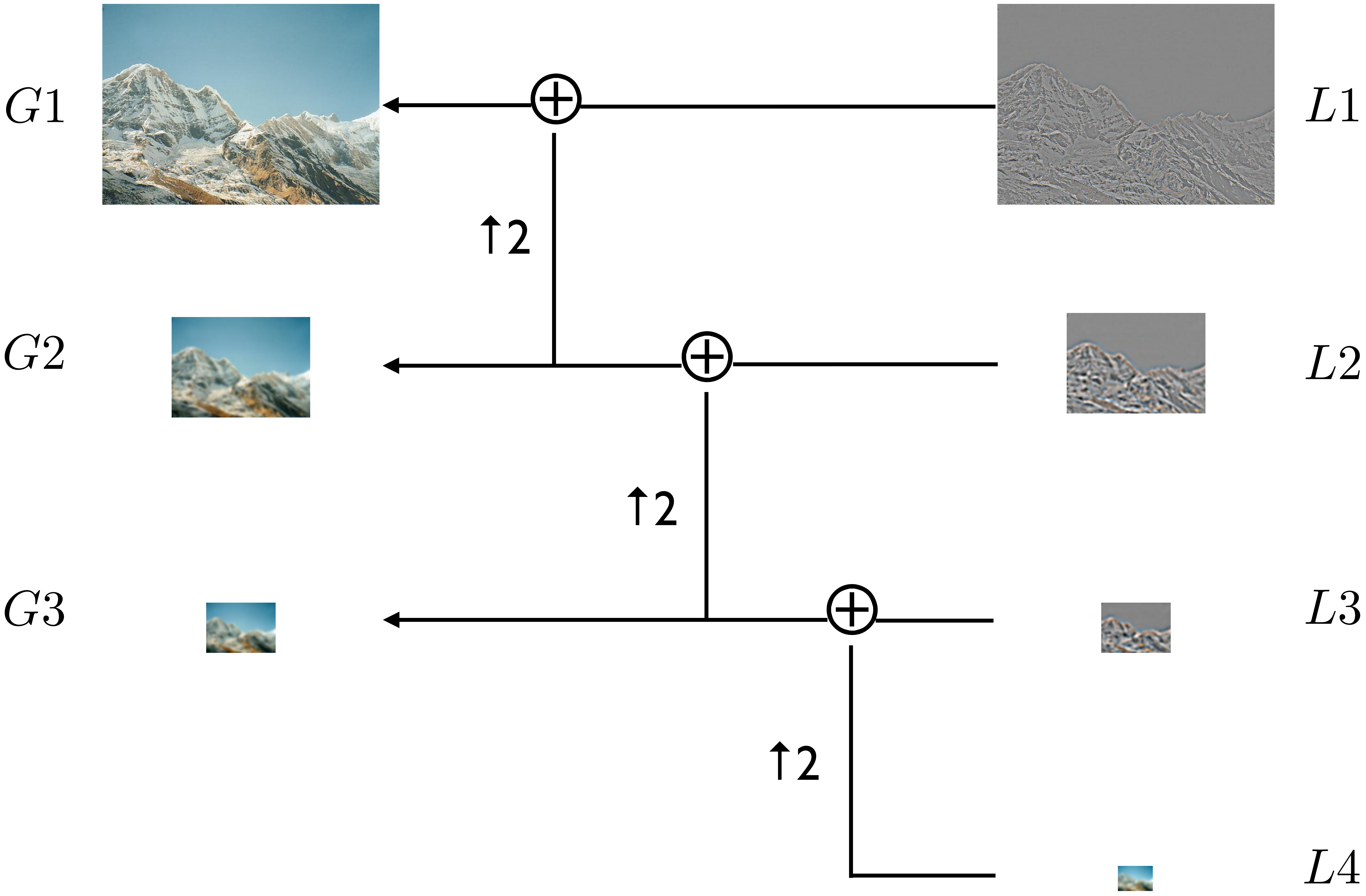
Laplacian Pyramid



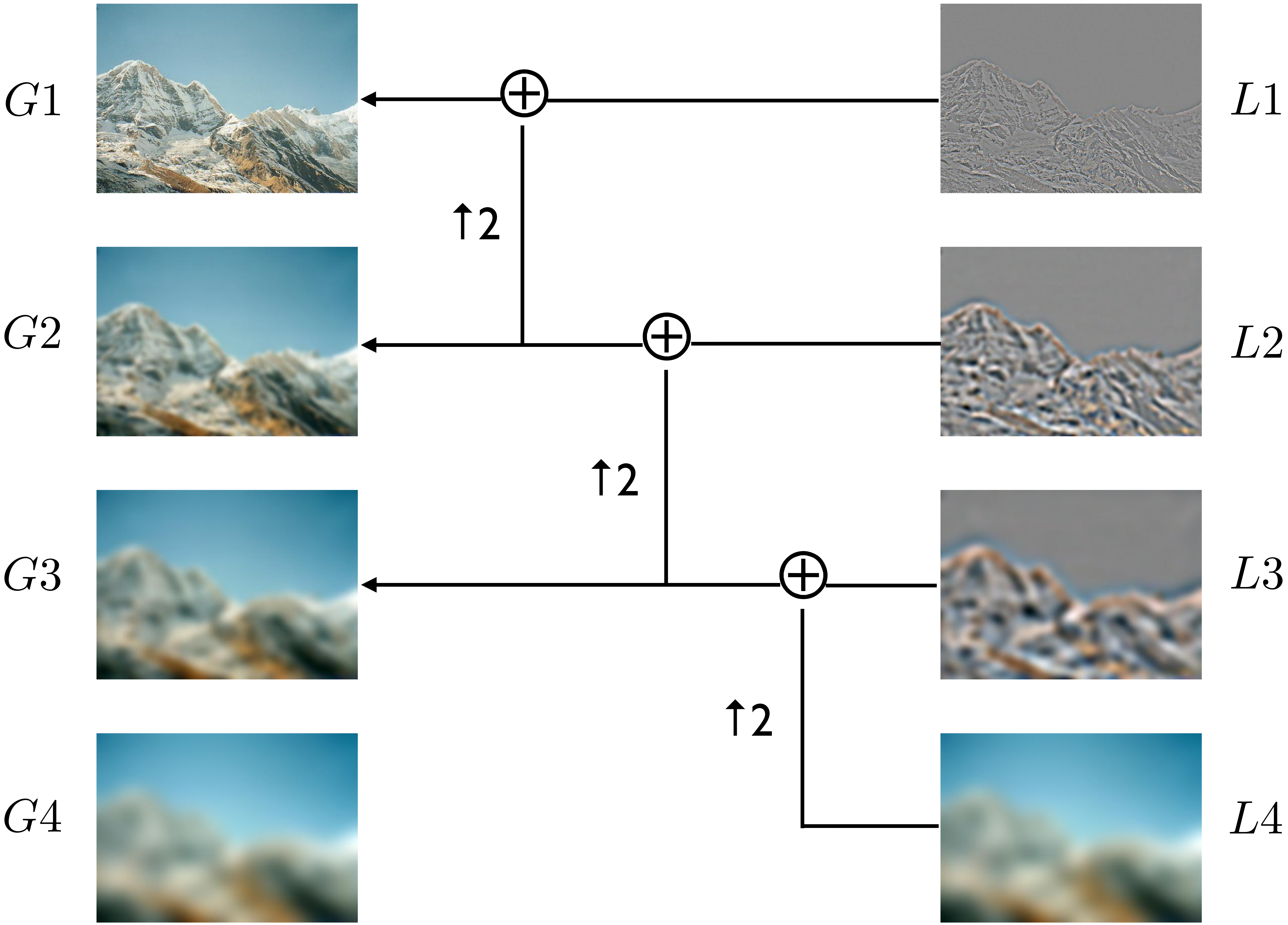
Laplacian Pyramid



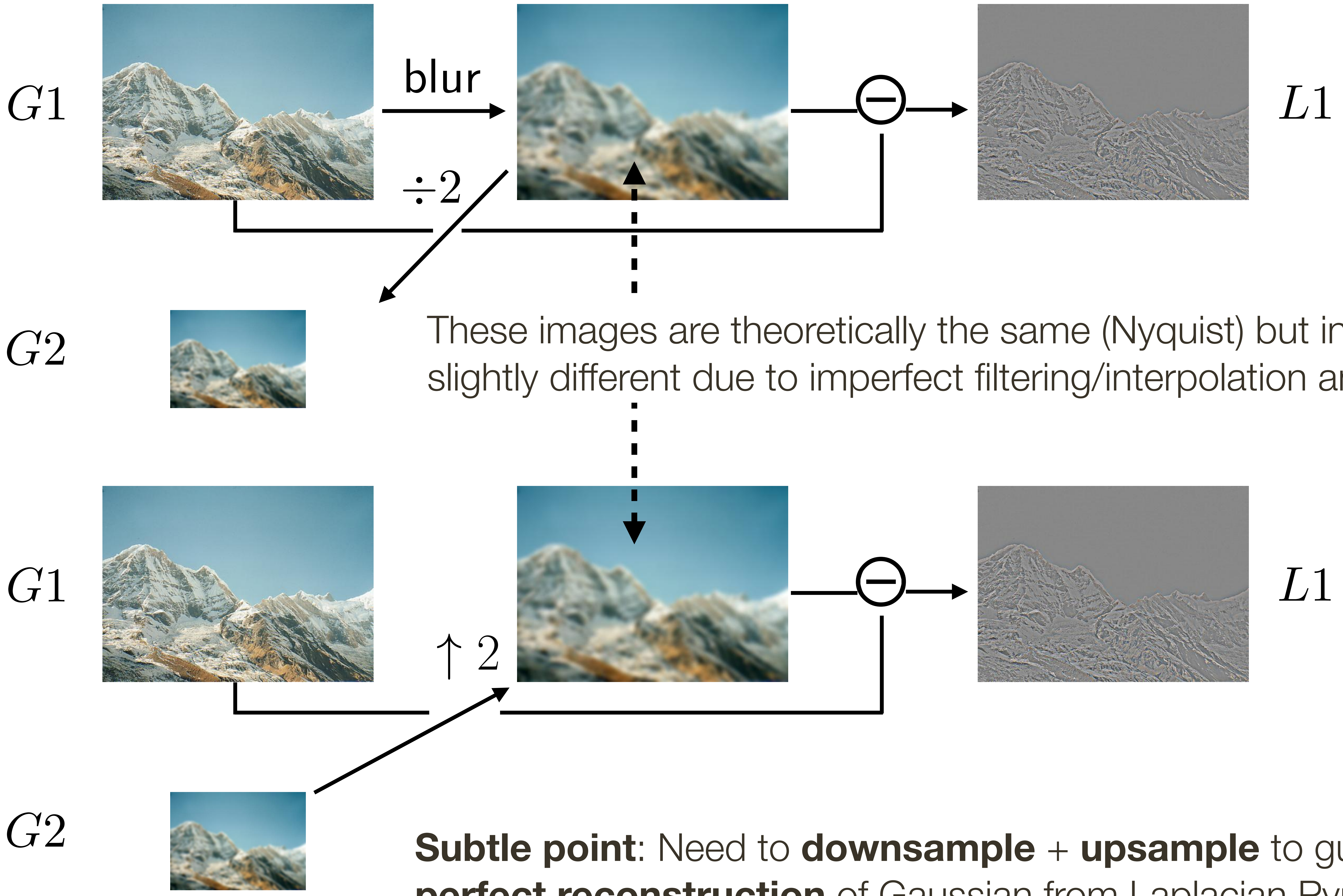
Laplacian Pyramid



Laplacian Pyramid



$G1$
$G2$
$G3$
$G4$
$L1$
$L2$
$L3$
$L4$
↑2
↑2
↑2



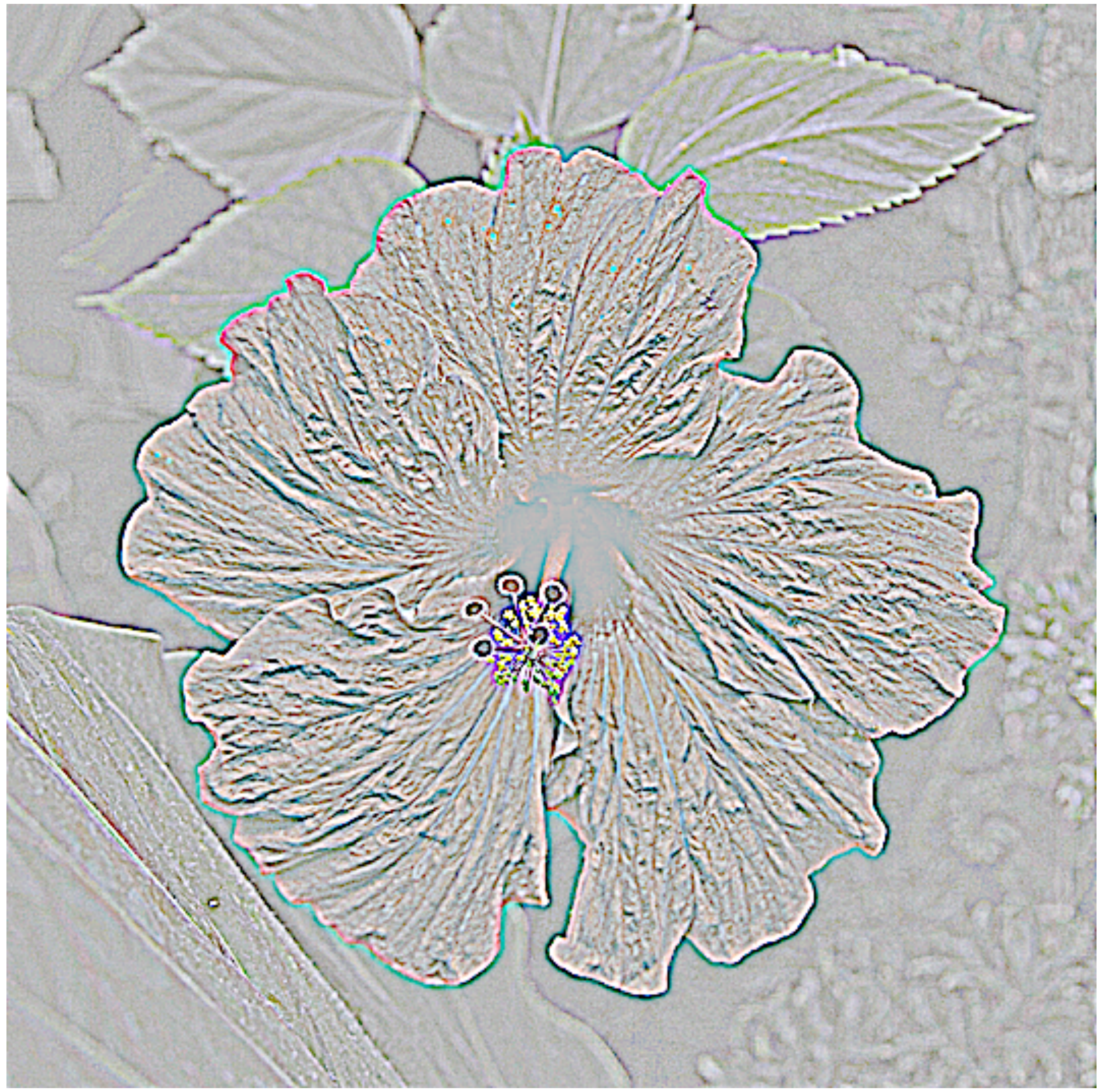
These images are theoretically the same (Nyquist) but in practice slightly different due to imperfect filtering/interpolation and edge effects

Subtle point: Need to **downsample** + **upsample** to guarantee **perfect reconstruction** of Gaussian from Laplacian Pyramid

Gaussian vs **Laplacian** Pyramid

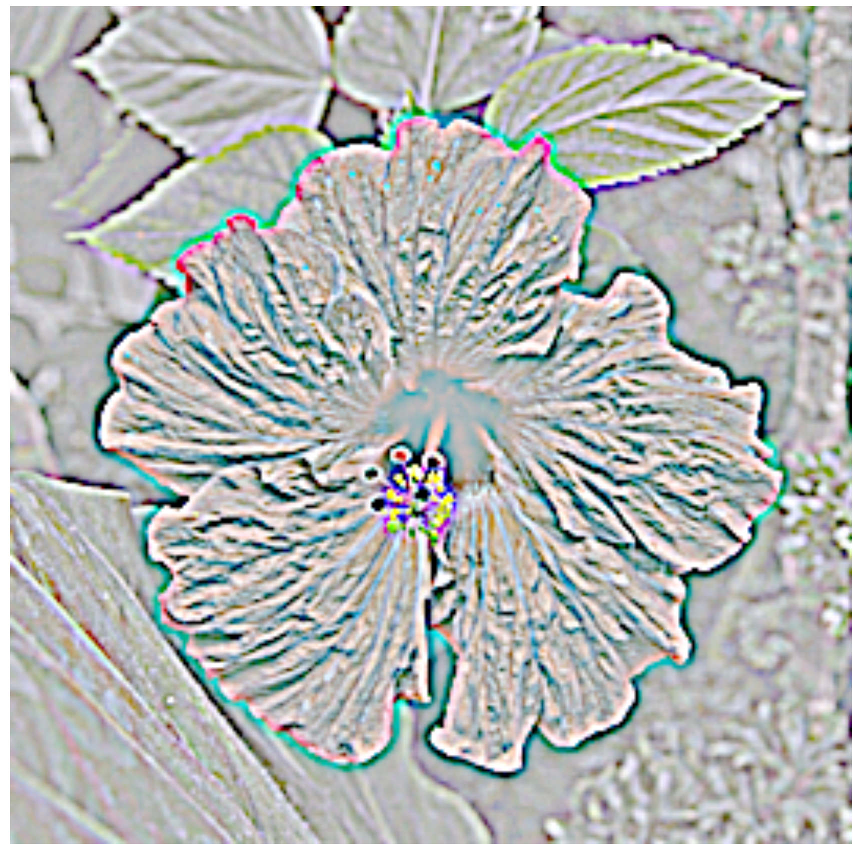



Shown in opposite order for space



Which one takes more space to store?

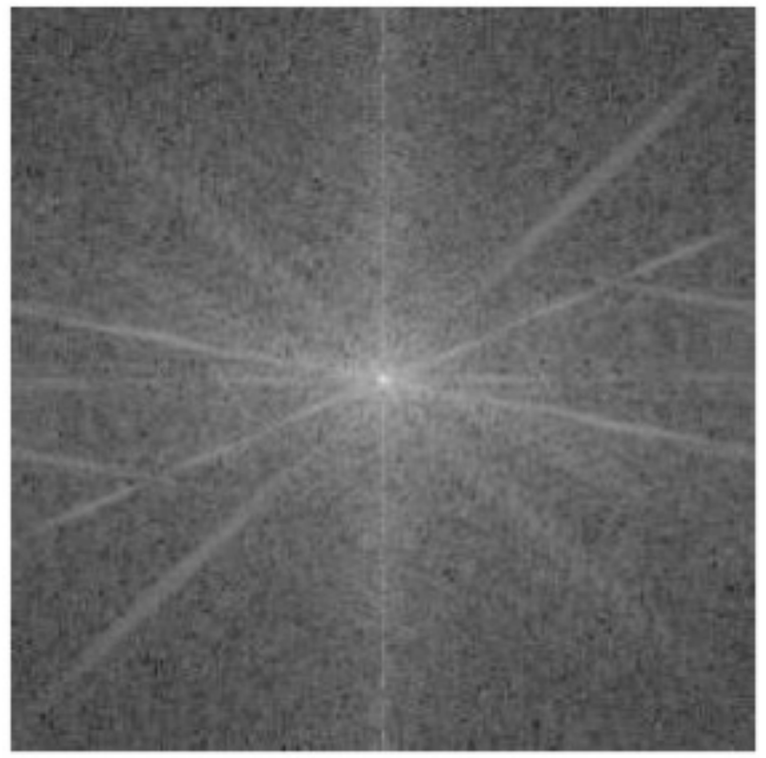




Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Laplacian is a Bandpass Filter

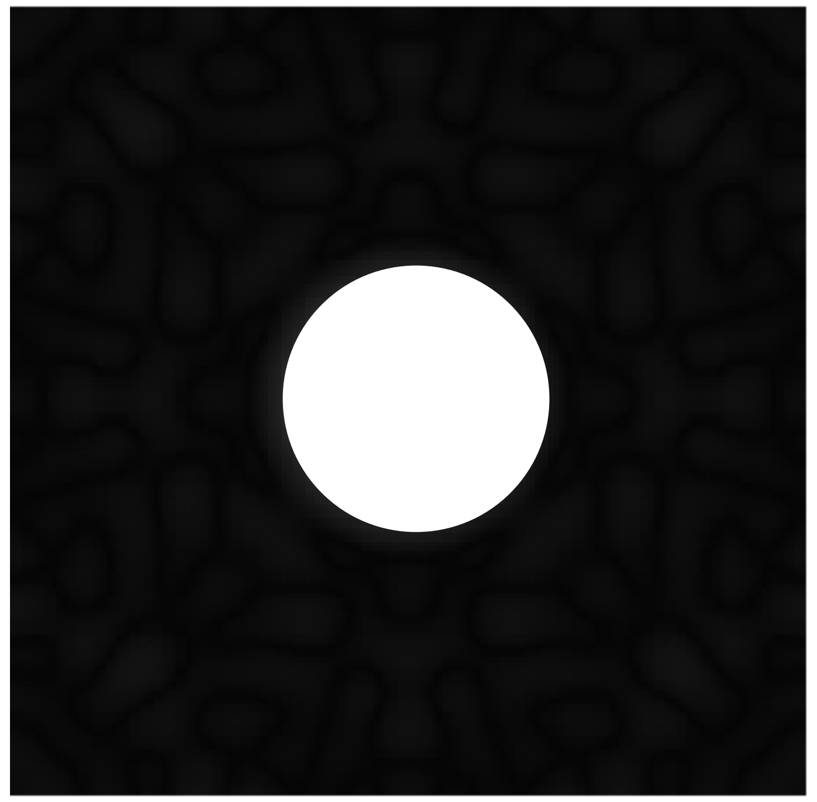
image



FFT (Mag)

complex
element-wise
multiplication

lower sigma

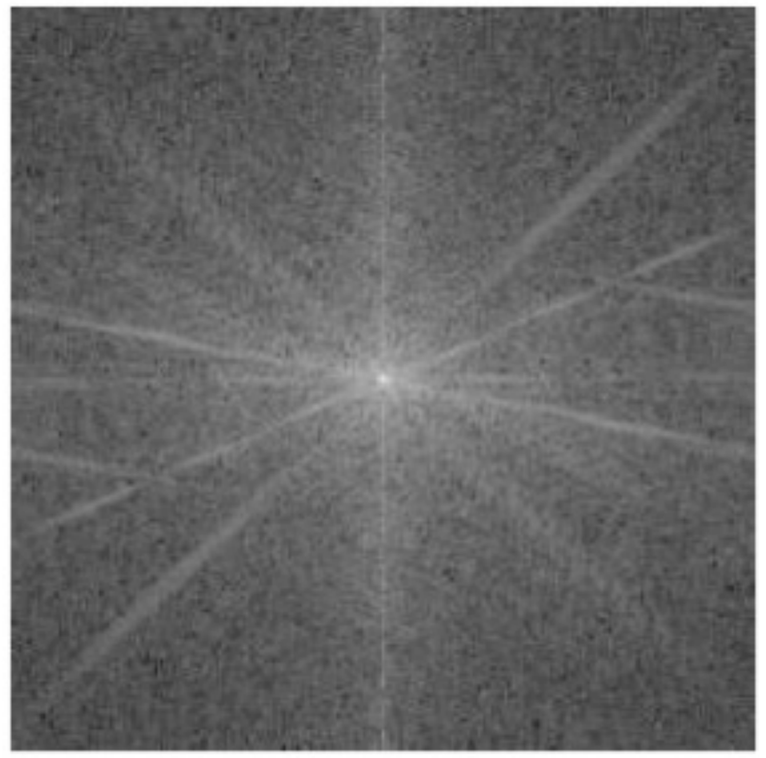


Low pass

filtered **image**

-

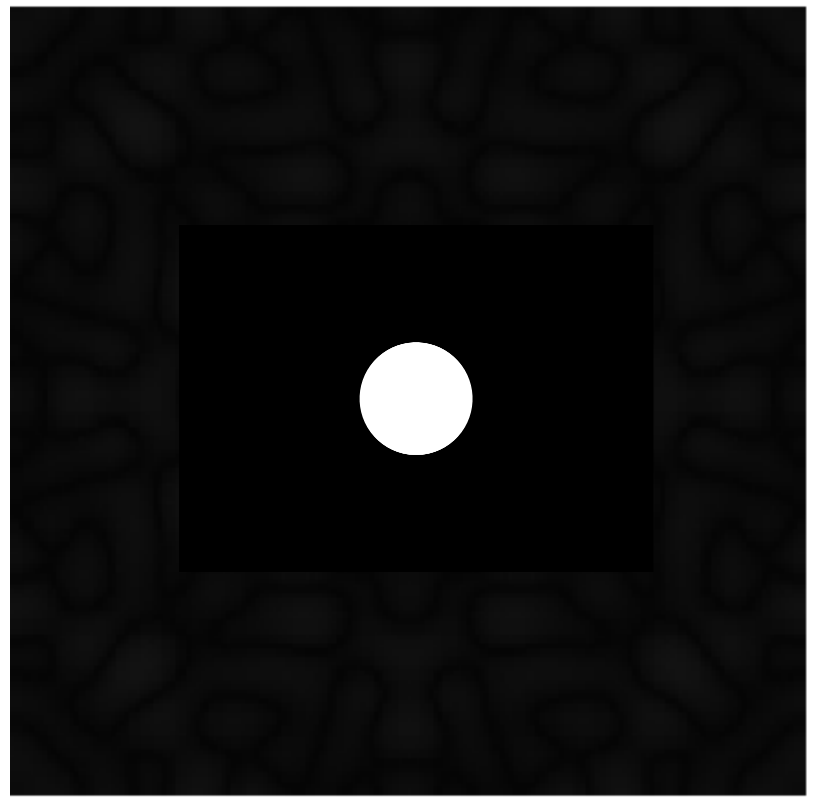
-



FFT (Mag)

complex
element-wise
multiplication

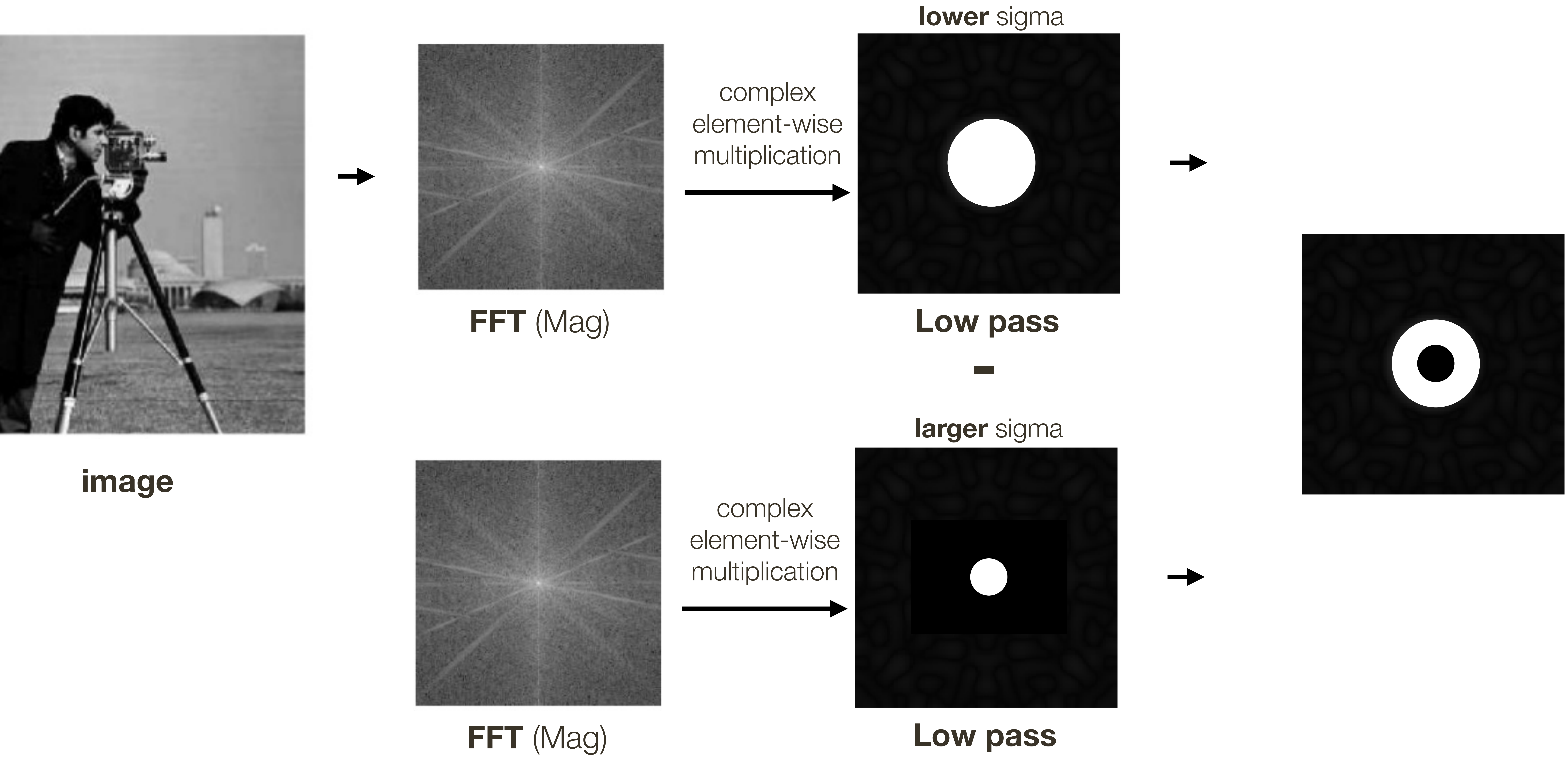
larger sigma



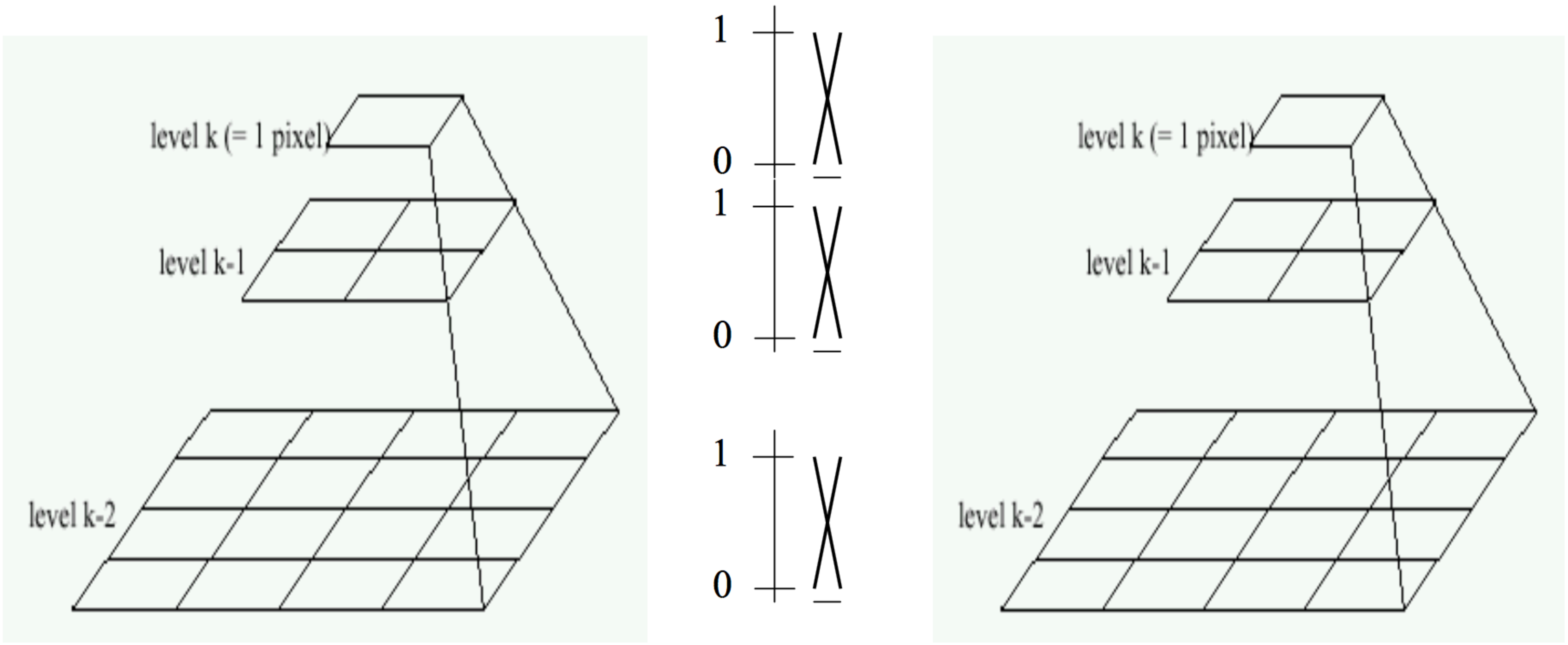
Low pass

filtered **image**

Laplacian is a Bandpass Filter



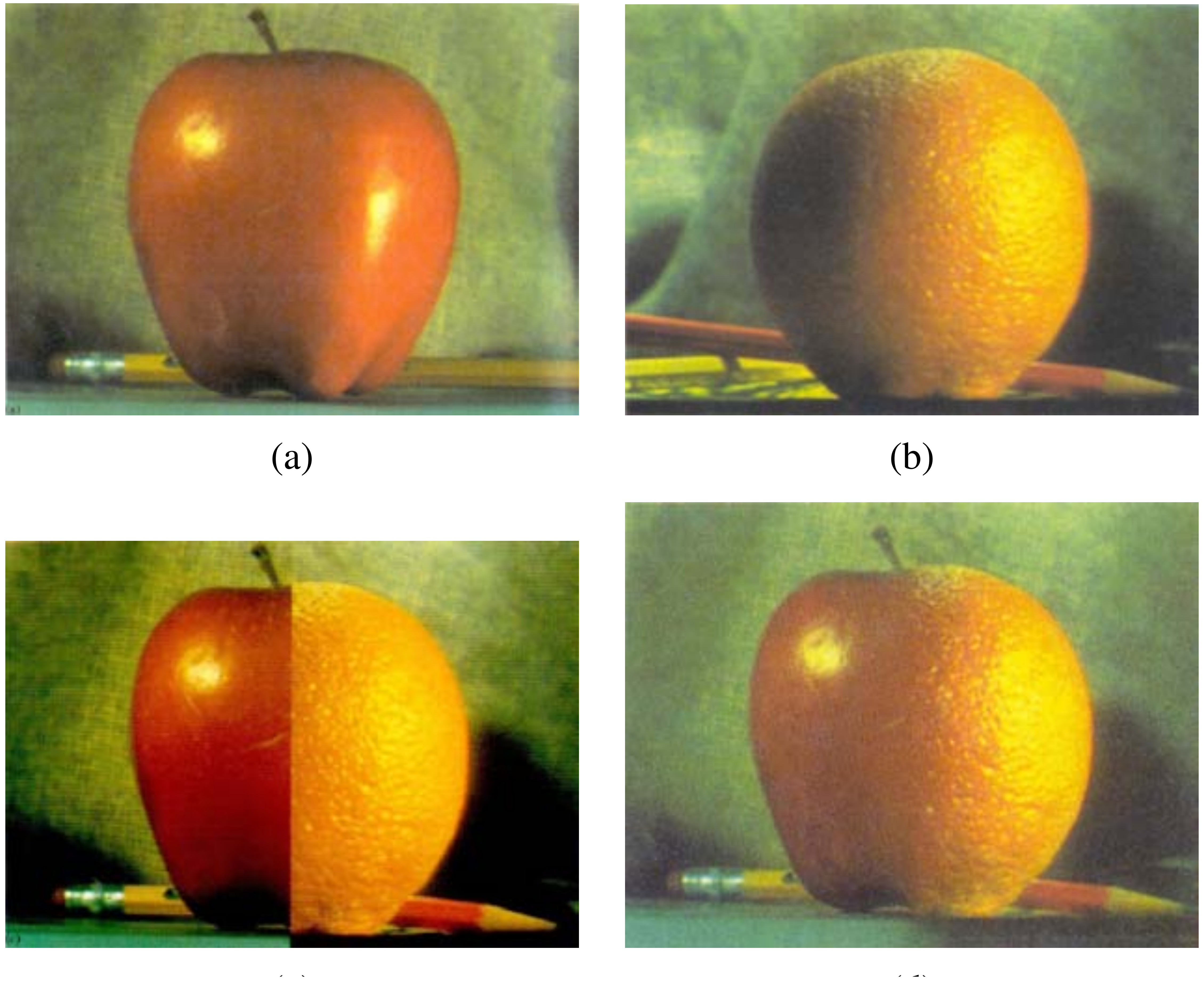
Application: Image Blending



Left pyramid blend Right pyramid

Burt and Adelson, “A multiresolution spline with application to image mosaics,”ACM Transactions on Graphics, 1983, Vol.2, pp.217-236.

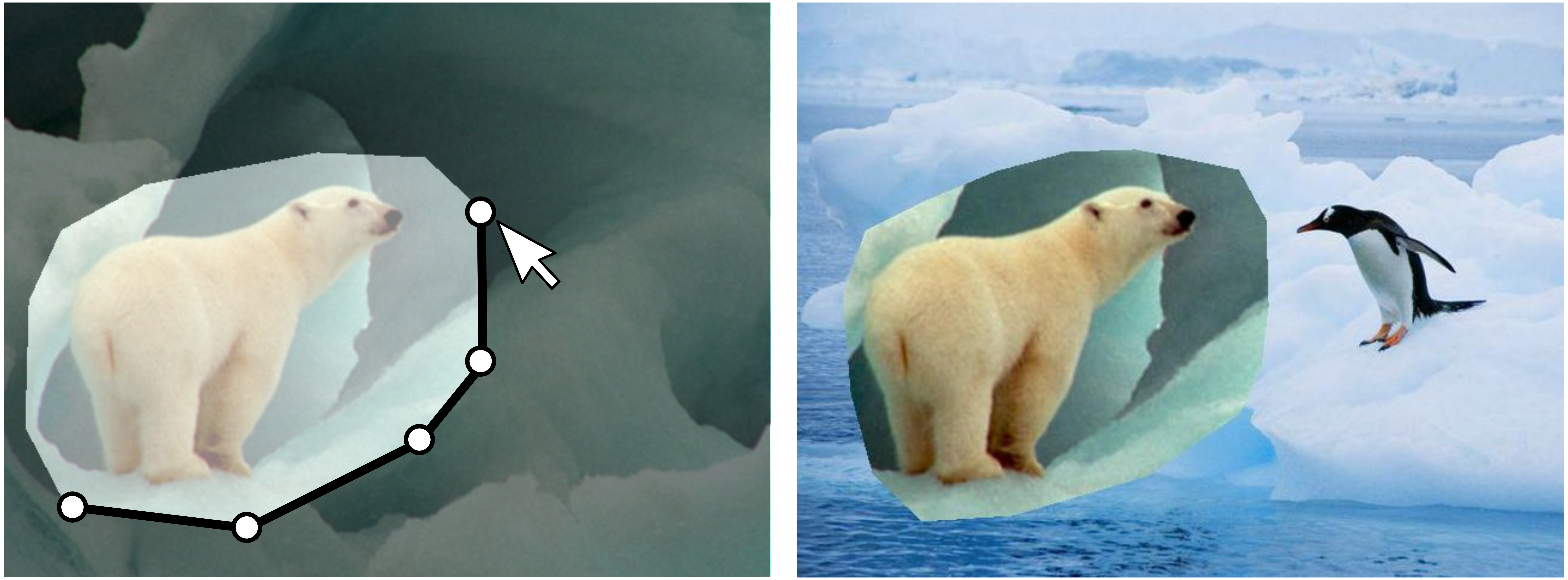
Application: Image Pyramid Blending

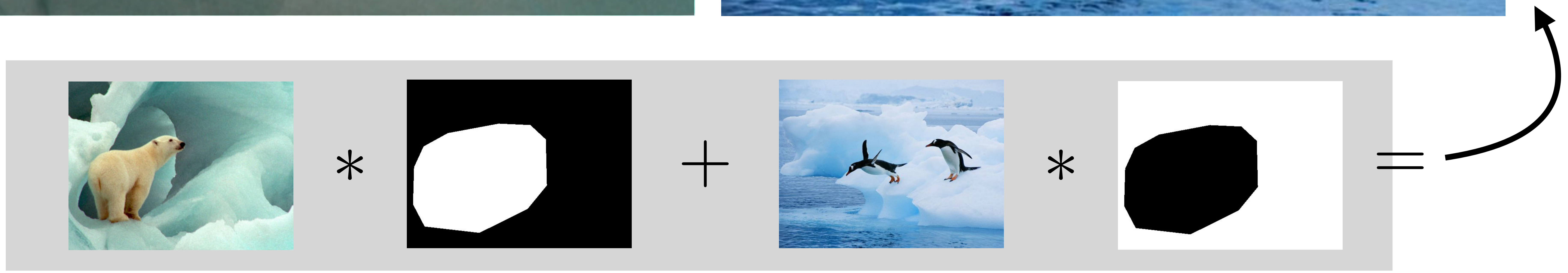


Burt and Adelson, “A multiresolution spline with application to image mosaics,” ACM Transactions on Graphics, 1983, Vol.2, pp.217-236.

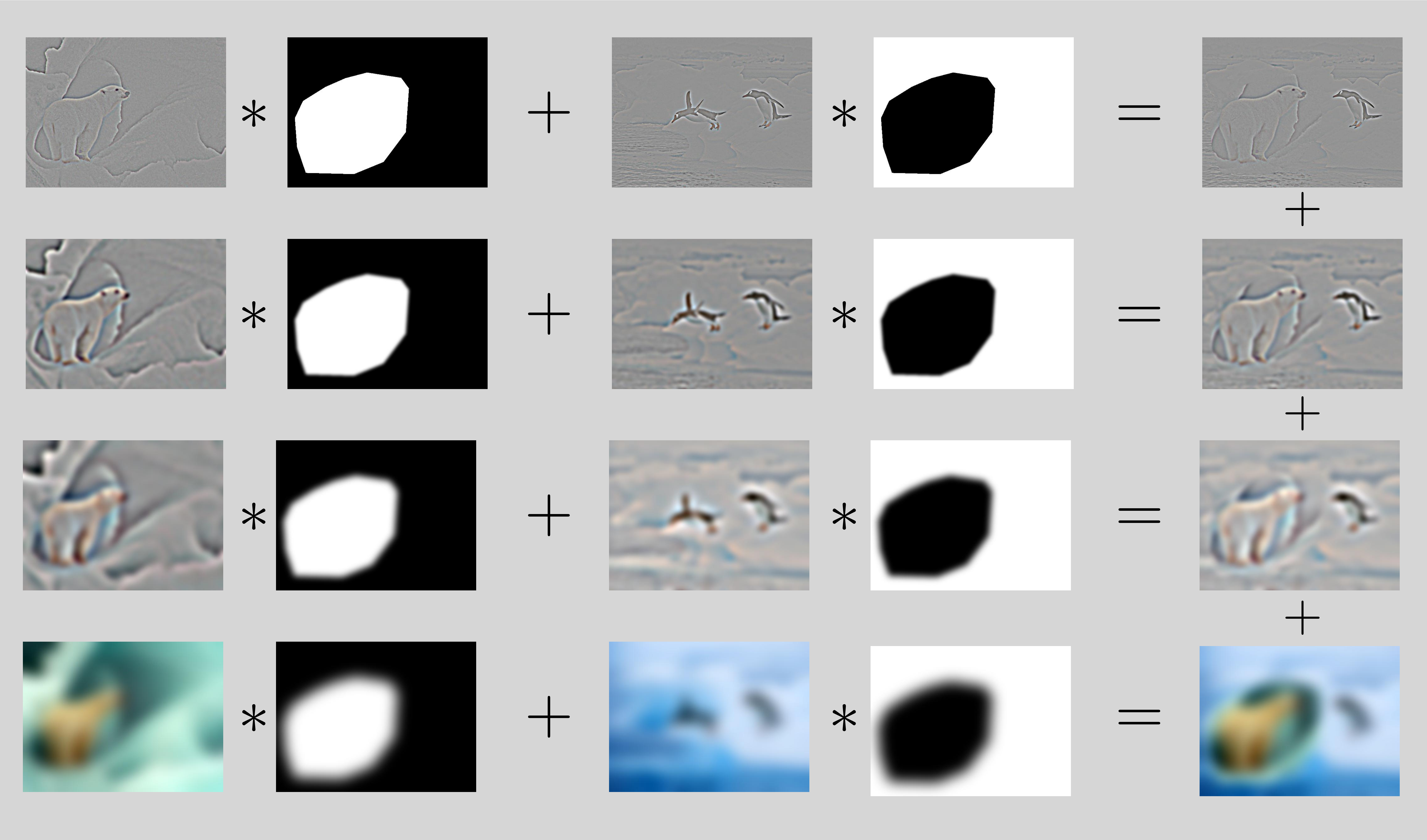
Application: Image Pyramid Blending

Application: Image Pyramid Blending





Step 1: Specify an Image Mask



Step 2: blend lower frequency bands over larger spatial ranges, high frequency bands over small spatial ranges

Application: Image Blending

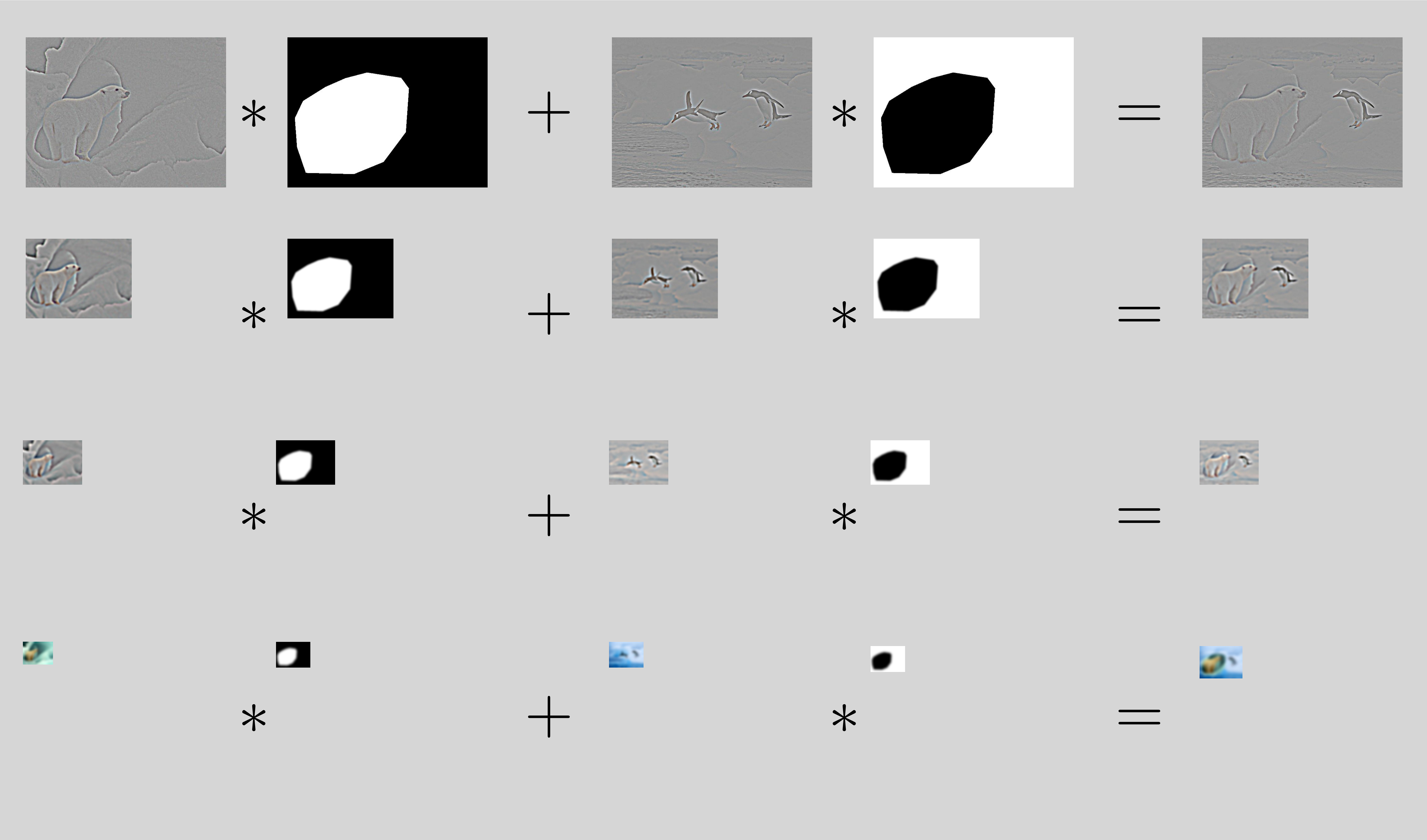
Algorithm:

1. Build Laplacian pyramid LA and LB from images A and B

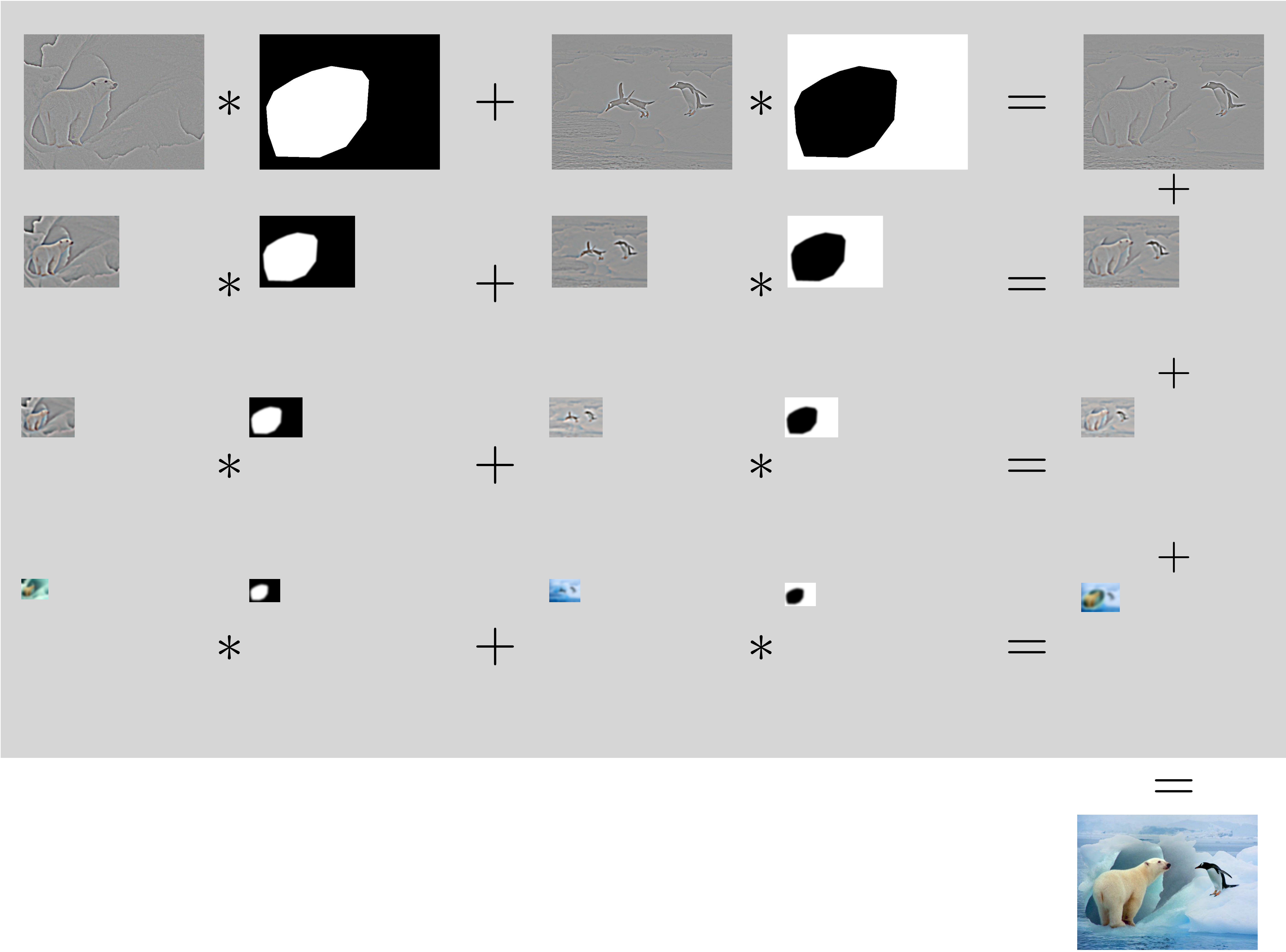
2. Build a Gaussian pyramid GR from mask image R (the mask defines which image pixels should be coming from A or B)

3. From a combined (blended) Laplacian pyramid LS, using nodes of GR as weights: LS(i,j) = GR(i,j) * LA(i,j) + (1-GR(i,j)) * LB(i,j)

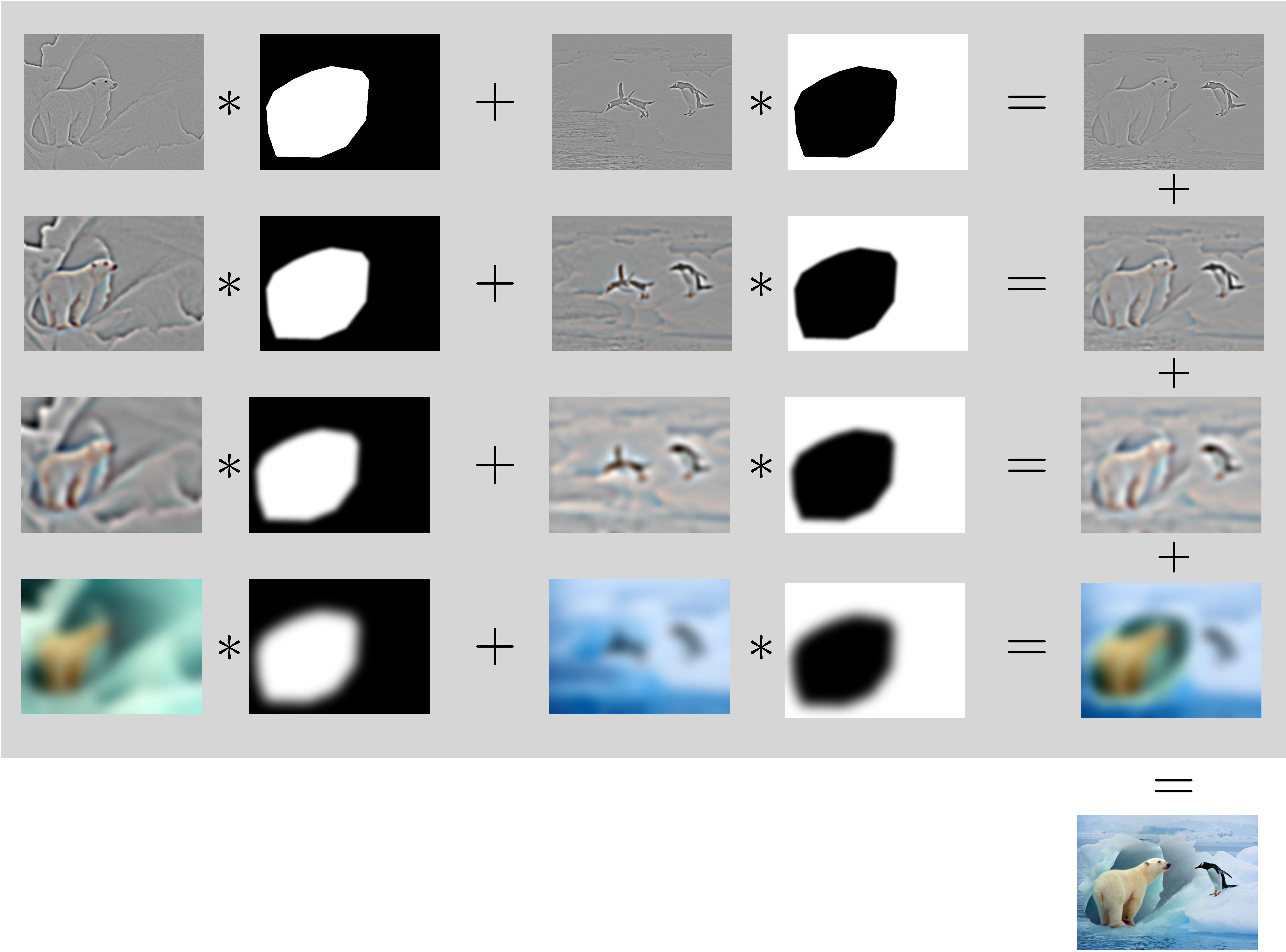
4. Reconstruct the final blended image from LS



Polar Bear Laplacian Pyramid	Mask Gaussian Pyramid	Penguin Laplacian Pyramid	1 - Mask Gaussian Pyramid	Result Pyramid



Reconstruct Result

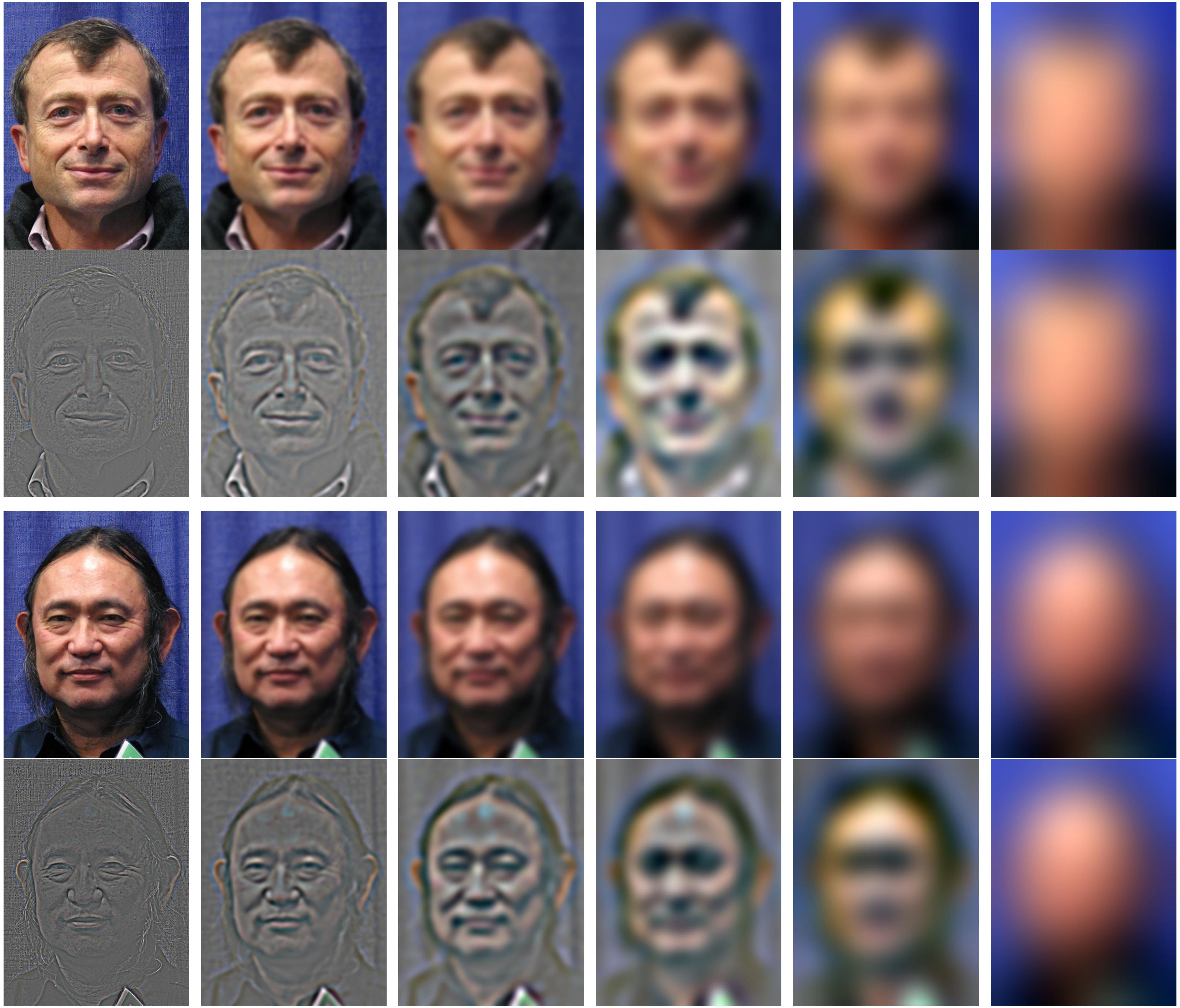


Reconstruct
Result

[Jim Kajiya, Andries van Dam]

[Jim Kajiya, Andries van Dam]



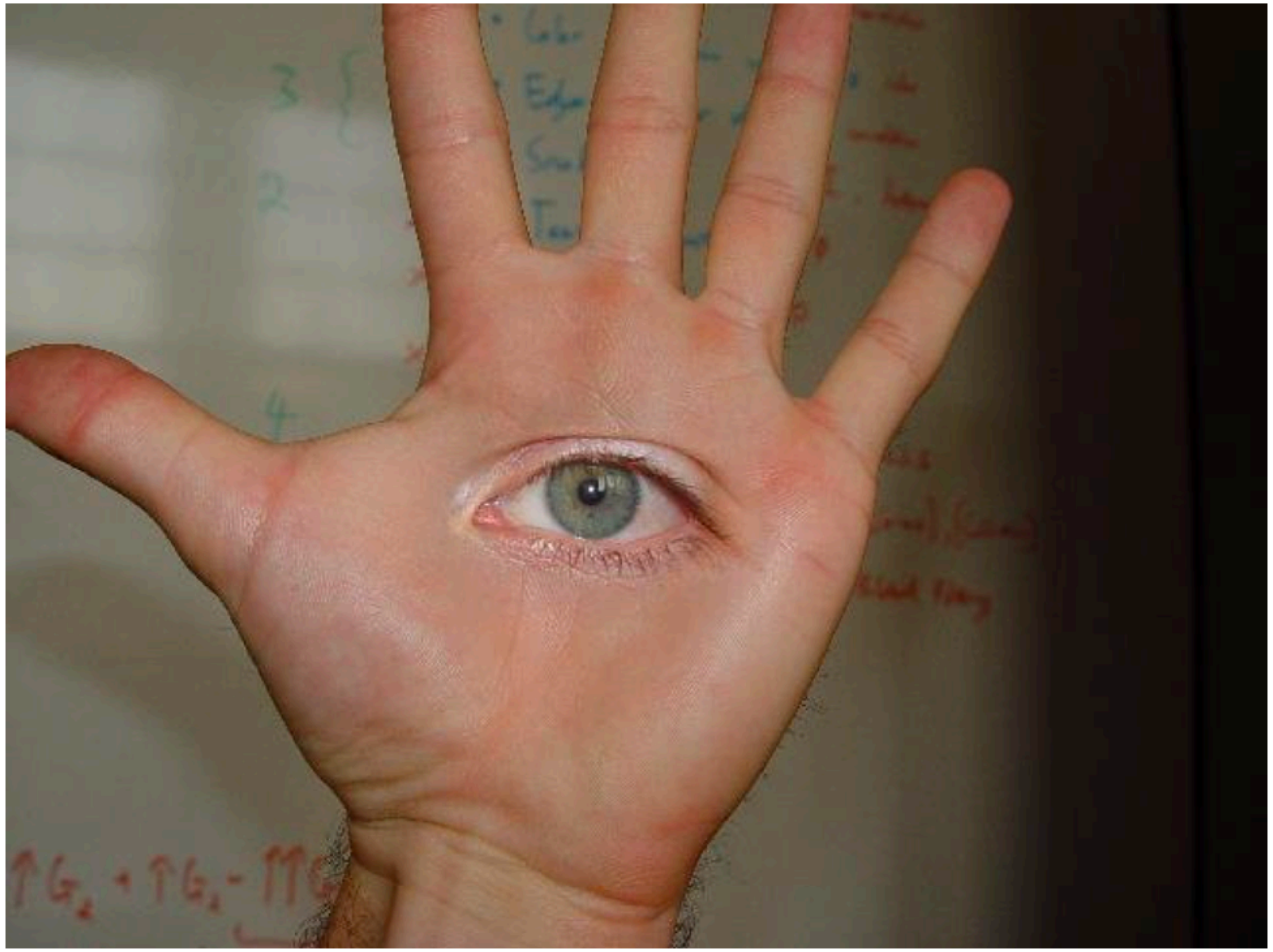
 

Alpha blend with sharp fall-off

Alpha blend with gradual fall-off

Pyramid Blend

More examples …



© david dmartin (Boston College)

More examples …

© Chris Cameron

Summary: **Scaled Representations**

Gaussian Pyramid

—Each level represents a **low-pass** filtered image at a different scale

—Generated by successive Gaussian blurring and downsampling

—Useful for image resizing, sampling

Laplacian Pyramid

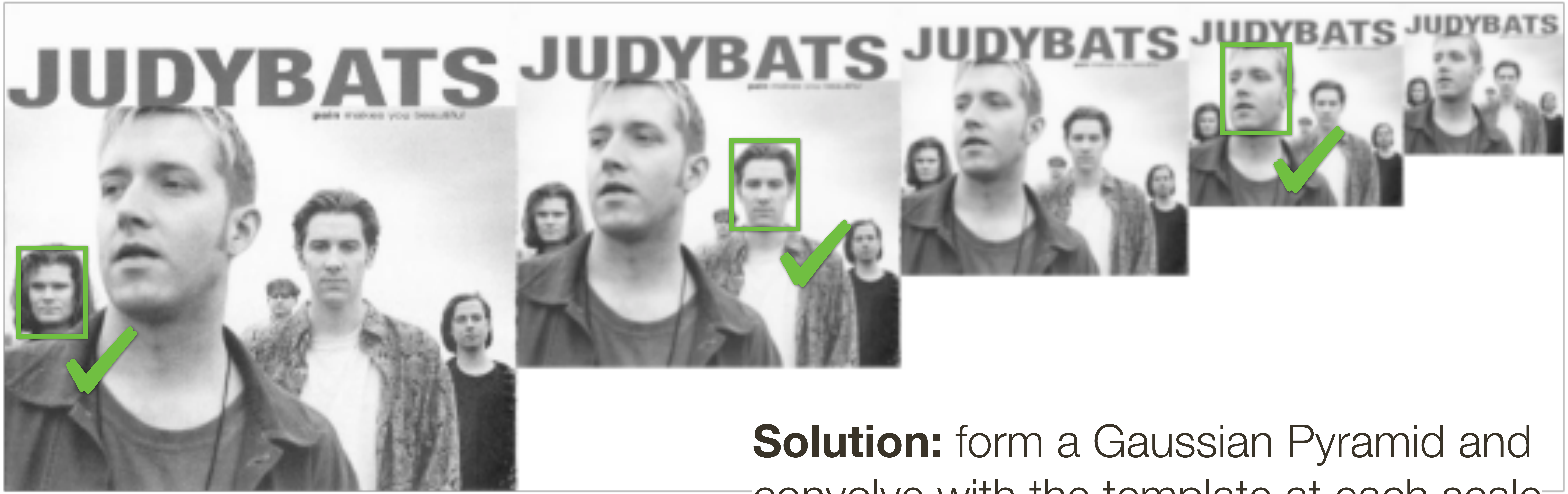
—Each level is a **band-pass** image at a different scale

—Generated by differences between successive levels of a Gaussian Pyramid

—Used for pyramid blending, feature extraction etc.

Recap: **Multi-Scale** Template Matching

Correlation with a **fixed-sized image** only detects faces at **specific scales**

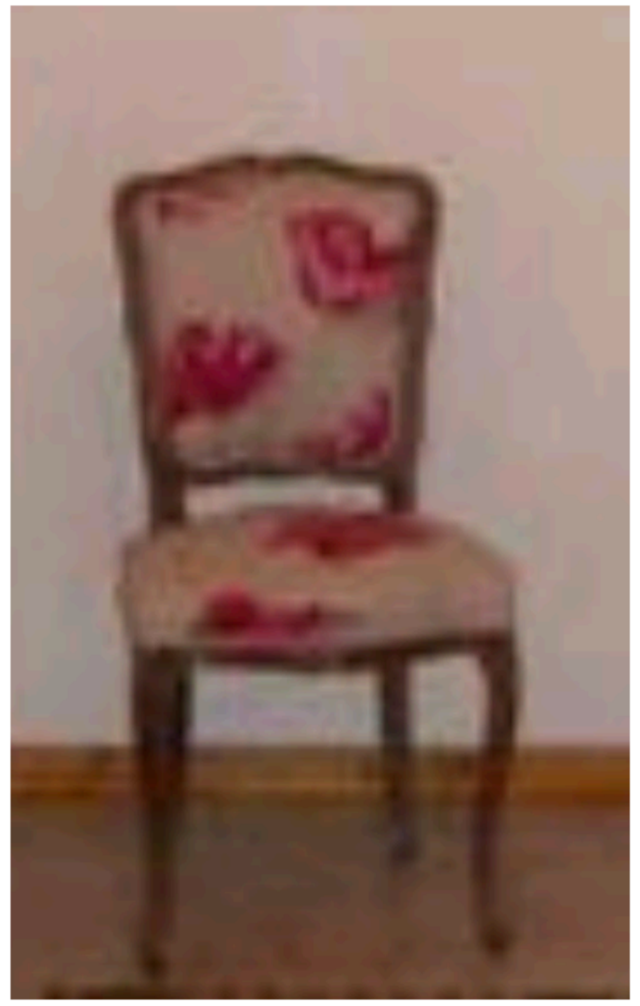


Solution: form a Gaussian Pyramid and convolve with the template at each scale

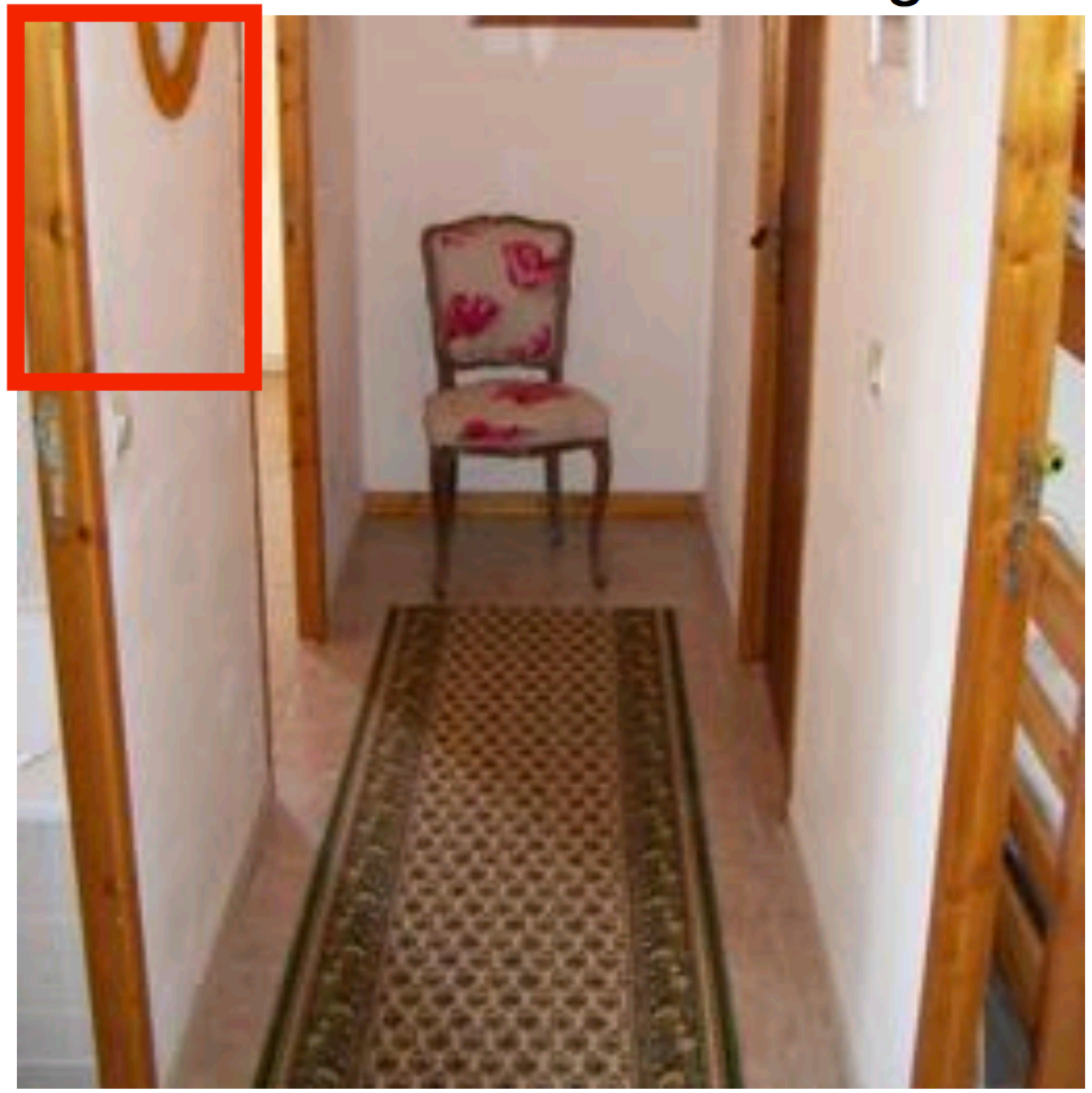
 Q. **Why scale** the **image** and not the **template**?  = Template

Improving Template Matching

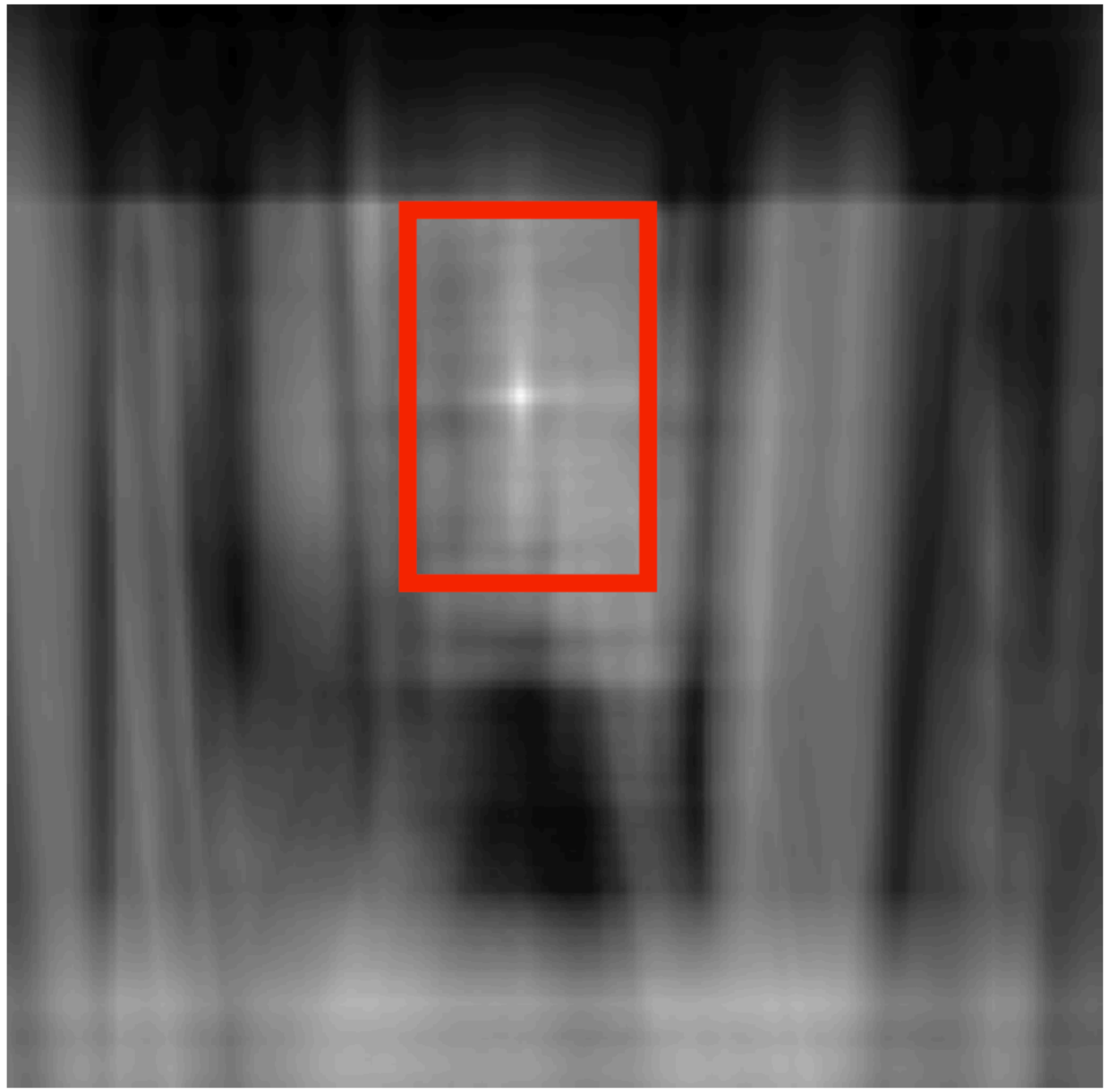
This is a chair



Find the chair in this image

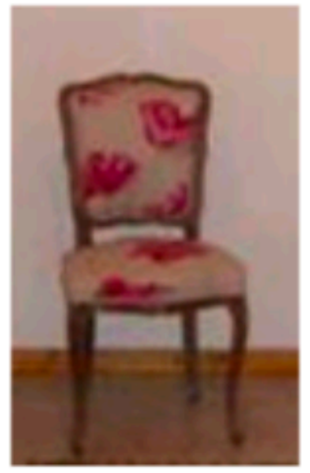


Output of normalized correlation

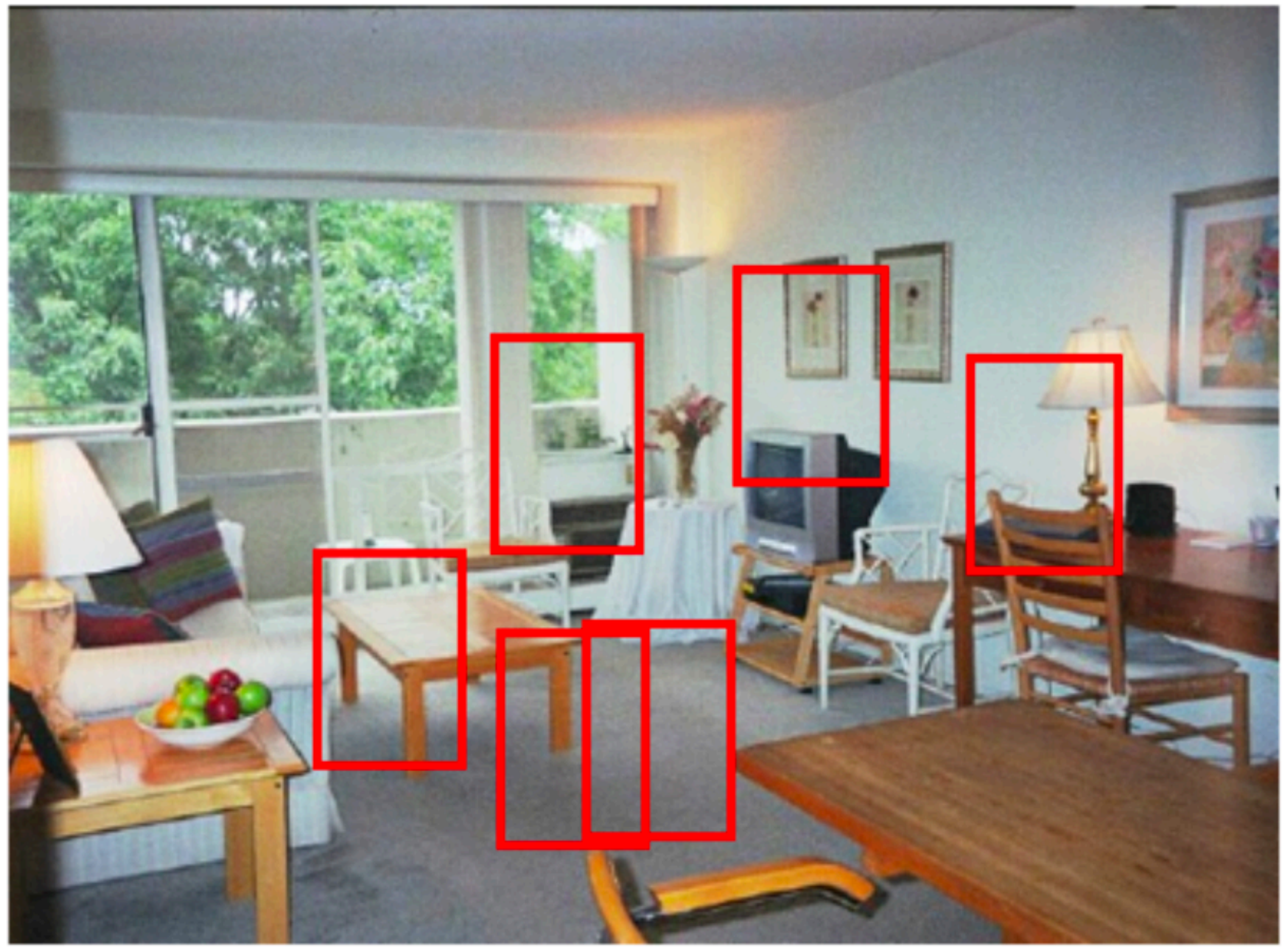


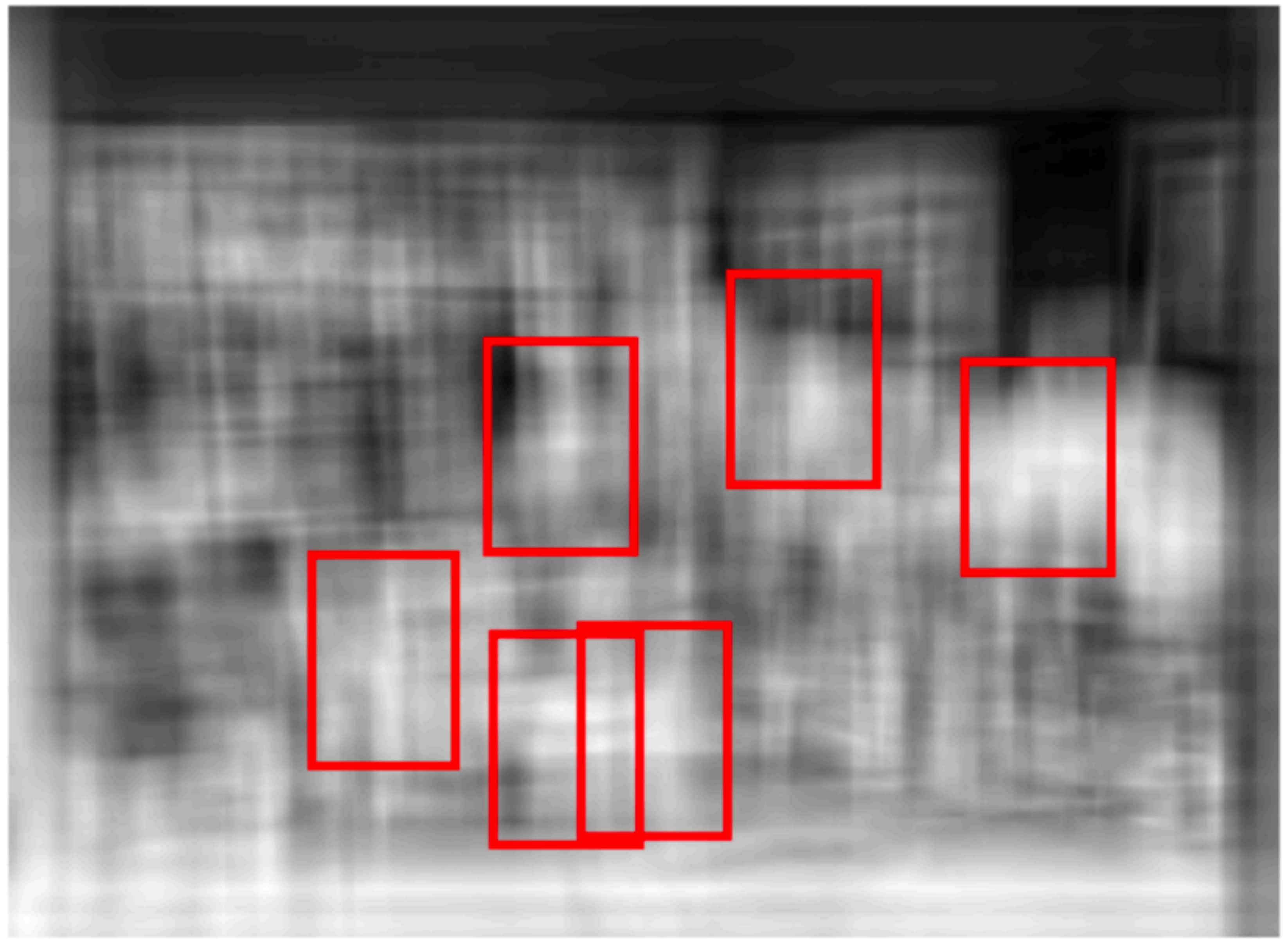
Slide Credit: Li Fei-Fei, Rob Fergus, and Antonio Torralba

Improving Template Matching



Find the chair in this image





Pretty much garbage
Simple template matching is not going to make it

Slide Credit: Li Fei-Fei, Rob Fergus, and Antonio Torralba

Improving Template Matching

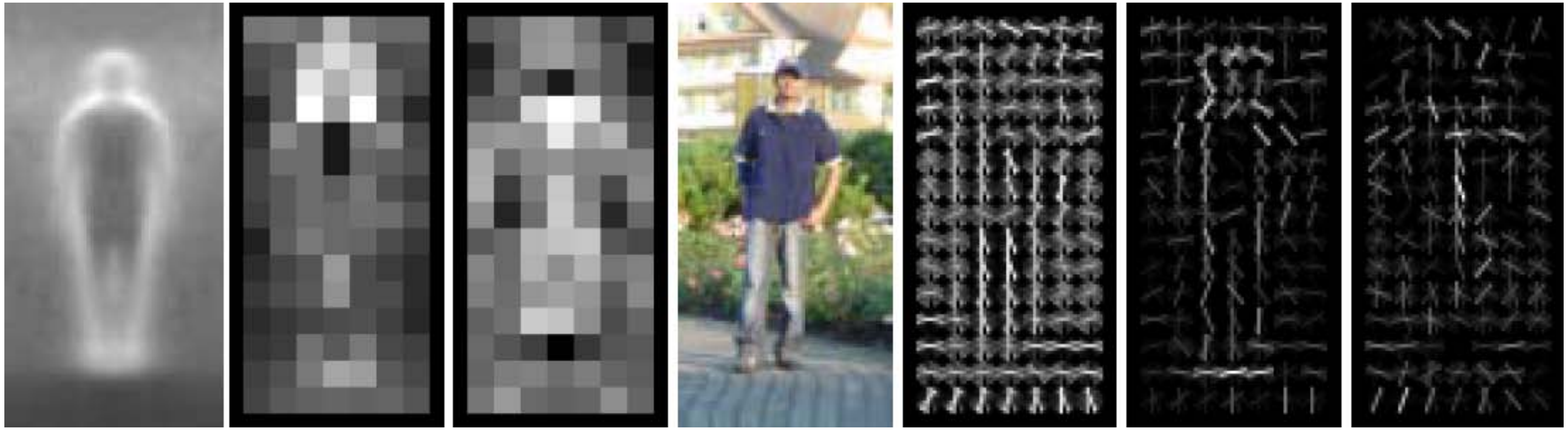
Improved detection algorithms make use of **image features**

These can be **hand coded** or **learned**

Template Matching with **HoG**

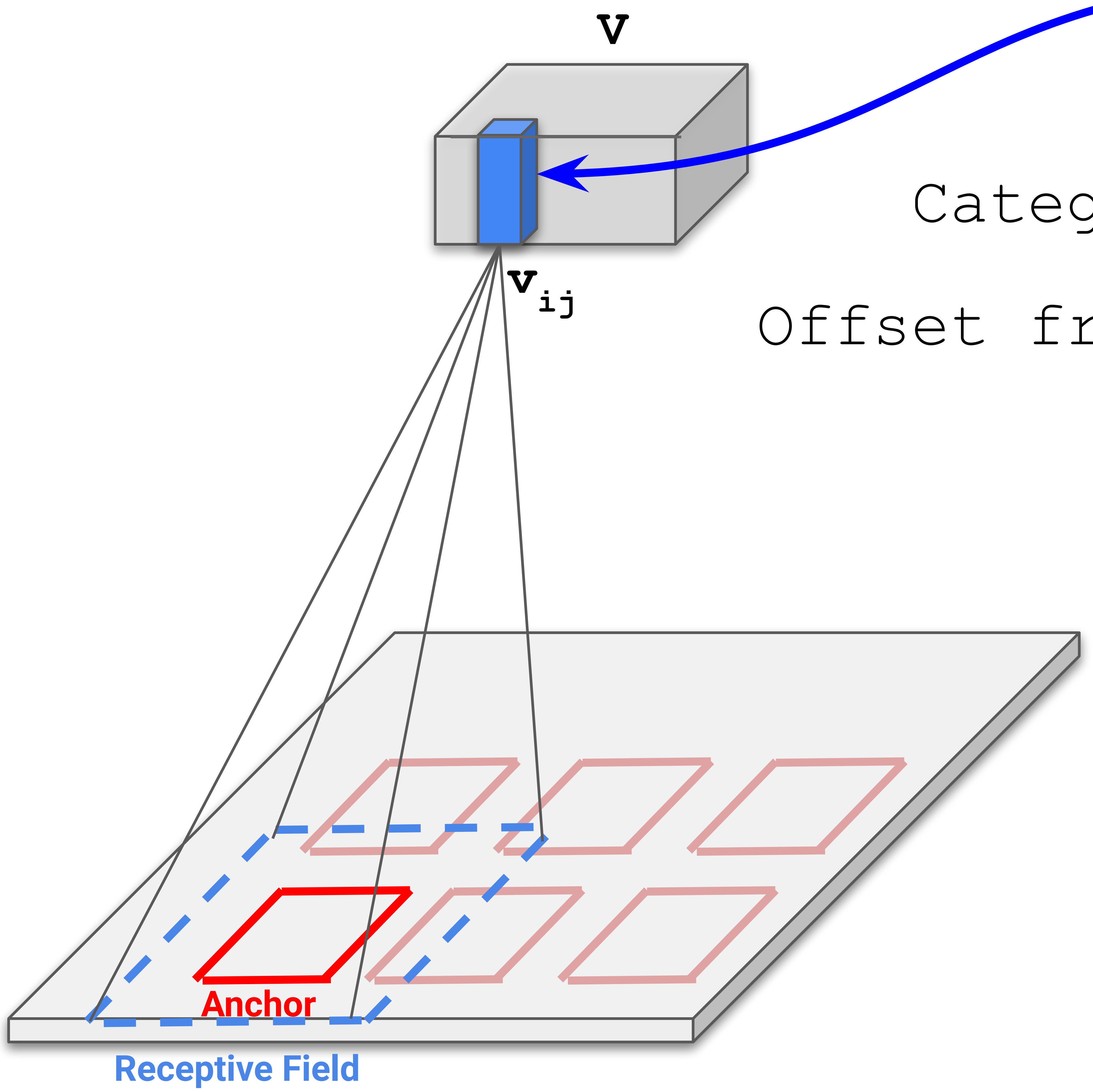
Template matching can be improved by using better features, e.g., Histograms of Gradients (HOG) [Dalal Triggs 2005]

The authors use a Learning-based approach (Support Vector Machine) to find an optimally weighted template



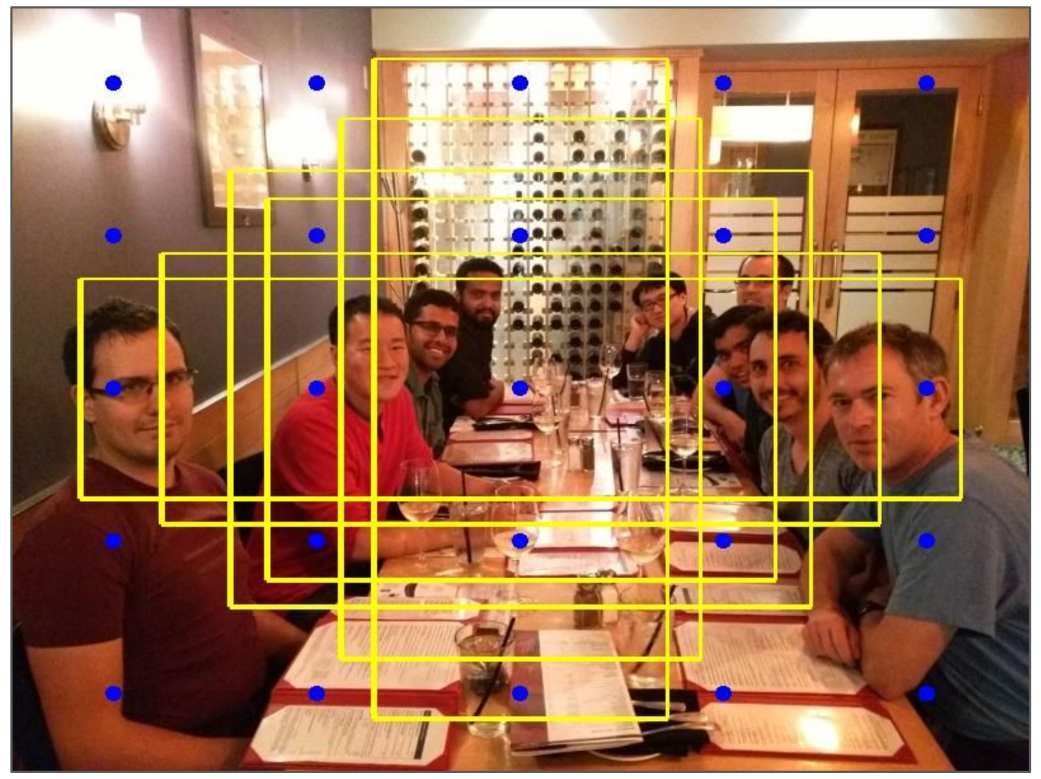
Convnet Object Detection

Think of each feature vector $\mathbf{v}_{ij}$ as corresponding to a sliding window (anchor).



$$\texttt{Category score} = \texttt{SoftMax}(W^{cls} \cdot \mathbf{v}_{ij})$$

$$\texttt{Offset from anchor} = W^{loc} \cdot \mathbf{v}_{ij}$$



— Convnet based object detectors resemble sliding window template matching in feature space

— Architectures typically involve multiple scales and aspect ratios, and regress to a 2D offset in addition to category scores

[Images: Jonathan Huang]

Summary

Template matching as (normalized) correlation. Template matching is not robust to changes in:

— 2D spatial scale and 2D orientation

— 3D pose and viewing direction

— illumination

Scaled representations facilitate

— template matching at multiple scales

— efficient search for image–to–image correspondences

— image analysis at multiple levels of detail

A **Gaussian pyramid** reduces artifacts introduced when sub-sampling to coarser scales

From Template Matching to **Local Feature Detection**

We'll now shift from global template matching to **local feature detection**

Consider the problem of finding images of an elephant using a template

From Template Matching to **Local Feature Detection**

We’ll now shift from global template matching to **local feature detection**

Consider the problem of finding images of an elephant using a template

An elephant looks different from different viewpoints
— from above (as in an aerial photograph or satellite image)
— head on
— sideways (i.e., in profile)
— rear on

What happens if parts of an elephant are obscured from view by trees, rocks, other elephants?

From Template Matching to **Local Feature Detection**

- Move from global template matching to **local template matching**
- Local template matching also called local **feature detection**
- Obvious local features to detect are **edges** and **corners**