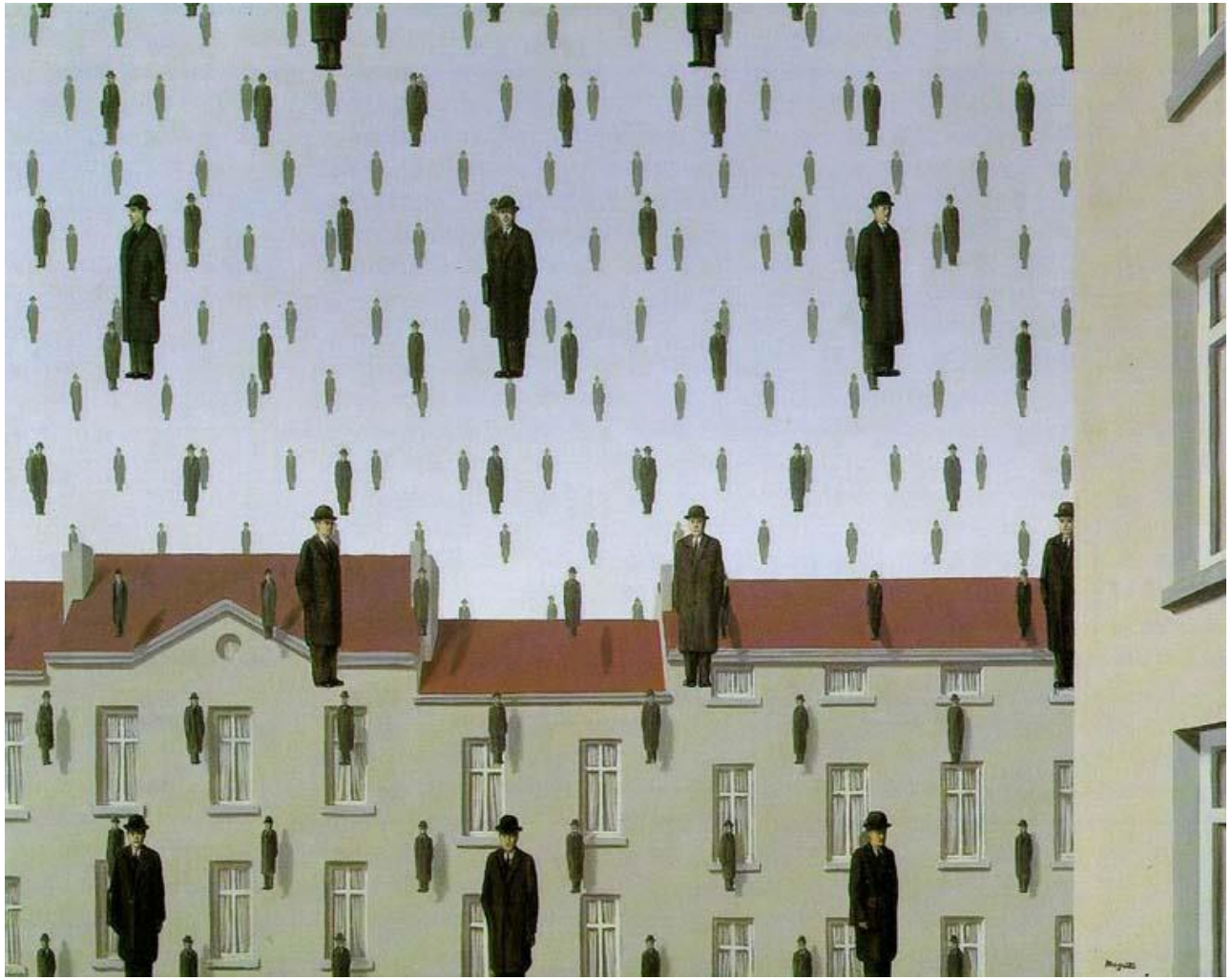
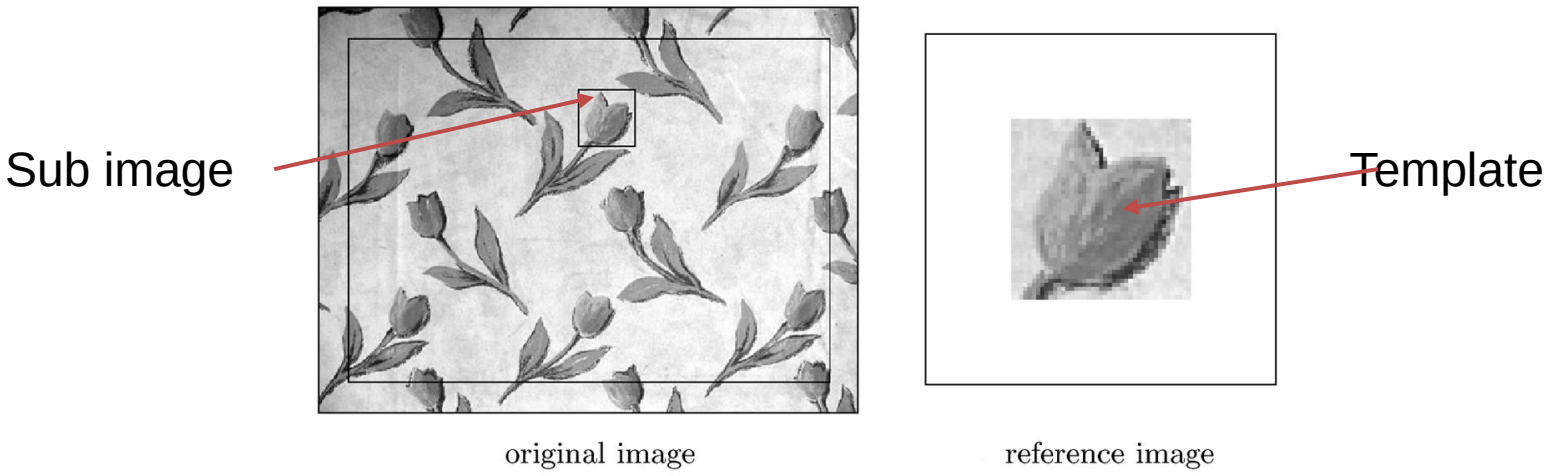
Templates, Image Pyramids



Template matching

- Move given pattern (template) over search image
- Measure difference between template and sub-images at different positions
- Record positions where highest similarity is found

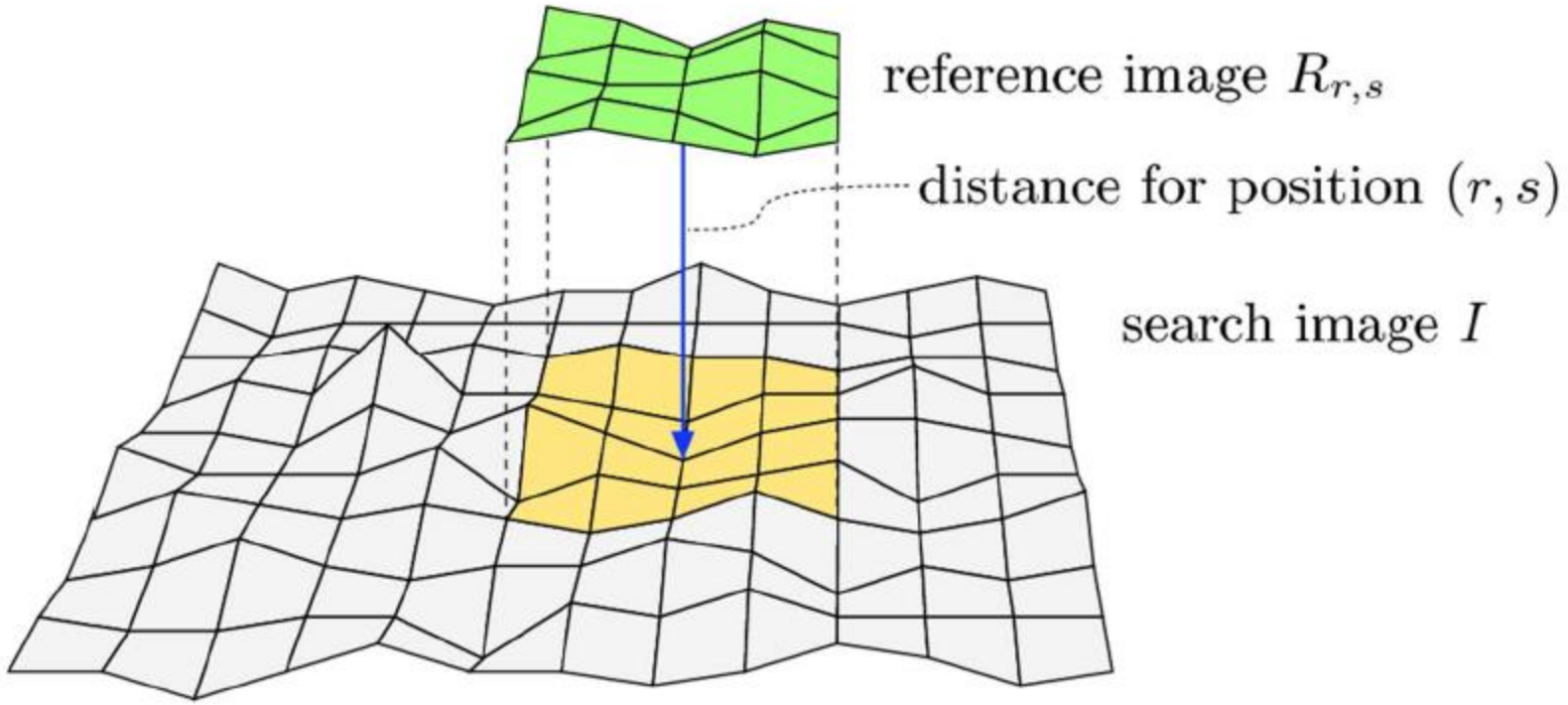


Template matching

- What is distance (difference) measure?
- What levels of difference should be considered a match?
- How are results affected when brightness or contrast changes?
- Solving this problem involves solving many sub-problems

Distance between Image Patterns

- Many measures proposed to compute distance between the shifted reference image (template) and corresponding sub image



Distance between Image Patterns

- Many measures proposed to compute distance between the shifted reference image $R_{r,s}$ and corresponding subimage I
- Sum of absolute differences:

$$\mathrm{d}_A(r,s) = \sum_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

- Maximum difference:

$$\mathrm{d}_M(r,s) = \max_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

- Sum of squared differences (also called N-dimensional Euclidean distance):

$$\mathrm{d}_E(r,s) = \Big[\sum_{(i,j)\in R} \big(I(r+i, s+j) - R(i,j)\big)^2 \Big]^{1/2}$$

Distance and Correlation

- Best matching position between shifted reference image $R_{r,s}$ and subimage *I* minimizes square of d_E which can be expanded as:

$$\begin{aligned} \mathrm{d}_E^2(r,s) &= \sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2 \\ &= \underbrace{\sum_{(i,j)\in R} I^2(r+i,s+j)}_{A(r,s)} + \underbrace{\sum_{(i,j)\in R} R^2(i,j)}_{B} - \underbrace{2\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}_{C(r,s)} \end{aligned}$$

- *B* term is a constant, independent of *r, s* and can be ignored
- *A* term is sum of squared values within subimage *I* at current offset *r, s*

Distance and Correlation

$$\mathrm{d}_E^2(r,s) = \sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2$$

$$= \underbrace{\sum_{(i,j)\in R} I^2(r+i,s+j)}_{A(r,s)} + \underbrace{\sum_{(i,j)\in R} R^2(i,j)}_{B} - \underbrace{2\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}_{C(r,s)}$$

- *C(r,s)* term is **linear cross correlation** between *I* and *R* defined as

$$(I \circledast R)(r,s) = \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} I(r+i,s+j)\cdot R(i,j)$$

- Since *R* and *I* are assumed to be zero outside their boundaries

$$\sum_{i=0}^{w_R-1}\sum_{j=0}^{h_R-1} I(r+i,s+j)\cdot R(i,j) = \sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)$$

- **Note:** Correlation is similar to linear convolution
- Min value of $d^2_E(r,s)$ corresponds to max value of $(I \circledast R)(r,s)$

Normalized Cross Correlation

- Unfortunately, *A* term is not constant in most images
- Thus cross correlation result varies with intensity changes in image *I*
- **Normalized cross correlation** considers energy in *I* and *R*

$$C_N(r,s) = \frac{C(r,s)}{\sqrt{A(r,s)\cdot B}} = \frac{C(r,s)}{\sqrt{A(r,s)}\cdot\sqrt{B}}$$

$$= \frac{\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}{\left[\sum_{(i,j)\in R} I^2(r+i,s+j)\right]^{1/2}\cdot\left[\sum_{(i,j)\in R} R^2(i,j)\right]^{1/2}}$$

- $C_N\,(r,s)$ is a local distance measure, is in [0,1] range
- $C_N\,(r,s)$ = 1 indicates maximum match
- $C_N\,(r,s)$ = 0 indicates images are very dissimilar

Correlation Coefficient

- **Correlation coefficient:** Use differences between *I* and *R* and their average values

$$C_L(r,s) = \frac{\sum_{(i,j)\in R} \big(I(r+i,s+j) - \bar{I}(r,s)\big) \cdot \big(R(i,j) - \bar{R}\big)}{\Big[\sum_{(i,j)\in R} \big(I(r+i,s+j) - \bar{I}_{r,s}\big)^2\Big]^{1/2} \cdot \Big[\underbrace{\sum_{(i,j)\in R} \big(R(i,j) - \bar{R}\big)^2}_{S_R^2 = K\cdot\sigma_R^2}\Big]^{1/2}}$$

- where the average values are defined as

$$\bar{I}_{r,s} = \frac{1}{K} \cdot \sum_{(i,j)\in R} I(r+i,s+j) \quad \text{and} \quad \bar{R} = \frac{1}{K} \cdot \sum_{(i,j)\in R} R(i,j)$$

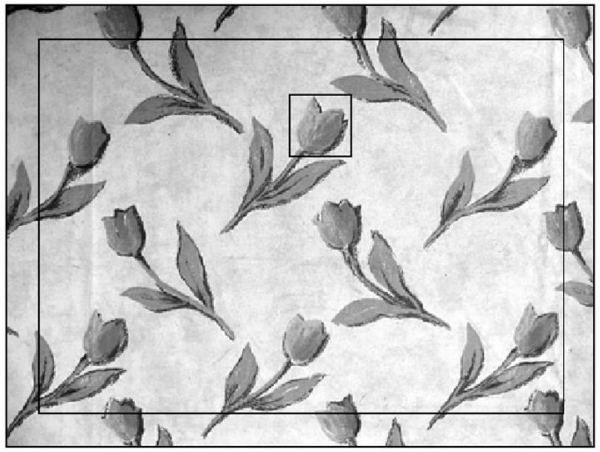
- *K* is number of pixels in reference image *R*

- $C_L(r,s)$ can be rewritten as

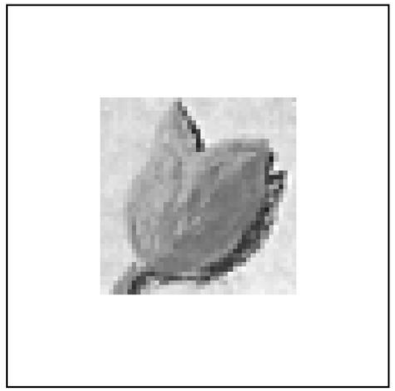
$$C_L(r,s) = \frac{\sum_{(i,j)\in R} \big(I(r+i,s+j)\cdot R(i,j)\big) \; - \; K\cdot\bar{I}_{r,s}\cdot\bar{R}}{\Big[\sum_{(i,j)\in R} I^2(r+i,s+j) \; - \; K\cdot\bar{I}_{r,s}^2\Big]^{1/2} \cdot S_R}$$

Examples and discussion

- We now compare these distance metrics
- **Original image *I*:** Repetitive flower pattern
- **Reference image *R*:** one instance of repetitive pattern extracted from *I*



original image I



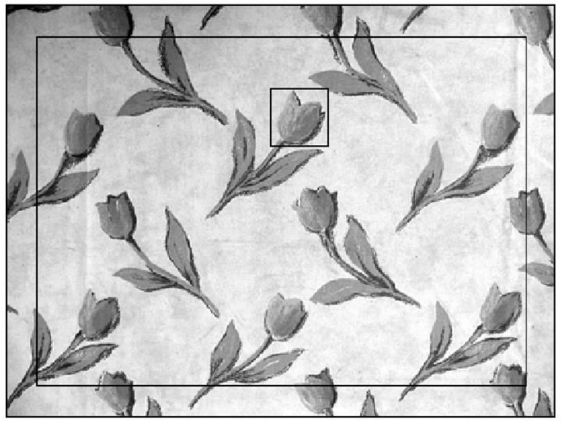
reference image R

- Now compute various distance measures for this *I* and *R*

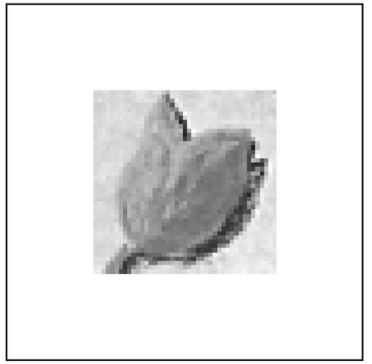
Examples and discussion

sum of absolute differences

maximum difference



original image I



reference image R

- **Sum of absolute differences**: performs good but affected by global intensity changes

$$\mathrm{d}_A(r,s) = \sum_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

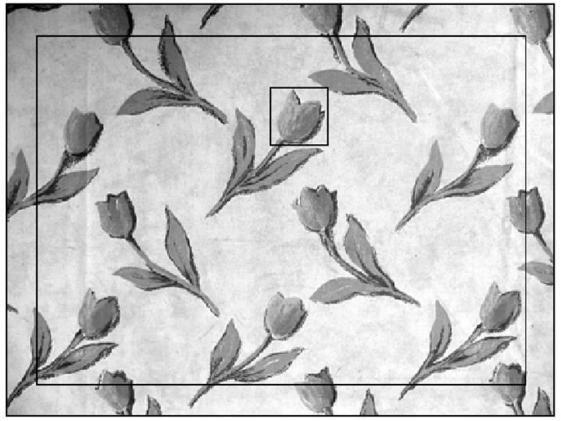
- **Maximum difference:** Responds more to lighting intensity changes than pattern similarity

$$\mathrm{d}_M(r,s) = \max_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

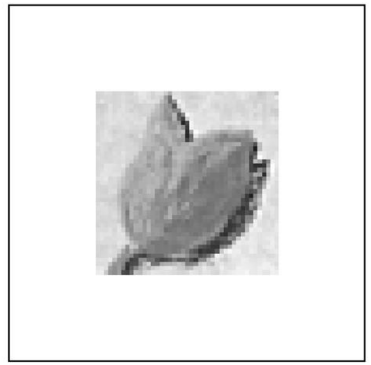
Examples and discussion

sum of squared distances

global cross correlation



original image I



reference image R

- **Sum of squared (euclidean) distances:** performs good but affected by global intensity changes

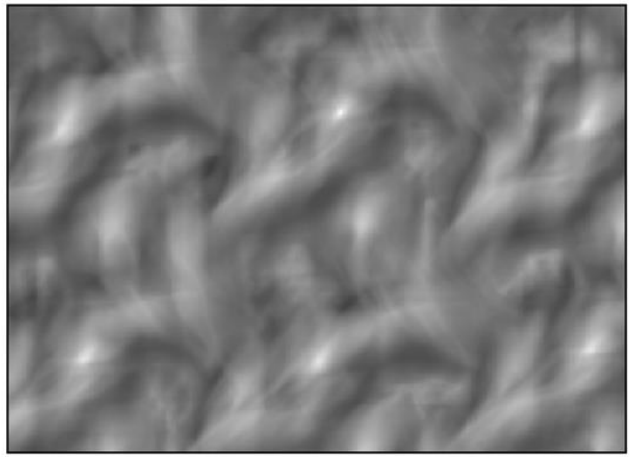
$$\mathrm{d}_E(r,s) = \Big[\sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2 \Big]^{1/2}$$

- **Global cross correlation:** Local maxima at true template position, but is dominated by high-intensity responses in brighter image parts

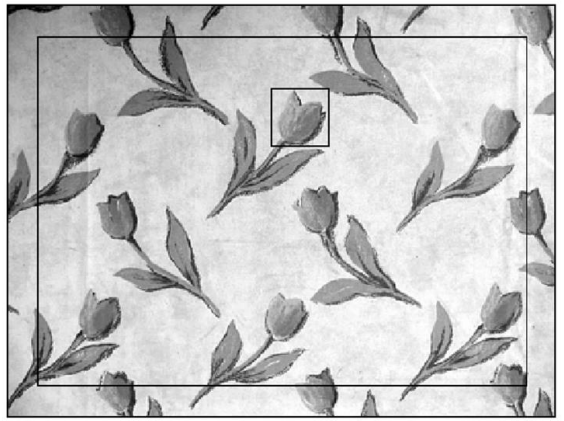
$$(I \circledast R)(r,s) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(r+i,s+j) \cdot R(i,j)$$

Examples and discussion

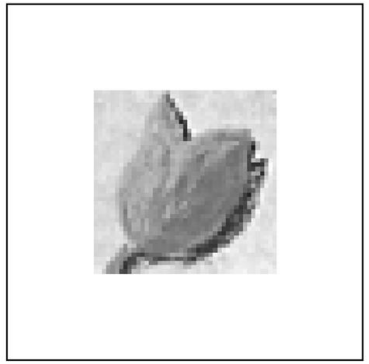
normalized cross correlation



correlation coefficient



original image I



reference image R

- **Normalized cross correlation**: results similar to Euclidean distance (affected by global intensity changes)

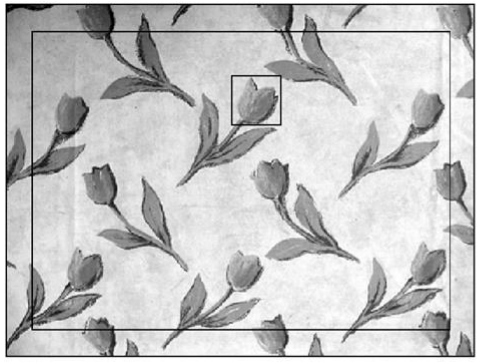
$$\frac{\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}{\Big[\sum_{(i,j)\in R} I^2(r+i,s+j)\Big]^{1/2}\cdot\Big[\sum_{(i,j)\in R} R^2(i,j)\Big]^{1/2}}$$

- **Correlation coefficient:** yields best results. Distinct peaks produced for all 6 template instances, unaffected by lighting

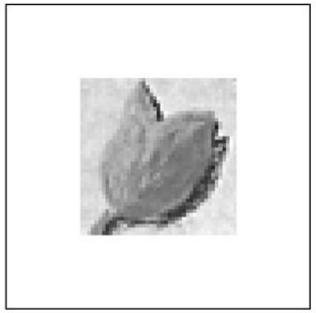
$$C_L(r,s)=\frac{\sum_{(i,j)\in R}\big(I(r+i,s+j)\cdot R(i,j)\big)\;-\;K\cdot\bar{I}_{r,s}\cdot\bar{R}}{\Big[\sum_{(i,j)\in R} I^2(r+i,s+j)\;-\;K\cdot\bar{I}^2_{r,s}\Big]^{1/2}\cdot S_R}$$

Effects of changing intensity

- To explore effects of globally changing intensity, raise intensity of reference image *R* by 50 units
- Distinct peaks disappear in **Euclidean distance**
- **Correlation coefficient** unchanged, robust measure in realistic lighting conditions

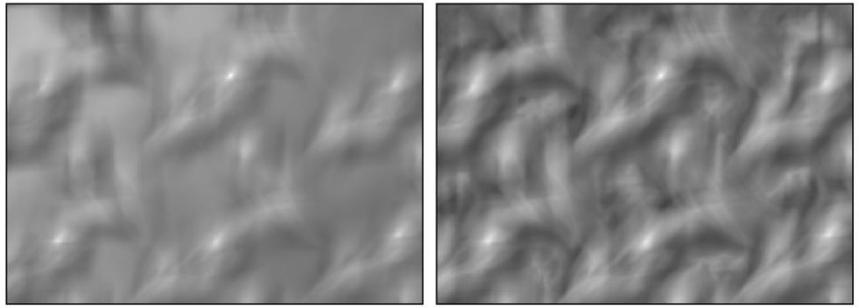


original image I



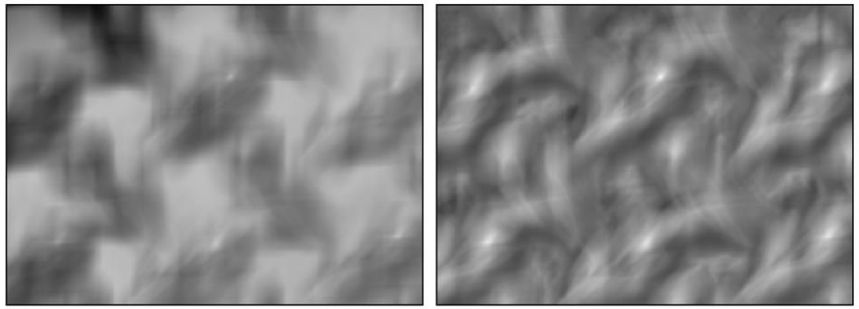
reference image R

Original reference image: R



Euclidean distance correlation coefficient

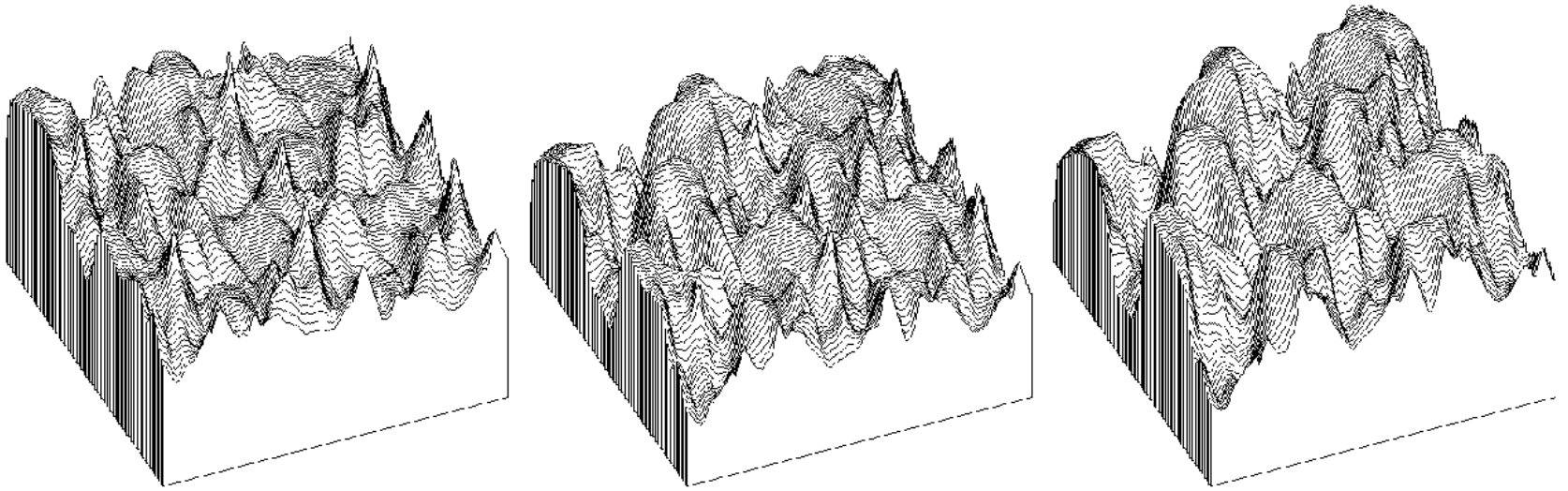
Modified reference image: $R' = R + 50$



Euclidean distance correlation coefficient

Euclidean distance under global intensity changes

- Local peaks disappear as template intensity (and thus distance) is increased



R

Distance function for original template *R*

$R + 25$

Distance function with intensity increased by 25 units

$R + 50$

Distance function with intensity increased by 50 units

Shape of template

- Template does not have to be rectangular
- Some applications use circular, elliptical or custom-shaped templates
- Non-rectangular templates stored in rectangular array, but pixels in template marked using a mask
- More generally, a weighted function can be applied to template elements

cv2.matchTemplate(), cv2.minMaxLoc()

- OpenCV comes with a function cv2.matchTemplate() that slides the template image over the input image (as in 2D convolution) and compares the template and patch of input image under the template image.
- Several comparison methods are implemented in OpenCV
  - cv2.TM_CCOEFF, cv2.TM_CCOEFF_NORMED, cv2.TM_CCORR, cv2.TM_CCORR_NORMED, cv2.TM_SQDIFF, cv2.TM_SQDIFF_NORMED
- It returns a grayscale image, where each pixel denotes how much does the neighborhood of that pixel match with template
- If input image is of size (WxH) and template image is of size (wxh), output image will have a size of (W-w+1, H-h+1). Once you got the result, you can use cv2.minMaxLoc() function to find where is the maximum/minimum value.
  - Take it as the top-left corner of rectangle and take (w,h) as width and height of the rectangle
  - That rectangle is your region of template

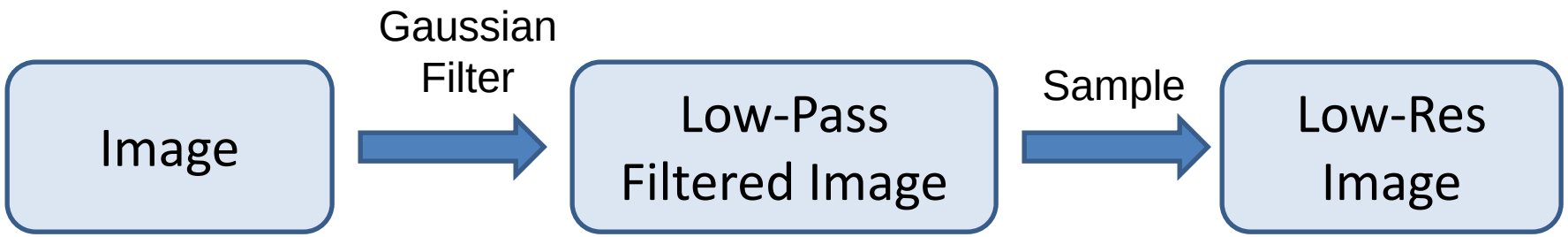
Q: What if we want to find larger or smaller eyes?

A: Image Pyramid

Image Pyramid

- Normally, we used to work with an image of constant size. But in some occasions, we need to work with images of different resolution of the same image
- For example, while searching for something in an image, like face, we are not sure at what size the object will be present in the image
- In that case, we will need to create a set of images with different resolution and search for object in all the images
- These set of images with different resolution are called Image Pyramids (because when they are kept in a stack with biggest image at bottom and smallest image at top look like a pyramid)
- There are two kinds of Image Pyramids
  - Gaussian Pyramid
  - Laplacian Pyramids

Review of Sampling



Gaussian pyramid

512 256 128 64 32 16 8

Gaussian pyramid creates versions of the input image at multiple resolutions

This is useful for analysis across different spatial scales, but doesn't separate the image into different frequency bands

Source: Forsyth

Template Matching with Image Pyramids

Input: Image, Template

1. Match template at current scale

2. Downsample image
   - In practice, scale step of 1.1 to 1.2

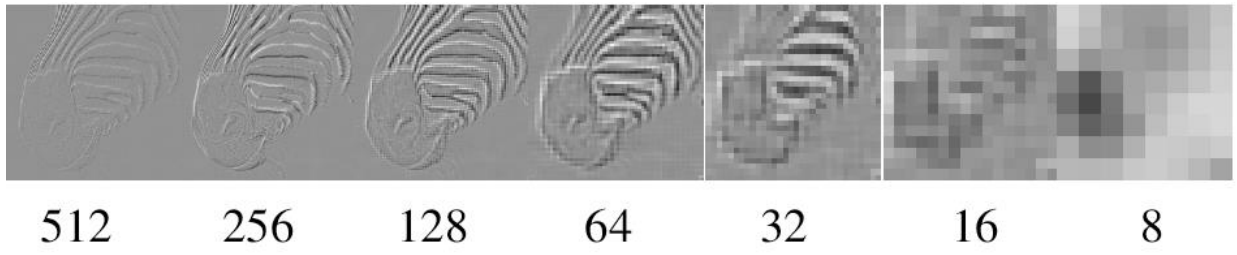
3. Repeat 1-2 until image is very small

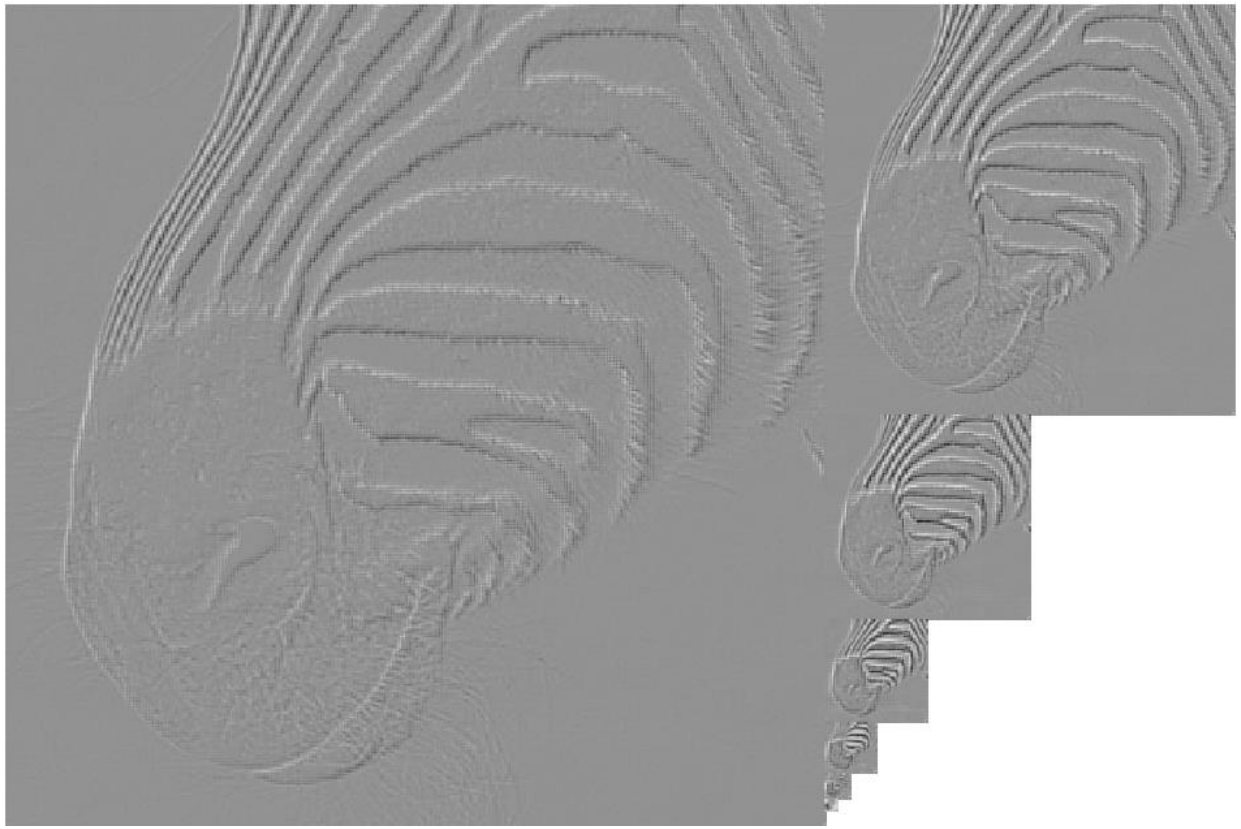
4. Take responses above some threshold, perhaps with non-maxima suppression

Laplacian pyramid

- Laplacian Pyramids are formed from the Gaussian Pyramids
- There is no exclusive function for that
- Laplacian pyramid images are like edge images only
- Most of its elements are zeros. They are used in image compression
- A level in Laplacian Pyramid is formed by the difference between that level in Gaussian Pyramid and expanded version of its upper level in Gaussian Pyramid

Laplacian pyramid

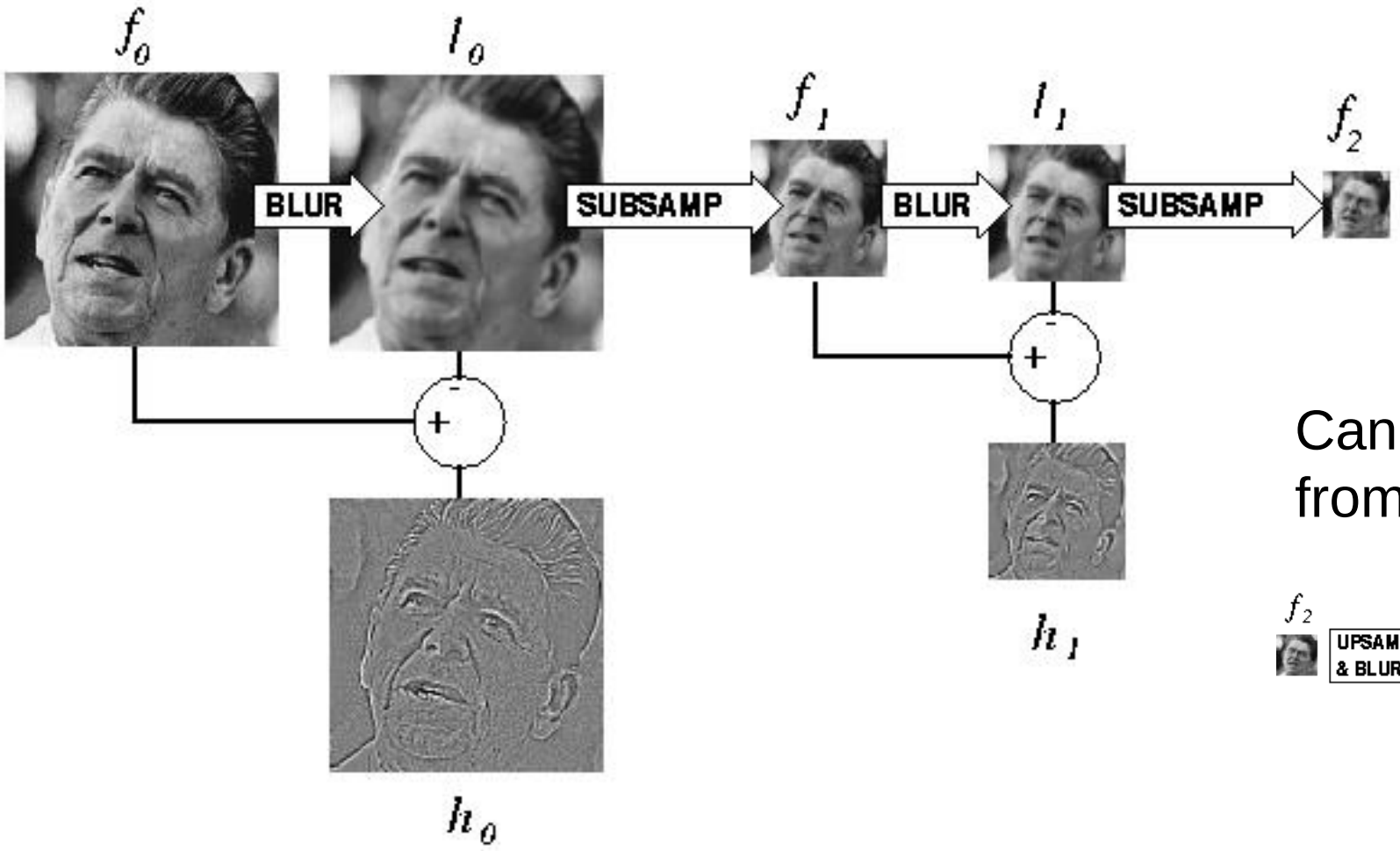




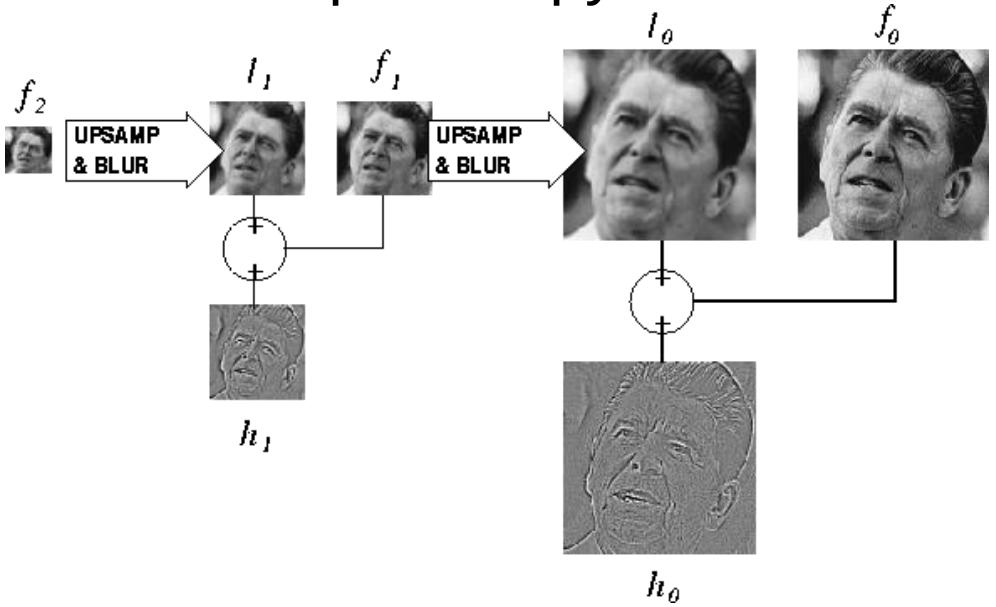
Laplacian pyramid provides an extra level of analysis as compared to Gaussian pyramid by breaking the image into different isotropic spatial frequency bands

Source: Forsyth

Computing Gaussian/Laplacian Pyramid



Can we reconstruct the original from the laplacian pyramid?



Pyramid in OpenCV

- **cv2.pyrUp()**, **cv2.pyrDown()**

- img = cv2.imread('messi5.jpg')
- lower_reso = cv2.pyrDown(higher_reso)

- higher_reso2 = cv2.pyrUp(lower_reso)

Image Blending using Pyramids

- One application of Pyramids is Image Blending
- For example, in image stitching, you will need to stack two images together, but it may not look good due to discontinuities between images
- In that case, image blending with Pyramids gives you seamless blending without leaving much data in the images
- One classical example of this is the blending of two fruits, Orange and Apple

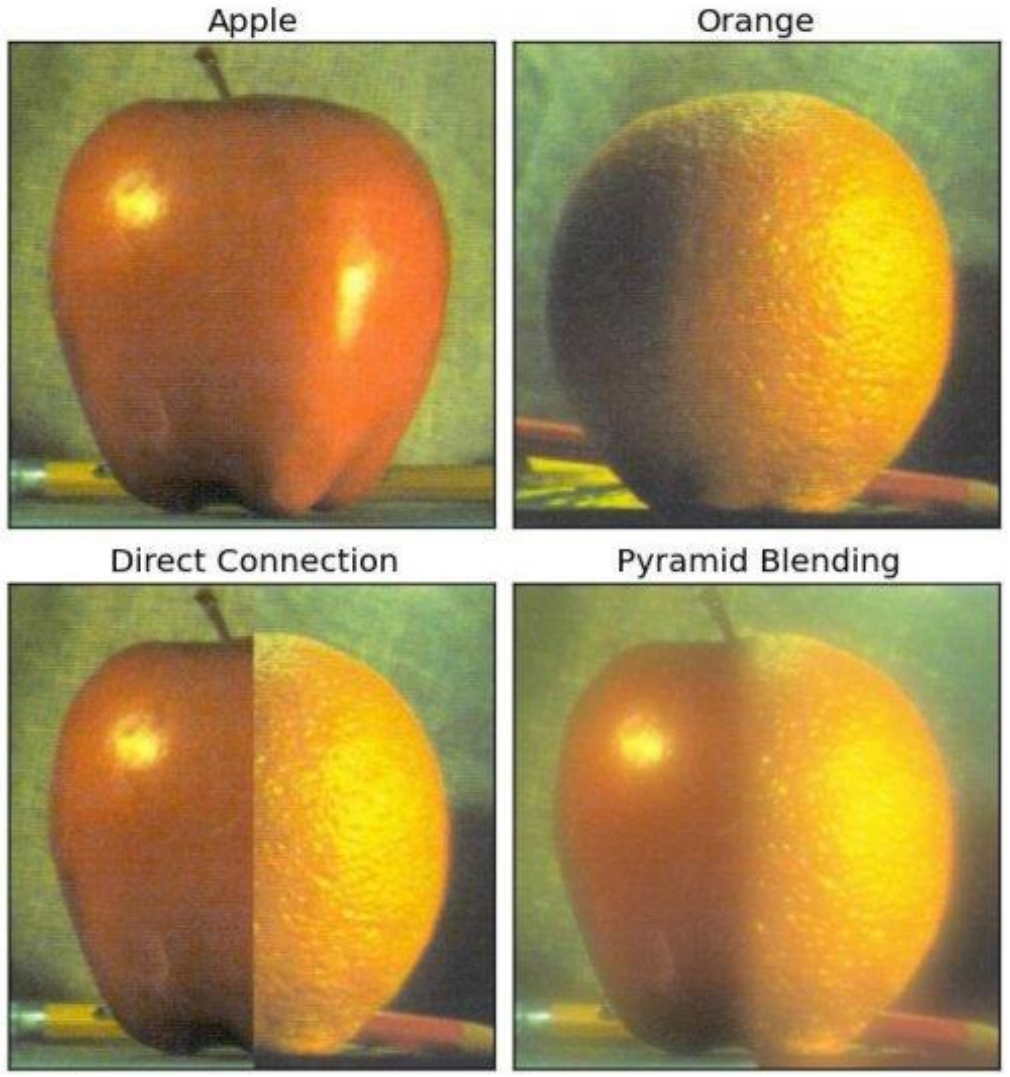


Image Blending using Pyramids

- Load the two images of apple and orange
- Find the Gaussian Pyramids for apple and orange (in this particular example, number of levels is 6)
- From Gaussian Pyramids, find their Laplacian Pyramids
- Now join the left half of apple and right half of orange in each levels of Laplacian Pyramids
- Finally from this joint image pyramids, reconstruct the original image.

Major uses of image pyramids

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Course-to-fine