Lec. 2
Introduction to Computer Vision II
Assist. Prof. Dr. Saad Albawi

Matrix Representation

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

$$\begin{bmatrix} 183 & 160 & 94 & 153 & 194 & 163 & 132 & 165 \\ 183 & 153 & 116 & 176 & 187 & 166 & 130 & 169 \\ 179 & 168 & 171 & 182 & 179 & 170 & 131 & 167 \\ 177 & 177 & 179 & 177 & 179 & 165 & 131 & 167 \\ 178 & 178 & 179 & 176 & 182 & 164 & 130 & 171 \\ 179 & 180 & 180 & 179 & 183 & 169 & 132 & 169 \\ 179 & 179 & 180 & 182 & 183 & 170 & 129 & 173 \\ 180 & 179 & 181 & 179 & 181 & 170 & 130 & 169 \end{bmatrix}$$

H=256

W=256

Divide into 8x8 blocks

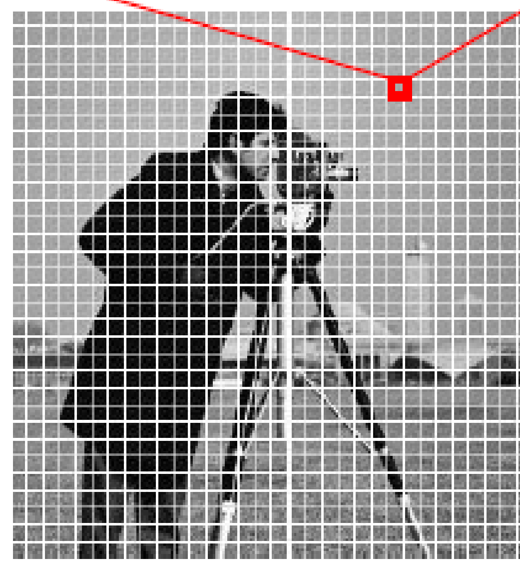


Image Resolution

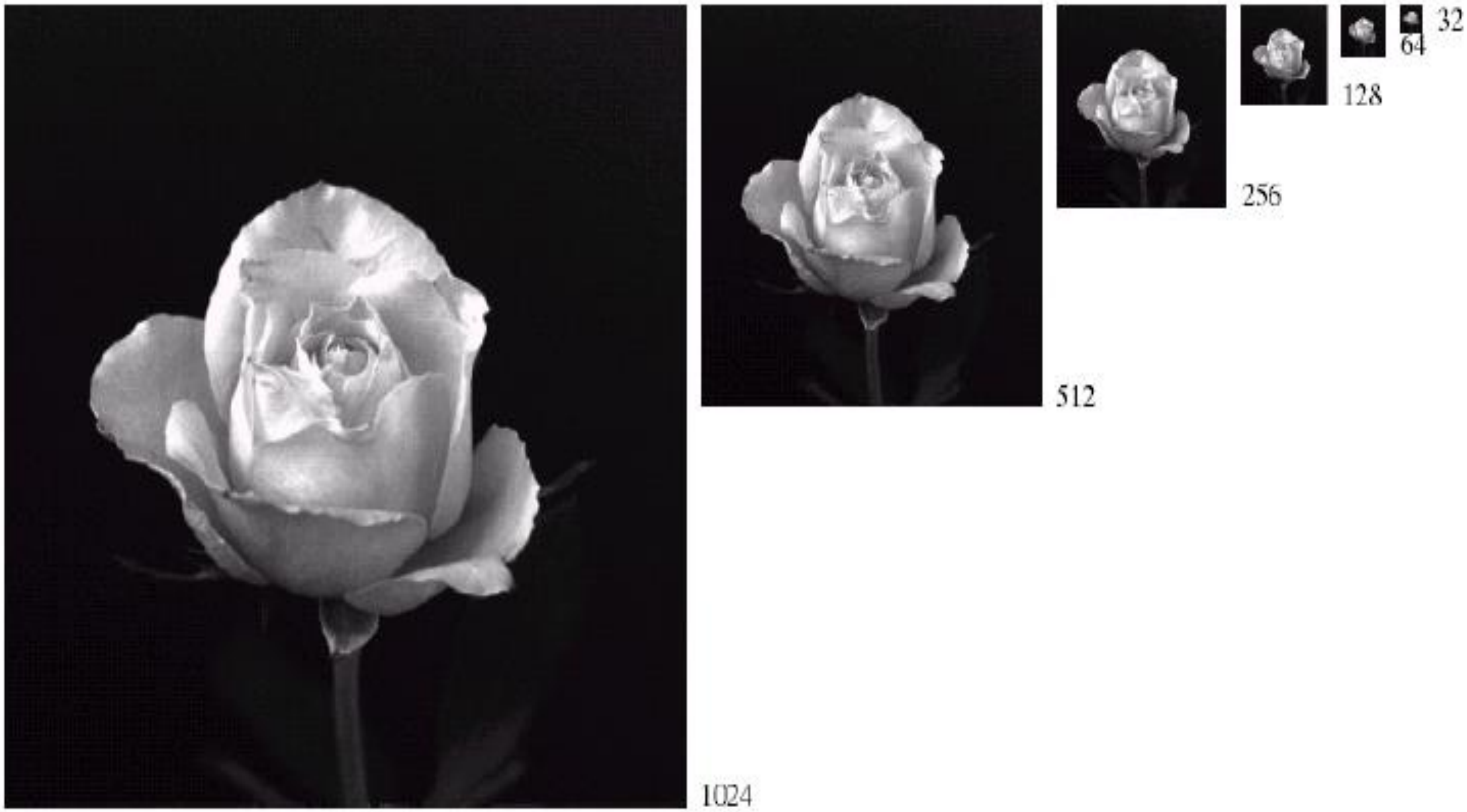
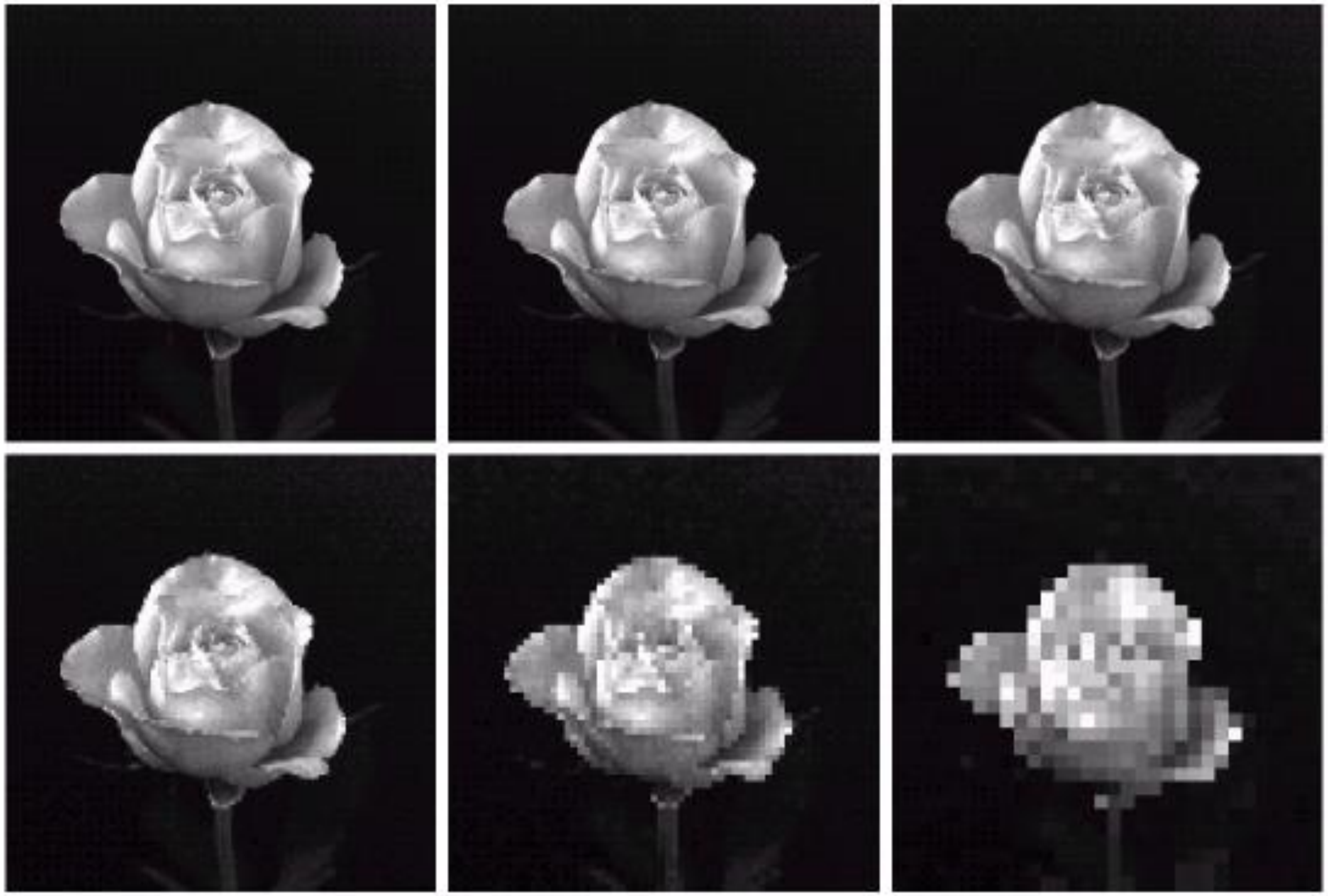


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

Image Resolution (cont.)



a b c
d e f

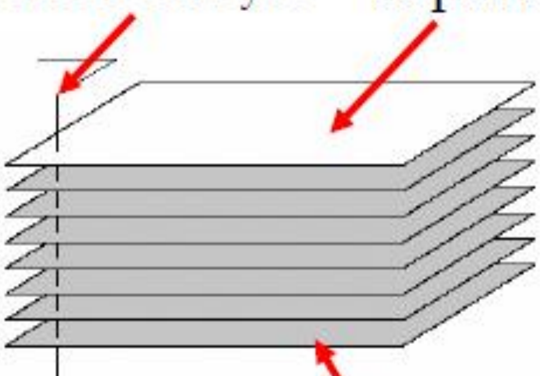
FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

Bitplanes

Original 8bits/pixel

Bitplane 7

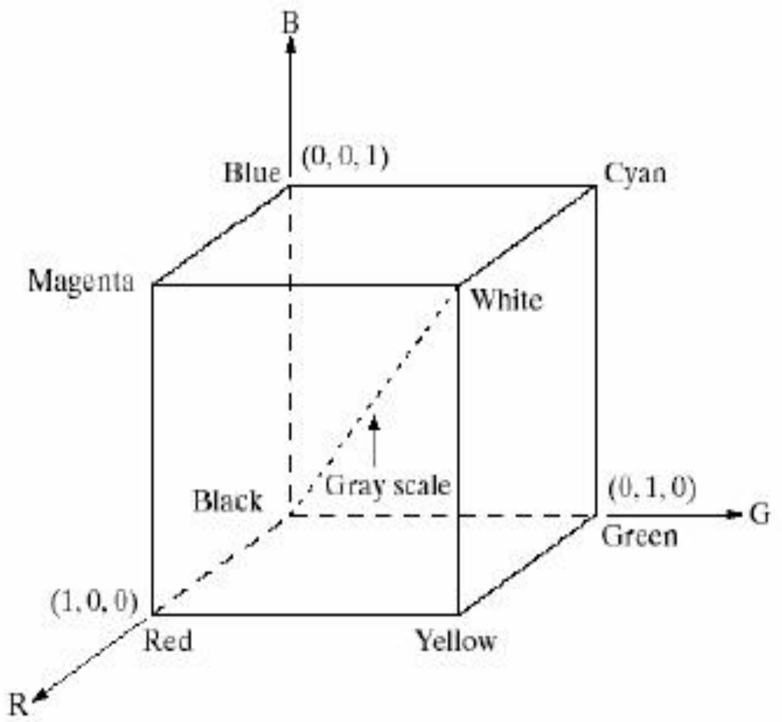
Bitplane 6

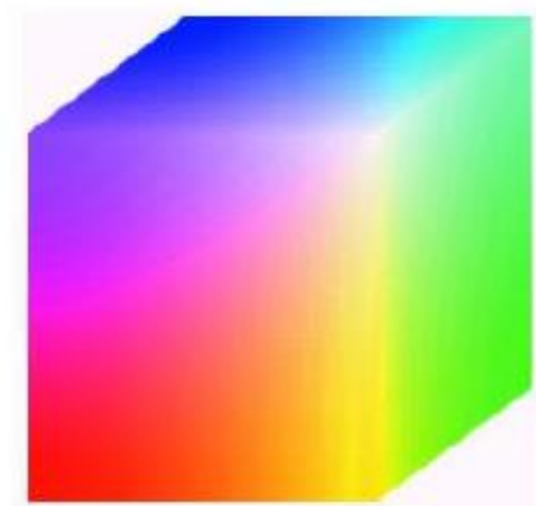


Bitplane 5

Bitplane 4

Color: RGB Cube

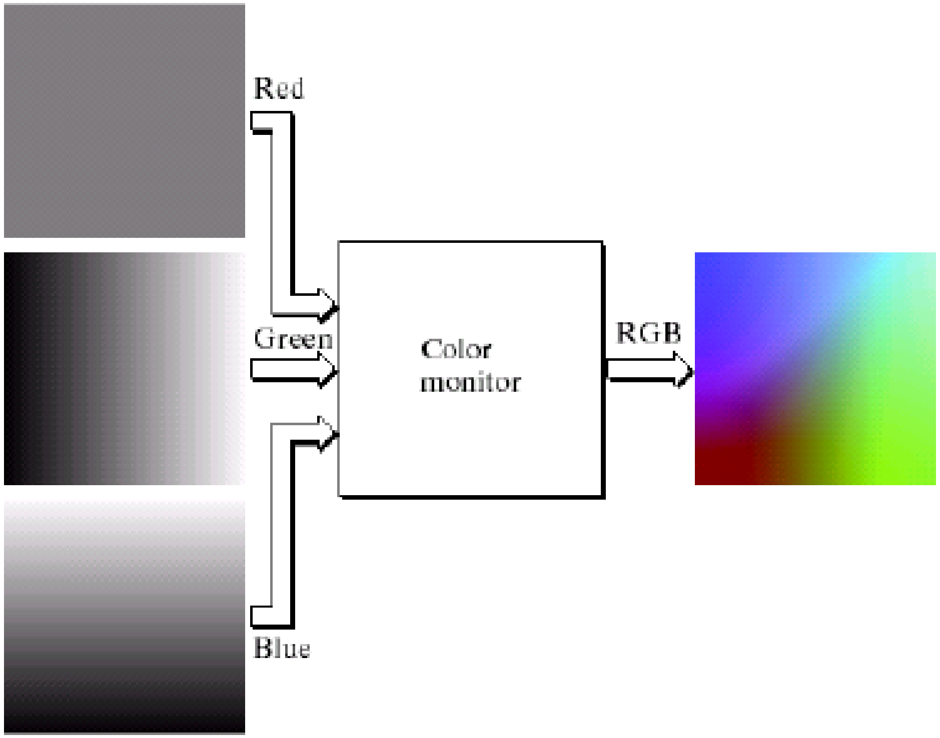


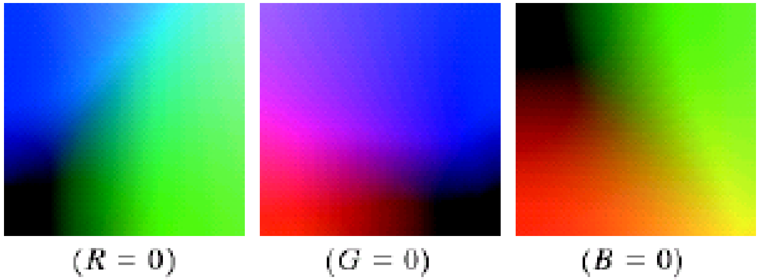


Color: RGB Representation

a
b

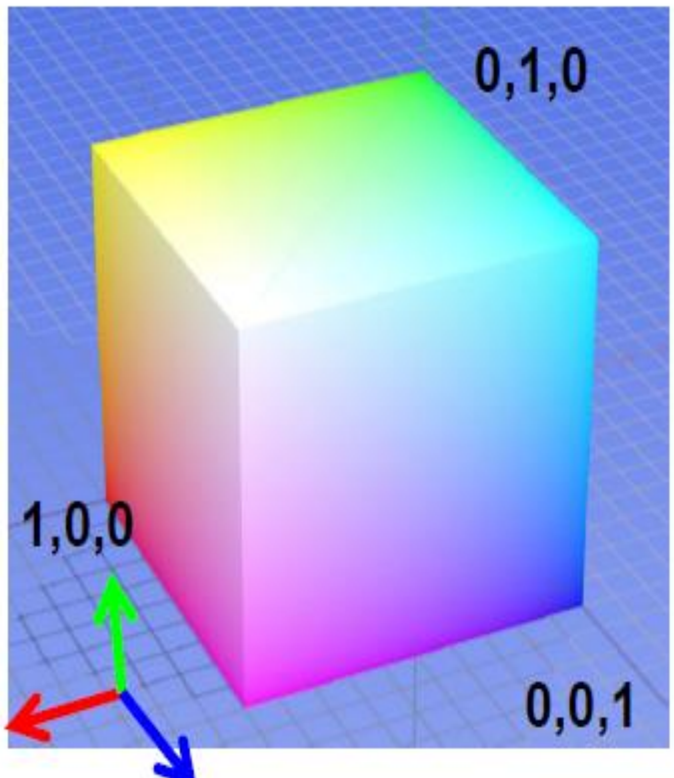
FIGURE 6.9
(a) Generating the RGB image of the cross-sectional color plane (127, *G*, *B*).
(b) The three hidden surface planes in the color cube of Fig. 6.8.

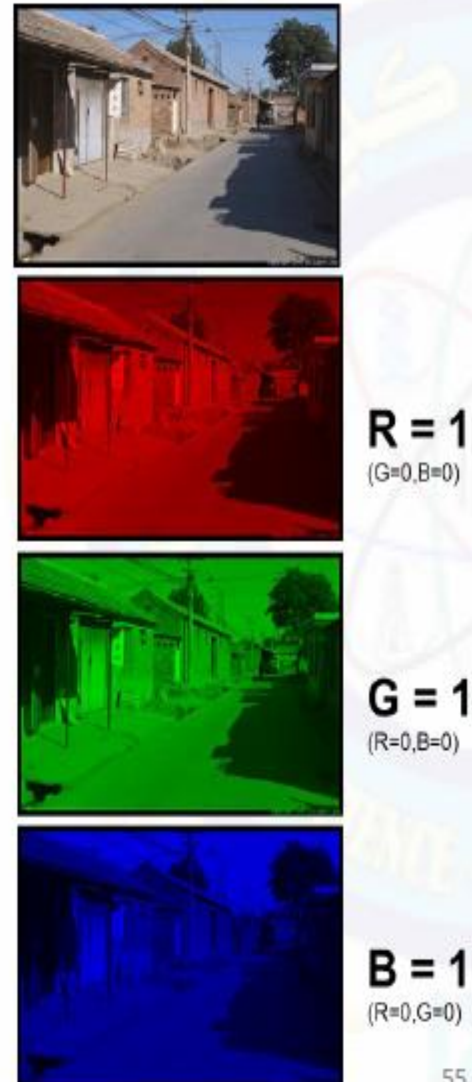




Color Sensing in Camera (RGB):

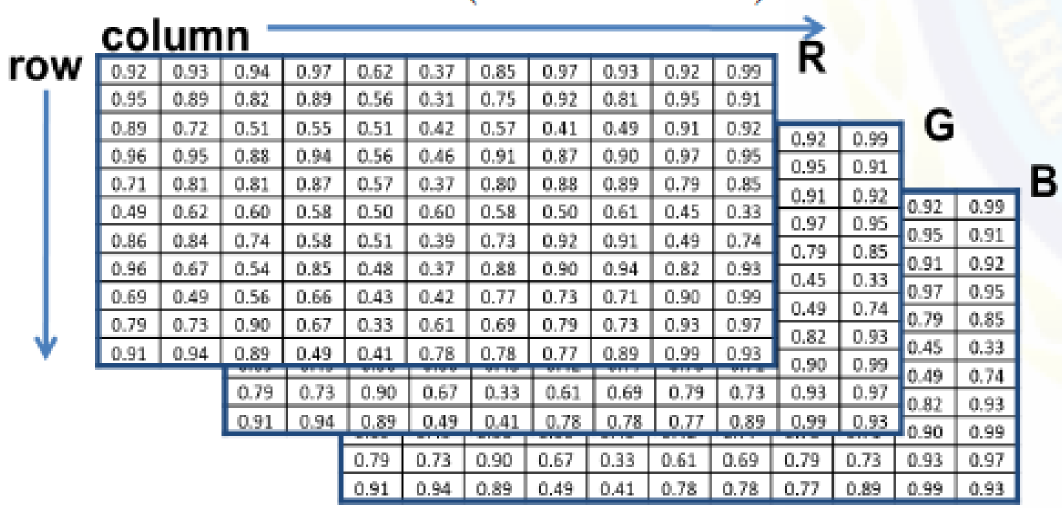
- Default color space:
  - Any color = r*R + g*G + b*B.
  - Strongly correlated channels.
  - Non-perceptual.





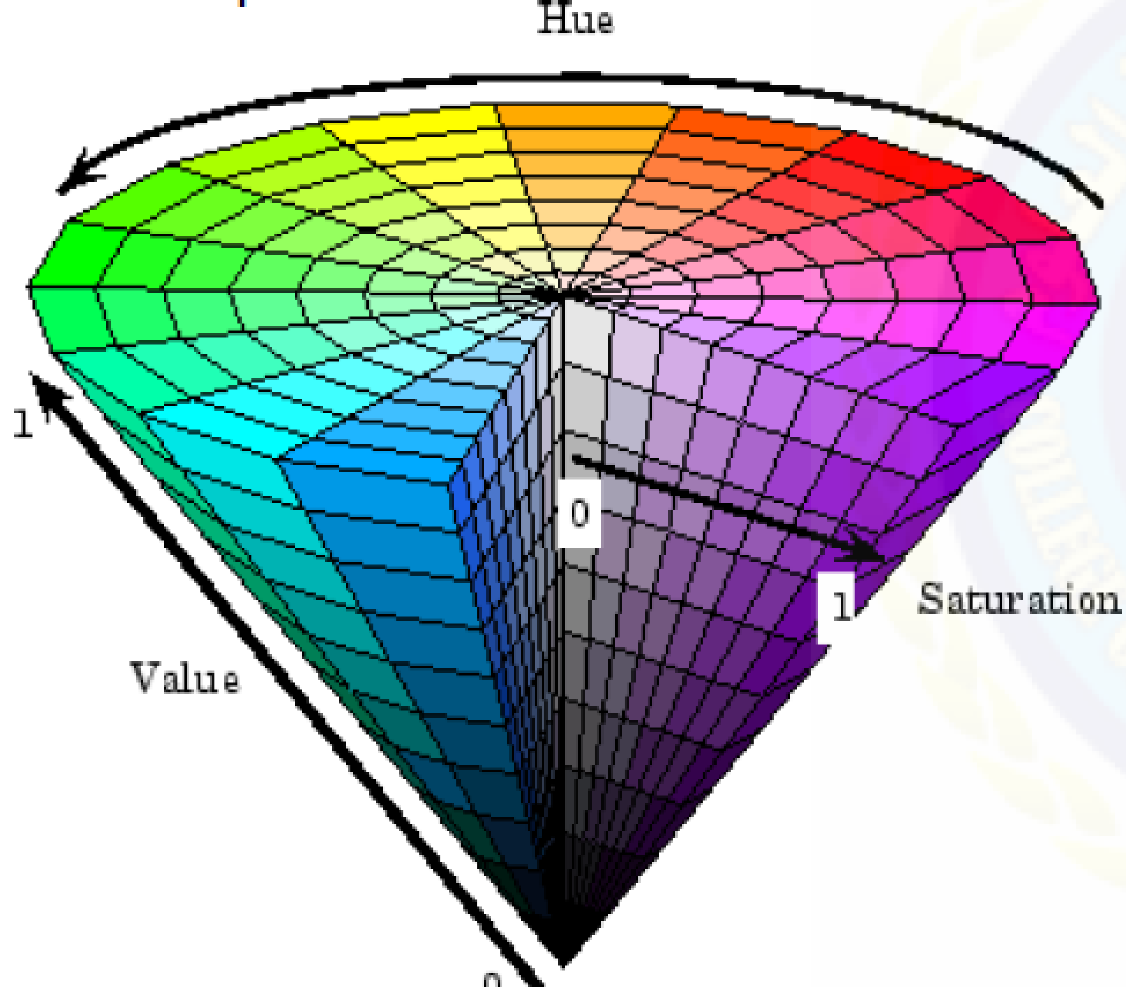
Color Image (RGB):

- Images represented as a matrix.
- Suppose we have a NxM RGB image called "im".
  - im(1,1,1) = top-left pixel value in R-channel.
  - im(y, x, b) = y pixels down, x pixels to right in the bth channel.
  - im(N, M, 3) = bottom-right pixel in B-channel
- imread(filename) returns a uint8 image (values 0 to 255).
  - Convert to double format (values 0 to 1) with im2double



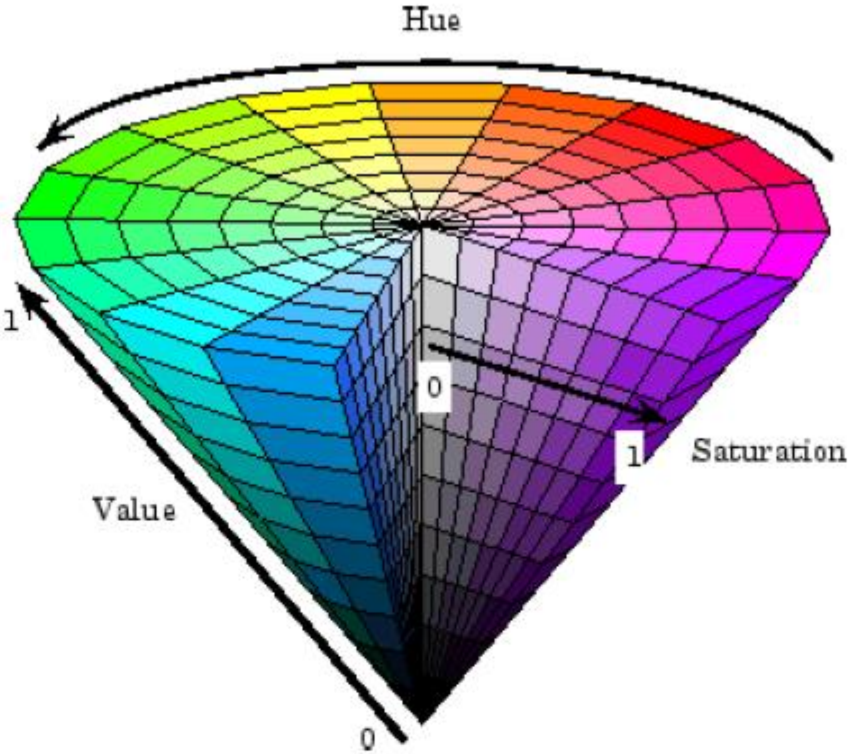
Color spaces: HSV:

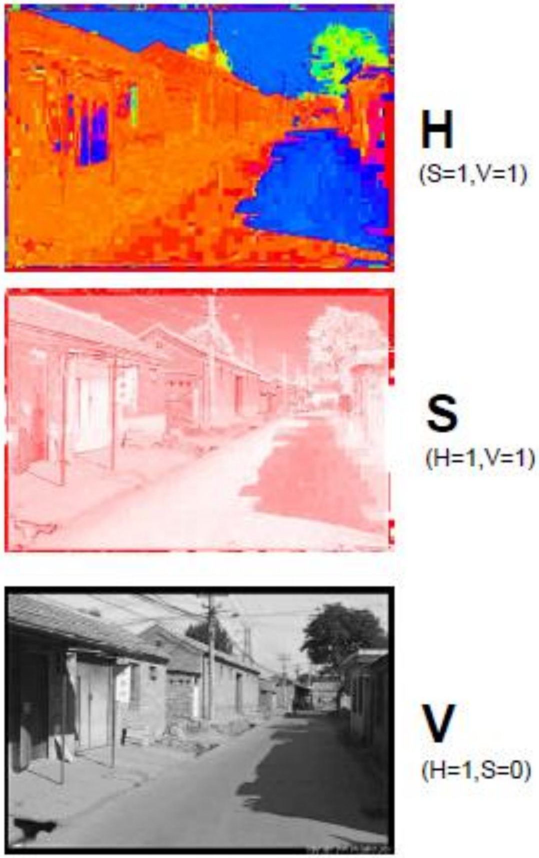
- Intuitive color space:



Color spaces: HSV

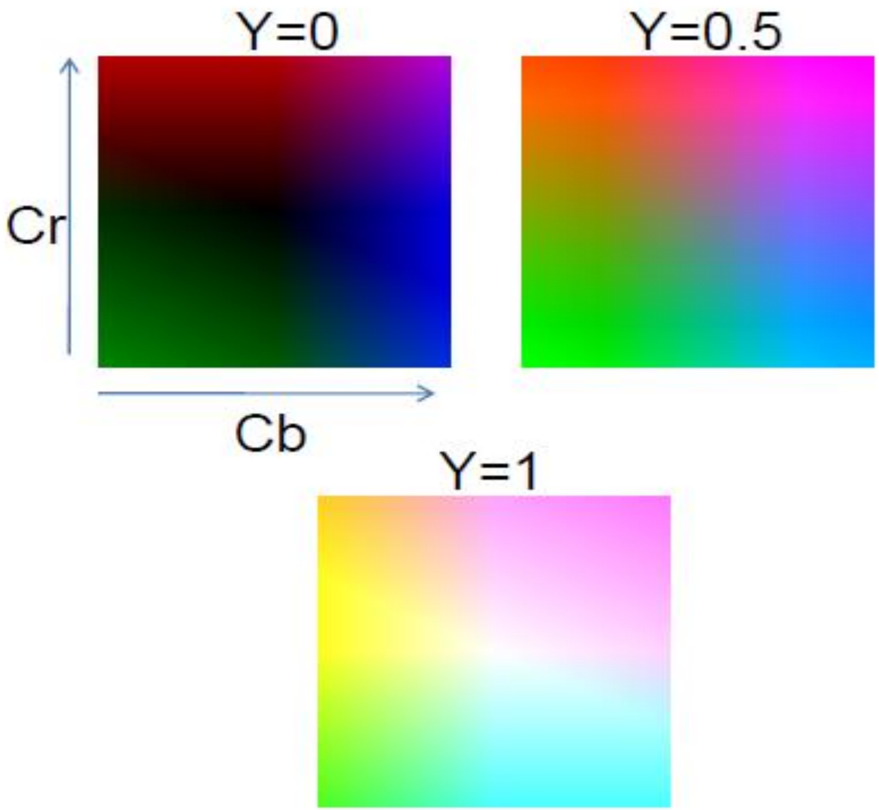
Intuitive color space





Color spaces: YCbCr

Fast to compute, good for compression, used by TV



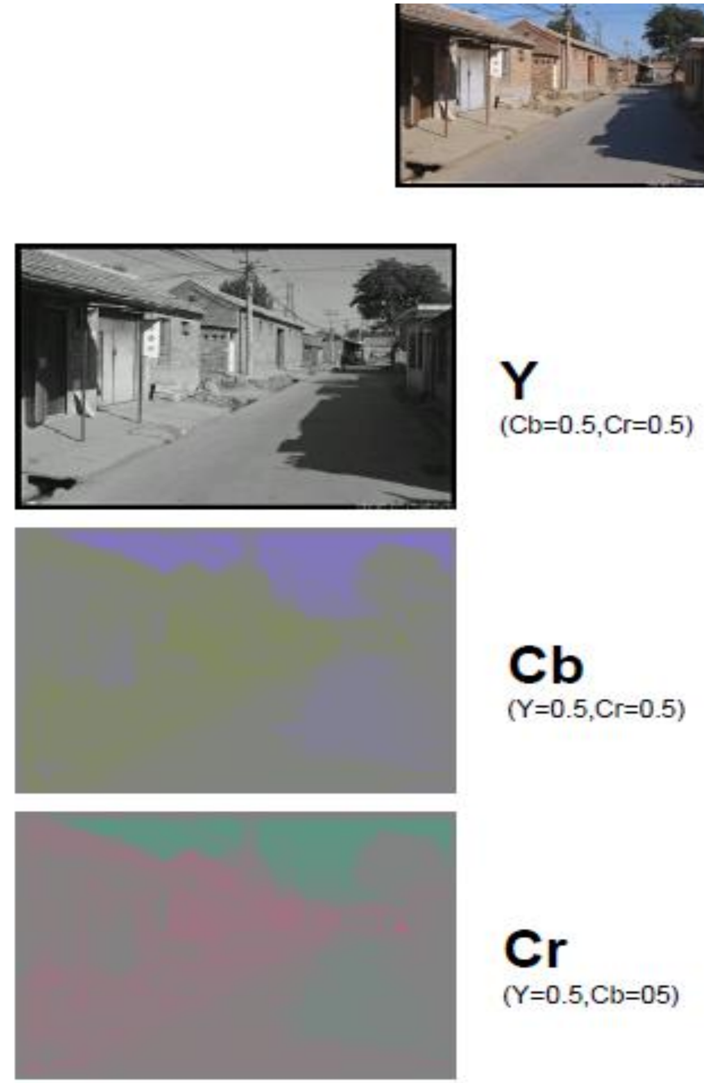
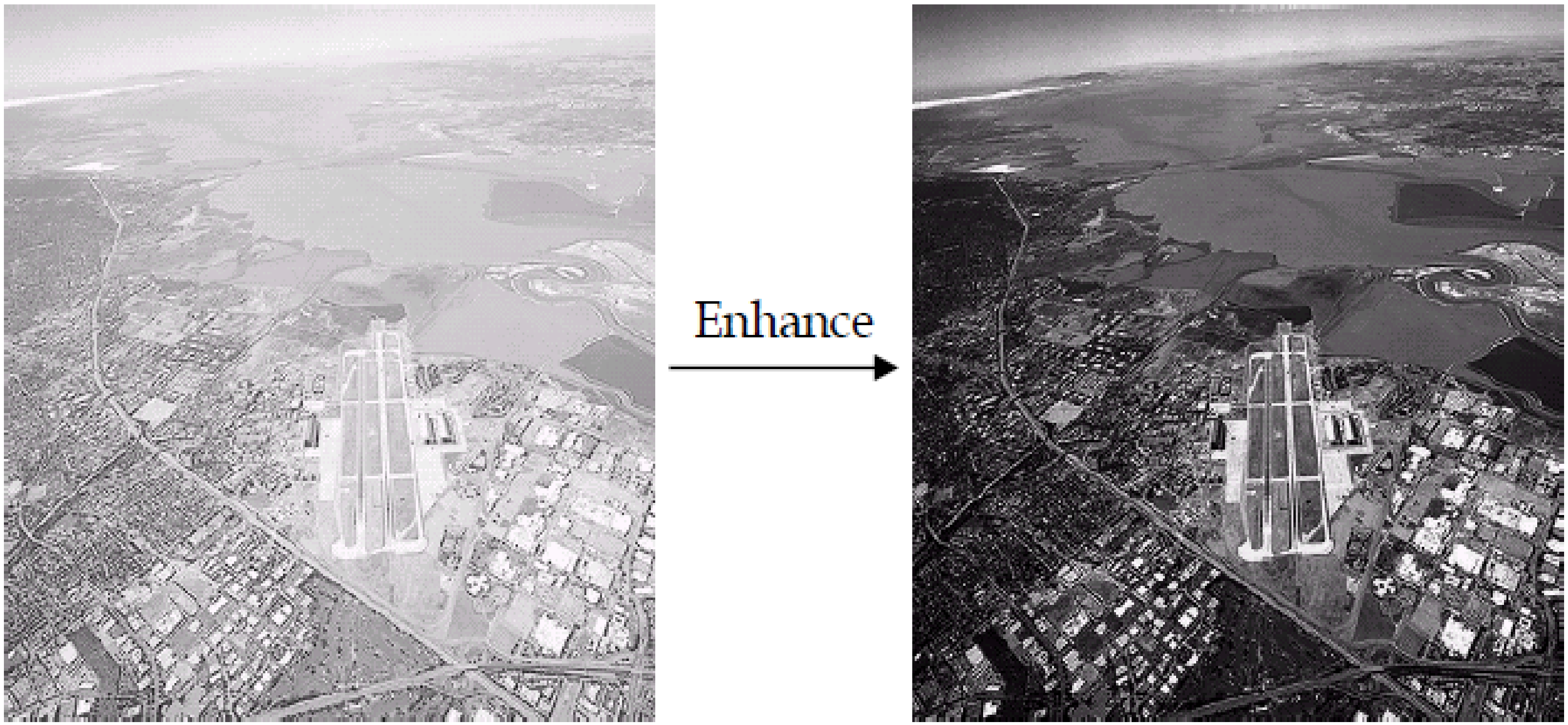


Image Enhancement



Demo from opencv: demhist.exe

Image Denoising

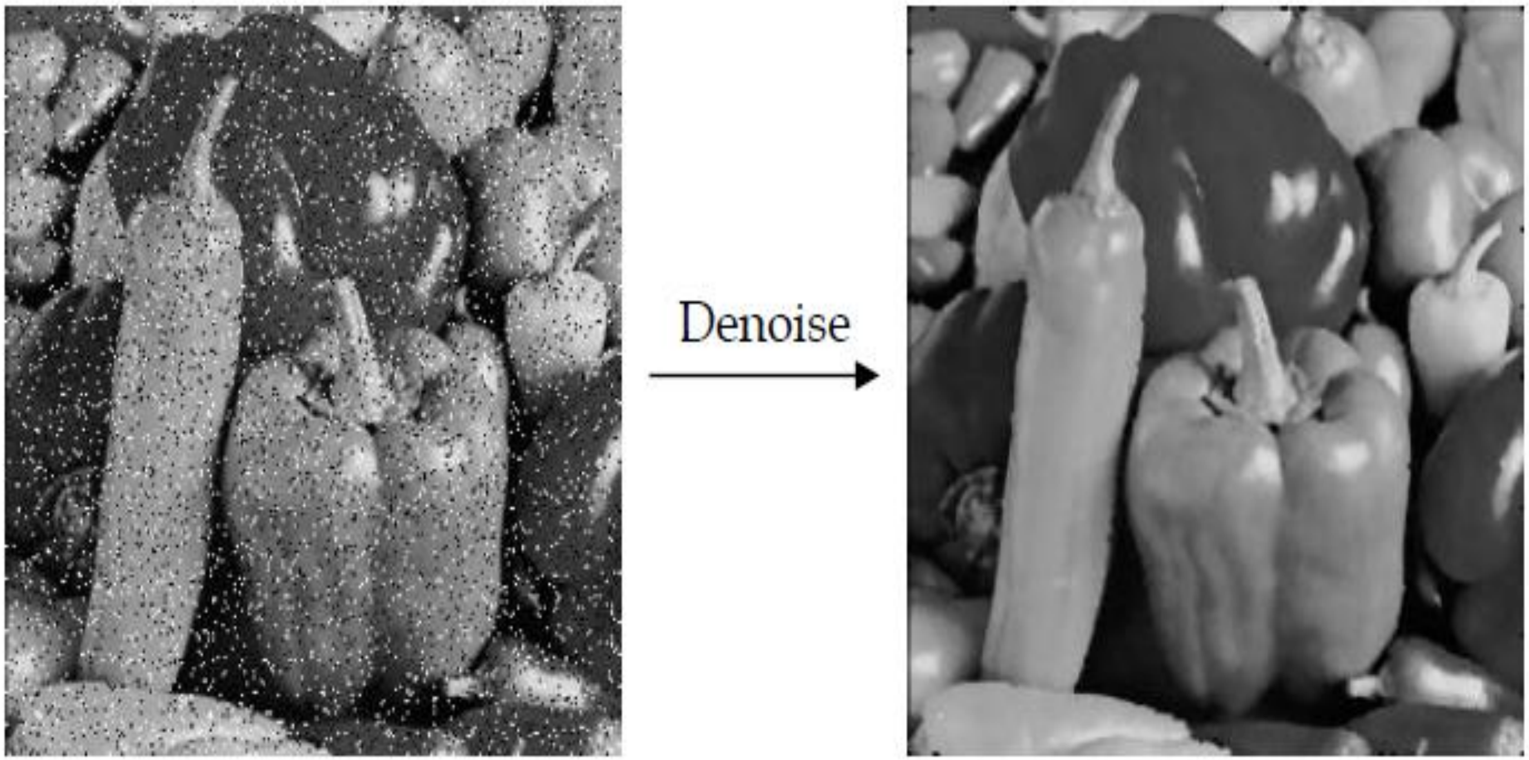
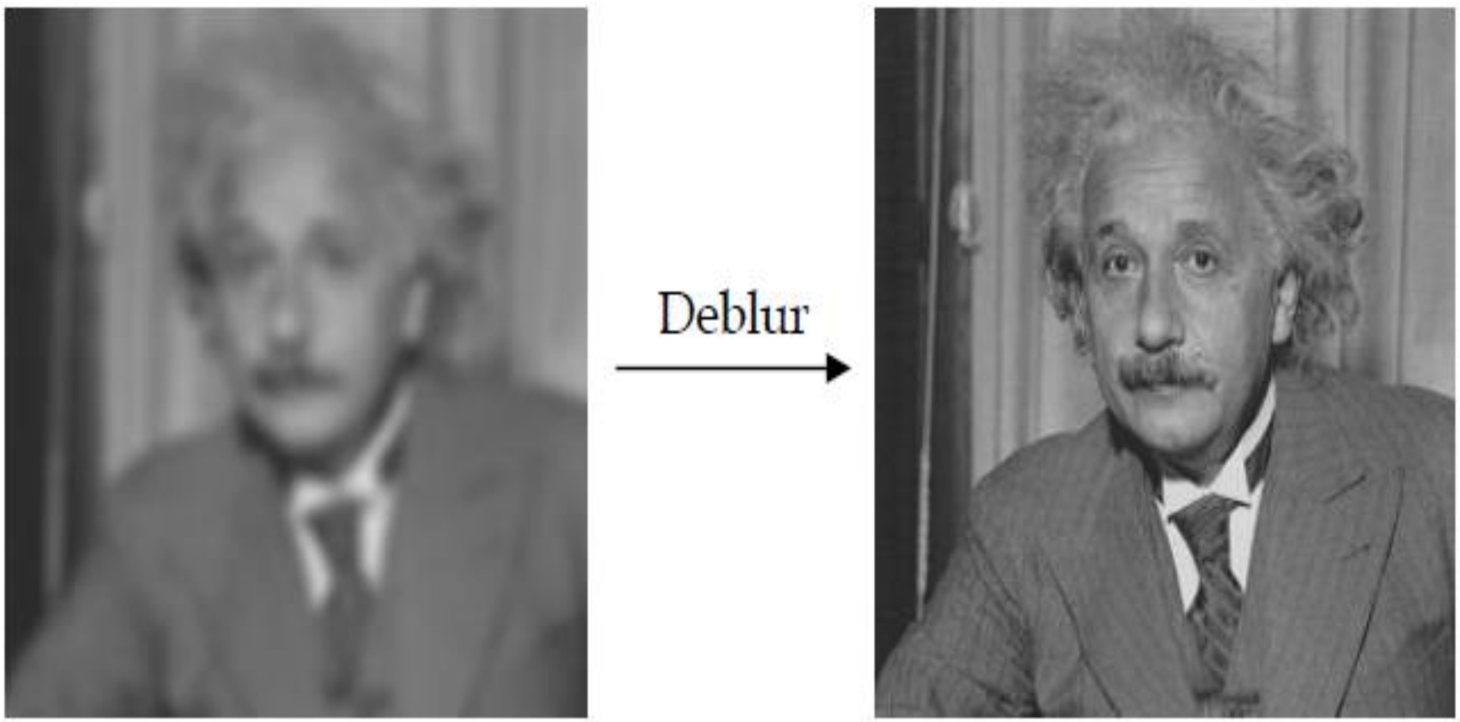
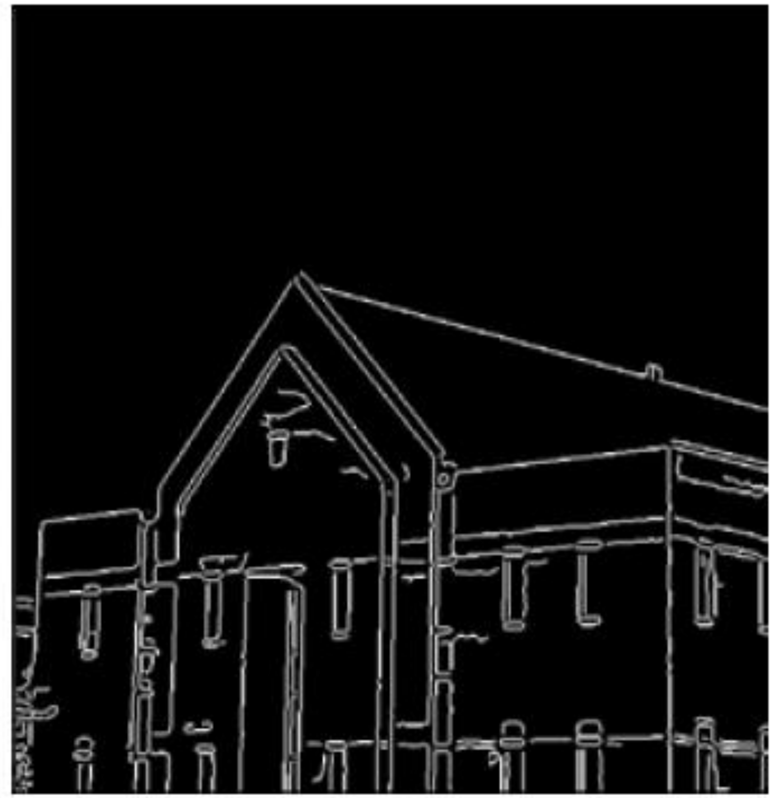


Image Deblurring



Edge Detection



Demo from opencv: edge.exe

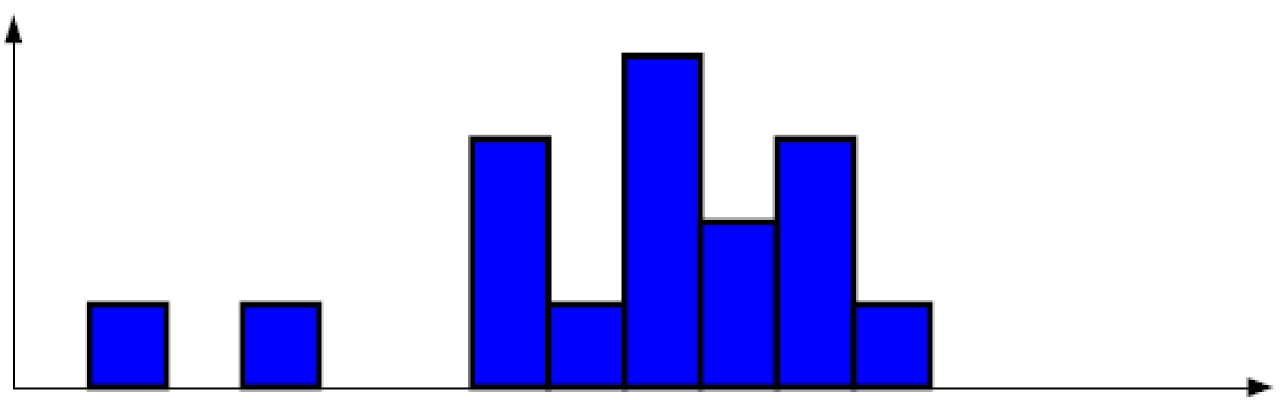
Intensity Histogram

- **Example**

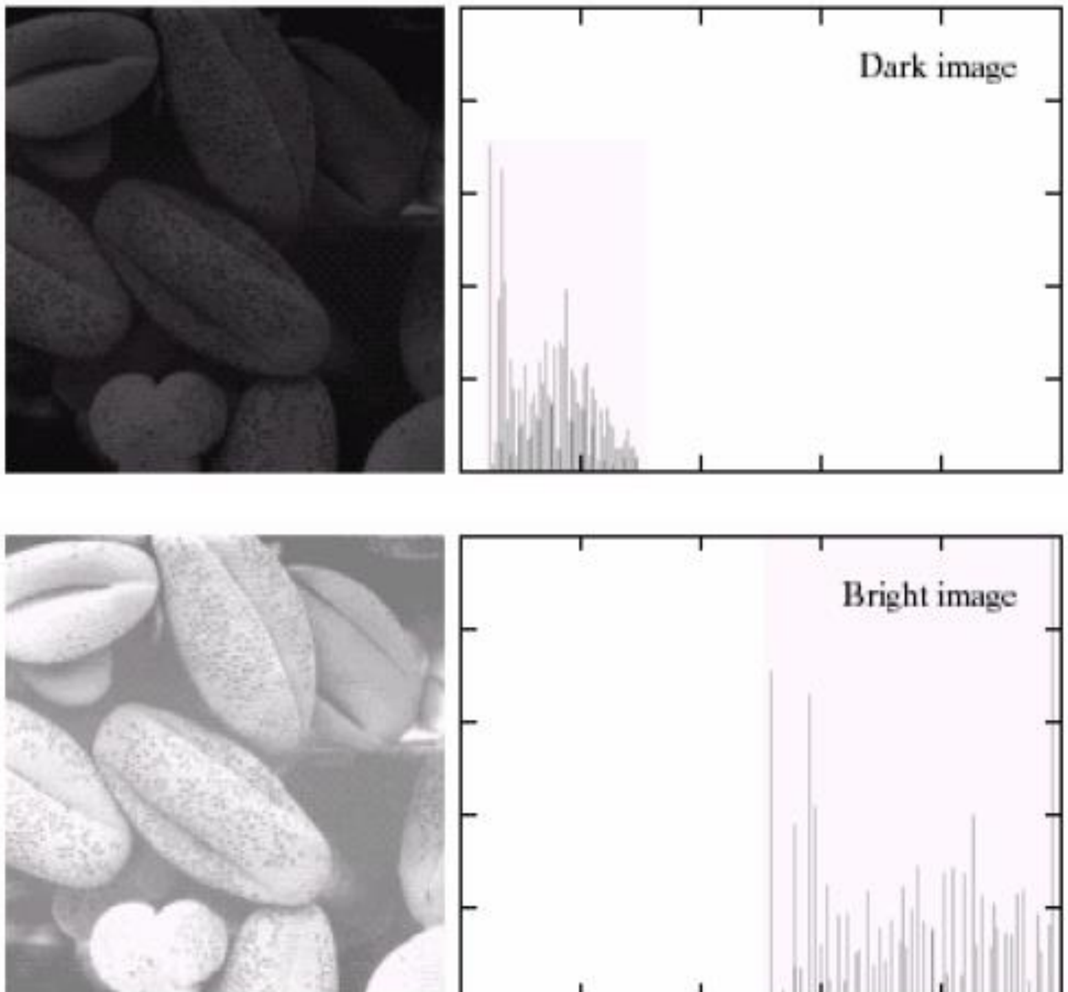
  a 4x4, 4bits/pixel image →

1	8	6	6
6	3	11	8
8	8	9	10
9	10	10	7

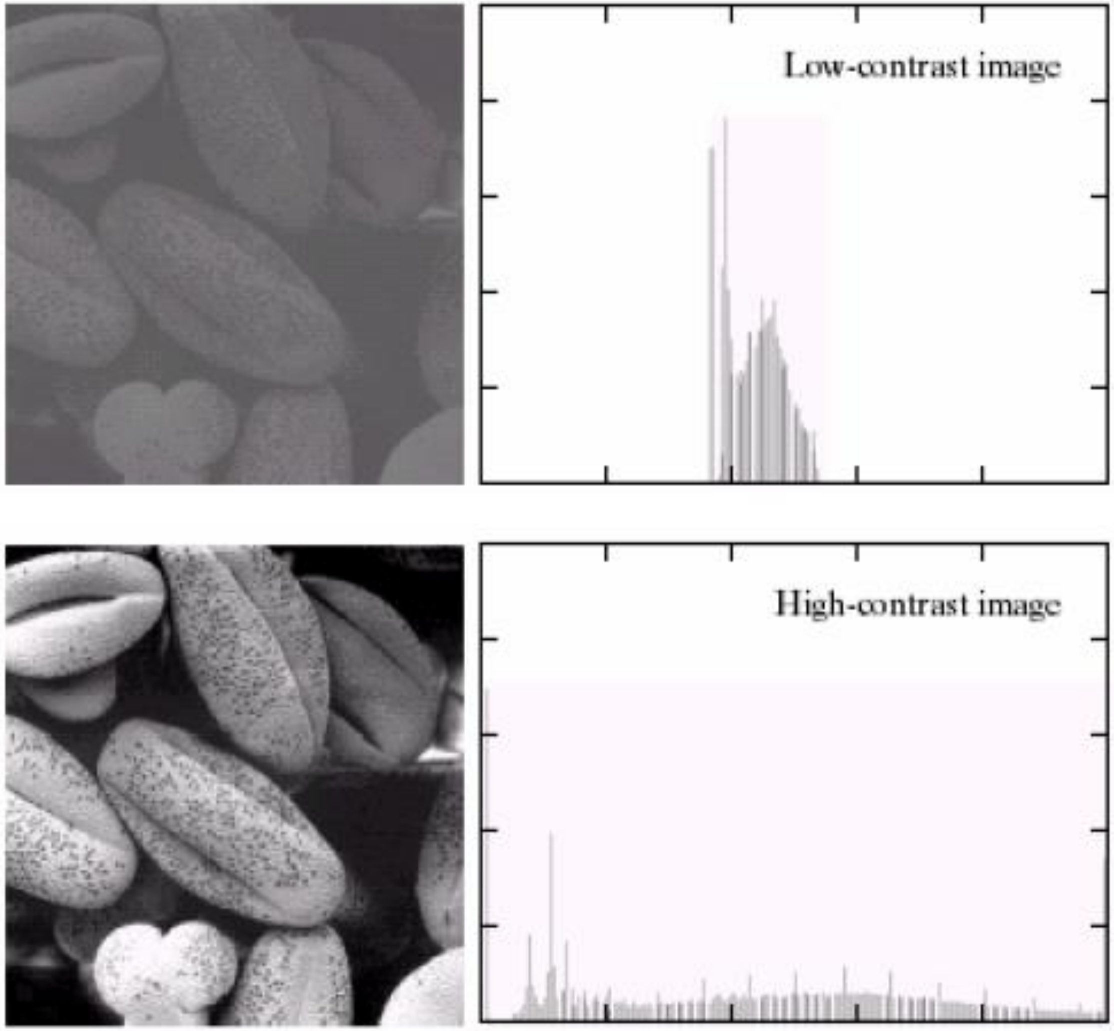
k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
H(k)	0	1	0	1	0	0	3	1	4	2	3	1	0	0	0	0



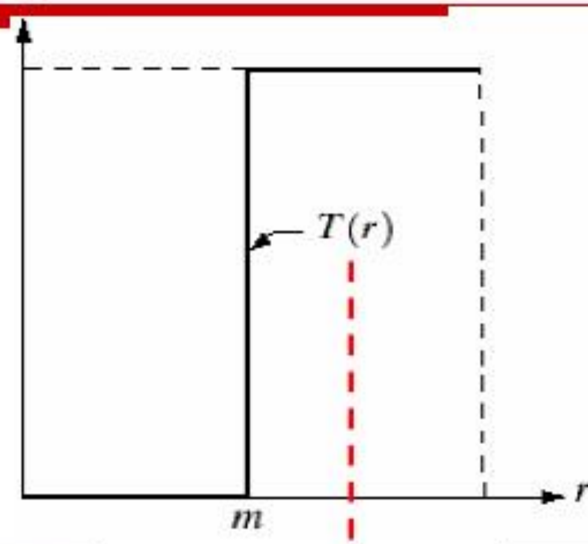
Intensity Histogram (cont.)



Intensity Histogram (cont.)

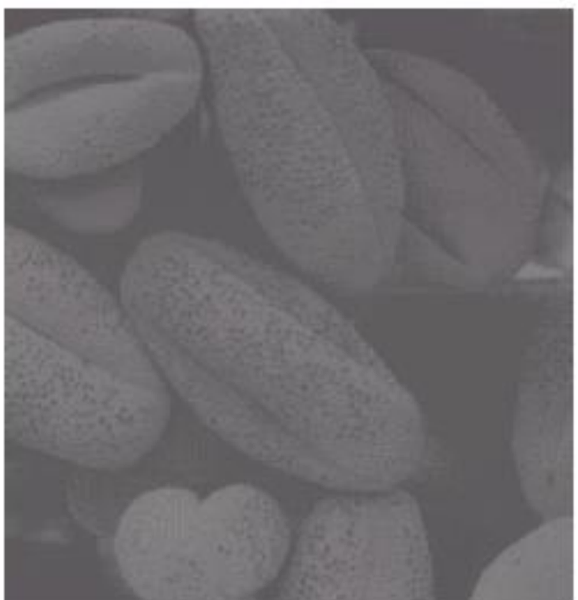


Thresholding

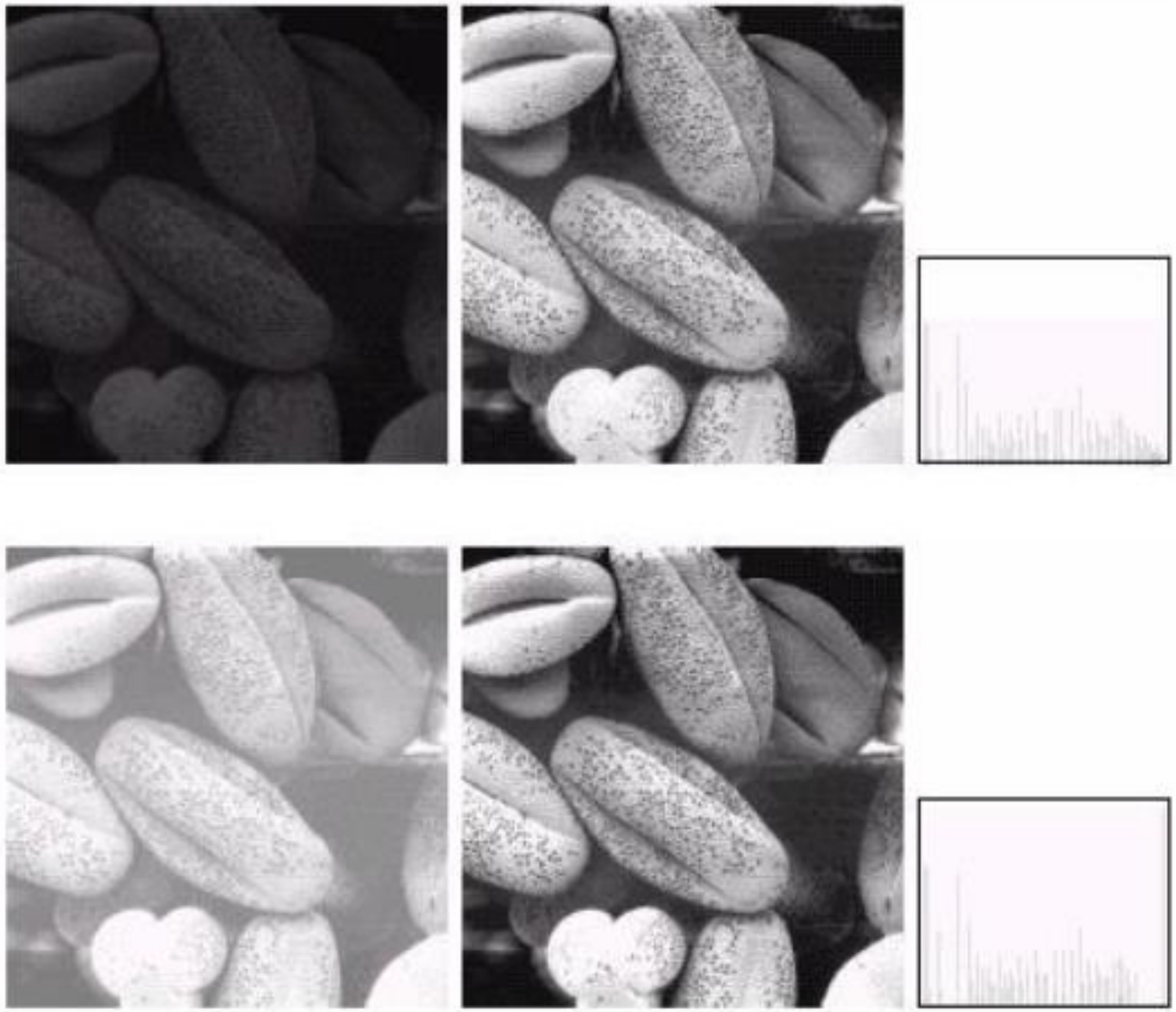


$$s = \begin{cases} 0 & \text{if } r \le m \\ c & \text{if } r > m \end{cases}$$

m : threshold



Histogram Equalization

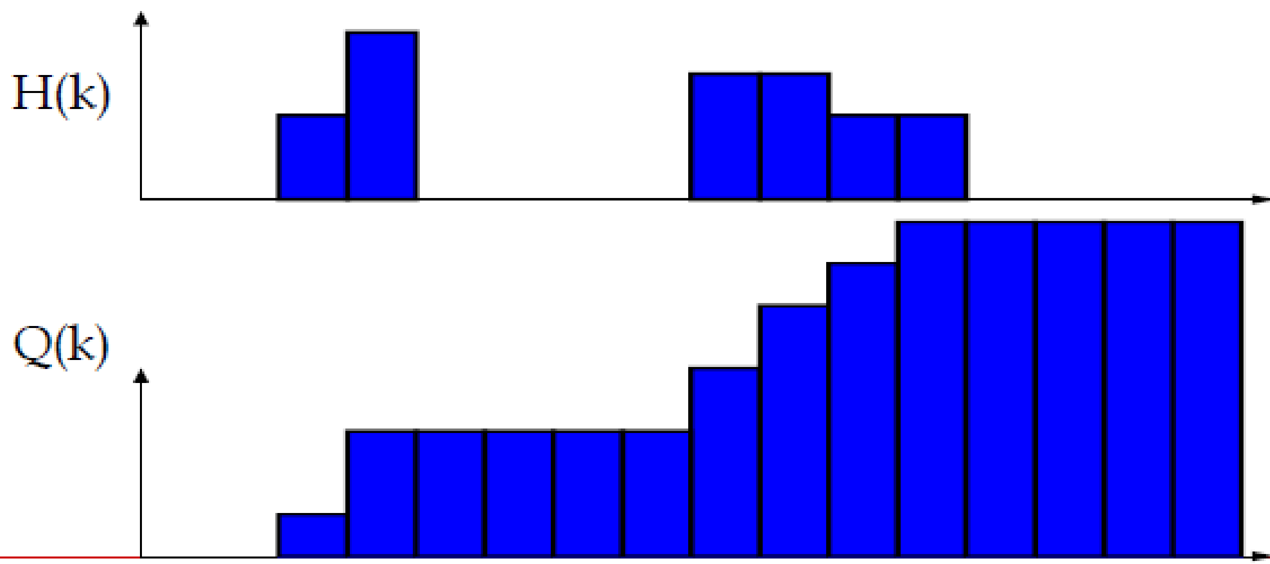


The left images: dark light; the right images:

Cumulative Histogram

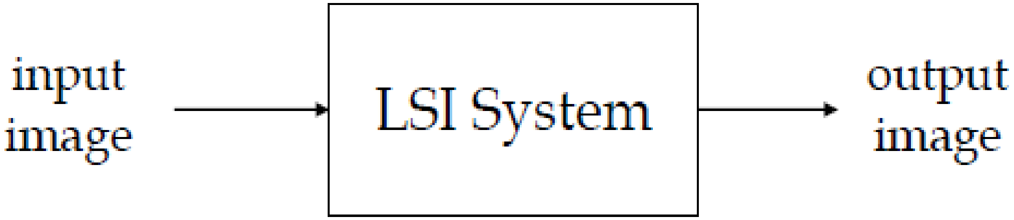
2	8	9	9
2	3	10	9
8	3	3	11
8	3	10	11

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
H(k)	0	0	2	4	0	0	0	0	3	3	2	2	0	0	0	0
Q(k)	0	0	2	6	6	6	6	6	9	12	14	16	16	16	16	16



Spatial Linear Filtering Systems

- **Linear Shift-Invariant (LSI) System**

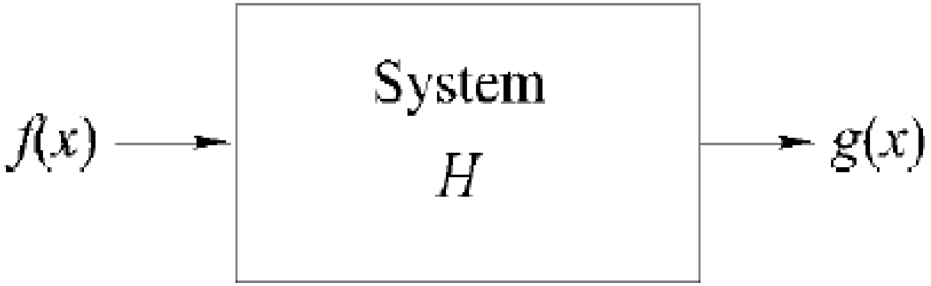


  - Linearity: "things can be added"
  - Shift-invariance: "things do not change over space"

- **Filtering operation with LSI System**
  - If performed in spatial domain → Convolution
  - If performed in frequency domain → Multiplication (Convolution Theorem)

What is a system?

With reference to the following figure, we define a ***system*** as a unit that converts an input function $f(x)$ into an output (or response) function $g(x)$, where x is an independent variable, such as time or, as in the case of images, spatial position. We assume for simplicity that x is a continuous variable, but the results that will be derived are equally applicable to discrete variables.



What is a system? (cont.)

It is required that the system output be determined completely by the input, the system properties, and a set of initial conditions. From the figure in the previous page, we write

$$g(x) = H[f(x)]$$

where H is the ***system operator***, defined as a mapping or assignment of a member of the set of possible outputs $\{g(x)\}$ to each member of the set of possible inputs $\{f(x)\}$. In other words, the system operator completely characterizes the system response for a given set of inputs $\{f(x)\}$.

Linear system

An operator H is called a *linear operator* for a class of inputs $\{f(x)\}$ if

$$\begin{aligned} H[a_i f_i(x) + a_j f_j(x)] &= a_i H[f_i(x)] + a_j H[f_{ji}(x)] \\ &= a_i g_i(x) + a_j g_j(x) \end{aligned}$$

for all $f_i(x)$ and $f_j(x)$ belonging to $\{f(x)\}$, where the a's are arbitrary constants and

$$g_i(x) = H[f_i(x)]$$

is the output for an arbitrary input $f_i(x) \in \{f(x)\}$.

2D Convolution

$$x(m,n) \longrightarrow \boxed{h(m,n)} \longrightarrow y(m,n)$$

$$y(m,n) = \sum_{k,l=-\infty}^{\infty} h(k,l)\,x(m-k,n-l) = h(m,n) \otimes x(m,n)$$

$$y(m,n) = \sum_{k,l=-\infty}^{\infty} h(m-k,n-l)\,x(k,l) = x(m,n) \otimes h(m,n)$$

$h(m, n)$ → impulse response (spatial linear filter)

x(m, n) → input image

y(m, n) → output image

Spatial Neighborhood

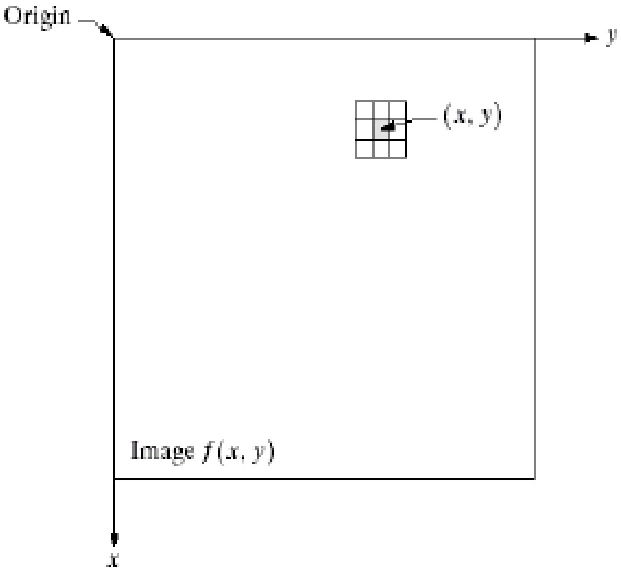
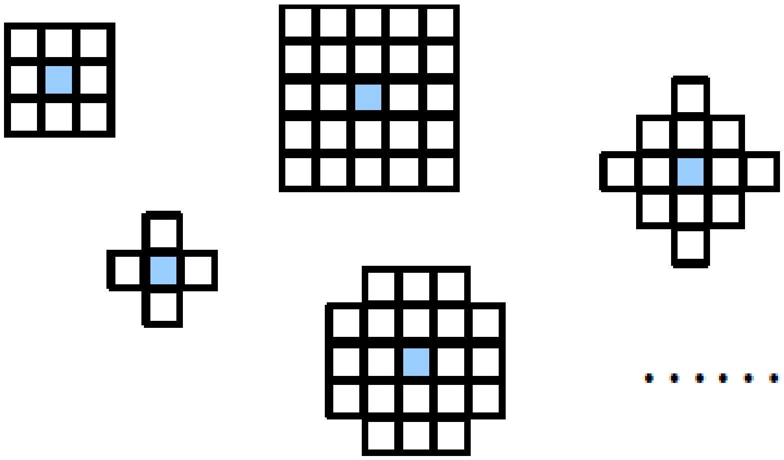


FIGURE 3.1 A 3 × 3 neighborhood about a point (x, y) in an image.

From Gonzalez & Woods

choices of neighborhood:



Masks, Windows, Filters and the Impulse Responses

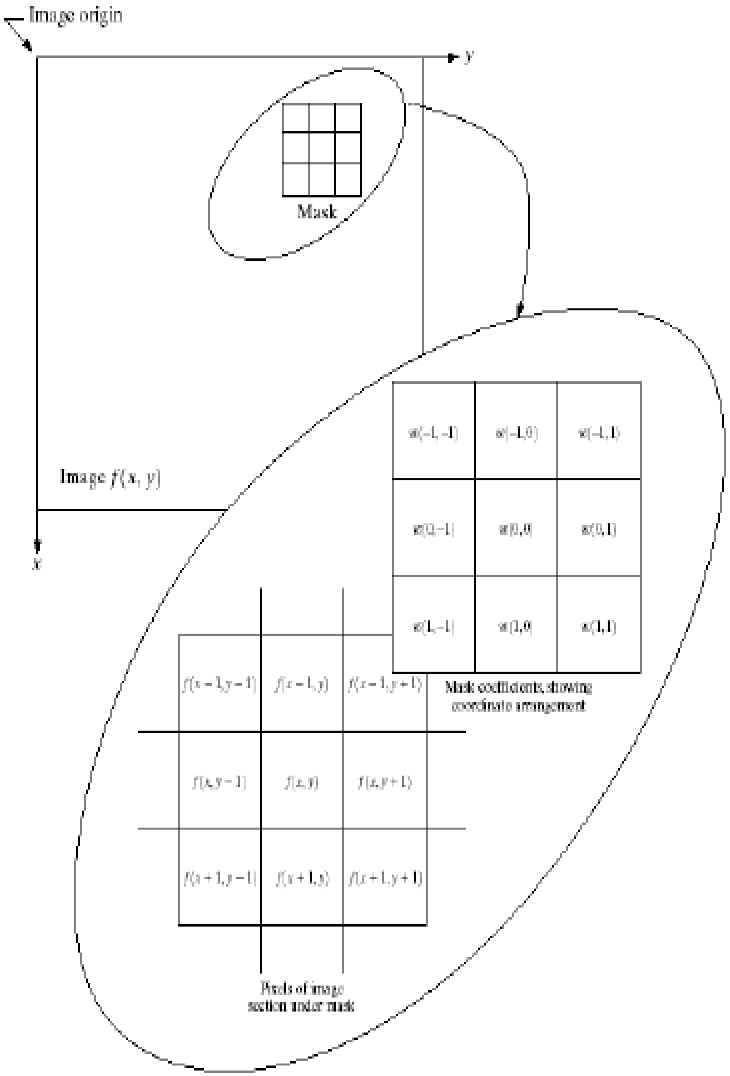


FIGURE 3.32 The mechanics of spatial filtering. The magnified drawing shows a 3 × 3 mask and the image section directly under it; the image section is shown displaced out from under the mask for ease of readability.

- **Spatial LSI Filter:**

  impulse response constrained within a local neighborhood

- **"Filter"**

  "Mask"

  "Window"

  "Impulse Response"

  often used interchangeably for LSI

Applications of Spatial Linear Filtering

- Image Smoothing
- Image Enhancement
- Image Restoration
  - Image de-noising
  - Image de-blurring
- Edge Detection
- Filter Bank

Image Smoothing: Average Filters

- **Average Filter**

$$h(m,n) = \frac{1}{N^2}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$$

NxN: filter size

noisy

PSNR=20.2dB
noise std = 25

smoothed

PSNR=23.8dB
3x3 window

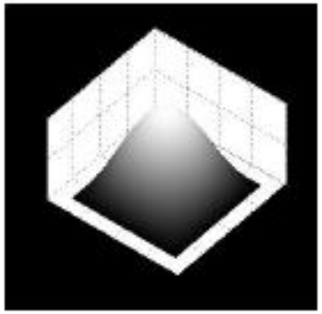
smoothed

PSNR=22.0dB
5x5 window

Image Smoothing: Gaussian Filters

- **Gaussian Filter** $h(m,n) = \frac{1}{Z}\exp\left[-\frac{m^2+n^2}{2\sigma^2}\right]$

$$-N \le m,n \le N$$



noisy | smoothed | smoothed

PSNR=20.2dB, noise std = 25 | PSNR=24.4dB, σ=1 | PSNR=22.8dB, σ=1.5

Image Smoothing Filter Example

- **Filter** $\frac{1}{6}$

0	1	0
1	2	1
0	1	0

- **Input image: A 4x4, 4 bits/pixel**

1	8	6	6
6	3	11	8
8	8	9	10
9	10	10	7

- **Preprocessing: Zero-padding**

1	8	6	6
6	3	11	8
8	8	9	10
9	10	10	7

→

0	0	0	0	0	0
0	1	8	6	6	0
0	6	3	11	8	0
0	8	8	9	10	0
0	9	10	10	7	0
0	0	0	0	0	0

Image Smoothing Filter Example

- **Move mask across the zero-padded image**

$$\frac{1}{6}$$

0	1	0
1	2	1
0	1	0

0	0	0	0	0	0
0	1	8	6	6	0
0	6	3	11	8	0
0	8	8	9	10	0
0	9	10	10	7	0
0	0	0	0	0	0

- **Compute weighted sum**

- **Result:**

2.6	4.3	6.2	4.3
4.0	6.5	8.0	7.2
6.5	7.7	9.5	7.3
6.0	7.8	7.7	5.7

rounding →

3	4	6	4
4	7	8	7
7	8	10	7
6	8	8	6

Sharpening Linear Filters

- **Laplacian** $$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$
  - Zero at border of regions
  - Sensitive to image details

- **Discrete approximation of Laplacian:**

0	-1	0
-1	4	-1
0	-1	0

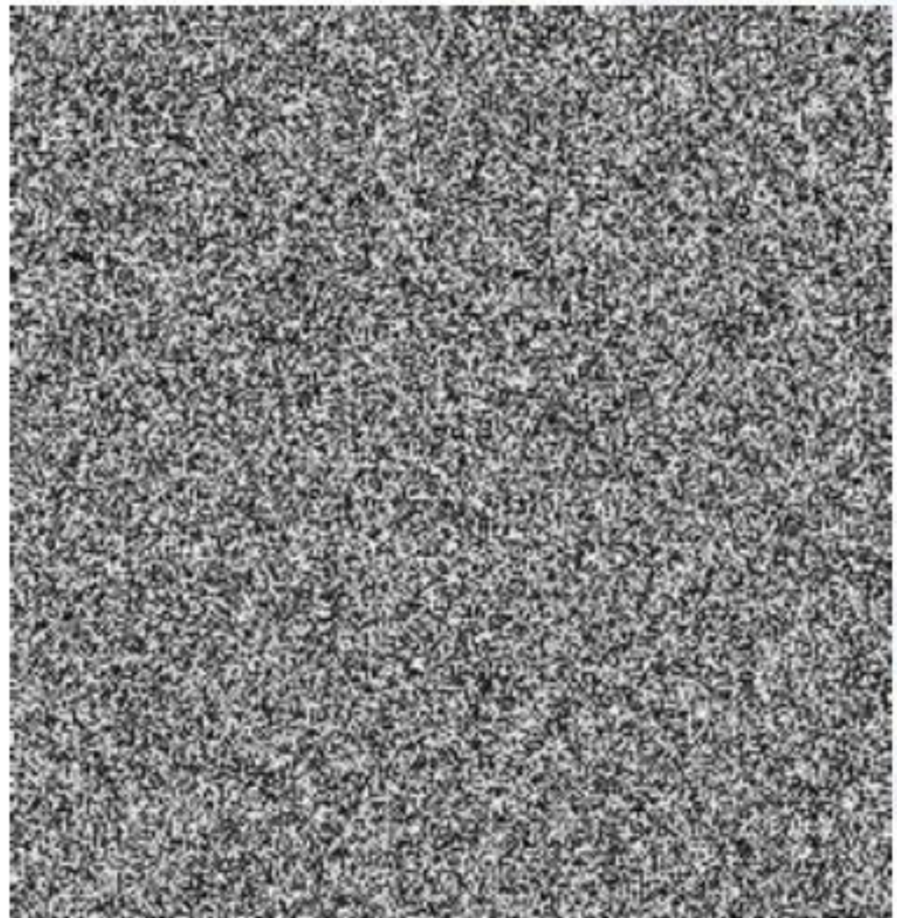
-1	-1	-1
-1	8	-1
-1	-1	-1

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

First MATLAB:

- If we use the following instruction:
  - >> I = rand(256,256);
  - >> imshow(I);



What is each part of an image?

- What does it represent in terms of cameras?
  - Pixel -> picture element

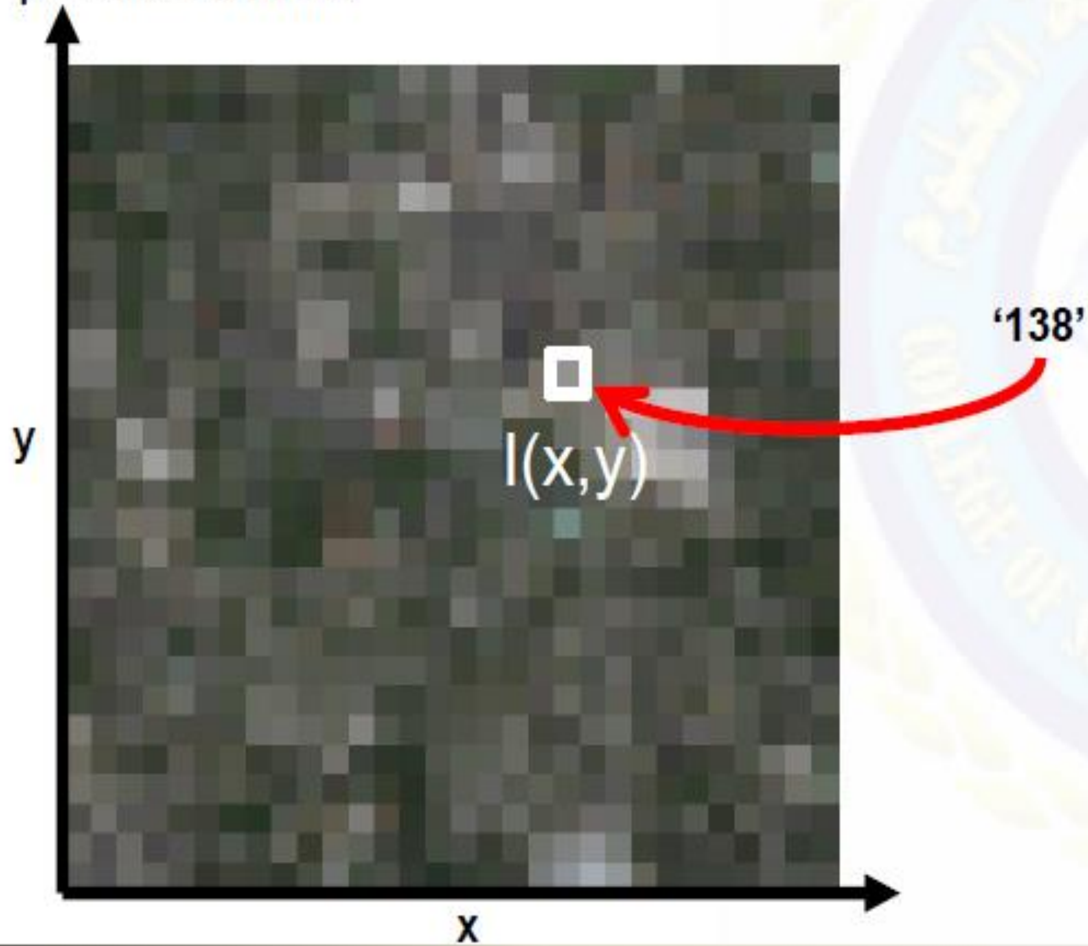


Image as a 2D Sampling of Signal:

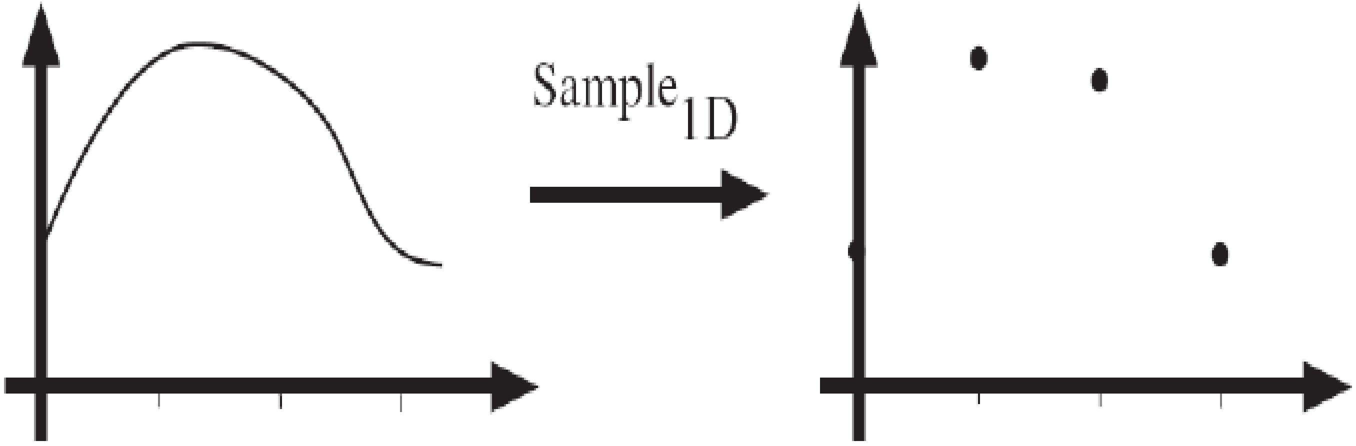
- Signal: function depending on some variable with physical meaning.

- Image: sampling of that function.
  - 2 variables: xy coordinates.
  - 3 variables: xy + time (video).
  - 'Brightness' is the value of the function for visible light.

- Can be other physical values too: temperature, pressure, depth.

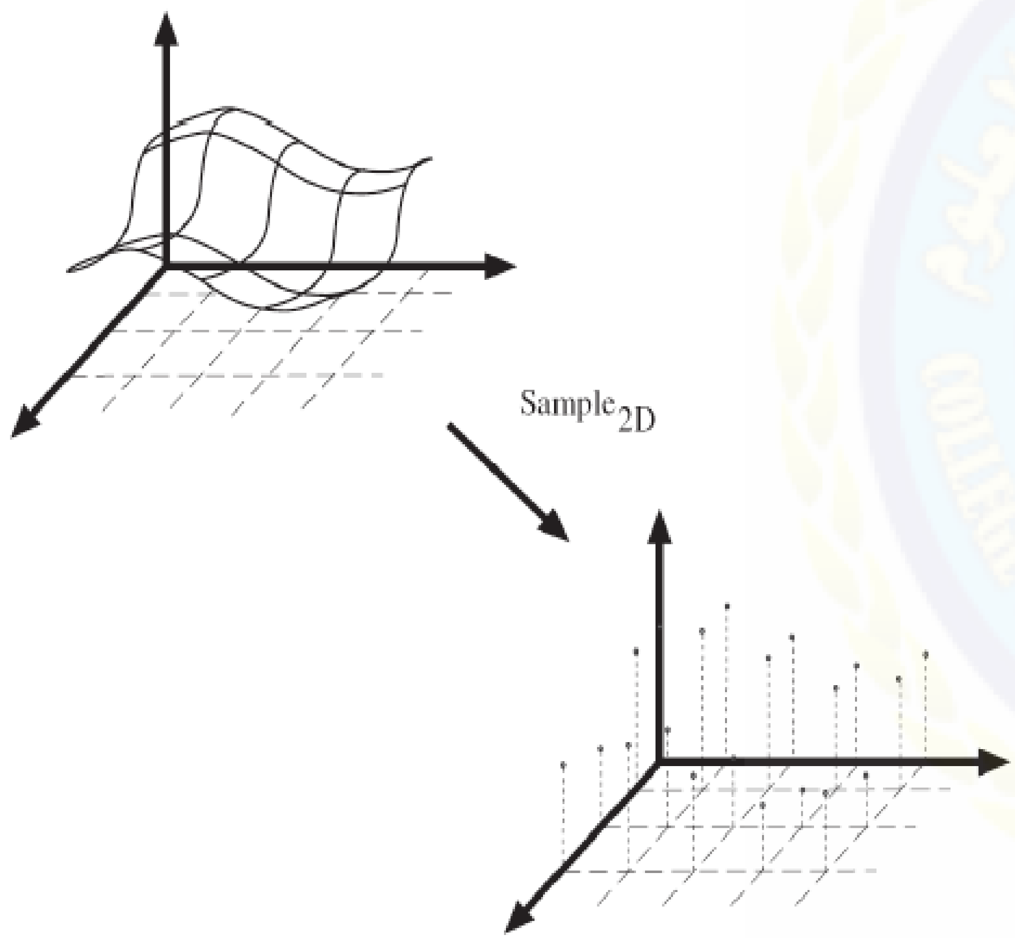
Sampling in 1D:

- Sampling in 1D takes a function and returns a vector whose elements are values of that function at the sample points.



Sampling in 2D:

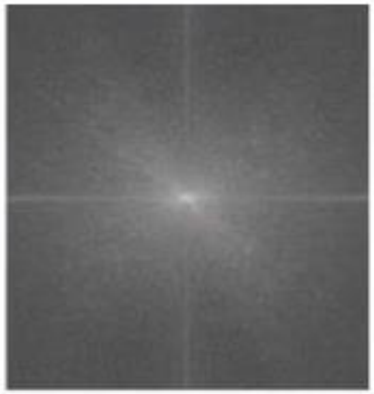
- Sampling in 2D takes a function and returns a matrix.



Fourier Transform:

- Basic Operations (low-high pass filters):

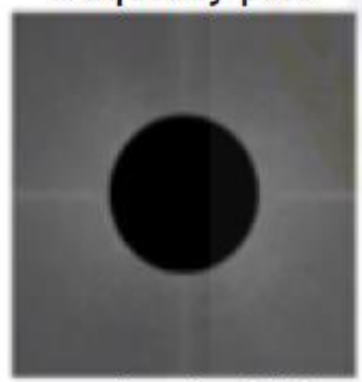
Original image



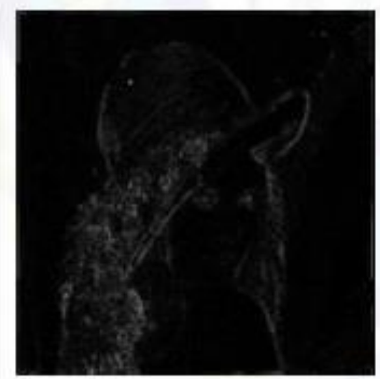
Fourier Transform

Letting the low frequency pass

Output image



Letting the high frequency pass

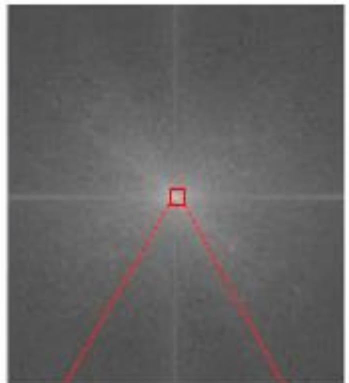


Output image

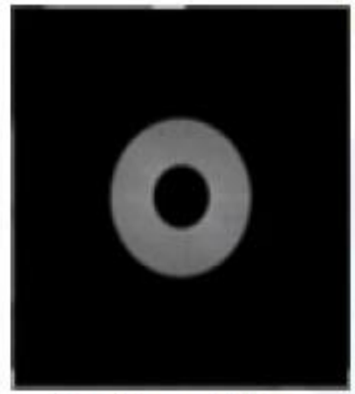
Fourier Transform:

- DC Component:

Original image

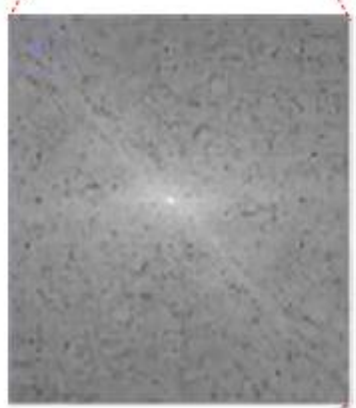


Fourier Transform

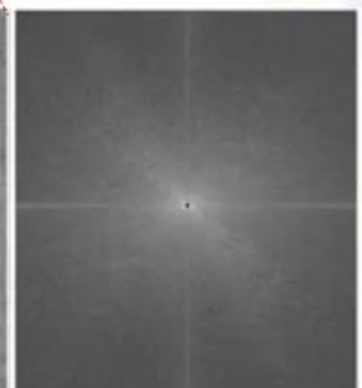


Bandpass Filter

Output image



DC Component

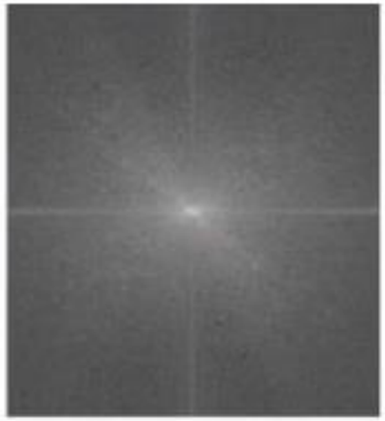


DC component removed

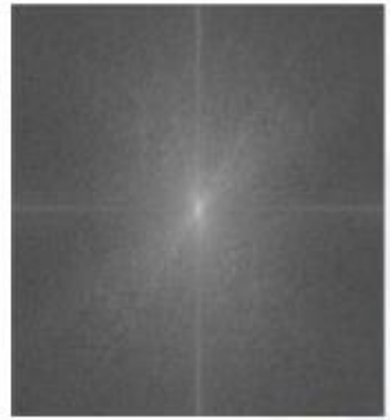
Fourier Transform:

- Basic Operations (image rotation and translation):

Original image

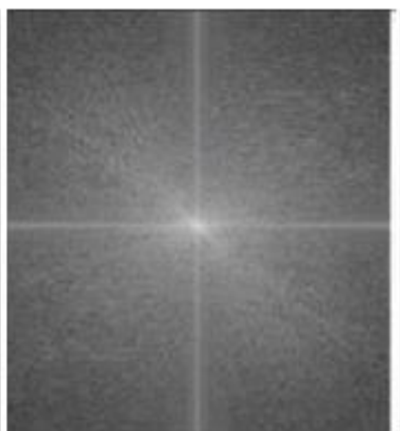


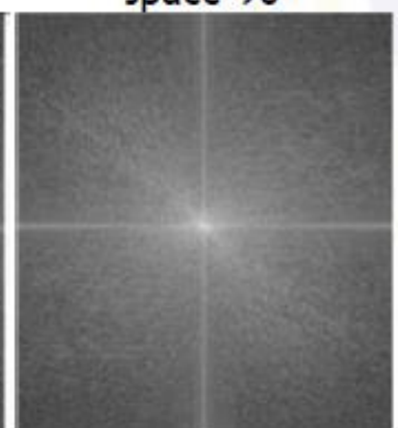
Fourier Transform



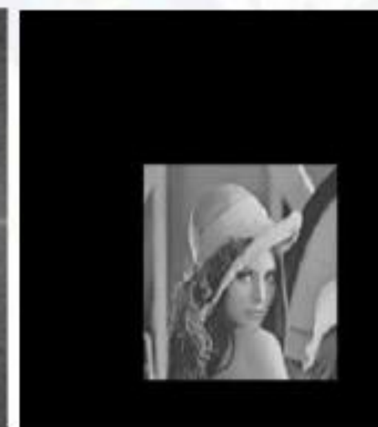
Rorate the Reciprocal space 90

Output image





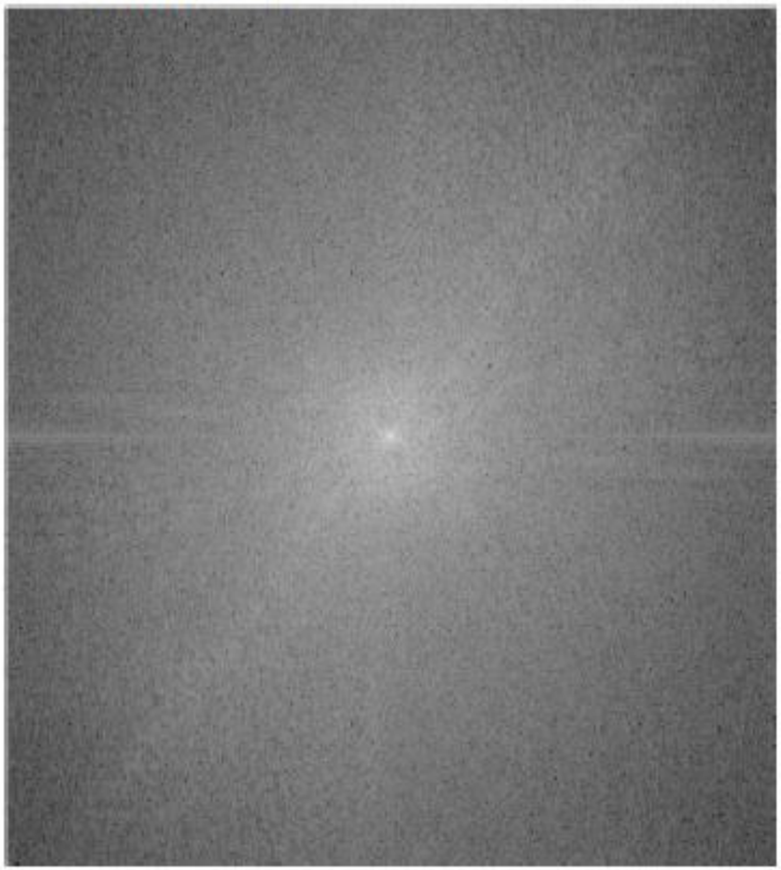
Phase shift



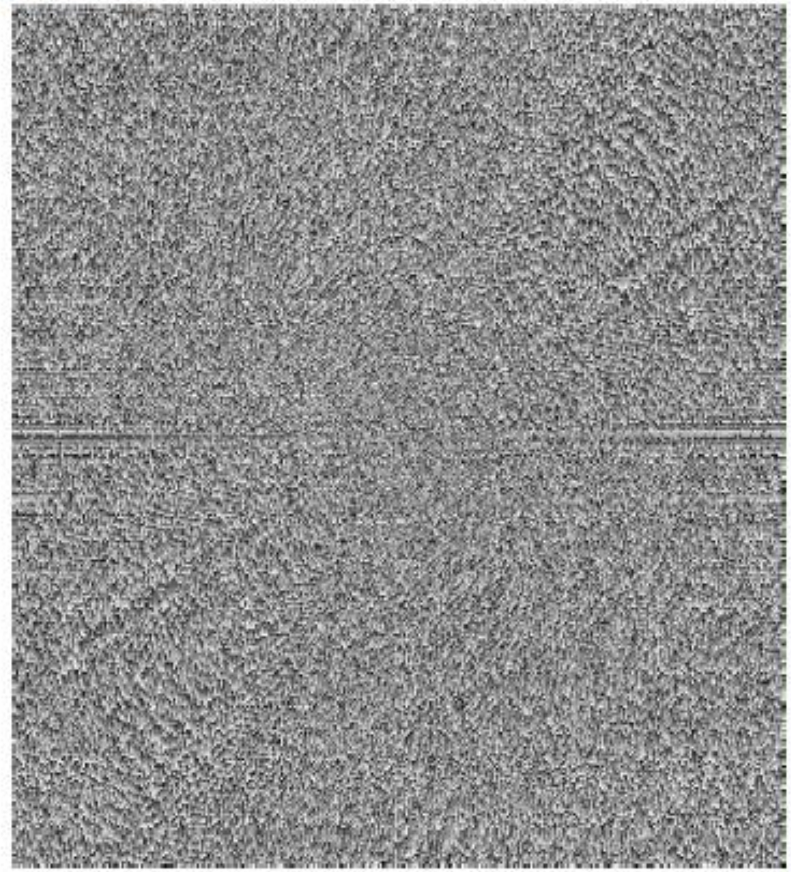
Output image

What about the phase?

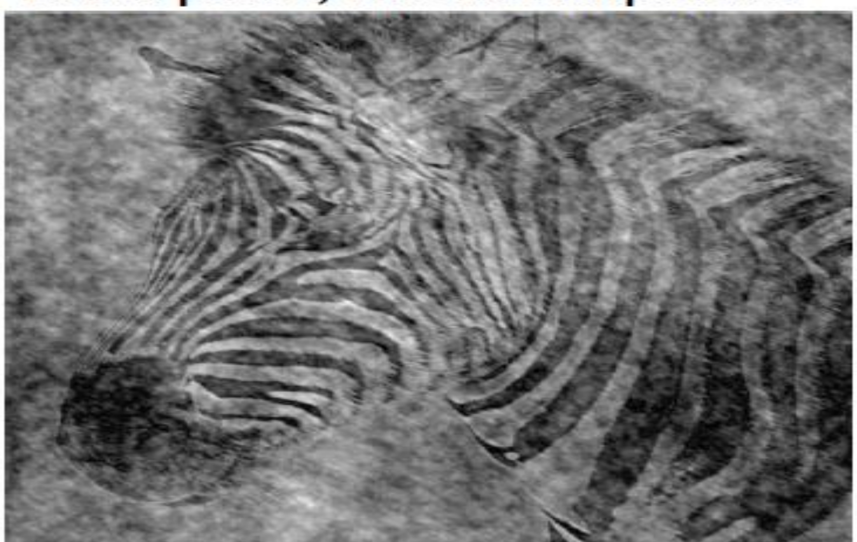
Amplitude



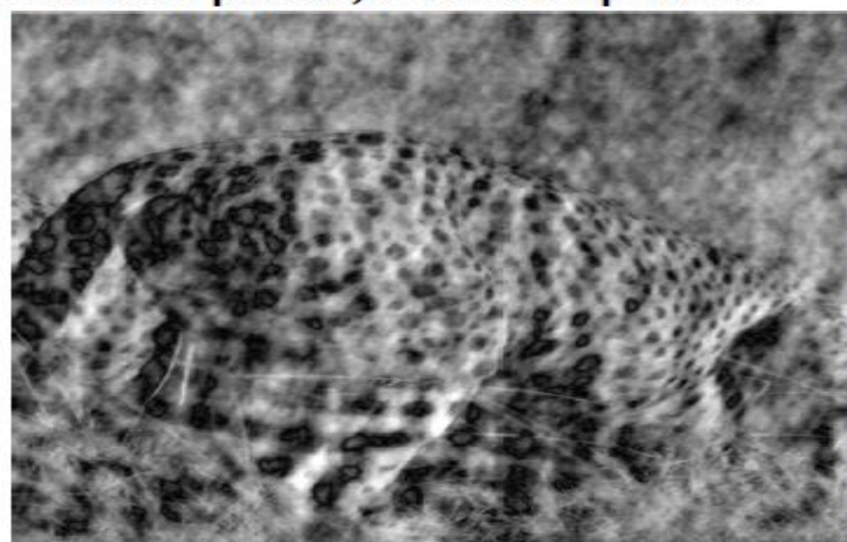
Phase



Zebra phase, cheetah amplitude

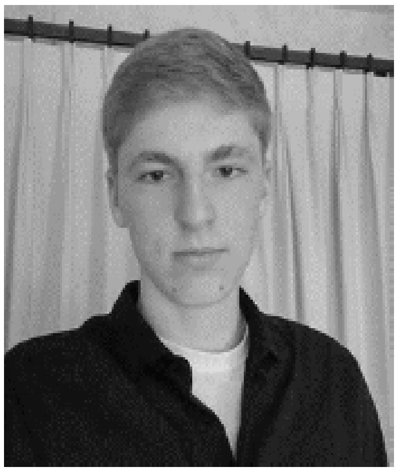


Cheetah phase, zebra amplitude



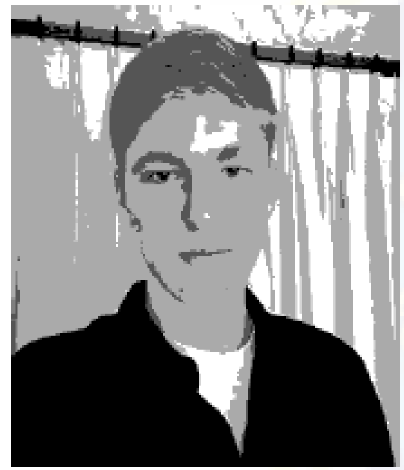
Quantization:

- Quantization Effects - Radiometric Resolution



8 bit – 256 levels

4 bit – 16 levels



2 bit – 4 levels

1 bit – 2 levels

Median Filters:

- Operates over a window by selecting the median intensity in the window.
- 'Rank' filter as based on ordering of gray levels
  - E.G., min, max, range filters.
- Better at salt and pepper noise.
  - Not convolution: try a region with 1's and a 2, and then 1's and a 3.

Image filtering - mean

$$f[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$I[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				
				?					

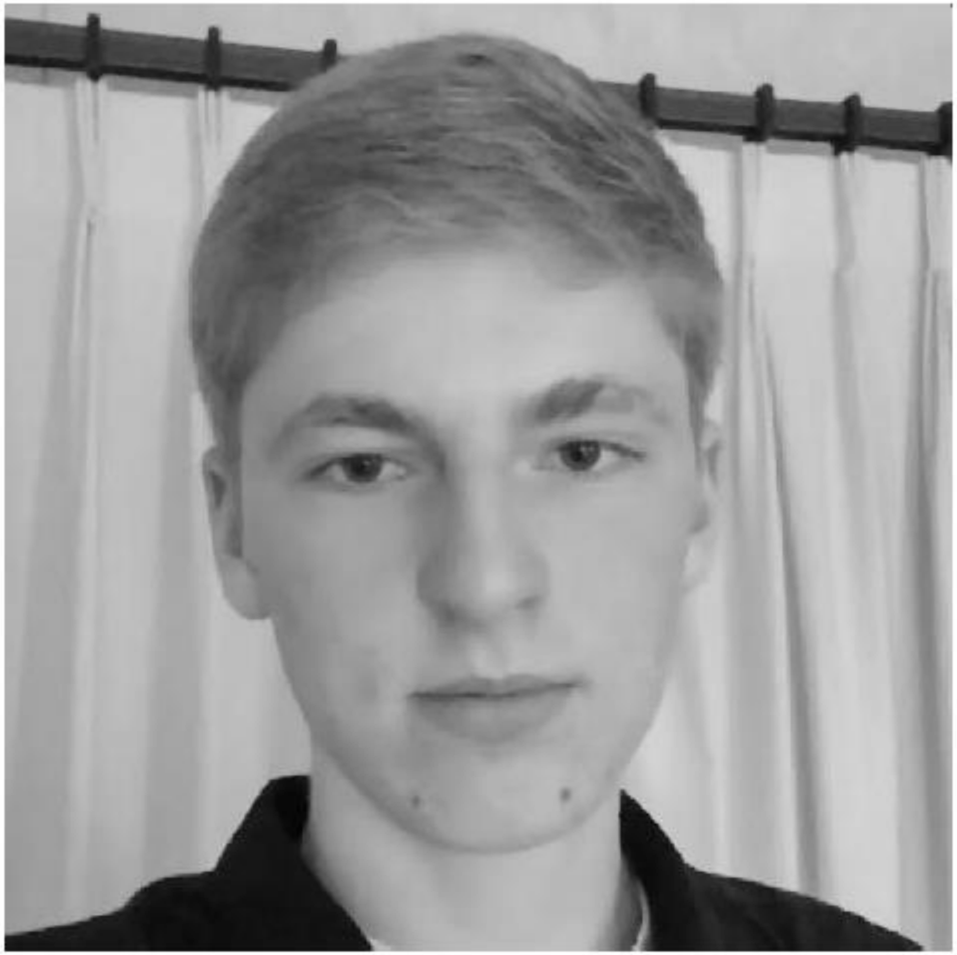
$$h[m,n]=\sum_{k,l} f[k,l]\, I[m+k,n+l]$$

Credit: S. Seitz

Noisy Jack – Salt and Pepper

Mean Jack – 3 x 3 filter

Median Jack – 3 x 3



Median Filters:

- Operates over a window by selecting the median intensity in the window.
- What advantage does a median filter have over a mean filter?
- Is a median filter a kind of convolution?
  - Answer: Not convolution: try a region with 1's and a 2, and then 1's and a 3.

Sobel Filter Visualization:

- Write down a 3x3 filter that both:
  - What happens to negative numbers?
  - For visualization:
    - Shift image + 0.5.
    - If gradients are small, scale edge response.

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

Sobel

```
>> I = img_to_float32( io.imread( 'luke.jpg' ) );
>> h = convolve2d( I, sobelKernel );
```

Sobel Filter Visualization:

- Write down a 3x3 filter that both:
  - What happens to negative numbers?
  - For visualization:

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

Sobel

`plt.imshow( h );`

`plt.imshow( h + 0.5 );`

Image Pyramid:

- A 'bar' in the big images is a hair on the zebra's nose; in smaller images, a stripe; in the smallest, the animal's nose.

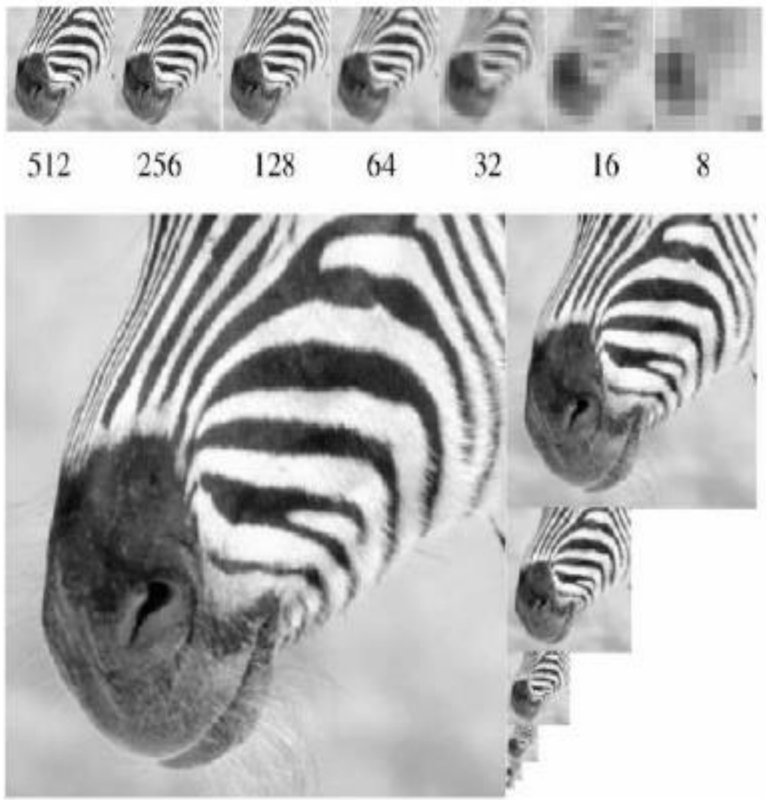
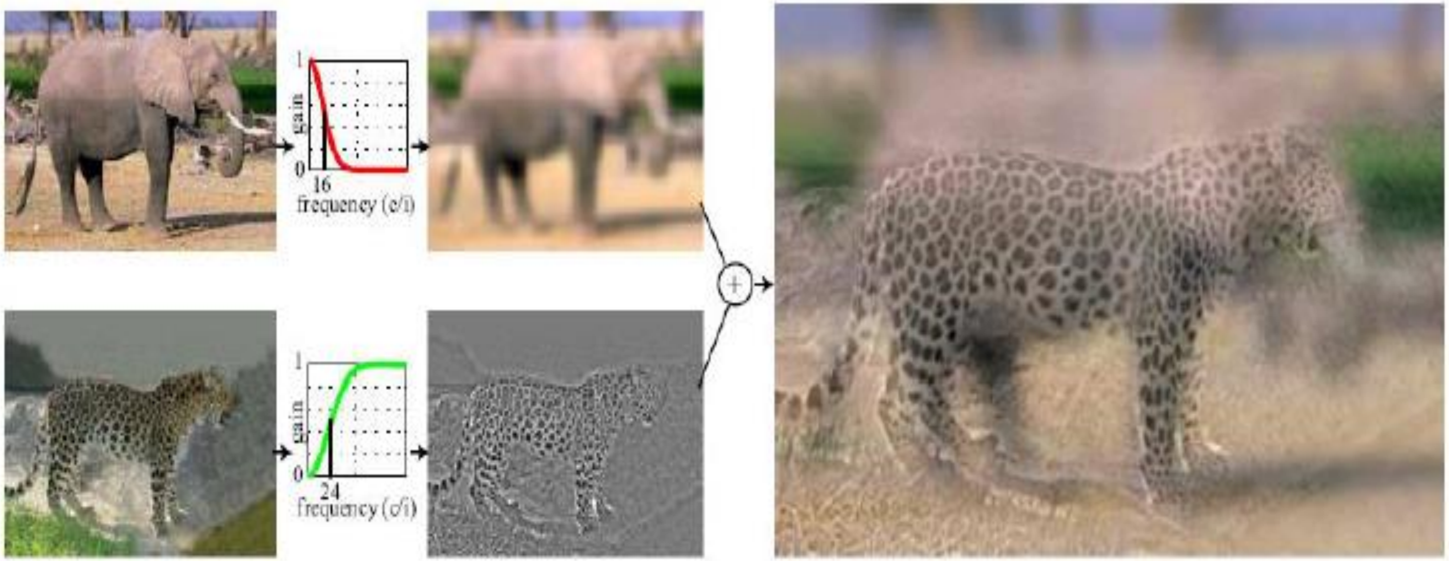


Image Pyramid:

- Algorithm for down sampling by factor of 2.
  1. Start with image of w x h
  2. Sample every other pixel.
     - im_small = image[::2, ::2]
  3. To build a pyramid,
     - Repeat Steps 1 & 2 until im_small is 1 pixel large.
- Image sub-sampling
  - Throw away every other row and column to create a 1/2 size image.

Hybrid Images?

- Merging two images together.



Why do we get different, distance-dependent interpretations of hybrid images?

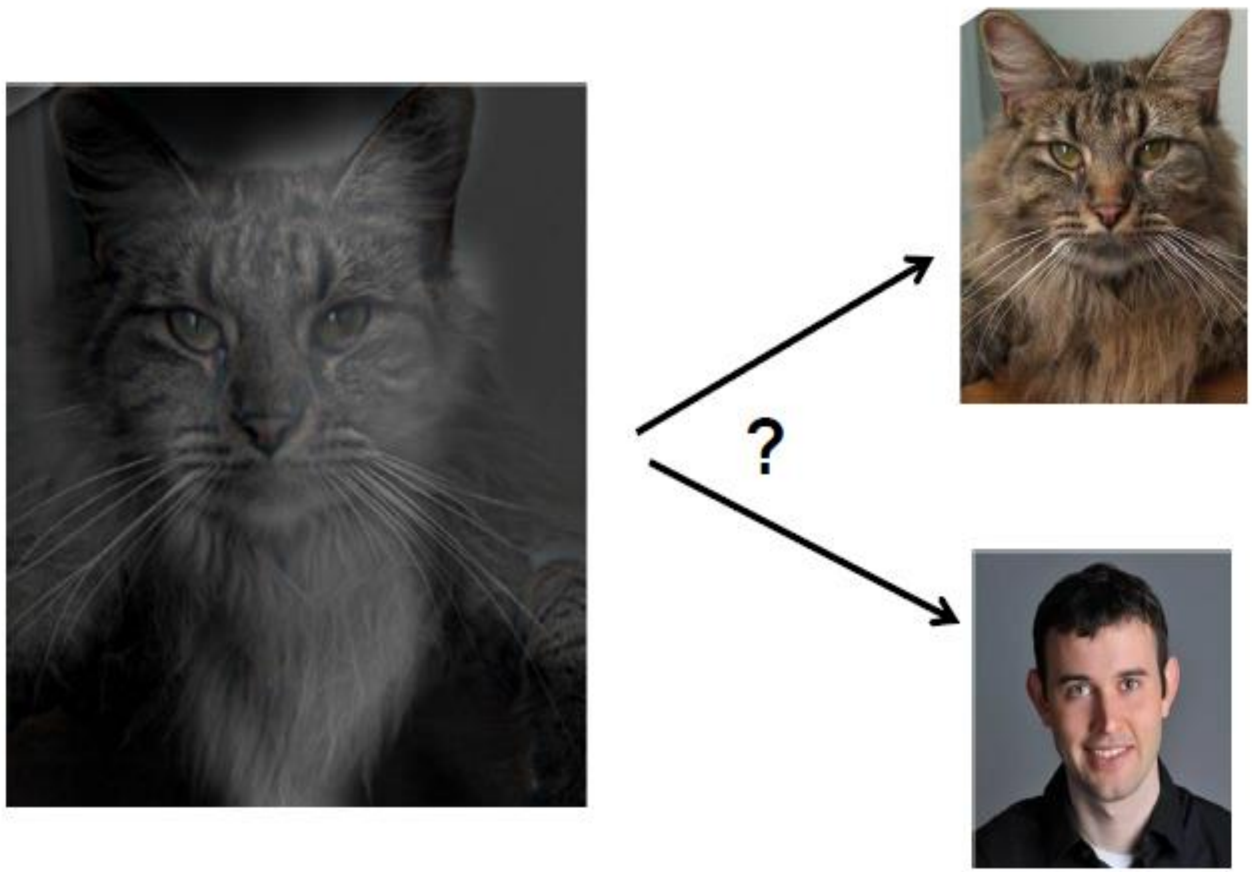
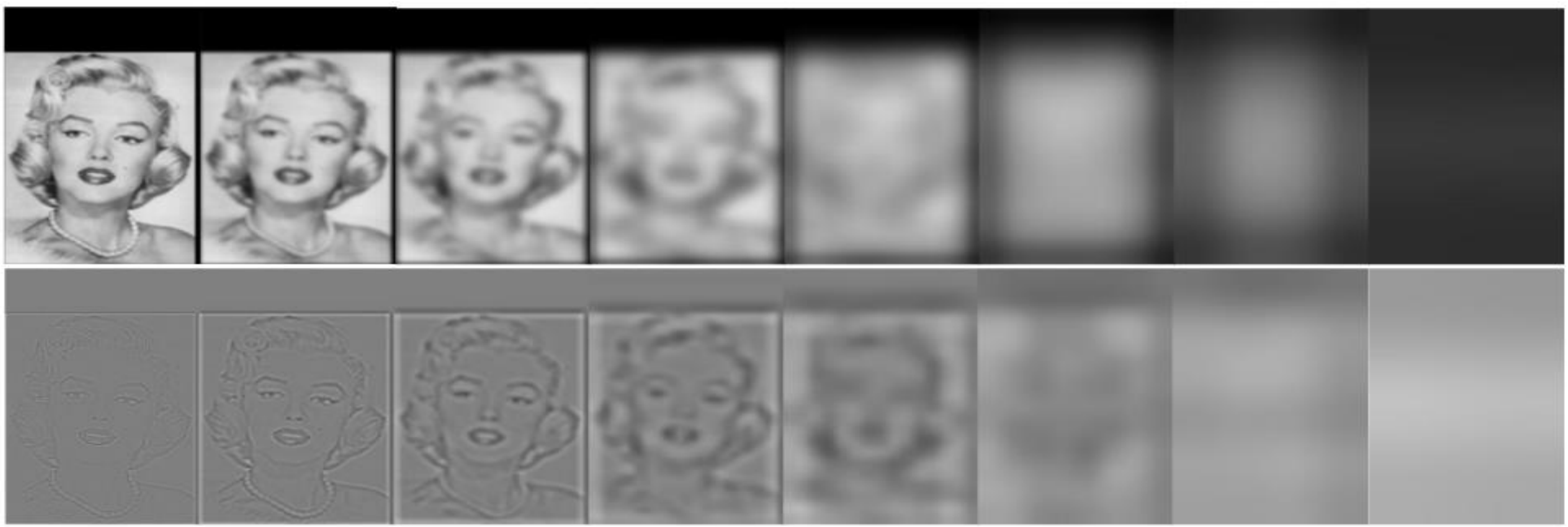


Image Pyramids:

Guassian and Laplacian pyramids represent a repeated low pass filtering and high pass filtering, respectively, of an image. Each pyramid level is formed as follows:

1) The original image is blurred with a Gaussian filter, and the blurred image is subtracted from the original, essentially extracting the highest frequencies in the image.
2) The blurred image is then downsampled, to negate the effects of the Gaussian filter, and step 1 is repeated for this new image.

Below is a sample Gaussian image pyramid, followed by a Laplacian image pyramid. The Laplacian pyramid has been brightened for clarity.



The process of creating a hybrid image breaks down into the following steps:

1) The two images are aligned such that similar features overlap and mask eachother. This aids in the visual effect, so that noticeable features of the less visible image do not distract the viewer's perception of the dominant image at a certain viewing distance.
2) Each image is then decomposed into a Gaussian pyramid and a Laplacian pyramid as described above.
3) Given a cutoff index N, the hybrid image is composed by combining the first 1 through N levels of the first image's Laplacian pyramid with the N+1 through last levels of the second image's Laplacian pyramid and the last level of the second image's Gaussian pyramid.
4) Finally, the hybrid image is cropped and exported.