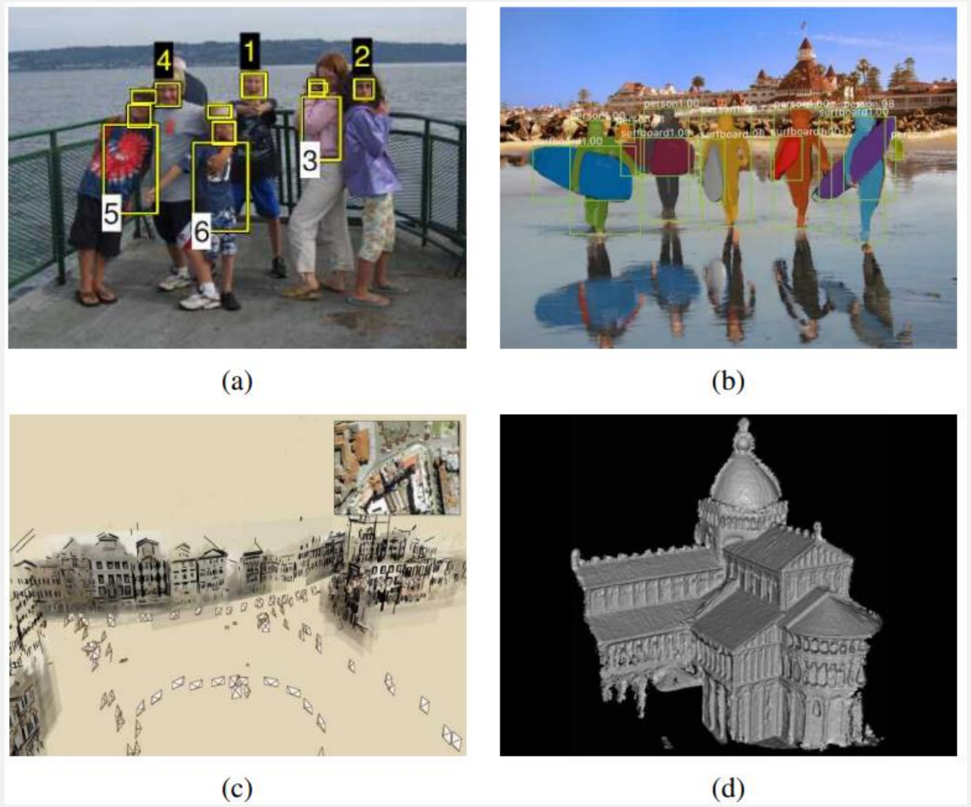
Threshold and Filter

01

What is CV?

What is CV?

Computer Vision (CV) is an interdisciplinary research field dedicated to enabling computers to understand and interpret content in the visual world



What is CV?

Purpose

- Automate vision tasks
  - Object recognition, face recognition, image classification, video analysis, etc.
- Understand and interpret visual data
  - Extract meaningful information, such as objects in the image, location, motion, etc.
- Practical Problems
  - Autonomous driving, medical image analysis, safety monitoring, industrial inspection, robot navigation, etc.

What is CV?

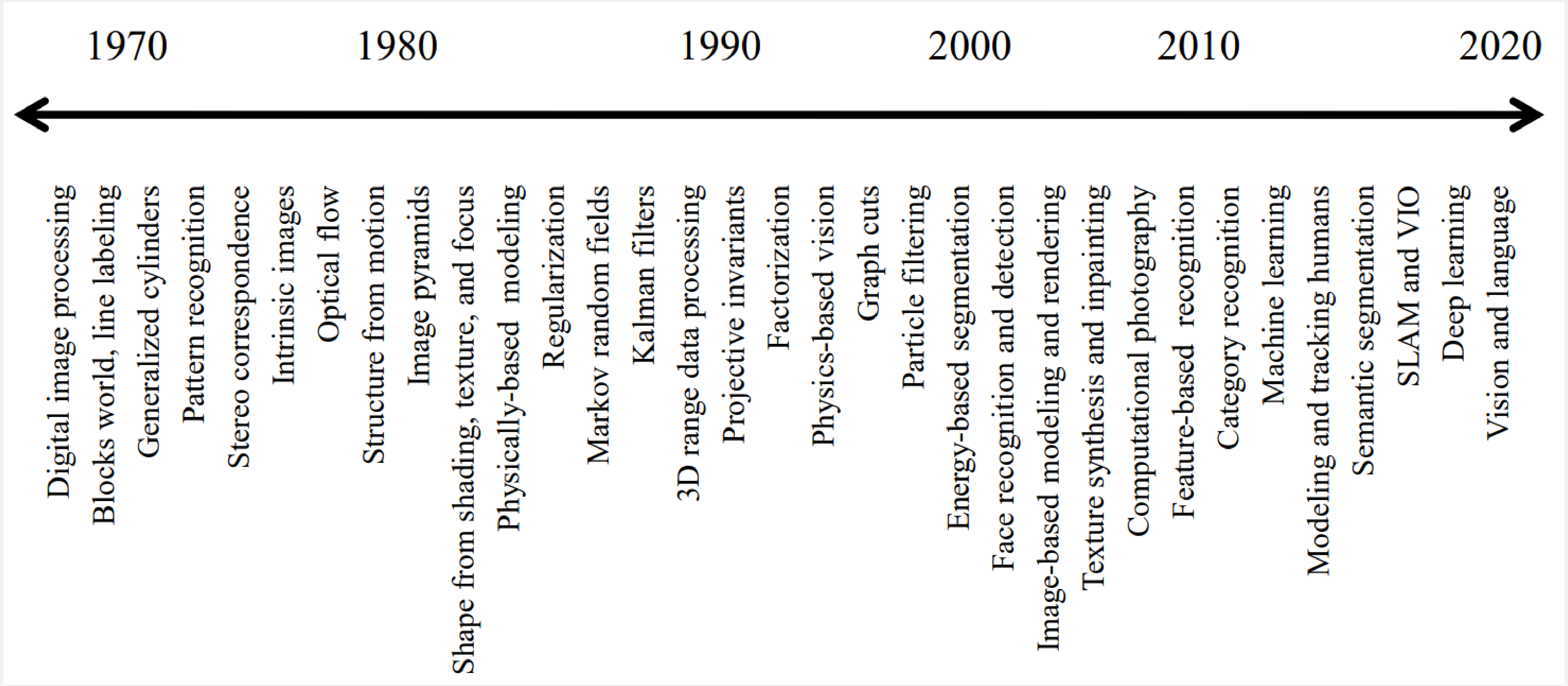
Differences between CV and Image Processing

- CV
  - Definition: Focus on how to extract and understand useful information
  - Target: Enable computers to "see" and "understand" information to make judgments and decisions
  - Operations: Object detection and recognition, image segmentation and classification, etc.

- Image Processing
  - Definition: Focus on how to process and improve the quality of digital images including image enhancement, restoration, filtering, etc.
  - Target: Improve image quality to make it more suitable for human observation
  - Operations: Denoising, sharpening, smoothing, contrast adjustment, edge detection, etc.

What is CV?

History

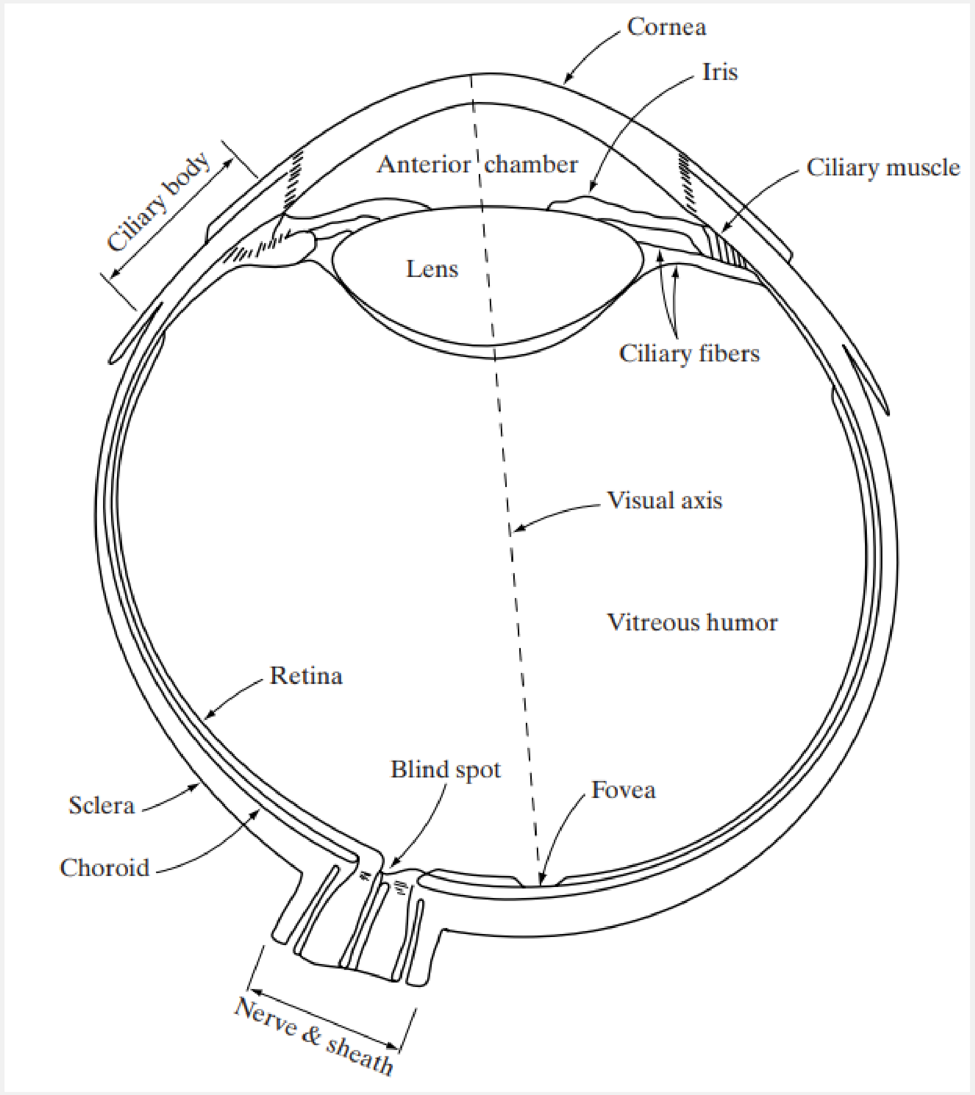


02

Visual Perception

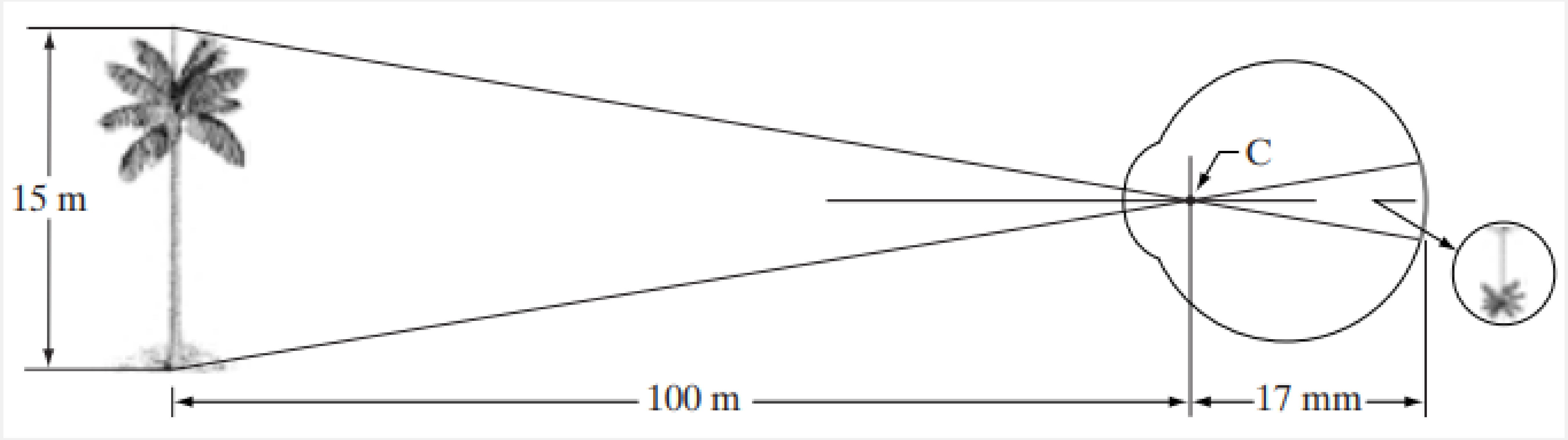
Visual Perception

Eye Structure



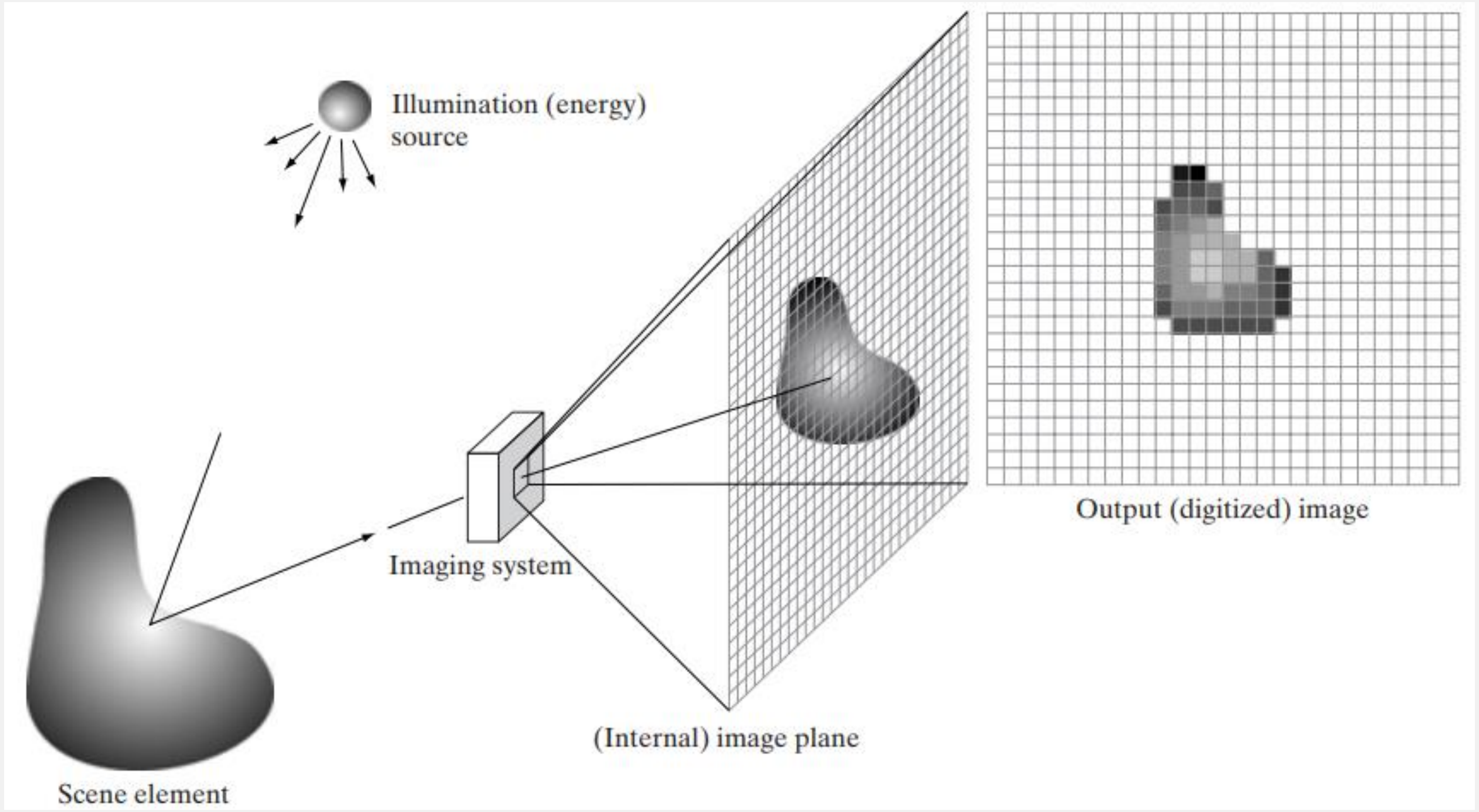
Visual Perception

Image formation in eye



Visual Perception

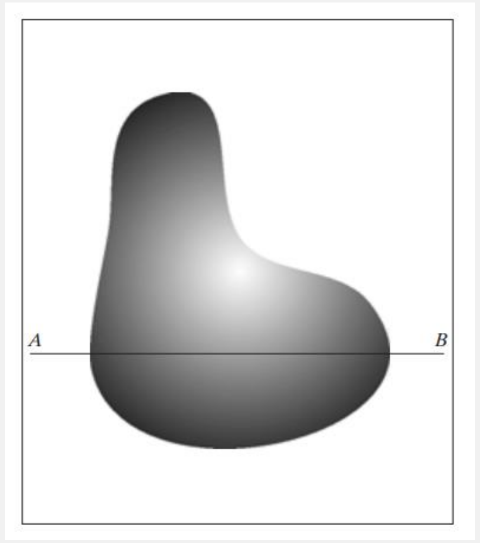
Image acquisition process

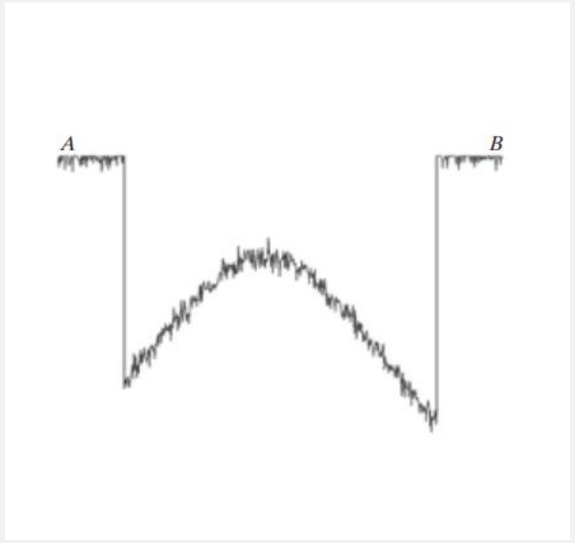


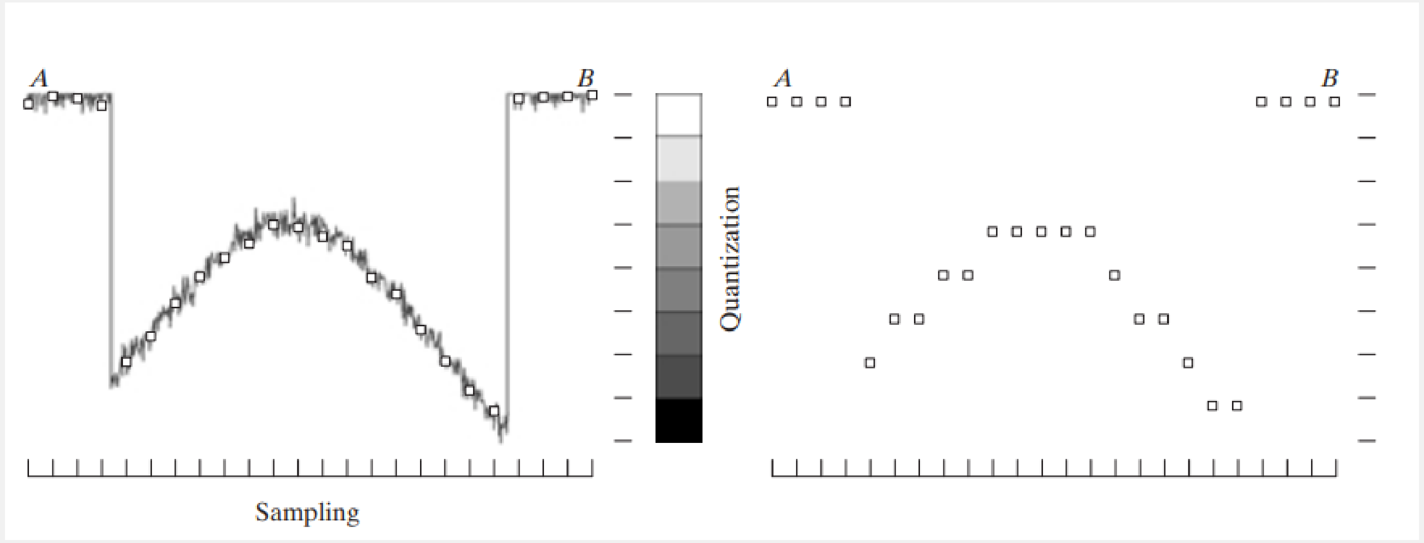
Visual Perception

Image sampling and quantization

- ➢ Sampling: digitizing the coordinate values
- ➢ Quantization: digitizing the amplitude values

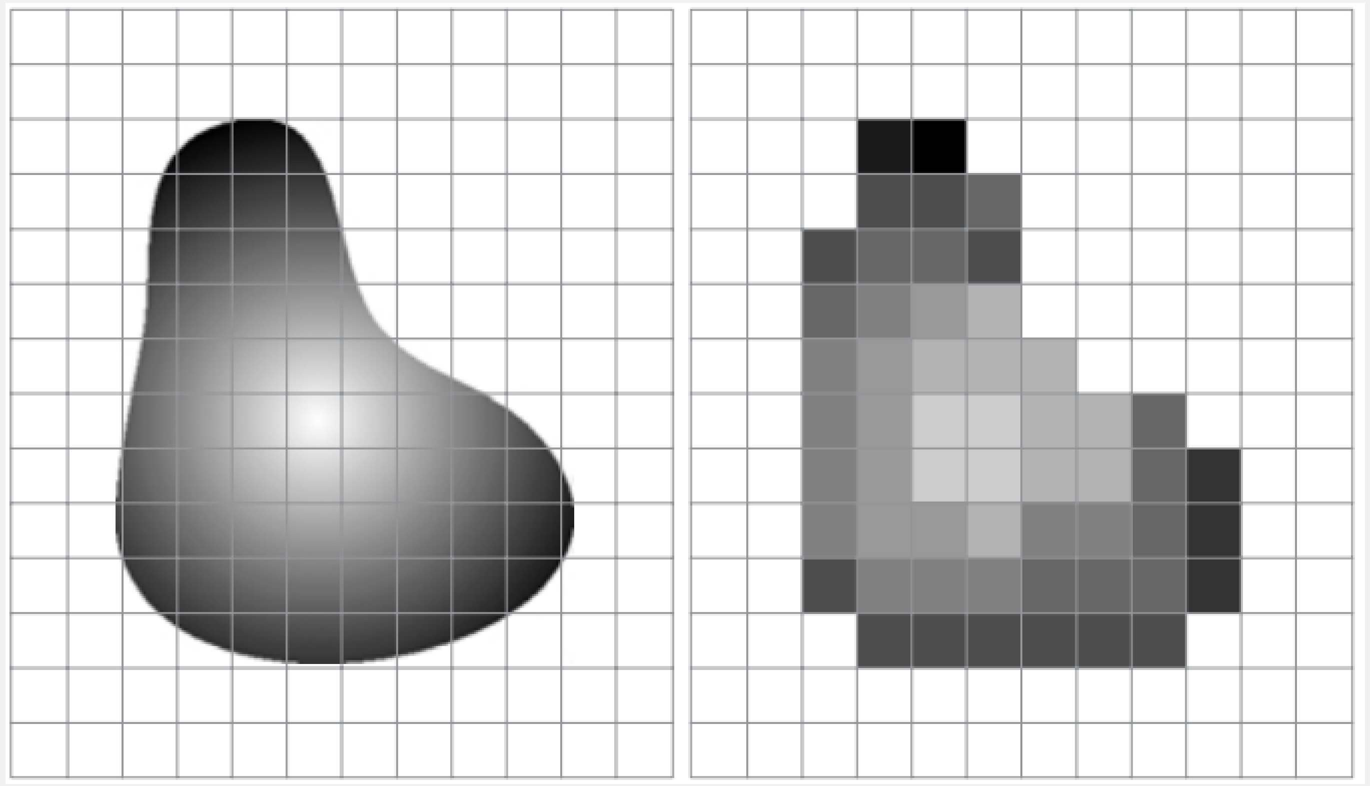


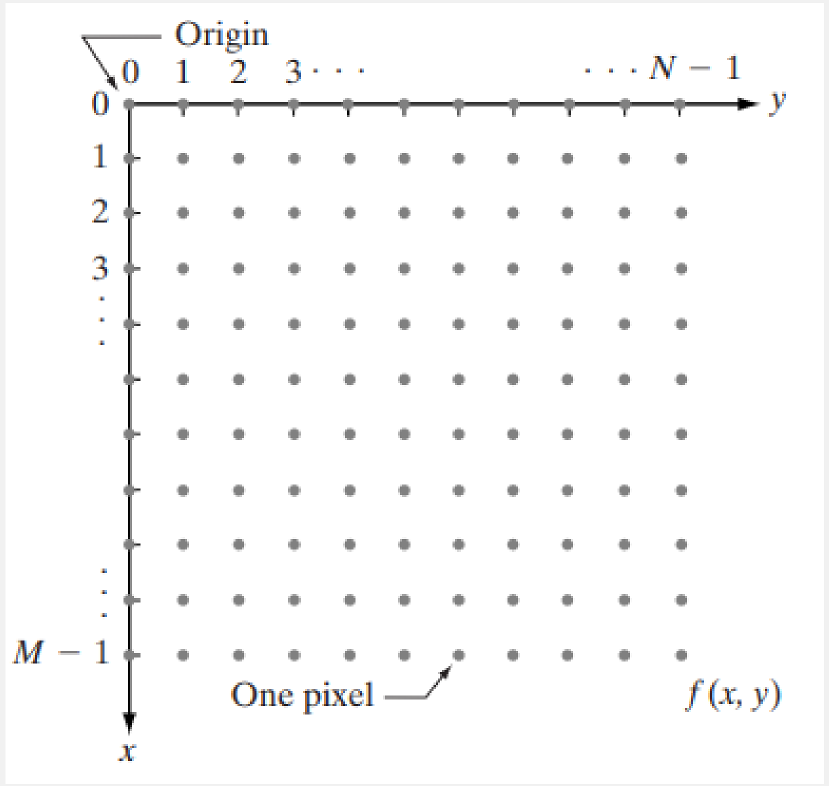




Visual Perception

Representing Digital Image

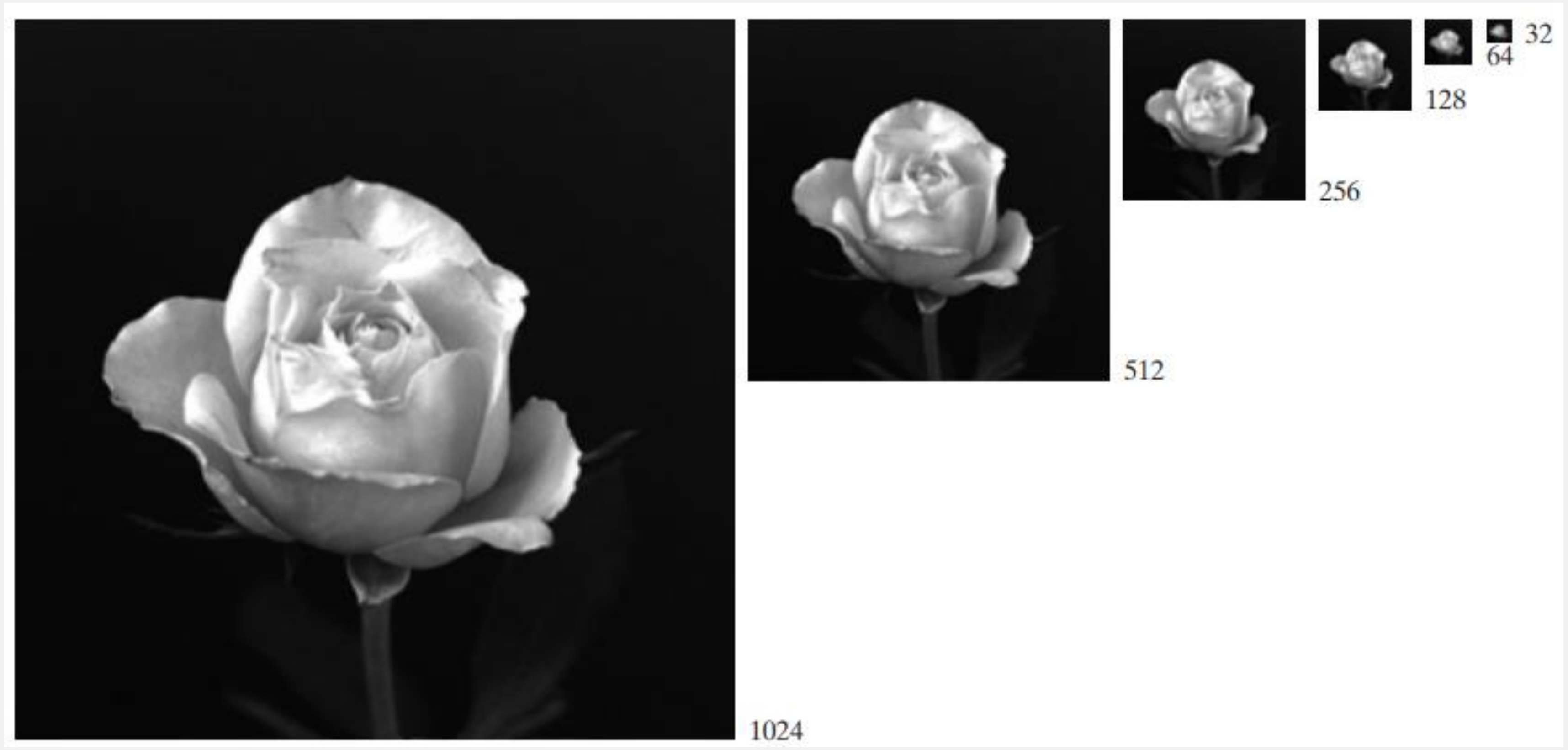




Visual Perception

Spatial Resolution

- A measure of an image or imaging system's ability to distinguish or present fine details



Visual Perception

Pixel Concept

- Each pixel in color image is constructed with three channels (RGB)
- Each pixel in gray image is only constructed with one channel

03

Threshold

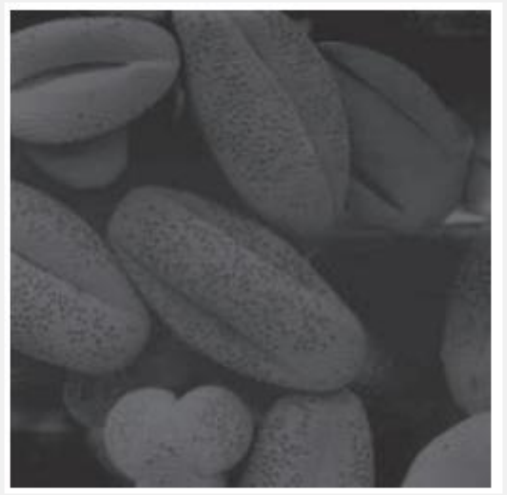
Threshold

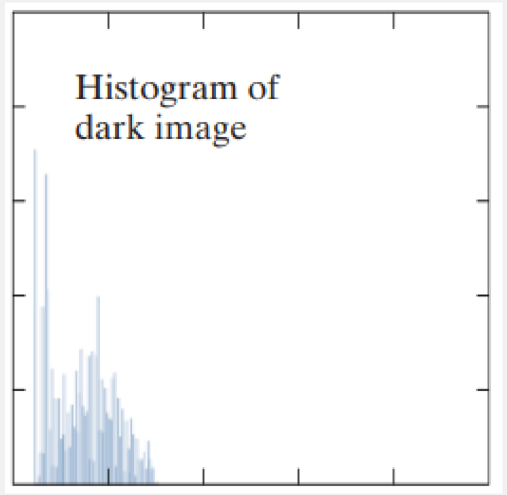
Histogram

- A visual representation of the distribution of quantitative data
  - Let r_k, for k=0, 1, 2, ..., L-1, denote the intensities of an L-level digital image, f(x, y)

    The unnormalized histogram $h(r_k) = n_k$ for k = 0, 1, 2, …, L-1

  - n_k is the number of pixels in image with intensity r_k, and the subdivisions of the intensity scale are called histogram bins
  - The normalized histogram $p(r_k) = \frac{n_k}{MN}$ where M and N are the row and column of image



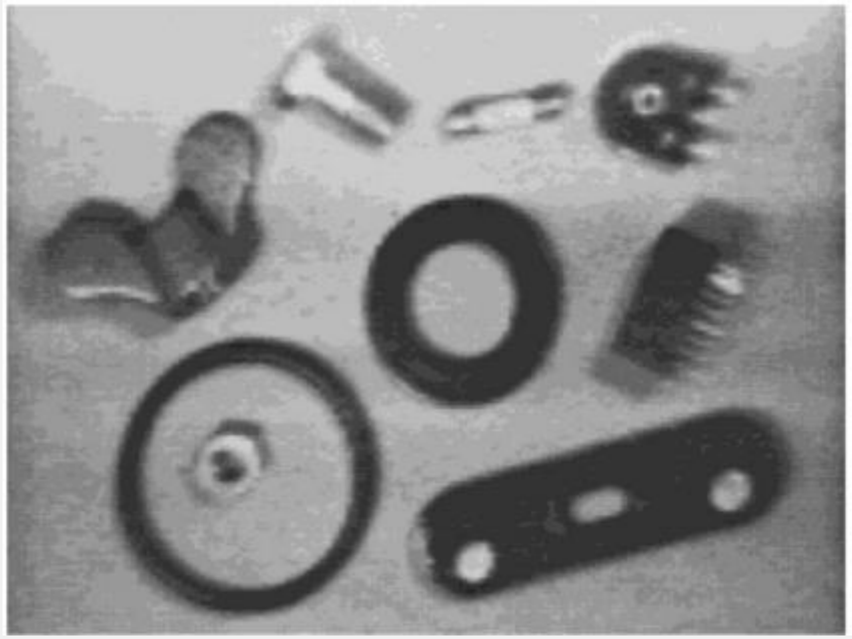


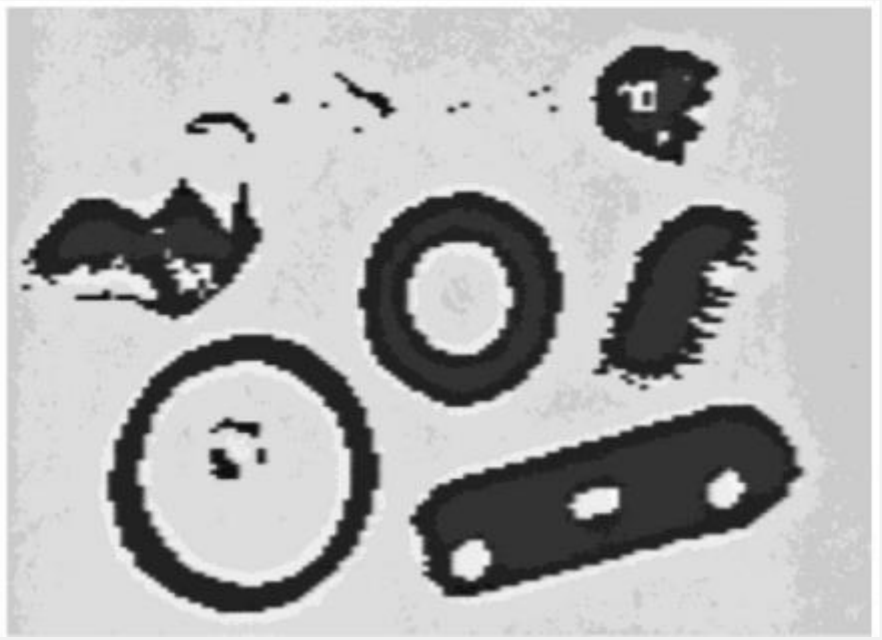
Threshold

Threshold

- A value to separate foreground from background

```
for all pixels in image do {
    if ($P0 > thresh$) $A0 = 1$; else $A0 = 0$;
}
for all pixels in image do {$R0 = 255*(1 - A0)$;}
```


Gray image

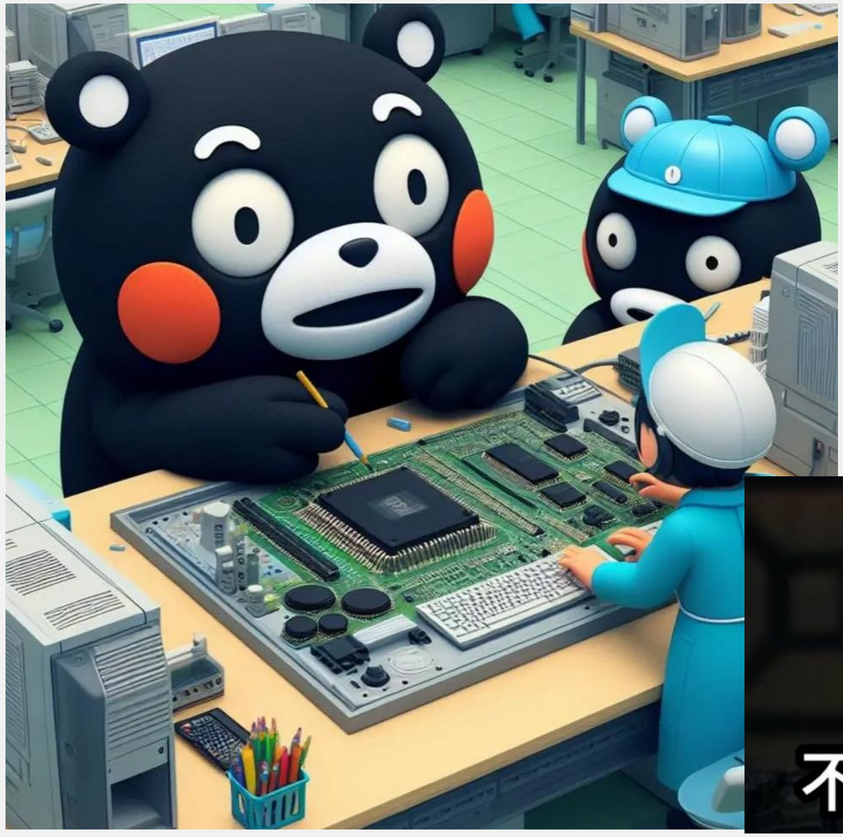

Gray image after threshold

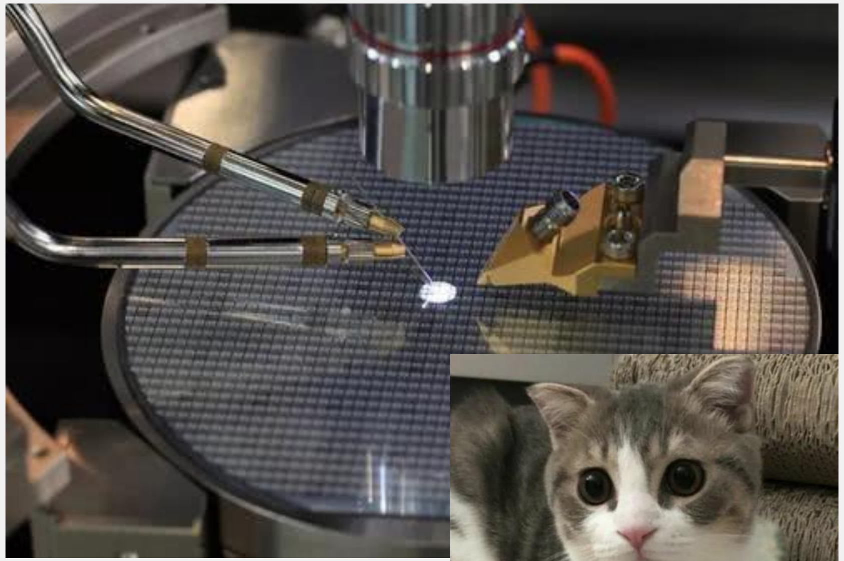
Threshold

But, how to set best threshold?

Manual?

Auto

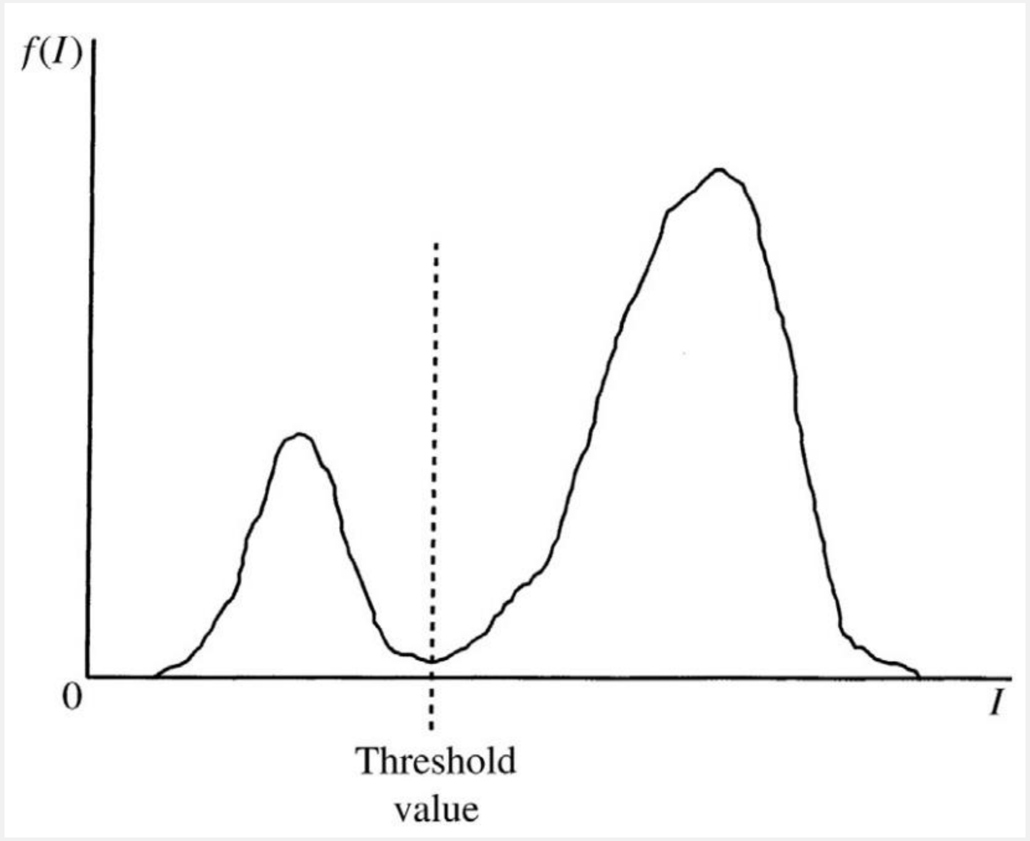




Threshold

A suitable threshold

- ➢ Analyze image histogram
- ➢ Drawbacks:
  - ◆ The valley may be so broad
  - ◆ There may be a number of minima because of the type of detail in the image, and selecting the most significant one will be difficult
  - ◆ Noise disturbance
  - ◆ There may be no clearly visible valley in the distribution
  - ◆ Either of the major peaks in the histogram (usually that due to the background) may be much larger than the other and this will then bias the position of the minimum
  - ◆ The histogram may be inherently multimodal



Threshold

A suitable threshold

- Local
  - Local thresholding
- Global
  - Variance-based thresholding
  - Entropy-based thresholding
  - Maximum likelihood thresholding

Threshold

Local Thresholding

- ➢ Segment an image into foreground and background regions based on local intensity variations
- ➢ Compute a threshold for each pixel based on the intensity values of its neighboring pixels
- ➢ Principles:
  - ◆ Neighborhood definition: A neighborhood around each pixel is defined, typically a square or circular region
  - ◆ Local statistics calculation: For each pixel, local statistics (mean, median, standard deviation, etc.) within the neighborhood are computed
  - ◆ Threshold calculation: A threshold value is determined with the local statistics for each pixel using approaches
    - ✓ Mean thresholding: Using the local mean intensity value
    - ✓ Adaptive mean thresholding: Using the local mean minus a constant
    - ✓ Niblack's method: T = $\mu + k\sigma$, where μ is the local mean, σ is the local standard deviation, and k is a constant
    - ✓ Sauvola's method: T = $\mu\,(1 + k(\sigma / R - 1))$, where R is the dynamic range of standard deviation (typically 128 for an 8-bit image)

Threshold

Local Thresholding

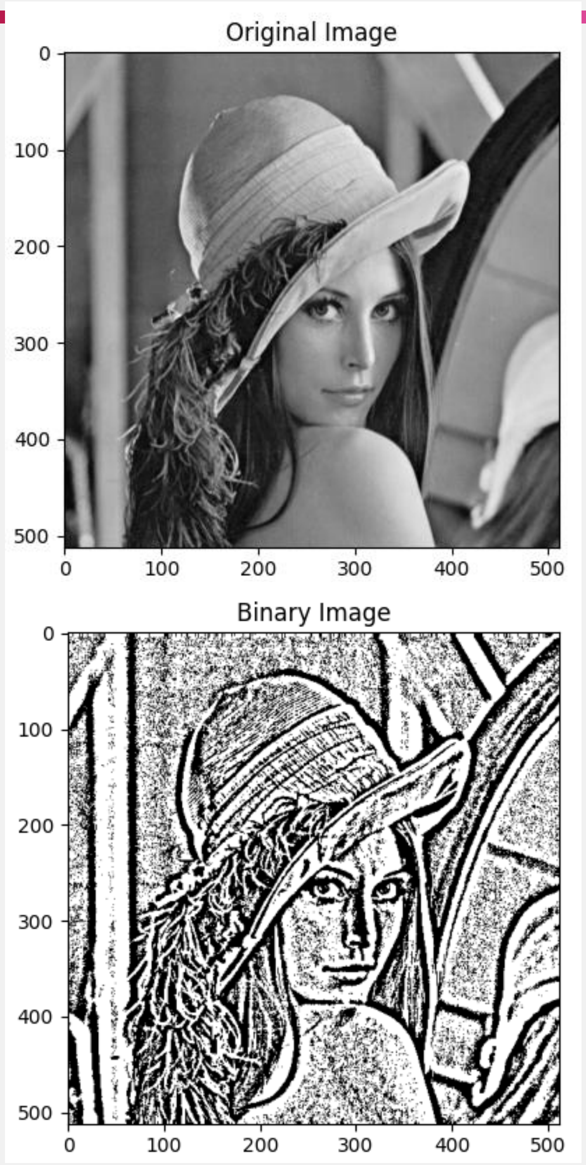
➢ Adaptive mean method: T = local_mean - c where c = 2

```
image = cv2.imread('lena.bmp', cv2.IMREAD_GRAYSCALE)

window_size = 15
c = 2

binary_image = cv2.adaptiveThreshold(
    image,
    255,                         # 最大值，閾值化的像素設置為此值
    cv2.ADAPTIVE_THRESH_MEAN_C, # 閾值類型：局部均值閾值法
    cv2.THRESH_BINARY,           # 閾值應用類型：二值化
    window_size,                 # 計算閾值時考慮的鄰域大小（必須是奇數）
    c                            # 常數，從計算出的均值中減去的值
)
```

Lena



Threshold

Global Thresholding

- It uses a single threshold to classify each pixel in the image as either foreground or background
- It is particular suitable for images with distinct contrast and uniform lighting
- Principle
    - Choosing a threshold T: it can be typically determined by analyzing the histogram of the image
        - ✓ Variance-based thresholding
        - ✓ Entropy-based thresholding
        - ✓ Maximum likelihood thresholding
    - Classifying pixels: if the pixel is greater than or equal to T, it is set to the foreground; otherwise it is set to the background
    - Formula representation:

$$Binary\ image = \begin{cases} Foreground\ (white)\ \ if\ pixel\ value\ \geq T \\ Background\ (black) if\ pixel\ value\ < T \end{cases}$$

Threshold

Global Thresholding

- Variance-based thresholding
  - Threshold is determined by maximizing the between-class variance between the foreground and background
  - Well known methods: Otsu algorithm
    - If a threshold T(k)=k, 0 < k < L-1 is selected, and it is used to threshold the input image into two classes, c_1 and c_2, where c_1 consists of all the pixels in the image with intensity values in the range [0, k] and c_2 consists of the pixels with values in the range [k+1, L-1]
    - The probability $P_i(k)$ is assigned to class c_i is determined by the cumulative sum

$$P_1(k) = \sum_{i=0}^{k} p_i \qquad P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$$

  - The mean intensity value of the pixels in c_i is

$$m_1(k) = \sum_{i=0}^{k} iP(i|c_1) = \sum_{i=0}^{k} iP(c_1|i)P(i)/P(c_1) = \frac{1}{P_1(k)}\sum_{i=0}^{k} ip_i \qquad m_2(k) = \frac{1}{P_2(k)}\sum_{i=k+1}^{L-1} ip_i \quad P(A|B) = P(B|A)P(A)/P(B)$$

Threshold

Global Thresholding

- Variance-based thresholding
  - The cumulative mean (average intensity) up to level k is

$$m(k) = \sum_{i=0}^{k} i p_i$$

  - The cumulative mean of the entire image (global intensity) is

$$m_G = \sum_{i=0}^{L-1} i p_i$$

  - To evaluate effectiveness of threshold at level k

Find the maximum

global variance

between-class variance

$$\eta = \frac{\sigma_B^2}{\sigma_G^2}$$

$$\sigma_G^2 = \sum_{i=0}^{L-1} (i - m_G)^2 p_i$$

$$\begin{aligned}\sigma_B^2 &= P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 \\ &= P_1 P_2 (m_1 - m_2)^2 \\ &= \frac{(m_G P_1 - m)^2}{P_1(1 - P_1)}\end{aligned}$$

Threshold

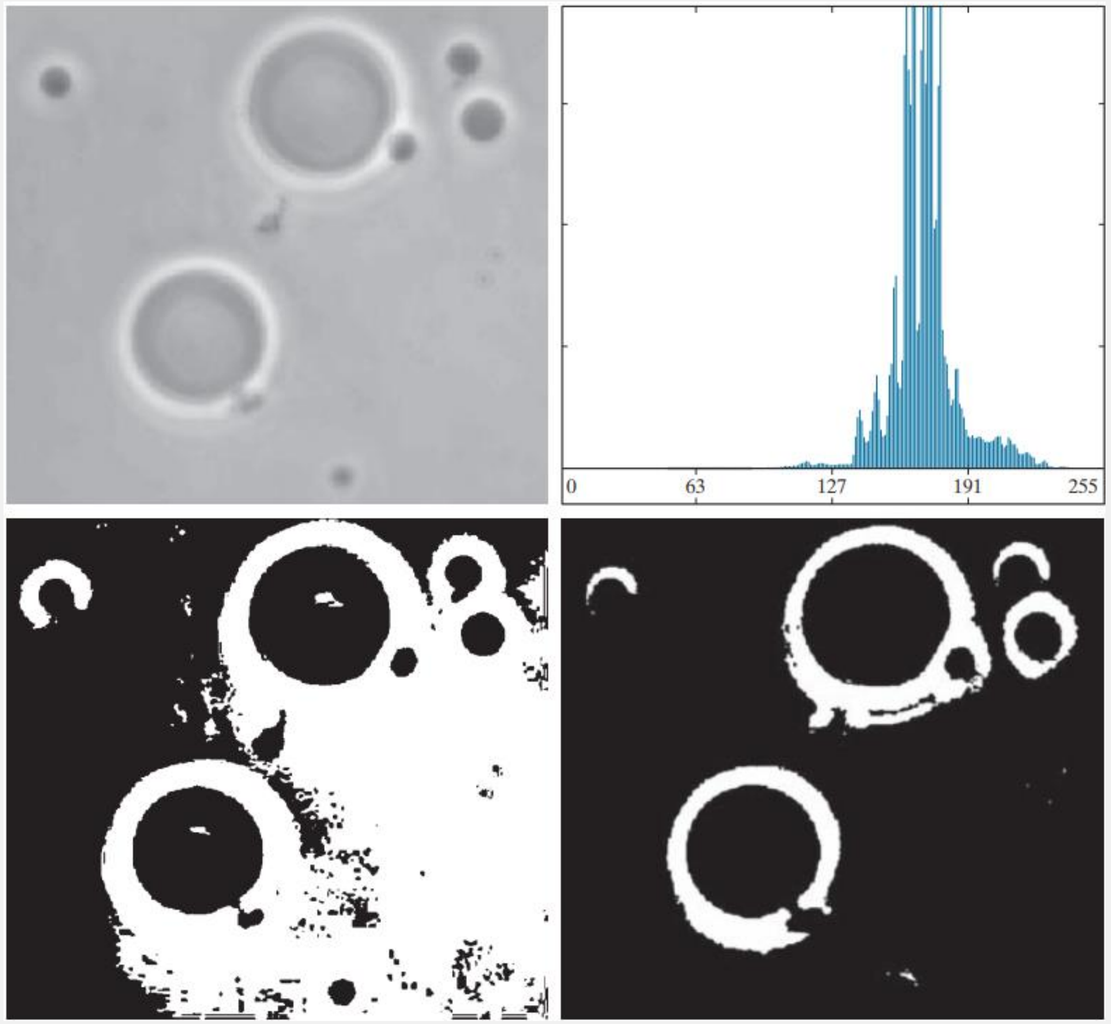
Global Thresholding

- Variance-based thresholding
  - Threshold is determined by maximizing the between-class variance between the foreground and background
  - Well known methods: Otsu algorithm
    - ✓ Compute normalized histogram
    - ✓ Calculate the cumulative sums, $P_1(k)$, for k=0, 1, 2, …, L-1
    - ✓ Calculate the cumulative means, m(k), for k=0, 1, 2, …, L-1
    - ✓ Calculate the global mean, m_G
    - ✓ Compute the between-class variance term, $\sigma_B^2(k)$, for k =0, 1, 2, …, L-1
    - ✓ Obtain the Otsu threshold, k^*, as the value of k for which $\sigma_B^2(k)$ is maximum
      - If the maximum is not unique, obtain k^* by averaging the values of k corresponding to the various maxima detected
    - ✓ Compute the global variance, σ_G^2, and obtain the separability measure, η^*

Threshold

Global Thresholding

- Otsu algorithm



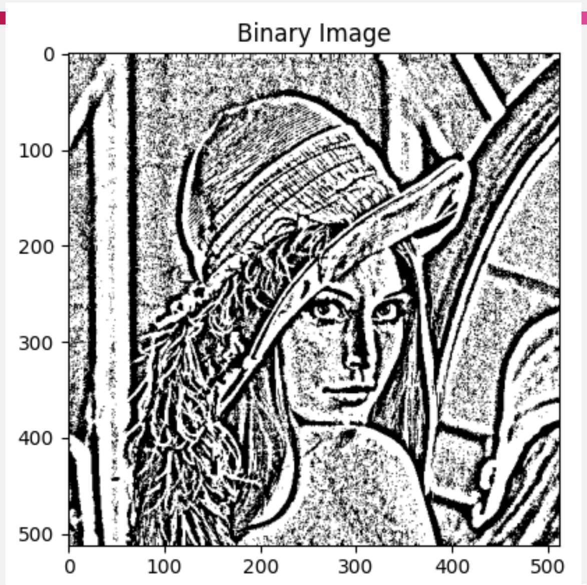
Threshold

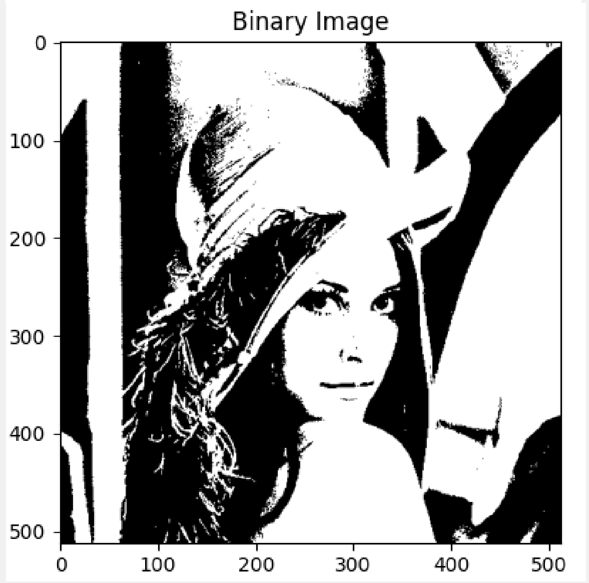
Global Thresholding

- Variance-based thresholding
  - Otsu algorithm

```
image = cv2.imread('lena.bmp', cv2.IMREAD_GRAYSCALE)

ret, otsu_threshold = cv2.threshold(
    image,
    0,                          # 使用Otsu時要忽略的值
    255,
    cv2.THRESH_BINARY + cv2.THRESH_OTSU
)
```





Threshold

Global Thresholding

- Entropy-based thresholding
  - Threshold is determined by maximizing the sum of entropy values of the foreground and background of image
  - Method
    - ✓ Histogram calculation
    - ✓ Probability distribution is computed by normalizing the histogram

      calculate the probability P(*i*) of each gray level *i*
    - ✓ Foreground and background probability

      For each possible threshold T, calculate the probabilities of the foreground and background
      Foreground probability $P_1(T)$ is the cumulative probability of gray levels from 0 to T
      Background probability $P_2(T)$ is the cumulative probability of gray levels from T+1 to maximum gray level
    - ✓ Calculate entropy values

      Foreground entropy $H_1(T)$ and background entropy $H_2(T)$ are calculated based on the probabilities of the foreground and background
      The entropy calculation formula is $H = -\Sigma(P(i) \times \log P(i))$
    - ✓ Maximize entropy

      $H(T) = H_1(T) + H_2(T)$

Threshold

Global Thresholding

- Maximum likelihood thresholding
  - It is a statistical method used to segment an image into foreground and background
  - Each class's pixel values follow a certain probability distribution, such as Gaussian distribution
  - Method
    - ✓ Histogram calculation
    - ✓ Initialize the parameters of the probability distributions for the foreground and background, such as mean and variance
    - ✓ Compute Within-class likelihood function (defined by user)
    - ✓ Compute joint likelihood function
    - ✓ Select optimal threshold

Threshold

Global Thresholding

- Maximum likelihood thresholding

```python
def maximum_likelihood_thresholding(image):
    # Convert image to grayscale if it is not
    if len(image.shape) > 2:
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Flatten the image to 1D array
    pixel_values = image.flatten()

    # Calculate the histogram
    histogram, bin_edges = np.histogram(pixel_values, bins=256, range=(0, 256))

    # Normalize the histogram
    histogram = histogram / float(np.sum(histogram))

    # Initialize variables
    max_likelihood = -np.inf
    optimal_threshold = -1
    epsilon = 1e-10
```

Threshold

Global Thresholding

- Maximum likelihood thresholding

```
for threshold in range(1, 256):
        # Split the histogram into two classes
        w0 = np.sum(histogram[:threshold])
        w1 = np.sum(histogram[threshold:])
        if w0 == 0 or w1 == 0:
            continue
        mean0 = np.sum(np.arange(0, threshold) * histogram[:threshold]) / w0
        mean1 = np.sum(np.arange(threshold, 256) * histogram[threshold:]) / w1
        variance0 = np.sum(((np.arange(0, threshold) - mean0) ** 2) * histogram[:threshold]) / w0
        variance1 = np.sum(((np.arange(threshold, 256) - mean1) ** 2) * histogram[threshold:]) / w1

        within_class_likelihood = w0 * np.log(variance0 + epsilon) + w1 * np.log(variance1 + epsilon)

        if within_class_likelihood > max_likelihood:
            max_likelihood = within_class_likelihood
            optimal_threshold = threshold

    return optimal_threshold
```

04

Filter

Filter

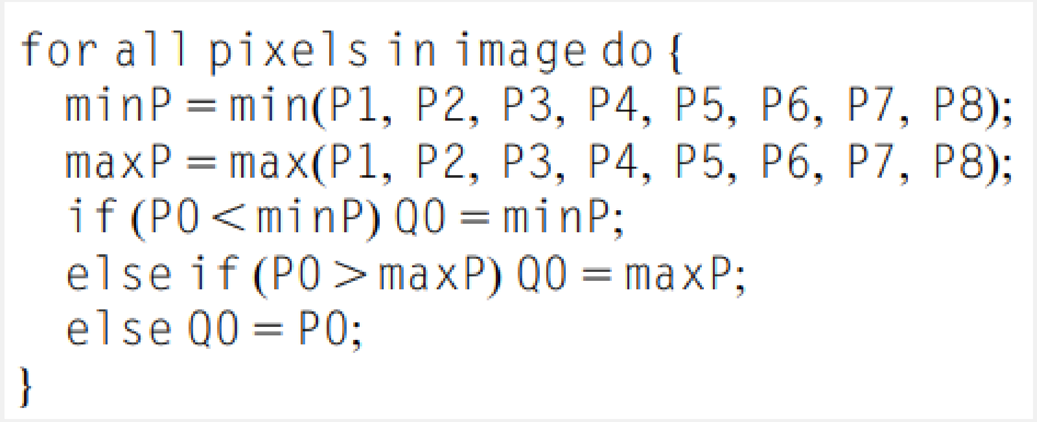
A filter is a tool used in digital signal processing to alter specific characteristics of the signal

- It can emphasize or suppress certain frequency components of a signal to achieve effects
  - Noise reduction
  - Edge detection
  - Smoothing
  - …
- Types
  - Low-pass filter (LPF)
  - High-pass filter (HPF)
  - Band-pass filter (BPF)
  - Directional filter

Filter

Low-pass filter allows low frequency signals to pass through while suppressing high frequency signals

- ➢ Median filter
  - ◆ It locate pixels in image which have extreme and therefore highly improbable intensities, and to ignore their actual intensities, replacing them with more suitable values



```
for all pixels in image do {
  minP = min(P1, P2, P3, P4, P5, P6, P7, P8);
  maxP = max(P1, P2, P3, P4, P5, P6, P7, P8);
  if (P0 < minP) Q0 = minP;
  else if (P0 > maxP) Q0 = maxP;
  else Q0 = P0;
}
```

Filter

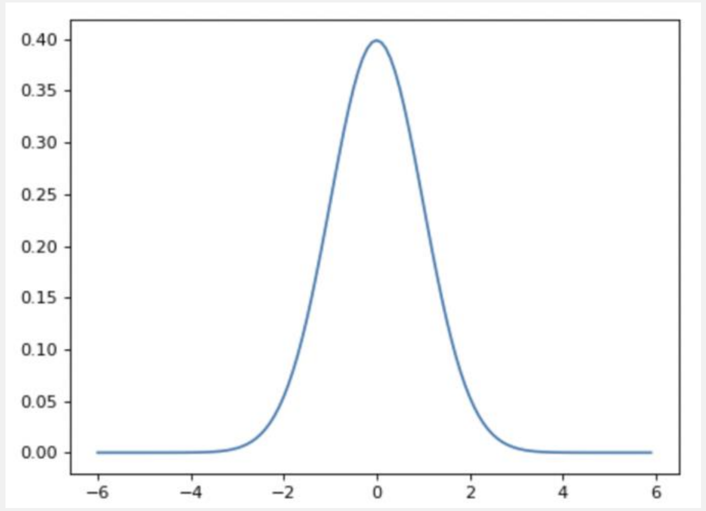
Low-pass filter allows low frequency signals to pass through while suppressing high frequency signals

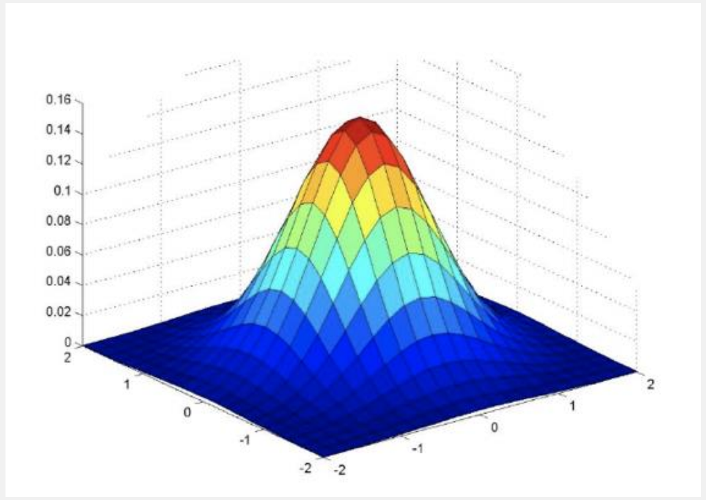
- Gaussian filter
  - 1-D dimension

$$f(x) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp(-\frac{x^2}{2\sigma^2})$$

  - 2-D dimension

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$





Filter

Low-pass filter allows low frequency signals to pass through while suppressing high frequency signals

- ➢ Gaussian filter
  - ◆ 2-D dimension
    - ✓ Kernel_size = 3, sigma = 0.707 ($\sqrt{0.5}$)

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \text{ and Normalize}$$

$$\begin{bmatrix} (-1,-1) & (0,-1) & (1,-1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,1) & (0,1) & (1,1) \end{bmatrix} \longrightarrow \begin{bmatrix} 0.045 & 0.122 & 0.045 \\ 0.122 & 0.332 & 0.122 \\ 0.045 & 0.122 & 0.045 \end{bmatrix}$$

Filter

Low-pass filter allows low frequency signals to pass through while suppressing high frequency signals

- ➢ Gaussian filter
  - ◆ 2-D dimension
    - ✓ Kernel_size = 3, σ= 0.707 ($\sqrt{0.5}$)

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$ and Normalize

$$\begin{bmatrix} (-1,-1) & (0,-1) & (1,-1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,1) & (0,1) & (1,1) \end{bmatrix} \longrightarrow \begin{bmatrix} 0.045 & 0.122 & 0.045 \\ 0.122 & 0.332 & 0.122 \\ 0.045 & 0.122 & 0.045 \end{bmatrix}$$

$$\begin{bmatrix} (-1,-1) & (0,-1) & (1,-1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,1) & (0,1) & (1,1) \end{bmatrix} \longrightarrow \begin{bmatrix} 0.135 & 0.368 & 0.135 \\ 0.368 & 1 & 0.368 \\ 0.135 & 0.368 & 0.135 \end{bmatrix} \xrightarrow{/\ 3.012} \begin{bmatrix} 0.045 & 0.122 & 0.045 \\ 0.122 & 0.332 & 0.122 \\ 0.045 & 0.122 & 0.045 \end{bmatrix}$$

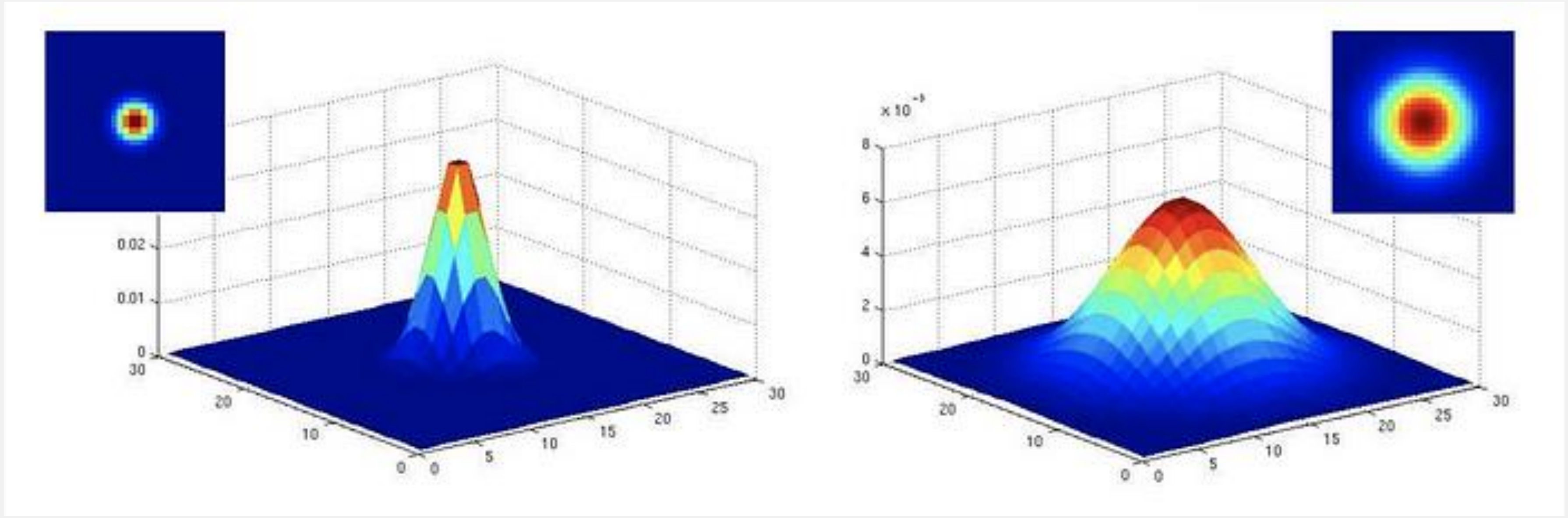
$$sum\left(\begin{bmatrix} 0.135 & 0.368 & 0.135 \\ 0.368 & 1 & 0.368 \\ 0.135 & 0.368 & 0.135 \end{bmatrix}\right) = 3.012$$

Filter

Low-pass filter allows low frequency signals to pass through while suppressing high frequency signals

- ➢ Gaussian filter
  - ◆ How to determine kernel size? The kernel size is dependent of standard deviation σ of Gaussian function -> k ≈ 6σ+1

σ = 2 σ = 5



Filter

High-pass filter (HPF) allows high frequency signals to pass through while suppressing low frequency signals

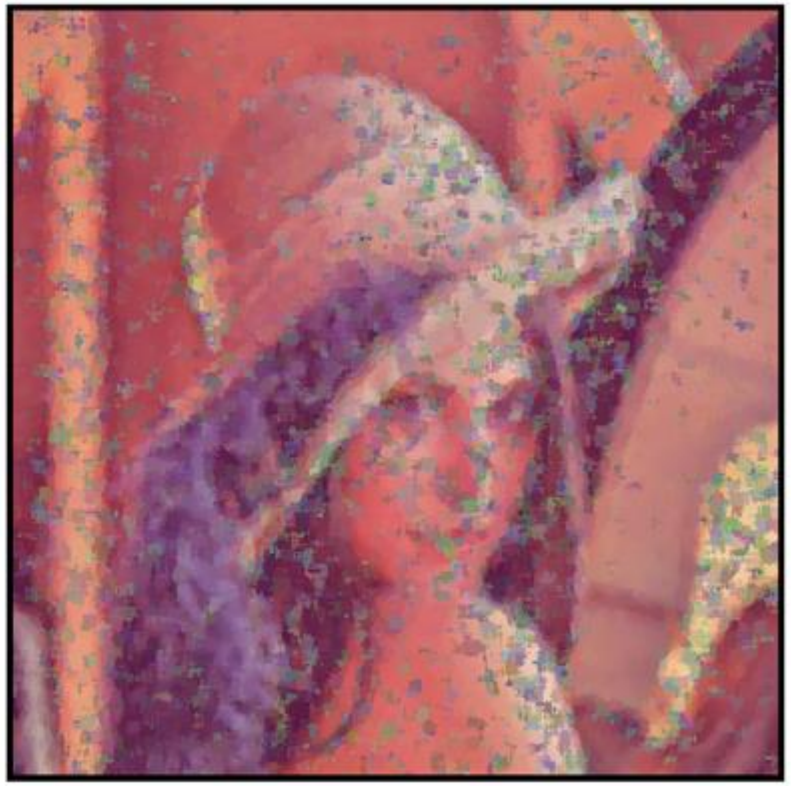
In image preprocessing, HPF is used for edge detection and detail enhancement which could highlight rapid changes and detailed features

- ➢ Sobel filter
- ➢ Laplacian filter
- ➢ Prewitt filter
- ➢ Unsharp filter
  - ◆ Enhance details by subtracting blurry images
  - ◆ Method
    - ✓ Smooth image
    - ✓ Result in high-frequency details by subtracting original image from smoothed image
    - ✓ Add high-frequency details back to original image

Filter

- Sharp and unsharp masking

(A)


(B)

(C)

Filter

Color in image filtering

- Problem
  - Color distortion
    - ✓ Most image filters are designed for grayscale images and may process each color channel (e.g., RGB) separately when applied to color images
    - ✓ This can lead to inconsistencies between the color channels, resulting in unnatural color changes and distortion
  - Improper color space selection
    - ✓ Processing images directly in the RGB color space can amplify differences between the color channels
  - Computation complexity
    - ✓ Color images have multiple color channels (typically three: red, green, blue), and each channel needs to be processed individually, increasing computational complexity and processing time
  - Boundary effect
    - ✓ Some filters may produce artifacts when processing edge regions, leading to the loss or blurring of edge details

Filter

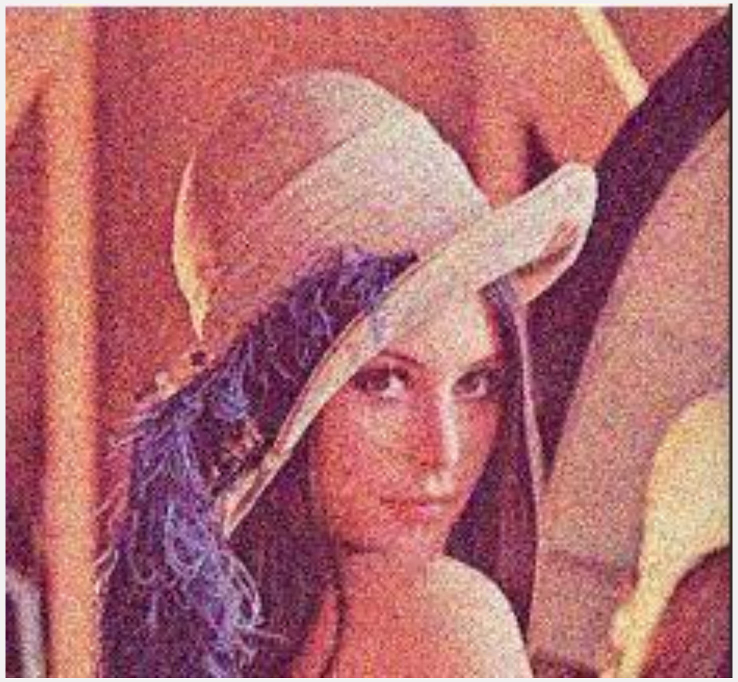
Color in image filtering

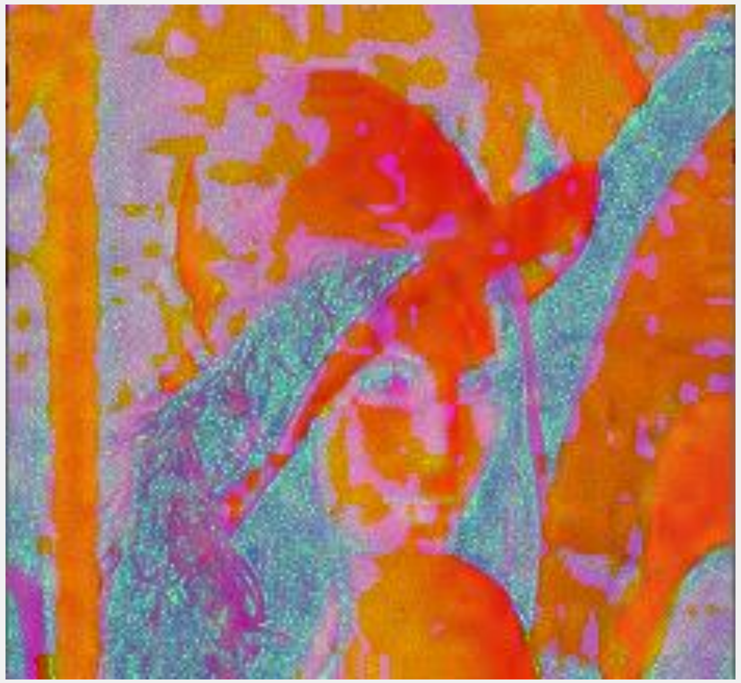
- ➢ Some solutions
  - ◆ Change to other color space
    - ✓ ex: RGB to HSV or YCbCr
  - ◆ Joint processing
    - ✓ Process all color channels simultaneously to maintain the relative relationships between colors
    - ✓ ex: Joint bilateral filtering, Joint Gaussian filtering, …etc.
  - ◆ Specially designed color filter
    - ✓ Use filters specifically designed for color images that consider the relationships between color channels, effectively reducing color distortion and artifacts
    - ✓ ex: Independent channel filtering, vector filtering, non-local means filter, …etc.

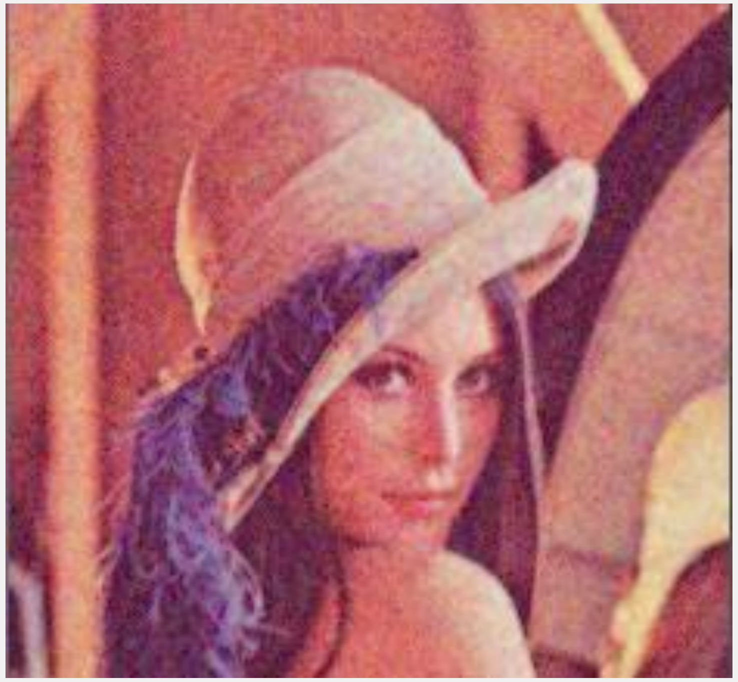
Filter

Color in image filtering

- RGB to HSV


Noise


HSV


Noise Reduction

Filter

Color in image filtering

- RGB to HSV
  - Methods
    - ✓ Channel normalize

      $R' = R / 255 \quad G' = G / 255 \quad B' = B / 255$

    - ✓ Find max and min

      $C_{max} = \max(R', G', B') \quad C_{min} = \min(R', G', B')$

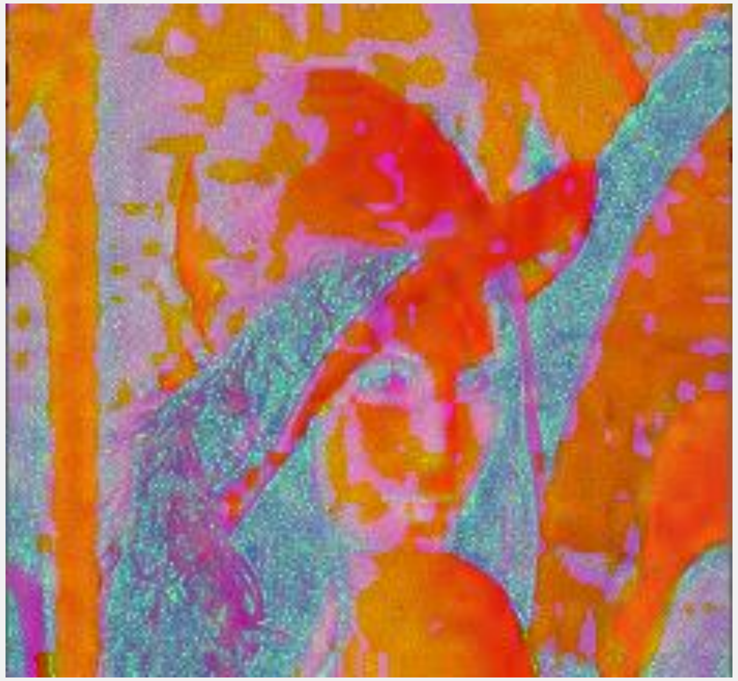
      $\Delta = C_{max} - C_{min}$

    - ✓ HSV

$$h = \begin{cases} 0 & if\ \Delta = 0 \\ 60 \times (\frac{G' - B'}{\Delta}\ mod\ 6) & if\ C_{max} = R' \\ 60 \times (\frac{B' - R'}{\Delta} + 2) & if\ C_{max} = G' \\ 60 \times (\frac{R' - G'}{\Delta} + 4) & if\ C_{max} = B' \end{cases}$$

$$s = \begin{cases} 0 & if\ C_{max} = 0 \\ \frac{\Delta}{C_{max}} & otherwise \end{cases}$$

$$v = C_{max}$$



Filter

Color in image filtering

- HSV to RGB
  - Methods
    - ✓ Calculate the interval where the hue lies

      $C = v \times s \qquad H' = H / 60 \qquad X = C \times (1-|(H' \bmod 2) -1|)$

    - ✓ Calculate the middle value of RGB

$$(R_1, G, B_1) = \begin{cases} (C, X, 0) & if\ 0 \le H' < 1 \\ (X, C, 0) & if\ 1 \le H' < 2 \\ (0, C, X) & if\ 2 \le H' < 3 \\ (0, X, C) & if\ 3 \le H' < 4 \\ (X, 0, C) & if\ 4 \le H' < 5 \\ (C, 0, X) & if\ 5 \le H' < 6 \end{cases}$$

    - ✓ Final RGB value

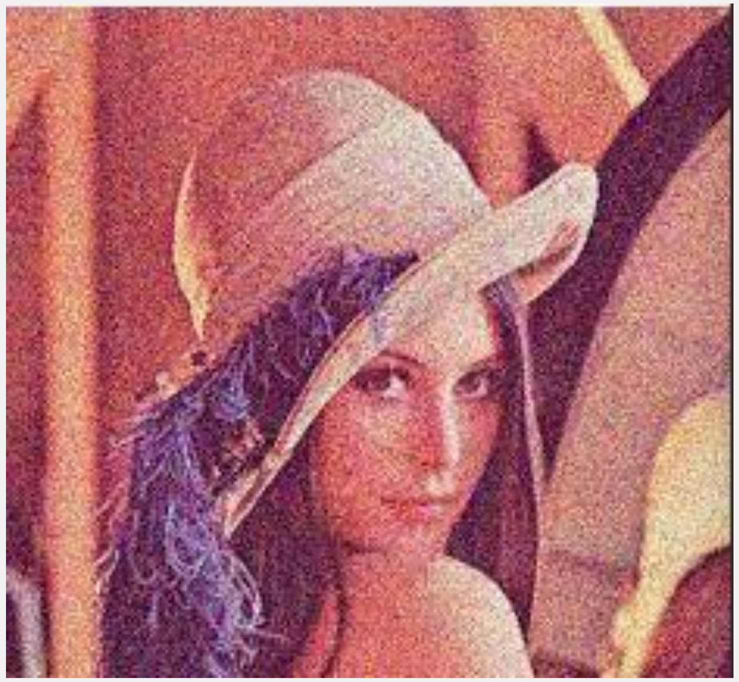
$$m = v - c$$

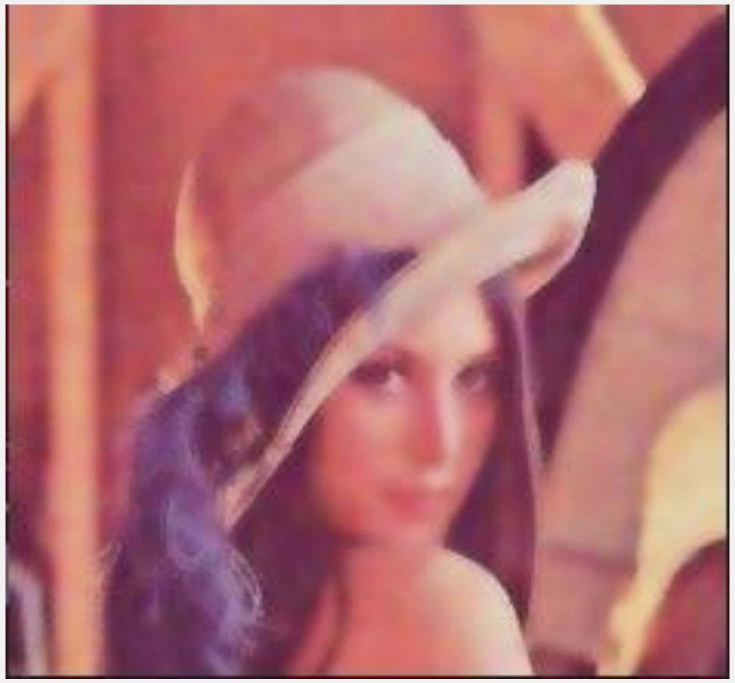
$$(R, G, B) = (R_1+m, G_1+m, B_1+m)$$

Filter

Color in image filtering

- Joint bilateral filtering


Noise


Noise Reduction

Filter

Color in image filtering

- Joint bilateral filtering
  - Spatial weight: determined by the Euclidean distance between pixels

$$\omega_s(p,q) = \exp(-\frac{||p-q||^2}{2\sigma_s^2})$$

  - Intensity weight: determined by the difference between pixels

$$\omega_r(I(p),G(q)) = \exp(-\frac{(I(p)-G(q))^2}{2\sigma_s^2})$$

where $\boxed{\text{G}}$ is the guide image

  - Joint weight: determined by the product of spatial weight and intensity weight

$$\omega(p,q) = \omega_s(p,q) \times \omega_r(I(p),G(q))$$

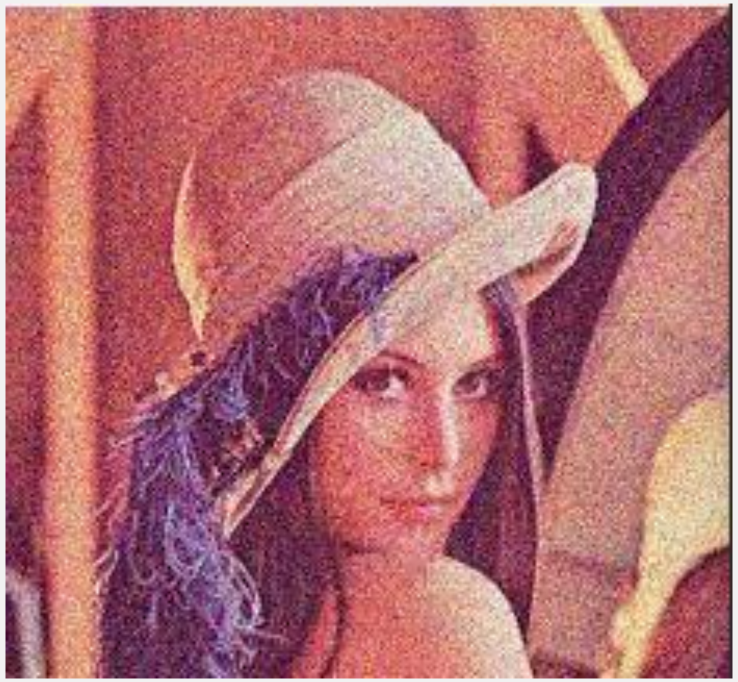
  - Filtered pixel: determined by the weighted average of all pixels in the neighborhood

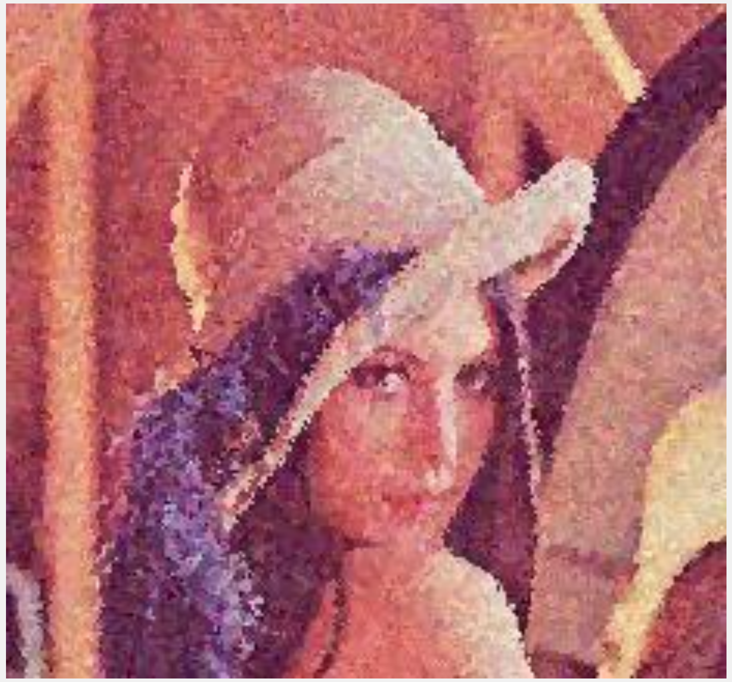
$$J(p) = \frac{\sum_{q\in N_p}\omega(p,q)\times I(p)}{\sum_{q\in N_p}\omega(p,q)}$$

Filter

Color in image filtering

- Vector (Median) filtering
  - It reduces noise by sorting each pixel and the pixel within its neighborhood and picking the pixel closest to the median


Noise


Noise Reduction

Filter

Color in image filtering

- Vector median filtering
  - Define vector data: suppose in a k×k window, the k^2 pixel vector are $v_i=(R_i, G_i, B_i)$
  - Distance calculation: For each vector vi, calculate the distance between it and all other vectors in the window by the Euclidean distance

$$d(v_i, v_j) = \sqrt{(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2}$$

  - Sum distance: determined by the sum of distances between center and neighborhood pixels

$$D(v_i) = \sum_j d(v_i, v_j)$$

  - Select median value: determined by the smallest distance sum as the vector median of the neighborhood