Class: III year – I sem
Dept: CSM
AY: 2025-2026
Subject: CVIP

**UNIT I: Introduction to Computer Vision and Image Processing**

Overview of Computer Vision and Image Processing: Definitions and scope, Historical development and applications, Image Formation and Representation: Image acquisition methods, Sampling and quantization, Color spaces and models, Fundamentals of Image Processing: Point operations (brightness and contrast adjustments), Histogram processing, Spatial filtering techniques Fourier Transform and Frequency Domain Processing: Discrete Fourier Transform (DFT), Filtering in the frequency domain, Image restoration concept.

## Definitions and Scope

### Computer Vision

**Definition:** A field of Artificial Intelligence (AI) and Computer Science that enables computers to interpret and make decisions based on visual data (images and videos).

**Goal:** To automate tasks that the human visual system can do — object detection, recognition, scene understanding, etc.

## Definitions and Scope

## Image Processing

**Definition:** A technique used to enhance or manipulate images to prepare them for further analysis or presentation.

**Goal:** To improve image quality or extract useful information using mathematical and algorithmic operations.

**Scope**

**Computer Vision:** High-level understanding — object recognition, scene interpretation, motion tracking, facial recognition.

**Image Processing:** Low-level operations — noise removal, filtering, enhancement, segmentation, compression.

## Historical development

| Time Period | Milestones |
| --- | --- |
| **1950s-1960s** | Early analog image processing and pattern recognition systems. |
| **1970s-1980s** | Digital image processing emerges with development in computers. Edge detection, filtering, and segmentation techniques developed. |
| **1990s** | Growth in object detection and recognition; real-time systems start appearing. |
| **2000s** | Rise of machine learning in vision; applications in biometrics, surveillance. |
| **2010s** | Deep Learning revolutionizes computer vision (CNNs, YOLO, Faster R-CNN). |
| **2020s** | Real-time and embedded vision, generative AI (e.g., diffusion models), multimodal systems (e.g., CLIP, SAM), and self-supervised learning. |

## Applications

**Computer Vision Applications:**

- **Healthcare:** Medical image analysis (e.g., tumor detection)
- **Automotive:** Self-driving cars (object and lane detection)
- **Retail:** Automated checkout, shelf stock monitoring
- **Agriculture:** Crop health monitoring via drone imaging
- **Security:** Surveillance, face recognition
- **Manufacturing:** Quality control and defect detection

**Image Processing Applications:**

- **Satellite Imaging:** Weather forecasting, environmental monitoring
- **Photo Editing:** Denoising, sharpening, artistic filters
- **Compression:** JPEG, PNG image compression for storage/transmission
- **Restoration:** Old photo recovery, deblurring

## Applications

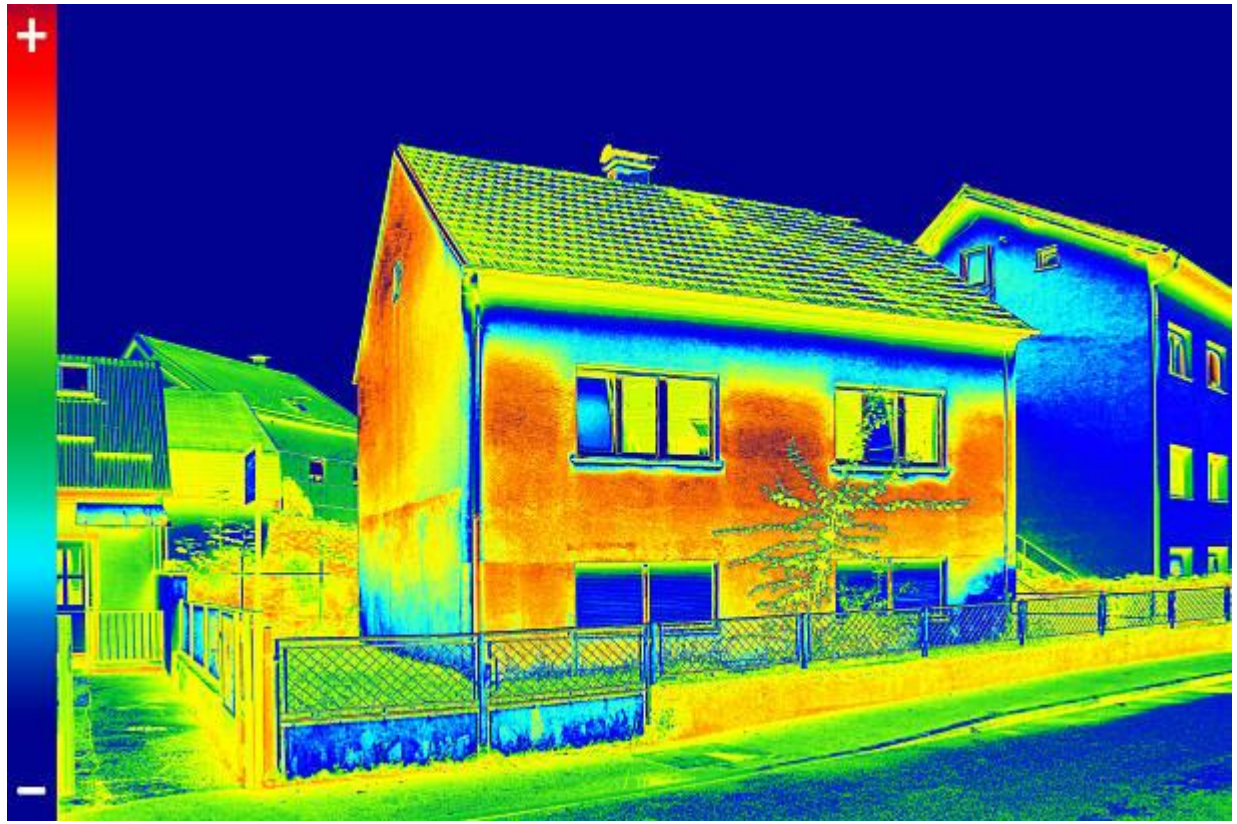



Figure: Thermal image

# Image Formation and Representation

## Image Formation and Representation

### Image Formation

•Images are formed by capturing light from a scene using sensors (CCD/CMOS) in cameras.
•The scene's **illumination**, **surface reflectance**, and **camera parameters** influence the image.

### Image Representation

•An image is a 2D function **f(x, y)**, where *x* and *y* denote spatial coordinates and *f* is the intensity or color.
•**Grayscale Image:** Single channel; values range from 0 (black) to 255 (white).
•**Color Image:** Typically 3 channels (RGB); each pixel has red, green, and blue components.
•**Binary Image:** Pixels are either 0 or 1 (black or white) — used in masks or segmentation.
•**Bit Depth:** Indicates color precision (e.g., 8-bit, 16-bit).

| Aspect | Image Processing | Computer Vision |
| --- | --- | --- |
| Focus | Image enhancement/analysis | Understanding visual content |
| Level | Low-level operations | High-level interpretation |
| Techniques | Filtering, thresholding, etc. | Object detection, classification |
| Tools/Libraries | OpenCV, PIL, skimage | OpenCV, TensorFlow, PyTorch, YOLO |

# COMPUTER VISION AND IMAGE PROCESSING

## DEFINITIONS AND SCOPE

**Computer Vision**
Field of artificial intelligence focused on en-abling comput-ers to interpret and understan tasks

**Image Processing**
Technique for enhancing manipulati-mg images to prepare them for higher-level tasks

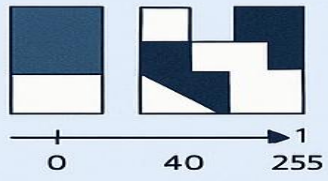
### SCOPE

| | |
|---|---|
| **1950s-1960s** | Early developments |
| **1970s-1980s** | Basic techniques |
| **1990s** | Object recognition |
| **2000s** | Machine learning |
| **2010s** | Deep learning |
| **2020s** | Real-time vision |

## HISTORICAL DEVELOPMENT AND APPLICATIONS

**Time peirods**
1150s-19660s
1970s-1980s
2000s
2010s
2020s

B-image

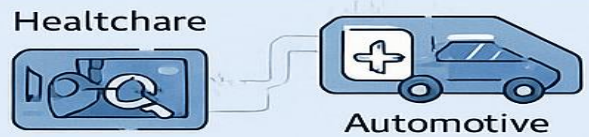



## IMAGE PROCESSING

Technique for enhancing and manipulating images to prepare thern for highei-levl tasks

### SCOPE

Healtchare



Automotive

Retail

Security

Manufact-uring

## IMAGE FORMATION AND REPRESENTATTON

Image as a 2D function $f(x.y)$

- Images are captured wita camera
- influences factors: illumination, refectance, camera seltings

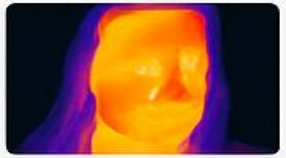
Grayscale Image

Color Image

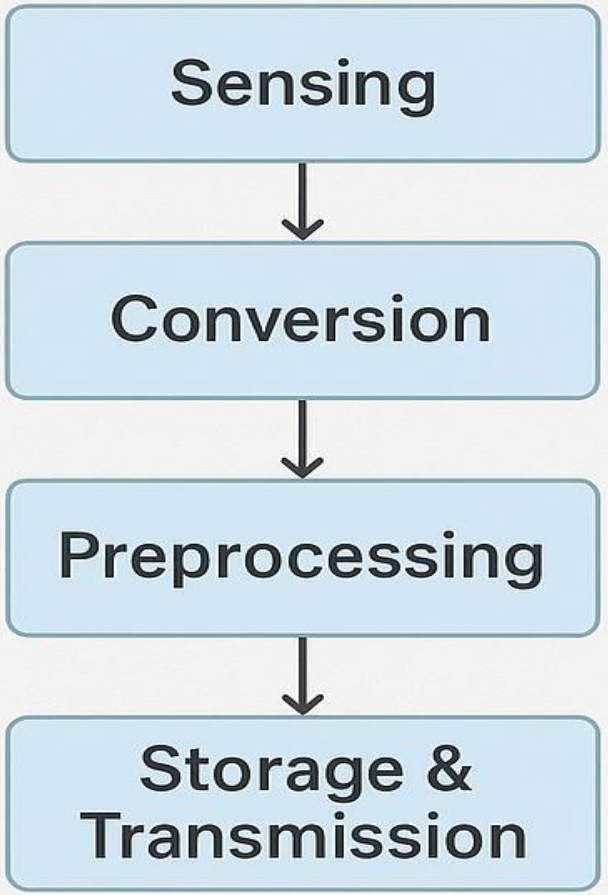
Binary Image

## Image Acquisition:

Image acquisition is the process of **capturing an image of a scene or object** using a device that converts real-world visual information into a digital form for further processing, analysis, or display.

| IMAGE ACQUISITION METHODS | |
|---|---|
| METHOD | EXAMPLE IMAGE |
| Optical Imaging |  |
| Radiological Imaging |  |
| Magnetic & Radiofrequency Imaging |  |
| Ultrasonic Imaging |  |
| Thermal & Infrared Imaging |  |
| Other Methods |  |

# IMAGE ACQUISITION



## Methods of Image Acquisition

### A. Optical Imaging

- **Cameras (Visible Light)**

*Standard digital cameras (DSLRs, smartphone cameras, webcams).*

- **Microscopy**

*Light microscopes, fluorescence microscopes, confocal microscopes.*

- **Endoscopy**

### B. Radiological Imaging

- **X-Ray**

*Captures images of dense structures inside the body.*

- **Computed Tomography (CT)**

*A series of X-ray images combined into cross-sectional views.*

- **Mammography**

*Specialized X-ray for breast tissue.*

## C. Magnetic & Radiofrequency Imaging

- **Magnetic Resonance Imaging (MRI)**
  *Uses strong magnetic fields and radio waves.*
- **Nuclear Medicine**
  *PET and SPECT: Use radioactive tracers to form images.*

## D. Ultrasonic Imaging

- **Ultrasound**
  *Uses high-frequency sound waves to produce real-time images (e.g., fetal scans, echocardiography).*

## E. Thermal & Infrared Imaging

- **Infrared Cameras**
  *Detect heat patterns for industrial, medical, or surveillance use.*

## F. Remote Sensing

- **Satellite Imaging**

*Earth observation via satellites.*

- **Aerial Drones**

*Capture high-resolution terrain images.*

## G. Other Methods

- **Scanning Electron Microscopy (SEM)**

*Captures surface details at high magnification.*

- **Atomic Force Microscopy (AFM)**

*Uses a physical probe to map surfaces at nanometer scale.*

## 3. General Steps in Image Acquisition

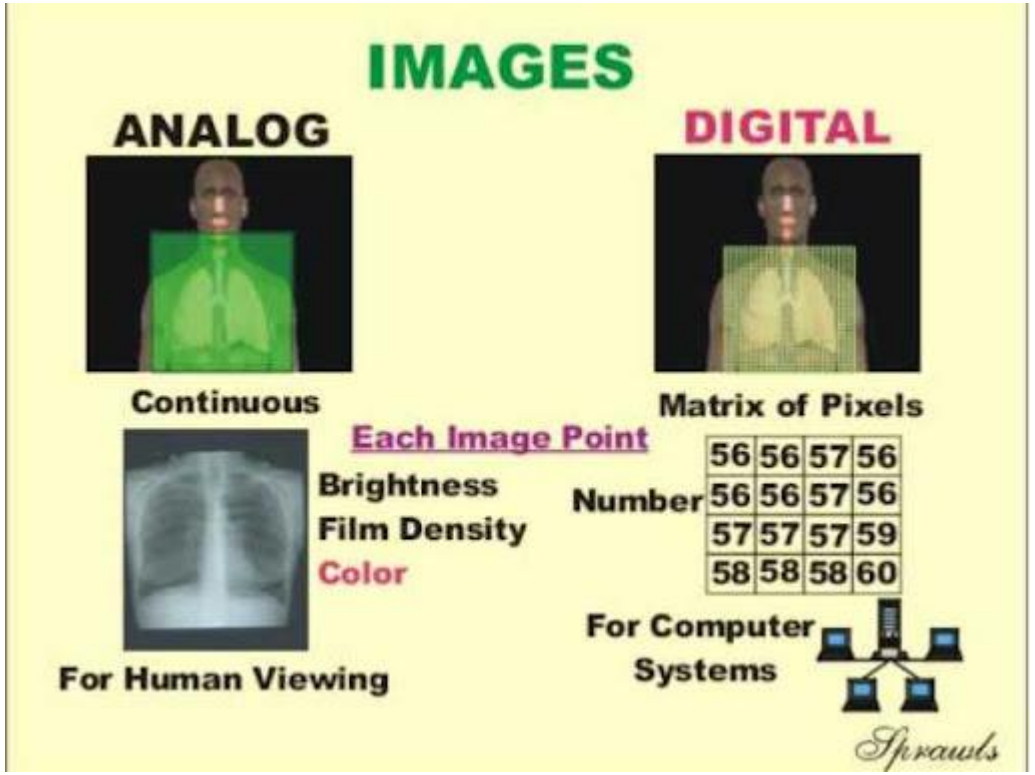
**Sensing** → Using a sensor (CCD, CMOS, etc.) to detect light, radiation, sound, or electrons.
**Conversion** → Analog signal converted to digital data.
**Preprocessing** → Noise reduction, contrast adjustment, calibration.
**Storage & Transmission** → Saving image data for analysis.

## Key Factors in Choosing a Method

- What physical property is being imaged? (Light, sound, radiation, electrons)
- Required resolution & contrast
- Cost & availability of equipment
- Safety (e.g., ionizing radiation)
- Application (medical, industrial, research, satellite, etc.)

## Sampling and Quantization in Image Processing

In digital image processing, sampling and quantization are two fundamental steps used to convert an analog image (or continuous image signal) into a digital image that a **computer can store, process, and display**.



# Sampling and Quantization in Image Processing

## Sampling

**Sampling** refers to measuring the intensity values of an image at specific, discrete points.

An analog image is continuous in both its spatial dimensions (x and y) and in intensity (brightness or color).

When we **sample**, we divide the image into a grid of pixels and measure the intensity at each grid point.

The result is a finite set of pixel values that approximates the original image.

**Key point:** The **sampling rate** (or resolution) determines how many samples are taken per unit area. Higher sampling rates mean more pixels, leading to better spatial resolution and more detail — but also larger file sizes.

## Sampling and Quantization in Image Processing

In digital image processing, sampling and quantization are two fundamental steps used to convert an analog image (or continuous image signal) into a digital image that a computer can store, process, and display.

### Quantization

**Quantization** is the process of assigning each sampled intensity value to a discrete digital value.

While the sampled points still have continuous intensity values, computers can only store discrete numbers.

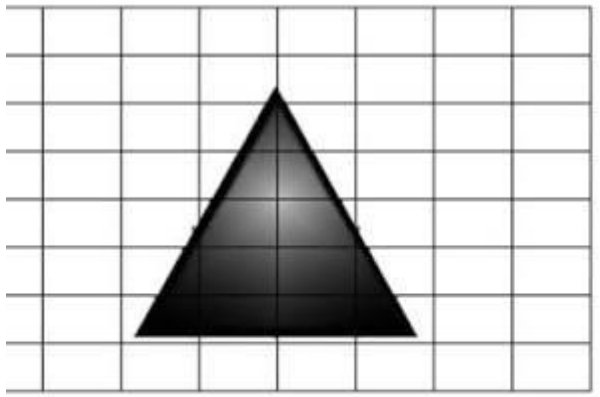
So, each intensity is mapped to the nearest level within a limited range of possible values.

For example:

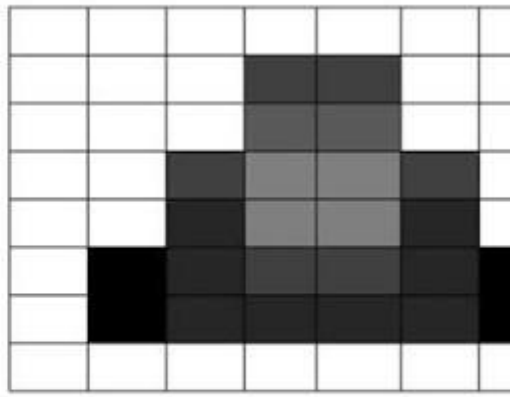
A grayscale image might use 8 bits per pixel, allowing 256 possible intensity levels (0–255).
More bits mean finer intensity levels, leading to smoother gradations in shades or colors.

## Sampling and Quantization in Image Processing

In digital image processing, sampling and quantization are two fundamental steps used to convert an analog image (or continuous image signal) into a digital image that a computer can store, process, and display.



Continuous image projected onto a sensor array.



Result of image sampling and quantization.

## Sampling and Quantization in Image Processing

In digital image processing, sampling and quantization are two fundamental steps used to convert an analog image (or continuous image signal) into a digital image that a computer can store, process, and display.

### Combined Effect

**Sampling** determines **how many pixels** represent the image.

**Quantization** determines **how many intensity or color levels** each pixel can have.

If the sampling is too low, the image appears **blocky or pixelated** (loss of detail).
If the quantization levels are too few, the image shows **banding or false contours** (loss of smooth gradients).

## Color spaces and Models

In image processing, **color spaces** and **color models** are essential for representing and manipulating colors in a consistent way

**Color Model**: A mathematical way of describing colors using a set of primary colors (e.g., RGB, CMY).

**Color Space**: A specific implementation of a color model with defined ranges and properties (e.g., sRGB, Adobe RGB).
Think of:

**Color model** = concept or theory.

**Color space** = actual instance or practical version of that model.

## Color spaces and Models

In image processing, **color spaces** and **color models** are essential for representing and manipulating colors in a consistent way

**What is a color ?**

# color Space and models

A **color model** is a mathematical way to represent colors as sets of numbers, typically as tuples of values.
It defines **how colors can be described** — but doesn't necessarily describe how they are displayed.
**Examples:**
- **RGB model:** Uses Red, Green, and Blue components.
- **CMY(K) model:** Cyan, Magenta, Yellow (+ Black for CMYK in printing).
- **HSI/HSV/HSL models:** Hue, Saturation, Intensity/Value/Lightness — closer to how humans perceive color.
- **YUV/YIQ:** Separates luminance (Y) and chrominance (U, V) — common in video.

A **color space** is a specific implementation of a color model — it **defines a range (gamut) of colors**, and how the numeric representation maps to real colors.
In other words, the **model is the theory**, the **space is the practical implementation**.
**Examples:**
- **sRGB:** A standard RGB color space for monitors, printers, web.
- **Adobe RGB:** A wider RGB color space used in photography and printing.
- **ProPhoto RGB:** Even wider gamut — professional photography.
- **YCbCr:** Used in JPEG and video compression — separates brightness (Y) from chroma (Cb, Cr).
- **CIELAB (Lab):** Perceptually uniform — used for color correction.

# Common Color spaces and Models

## 1. RGB (Red, Green, Blue)

**Type**: Additive color model.
**Used In**: Monitors, digital cameras, scanners.
**Range**: Each channel typically 0–255 in 8-bit images.
**sRGB**: A common RGB color space for standard displays.
**Limitation**: Not perceptually uniform — difficult to perform color analysis.

## 2. HSV / HSL (Hue, Saturation, Value/Lightness)

**Type**: Cylindrical transformation of RGB.
**Used In**: Color selection tools, image segmentation.
**Hue**: Color type (0–360°)
**Saturation**: Intensity (0–100%)
**Value / Lightness**: Brightness level

**Advantage**: Easier to isolate colors than RGB (e.g., detect red objects).

Common Color spaces and Models

## 3. CMY / CMYK (Cyan, Magenta, Yellow, Key/Black)

**Type**: Subtractive color model.
**Used In**: Printing industry.
**CMY**: For absorbing red, green, and blue light.
**CMYK**: Adds black to enhance contrast and save ink.

## 4. YCbCr / YUV

**Type**: Luminance and chrominance model.
**Used In**: Video compression (JPEG, MPEG).
**Y**: Luminance (brightness)
**Cb**: Blue-difference chroma
**Cr**: Red-difference chroma

Common Color spaces and Models

## 5. Lab Color (CIELAB)

**Type**: Perceptually uniform model by the CIE.
**Used In**: Color comparison, high-quality image editing.
**L***: Lightness
**a***: Green to Red
**b***: Blue to Yellow
**Key Feature**: Closely models human vision; great for measuring color differences (ΔE).

## 6. Grayscale

**Type**: Single-channel color model.
**Used In**: Simplified image processing (edge detection, segmentation).
**Value Range**: 0 (black) to 255 (white)

## Choosing the Right Color Space

| Task | Recommended Color Space |
| --- | --- |
| Object detection | HSV, Lab |
| Face detection | YCrCb |
| Printing | CMYK |
| Display | RGB / sRGB |
| Compression | YCbCr |
| Image comparison | Lab |

Choosing the Right Color Space

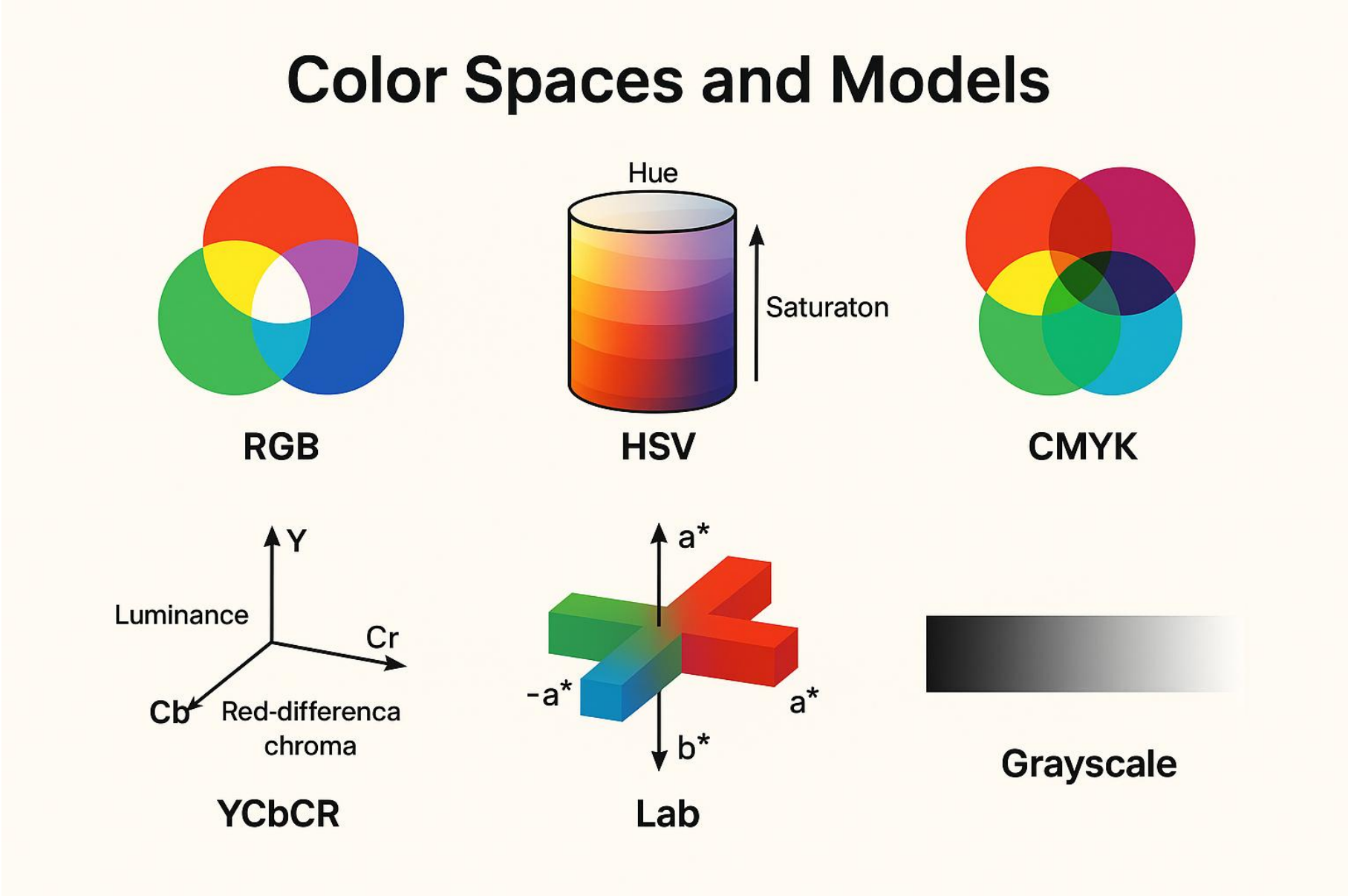


Image processing fundamentals:

**What are Point Operations?**

**Point Operations** are image processing operations where **each output pixel** depends **only on the value of the corresponding input pixel** — not its neighbors.

**Key idea:**
No context or neighborhood is involved — every pixel is transformed **independently**.

**Common examples:**

•Brightness adjustment

•Contrast adjustment

•Image negative

•Thresholding

Image processing fundamentals:

**Brightness Adjustment**

**Definition:**
Brightness means shifting pixel intensities up or down by adding/subtracting a constant.

**Mathematical expression:**

$g(x,y) = f(x,y) + b$

Where:
f(x,y) is the original pixel value.
b is the bias (brightness offset).
g(x,y) is the new pixel value.

**Effect:**
**Positive b:** image becomes brighter. **Negative b:** image becomes darker.

Image processing fundamentals:

**Contrast Adjustment**

**Definition:**
Contrast means stretching or compressing the range of pixel intensities.

**Mathematical expression:**
$g(x,y) = a \cdot f(x,y)$

Where:
a is the gain factor.

**Effect:**

**a > 1:** increases contrast.
**0 < a < 1:** reduces contrast.

Image processing fundamentals:

**Use Cases**

Brightness: Correct underexposed or overexposed images.
Contrast: Make features stand out — useful in medical or satellite images.
Common pre-processing step: Helps downstream tasks like edge detection.

**Related Point Operations**

Other point operations include:

- **Image negative:** g = 255 - f → Inverts colors.
- **Gamma correction:** Non-linear transformation for display adjustment.
- **Thresholding:** Convert to binary (black and white) image based on a cutoff.

Image processing fundamentals:

**Use Cases**

Brightness: Correct underexposed or overexposed images.
Contrast: Make features stand out — useful in medical or satellite images.
Common pre-processing step: Helps downstream tasks like edge detection.

**Related Point Operations**

Other point operations include:

- **Image negative:** g = 255 - f → Inverts colors.
- **Gamma correction:** Non-linear transformation for display adjustment.
- **Thresholding:** Convert to binary (black and white) image based on a cutoff.

# Image processing fundamentals:

Original Gradient

Brightness Increased (+50)

Contrast Increased (x1.5)

Point Operation Transfer Functions

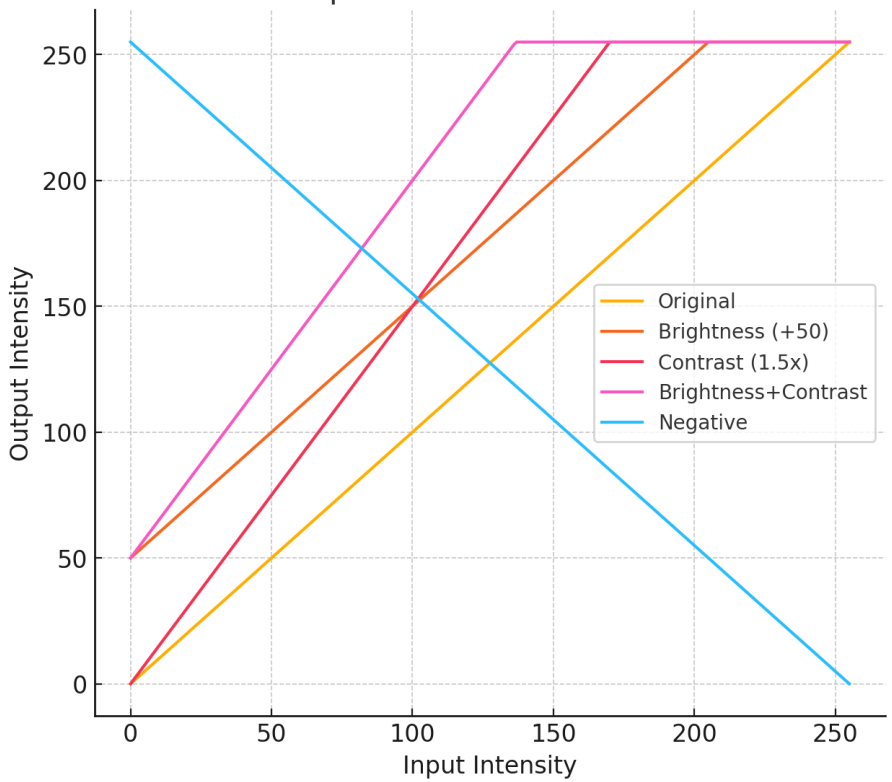


Image processing fundamentals:

**What is a Histogram in Image Processing?**

A **histogram** in image processing is a graphical representation of the distribution of pixel intensities in an image. For a grayscale image:

- The **x-axis**: intensity values (0–255 for 8-bit images).
- The **y-axis**: number of pixels at each intensity.

**Example:**

- A dark image → histogram bunched toward 0.
- A bright image → histogram shifted toward 255.
- Low-contrast → histogram is narrow.
- High-contrast → histogram is spread out.

Image processing fundamentals:

**Why Process the Histogram?**

**Histogram processing** adjusts the intensity distribution to:

- Improve contrast.
- Enhance details.
- Correct exposure problems.
- Normalize lighting conditions.

**Key idea:** Operate on pixel values to spread or redistribute them for better visibility.

## Common Histogram Processing Techniques

### A) Histogram Stretching (Contrast Stretching)

•Goal: **Expand** the range of pixel intensities.

•Example:
- Original: pixel values from 50–18

•Formula: $$I_{new} = \frac{I_{old} - I_{min}}{I_{max} - I_{min}} \times 255$$

•This is a **linear point operation**.
**Effect:** Makes darks darker and brights brighter.

**B) Histogram Equalization**

•Goal: Redistribute intensities to **flatten** the histogram → more uniform distribution.
•Non-linear operation.
•Enhances global contrast, especially for images that are too bright or too dark.

**How it works:**
•Calculate the **cumulative distribution function (CDF)**.
•Map old pixel values to new ones using the CDF.

**Result:** Details in underexposed or overexposed regions become more visible.

## C) Adaptive Histogram Equalization (AHE)

- Improves on basic equalization.
- Instead of using the whole image, it divides the image into small blocks (tiles) and equalizes each one.
- Handles varying lighting.

**CLAHE (Contrast Limited AHE):**

- Limits the amplification to avoid noise over-enhancement.

# Image processing fundamentals: Spatial filtering

## Spatial Filtering

**Spatial filtering** is an image processing technique where we **modify pixel values based on the values of neighboring pixels** using a **filter mask (kernel)**.

A spatial filter slides over an image and **computes a new value** for each pixel using the neighborhood defined by the filter.

Spatial filters are used for:

- **Smoothing** (noise reduction, blur)
- **Sharpening** (edge enhancement)
- **Edge detection** (detecting boundaries)
- **Feature extraction**
- **Image enhancement**

# Image processing fundamentals: Spatial filtering

**Spatial Filter Working:**

- Choose a **kernel** (mask) → a small matrix, e.g., 3×3, 5×5.
- Place the kernel over each pixel neighborhood.
- Multiply the kernel values with the corresponding image pixel values.
- Sum them → assign result to the central pixel.
- This is **spatial convolution** or **correlation**.

They are broadly divided into two categories:

- **A) Smoothing Filters**
- **B) Sharpening Filters**

# Image processing fundamentals: Spatial filtering

## A) Smoothing (Low-Pass) Filters

Reduce noise, smooth variations, blur the image.

- **1. Mean Filter (Averaging Filter)**

Replace each pixel with the **average** of its neighbors.

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Kernel example (3×3):

Effect: Blurs details, reduces noise.

# Image processing fundamentals: Spatial filtering

**1. Mean Filter Formula**

Given an image I(x,y) and a mean filter kernel of size N×N, the output image I'(x,y) at each pixel is calculated as:

$$I'(x,y) = \frac{1}{N^2} \sum_{i=-k}^{k} \sum_{j=-k}^{k} I(x+i, y+j)$$

Where:

- $N$ is the size of the kernel (e.g., $3$ for a $3 \times 3$ kernel)
- $k = \lfloor \frac{N}{2} \rfloor$
- $(x, y)$ is the coordinate of the pixel being processed
- $I(x+i, y+j)$ are the neighboring pixels in the $N \times N$ window

# Image processing fundamentals: Spatial filtering

Let:

- $I(x, y)$ be a 3x3 region centered at (x, y)
- The 3x3 values be:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Then the mean-filtered value at the center is:

$$I'(x, y) = \frac{1}{9}(a + b + c + d + e + f + g + h + i)$$

The **3×3 mean filter kernel** $H$ is:

$$H = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Then mean filtering is equivalent to **convolution** of the image $I$ with kernel $H$:

$$I' = I * H$$

Where $*$ denotes the convolution operation.

# Image processing fundamentals: Spatial filtering

Let:

- $I(x, y)$ be a 3x3 region centered at (x, y)
- The 3x3 values be:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Then the mean-filtered value at the center is:

$$I'(x, y) = \frac{1}{9}(a + b + c + d + e + f + g + h + i)$$

The **3×3 mean filter kernel** $H$ is:

$$H = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Then mean filtering is equivalent to **convolution** of the image $I$ with kernel $H$:

$$I' = I * H$$

Where $*$ denotes the convolution operation.

# Image processing fundamentals: Spatial filtering

**Smoothing filters** are a category of image processing filters used to **reduce noise**, **remove fine details**, and **blur images**. They work by modifying the pixel values based on their neighborhood, making the image look **softer** and **less sharp**.

- Reduce **random noise** (especially Gaussian or salt-and-pepper noise)
- Preprocess images before **edge detection**, **segmentation**, or **feature extraction**
- Eliminate small, irrelevant details in an image

# Image processing fundamentals: Spatial filtering

| Filter Type | Linear/Non-Linear | Description |
|---|---|---|
| **Mean Filter** | Linear | Replaces pixel with average of neighborhood |
| **Gaussian Filter** | Linear | Uses a Gaussian function to give more weight to central pixels |
| **Median Filter** | Non-linear | Replaces pixel with the **median** of neighborhood (great for salt & pepper noise) |
| **Bilateral Filter** | Non-linear | Smooths while **preserving edges** using both spatial and intensity differences |
| **Box Filter** | Linear | Simple averaging filter, similar to mean filter |

## A) Smoothing (Low-Pass) Filters

### 2. Box Filter

Same as mean filter — uses uniform weights.

Simple but can blur edges.

### 3. Gaussian Filter

- Uses a **Gaussian function** for weights.
- Gives more weight to center pixels → smoother, natural blur.

Kernel example (3×3, 5×5):

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} /16$$

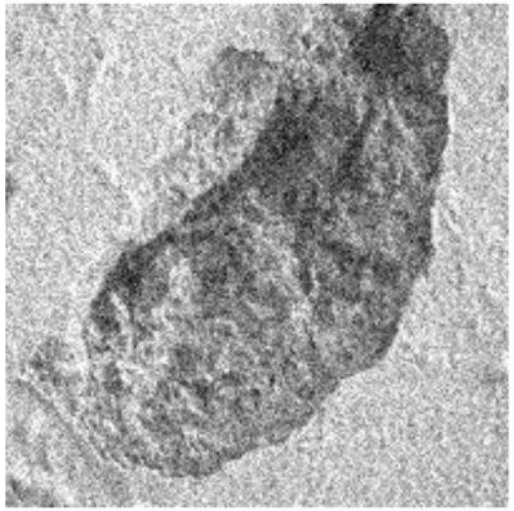
Effect: Better smoothing, preserves edges better than mean filter.

## A) Smoothing (Low-Pass) Filters

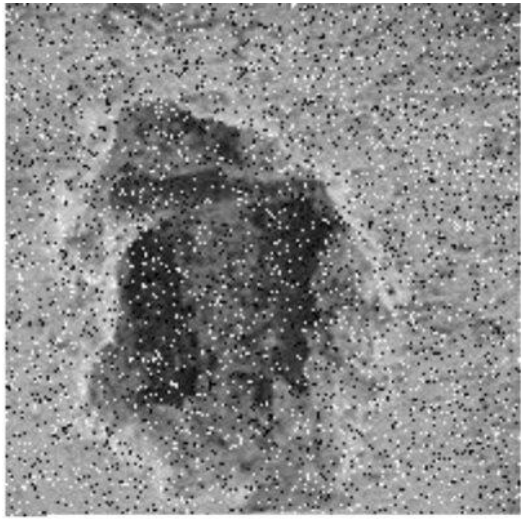
### 4. Median Filter

- **Non-linear smoothing**: replaces each pixel with the **median** of its neighborhood.
- Excellent for removing **salt-and-pepper noise**.
- Preserves edges better than mean filter.

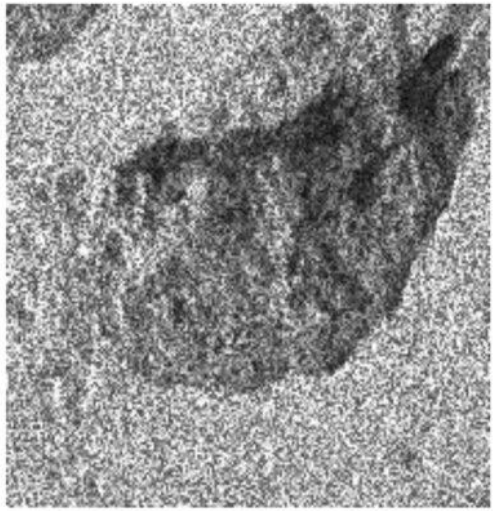
# Image processing fundamentals: Spatial filtering



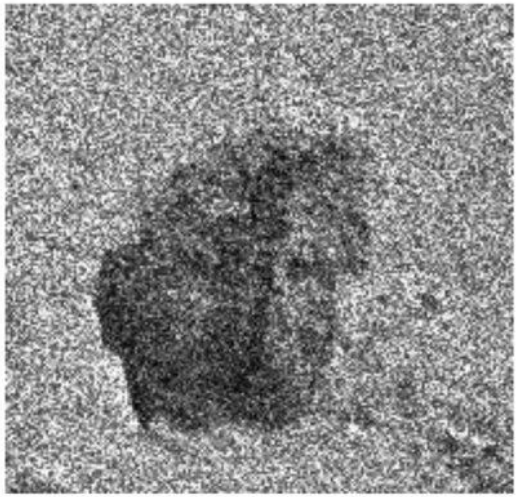
(a) Gaussian noise



(b) Salt and pepper noise



(c) Speckle noise



(d) Combination of Gaussian and speckle noises

# Image processing fundamentals: Spatial filtering

## 1. Gaussian Noise

**Definition:**

Gaussian noise is a type of statistical noise having a **probability density function (PDF)** equal to that of the **normal (Gaussian) distribution**.

$$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- $\mu$: Mean of the noise (typically 0)
- $\sigma$: Standard deviation (controls spread)

**Characteristics:**

- Adds **random intensity variation** to each pixel.
- **Common source:** Sensor temperature, electronic circuit noise.
- Affects **entire image** in a smooth, distributed manner.

---

## 🔸 2. Salt and Pepper Noise (Impulse Noise)

**Definition:**

Salt-and-pepper noise is a form of **impulse noise** where random pixels are replaced by **extreme values**—white (255, salt) and black (0, pepper).

**Characteristics:**

- Appears as **sparse white and black dots** on the image.
- Only a certain percentage of pixels are affected.
- **Common source:** Bit errors, faulty memory locations, transmission errors.

---

## 🔸 3. Speckle Noise

**Definition:**

Speckle noise is a **multiplicative noise** that occurs in coherent imaging systems like **radar, sonar, and ultrasound**.

$$g(x, y) = f(x, y) + f(x, y) \cdot n(x, y)$$

- $f(x, y)$: Original image
- $n(x, y)$: Multiplicative noise component (typically zero-mean Gaussian)

**Characteristics:**

- **Granular appearance** in the image.
- Depends on the intensity of the underlying pixel.
- Makes denoising more challenging than additive noise.

## 🔸 4. Combination of Gaussian and Salt-and-Pepper Noise

**Definition:**

This type of noise is a **hybrid model** where the image is affected by **both Gaussian and Salt-and-Pepper noise simultaneously**.

**Characteristics:**

- Gaussian noise adds **random brightness variations** across the image.
- Salt-and-pepper noise adds **impulse artifacts** (white/black dots).
- **Common in real-world scenarios** where images are affected by multiple sources of degradation simultaneously (e.g., medical images, old documents, noisy transmissions).

---

| Noise Type | Nature | Effect on Image | Common Source |
|---|---|---|---|
| **Gaussian Noise** | Additive | Smooth intensity fluctuations | Sensor noise, electronics |
| **Salt and Pepper Noise** | Impulsive | Random black/white pixel corruption | Data transmission errors |
| **Speckle Noise** | Multiplicative | Grainy texture, scale-dependent noise | Radar, ultrasound, SAR |
| **Gaussian + Salt & Pepper** | Mixed | Combination of intensity + impulse | Harsh real-world environments |

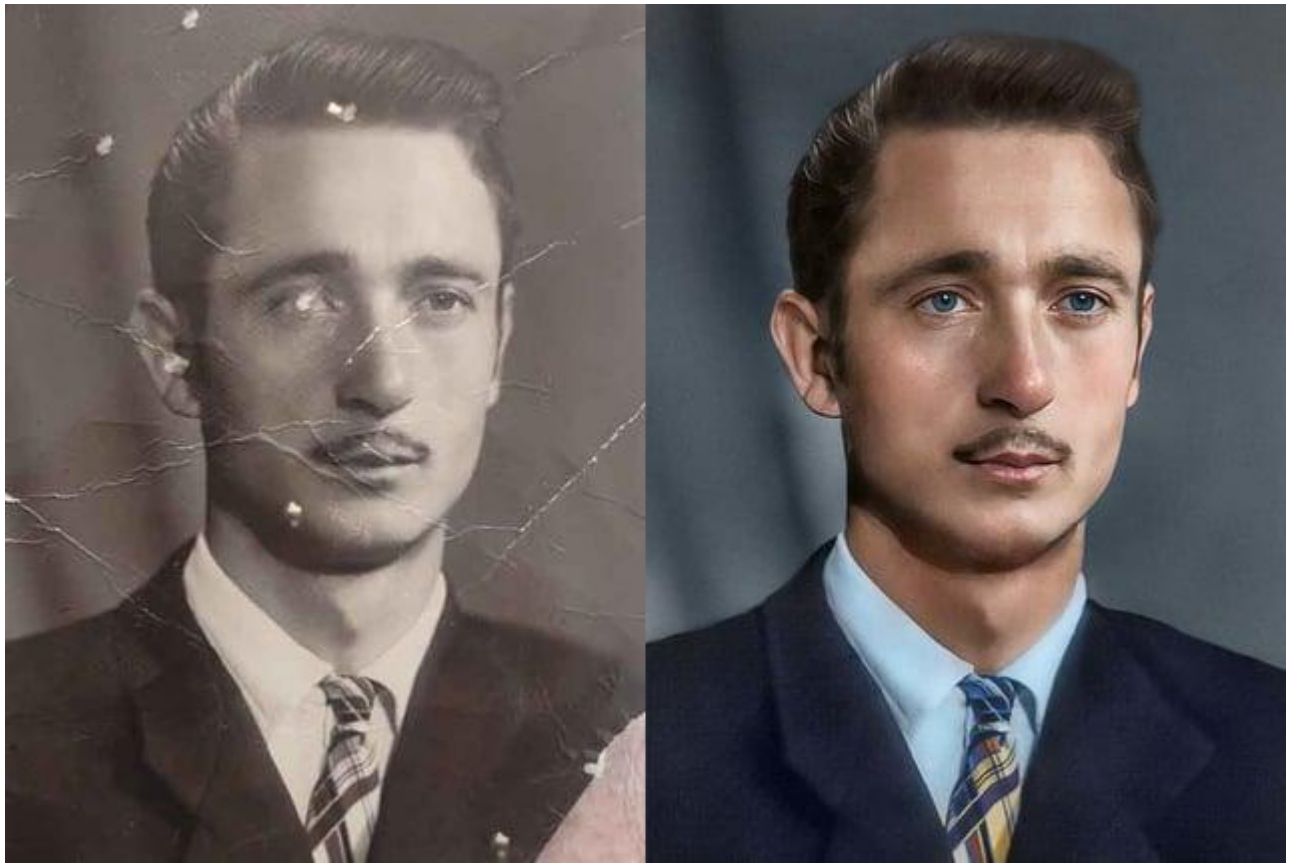
# Image restoration:

**Image Restoration** is a process of **recovering an original, clean image** from a **corrupted, degraded, or noisy version** by reversing known degradations.
Unlike enhancement (which improves appearance), restoration **aims to reconstruct the original image** using **mathematical or probabilistic models** of the degradation process.

Images can degrade due to:

- Motion blur
- Out-of-focus blur
- Sensor noise
- Atmospheric turbulence
- Low resolution
- Compression artifacts
- Missing pixels

# Image restoration:



# Image restoration:

## General degradation model:

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y)$$

Where:

- $g(x, y)$: degraded image
- $h(x, y)$: degradation function (blur kernel or PSF)
- $f(x, y)$: original image (to be recovered)
- $\eta(x, y)$: additive noise (e.g., Gaussian)
- $*$: convolution operation

# Image restoration:

**Restoration Techniques**

**1. Noise Reduction (Denoising)**
Removes unwanted random variation (noise) from the image.

**Spatial Domain Methods:**
- **Mean Filter**
- **Median Filter**
- **Gaussian Filter**
- **Adaptive Filters**

**Frequency Domain Methods: 301,17,21,26,29,37,42,50,52,58,62….**

- **Fourier Transform-based Filtering**

Image restoration:

## 2. Deblurring

- **Inverse Filtering:**

Assumes known degradation function $H(u, v)$. In frequency domain:

$$F(u, v) = \frac{G(u, v)}{H(u, v)}$$

- Highly sensitive to noise.
- Only useful when $H(u, v)$ is known and non-zero.

- **Wiener Filter:**

Balances noise reduction and inverse filtering:

$$\hat{F}(u, v) = \frac{H^*(u, v)}{|H(u, v)|^2 + \frac{S_\eta(u,v)}{S_f(u,v)}} \cdot G(u, v)$$

Where:

- $S_\eta$: Power spectral density of noise
- $S_f$: Power spectral density of original image

- **Lucy-Richardson Algorithm (Iterative):**
- Based on **maximum likelihood estimation**.
- Suitable for **Poisson noise**, commonly found in astronomy.

Image restoration:

### 3. Blind Deconvolution

- When the degradation function $h(x, y)$ is **unknown**.
- Both $h$ and $f$ are estimated iteratively.
- More complex and computationally expensive.

---

### 4. Regularization Methods

- Add constraints to stabilize inversion:

$$\min_f \|g - Hf\|^2 + \lambda \|R(f)\|$$

Where:

- $R(f)$: Regularization term (e.g., smoothness)
- $\lambda$: Trade-off parameter

Image restoration:

| Noise Type | Description | Suitable Filters |
|---|---|---|
| Gaussian Noise | Random variation in intensity | Gaussian, Wiener |
| Salt & Pepper Noise | White and black pixel noise | Median Filter |
| Speckle Noise | Multiplicative noise | Adaptive Filter, Anisotropic Diffusion |
| Poisson Noise | Due to photon counting | Lucy-Richardson |

Image restoration:

**Applications of Image Restoration**

- Medical Imaging (MRI, X-Ray)
- Satellite & Aerial Imagery
- Forensics and Surveillance
- Astrophotography
- Historical Image Archiving

Image restoration:

**Applications of Image Restoration**

- Medical Imaging (MRI, X-Ray)
- Satellite & Aerial Imagery
- Forensics and Surveillance
- Astrophotography
- Historical Image Archiving

# Fourier Transform and Frequency Domain Processing in Image Processing

**Why Fourier Transform in Image Processing?**

Image processing in the **frequency domain** focuses on analyzing how pixel intensities change spatially. While the spatial domain works directly on pixels, **Fourier Transform** analyzes the image in terms of

**sinusoidal components** — separating image content by **frequency**.

- **Low frequencies** → Smooth variations (e.g., background)

- **High frequencies** → Sharp changes (e.g., edges, noise)

# Fourier Transform and Frequency Domain Processing in Image Processing

## 🔢 1. Discrete Fourier Transform (DFT)

For an image $f(x, y)$ of size $M \times N$, the **2D Discrete Fourier Transform** is given by:

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \cdot e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

- $f(x, y)$: input image (spatial domain)
- $F(u, v)$: transformed image (frequency domain)
- $j$: imaginary unit

The **inverse DFT** is:

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) \cdot e^{j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

# Fourier Transform and Frequency Domain Processing in Image Processing

**Steps in Frequency Domain Processing**

1. **Compute DFT** of the image → get frequency spectrum

2. **Apply a filter** in the frequency domain (e.g., low-pass, high-pass)

3. **Compute inverse DFT** to get the processed image

# Fourier Transform and Frequency Domain Processing in Image Processing

| Filter Type | Purpose | Behavior in Frequency Domain |
|---|---|---|
| Low-Pass Filter | Blur / remove noise | Keep low frequencies, remove high |
| High-Pass Filter | Edge detection, sharpen | Remove low frequencies |
| Band-Pass | Keep only certain frequency ranges | Remove very low & very high |

# Fourier Transform and Frequency Domain Processing in Image Processing

**Applications in Image Processing**

- **Image filtering (blurring/sharpening)**
- **Image compression (e.g., JPEG uses DCT)**
- **Noise reduction**
- **Pattern and texture analysis**
- **Watermarking and authentication**

**3,10,15,17,18,25,26,32,38,47,49,52,57,63,Le**

## UNIT – II
## Image Analysis Techniques

Edge Detection and Feature Extraction: Gradient operators (Sobel, Prewitt), Canny edge detector, Corner and interest point detection, Image Segmentation: Thresholding methods, Region-based segmentation, Clustering techniques (K-means, Mean-Shift), Morphological Image Processing: Erosion and dilation, Opening and closing operations, Applications in shape analysis, Texture Analysis, Statistical methods (co-occurrence matrices), Transform-based methods (Gabor filters), Applications in pattern recognition.

**Edge Detection and Feature Extraction**

- Edge detection and feature extraction are fundamental tasks in **image processing** and **computer vision** that aim to identify meaningful structural information within images.
- These techniques are widely used in object detection, image segmentation, motion analysis, and 3D reconstruction.
- Edges are significant local changes in image intensity. They typically occur at the boundaries of objects, and detecting them helps in understanding the structure and geometry within an image.

**Objectives of Edge Detection:**

- Identify object boundaries
- Highlight shape, contour, or texture
- Reduce the amount of data while preserving essential structural information

**Gradient Operators:**

Gradient operators calculate the **first derivative** of the image intensity to detect edges. They highlight regions where intensity changes rapidly.

**Sobel Operator**

The Sobel operator combines Gaussian smoothing and differentiation.
It uses **two 3×3 convolution kernels**, one for the horizontal direction (**Gx**) and one for the vertical direction (**Gy**).

Kernels:

Gx = [-1 0 +1
-2 0 +2
-1 0 +1]

Gy = [+1 +2 +1
0 0 0
-1 -2 -1]

- **Gradient magnitude** is calculated as:

$$G = \sqrt{G_x^2 + G_y^2}$$

- Often approximated as:

$$G = |G_x| + |G_y|$$

- **Use:** Better noise suppression due to weighting and smoothing.

**Prewitt Operator**

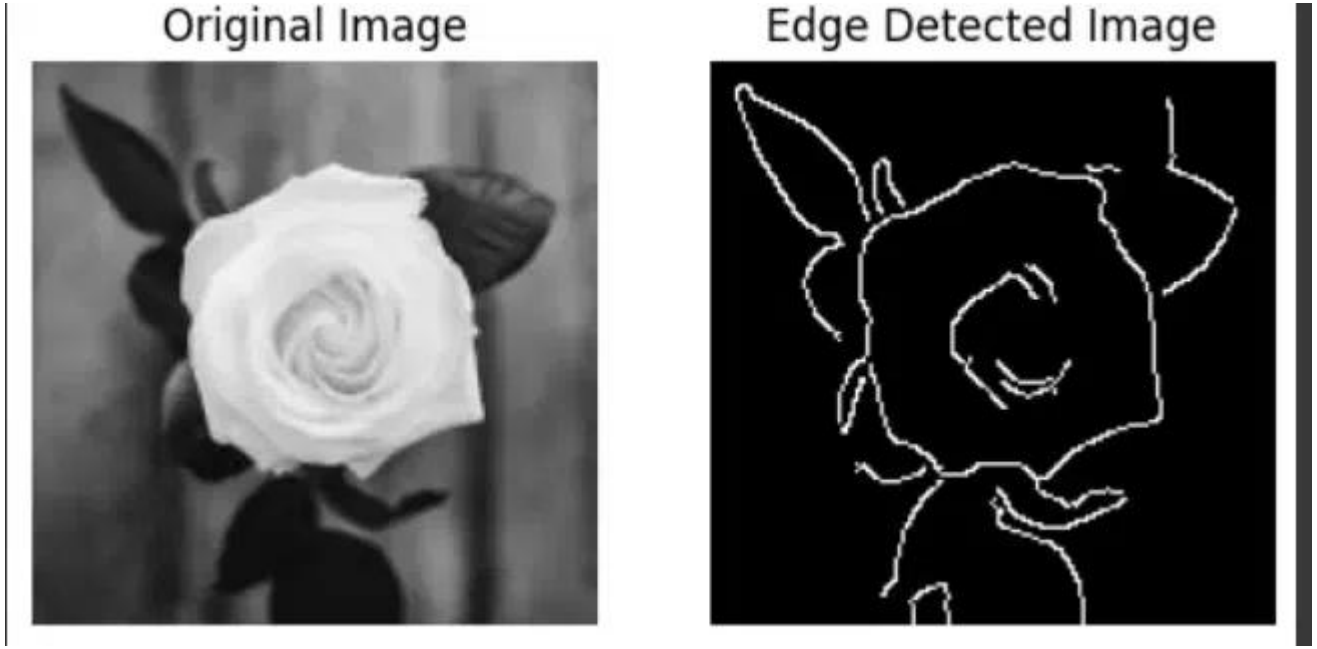
- Similar to Sobel but **less accurate** in edge detection, especially under noisy conditions.
- The kernel does not apply more weight to the center row or column.

Kernels:

Gx = [-1 0 +1
-1 0 +1
-1 0 +1]

Gy = [+1 +1 +1
0 0 0
-1 -1 -1]


origional image
Final Image


Original Image
Edge Detected Image

## 2. Canny Edge Detector

The **Canny edge detector** is a **multi-stage algorithm** that is widely regarded as one of the most robust edge detection techniques.

**Stages of Canny Edge Detection:**

**1.Noise Reduction:**

1. Apply a Gaussian filter to smooth the image and reduce noise.

**2.Gradient Calculation:**

1. Compute the gradient magnitude and direction using operators like Sobel.

**3.Non-Maximum Suppression:**

1. Thins the edges by suppressing non-maximum pixels in the gradient direction.

**4.Double Thresholding:**

1. Apply two thresholds: high and low.
2. Strong edges (above high threshold) are preserved.
3. Weak edges (between thresholds) are preserved only if connected to strong edges.

**5.Edge Tracking by Hysteresis:**

1. Ensures continuity by connecting weak edges that are part of the same structure.

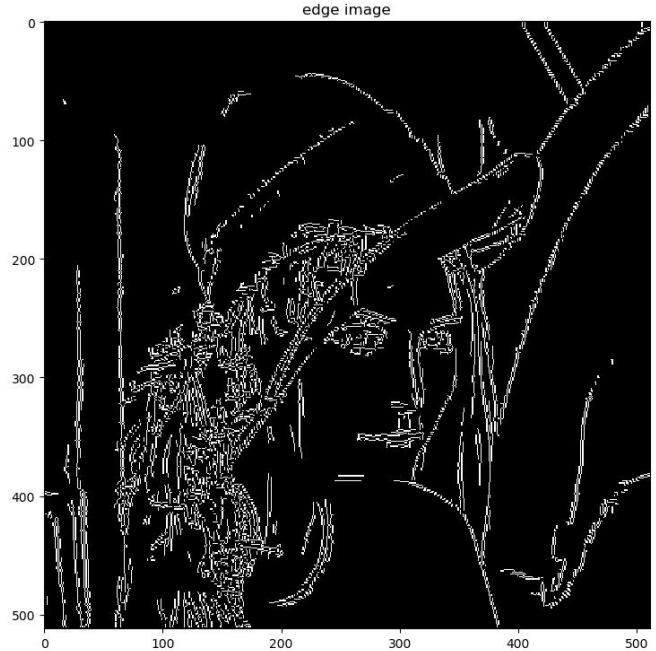
**Advantages:**

High accuracy and localization

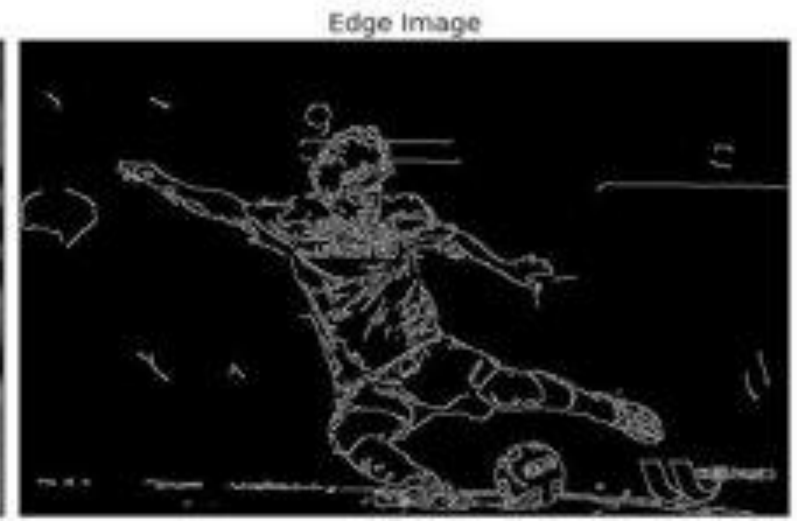
Reduced false positives

Robust to noise


gray image
0
100
200
300
400
500
0
100
200
300
400
500

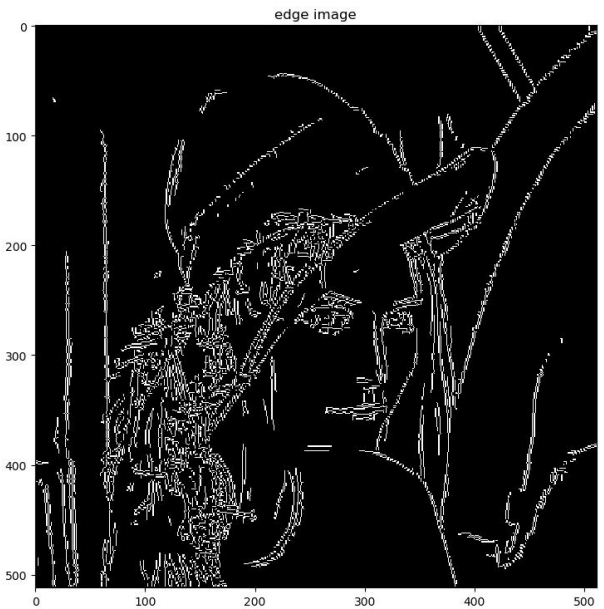

edge image
0
100
200
300
400
500
0
100
200
300
400
500


Original Image


Edge Image


Sobel Edge detection

Cany Edge detection

## 3. Corner and Interest Point Detection

While edges define boundaries, **corners** and **interest points** are areas where the gradient changes in **multiple directions**, offering more robust features for tasks like object recognition and tracking.

### Harris Corner Detector

- Detects corners by analyzing the local autocorrelation matrix of image gradients.
- Based on the change in intensity when the window shifts in different directions.

### Corner Response Function:

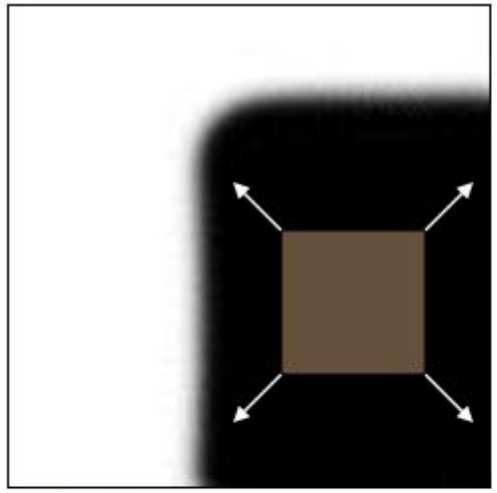
$$R = \det(M) - k \cdot (\text{trace}(M))^2$$

Where `M` is a matrix of image gradients and `k` is a constant (typically 0.04–0.06).

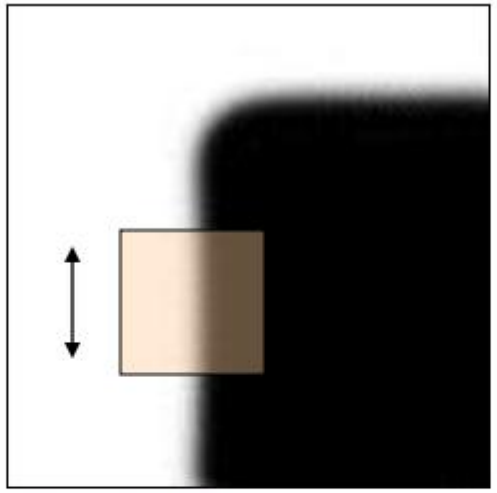
- If `R > threshold`: corner
- If `R ≈ 0`: flat region
- If `R < 0`: edge

## 3. Corner and Interest Point Detection

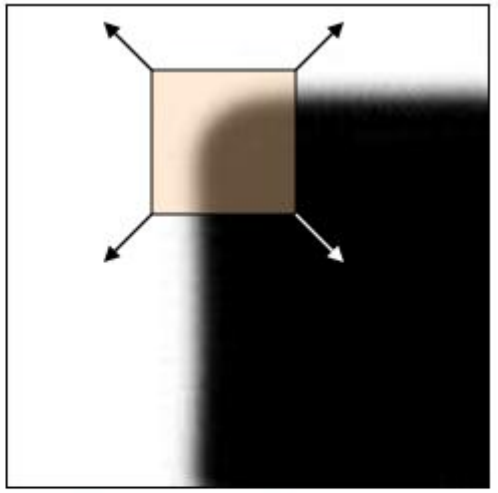
While edges define boundaries, **corners** and **interest points** are areas where the gradient changes in **multiple directions**, offering more robust features for tasks like object recognition and tracking.



"flat" region:
no change in
all directions



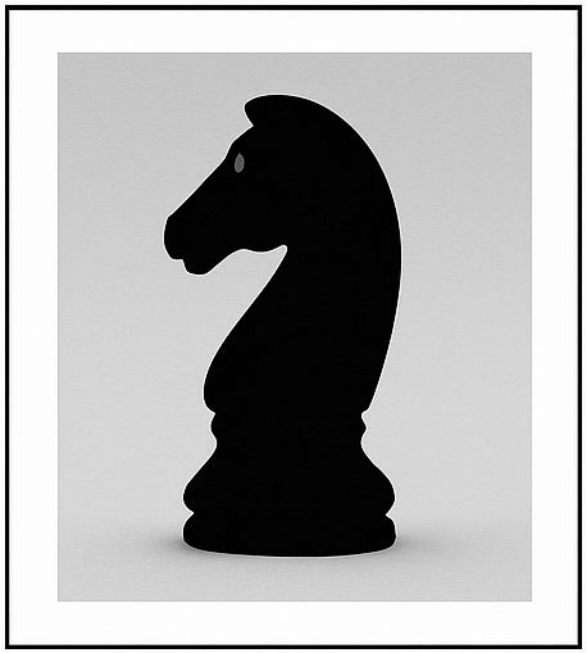
"edge":
no change
along the edge
direction



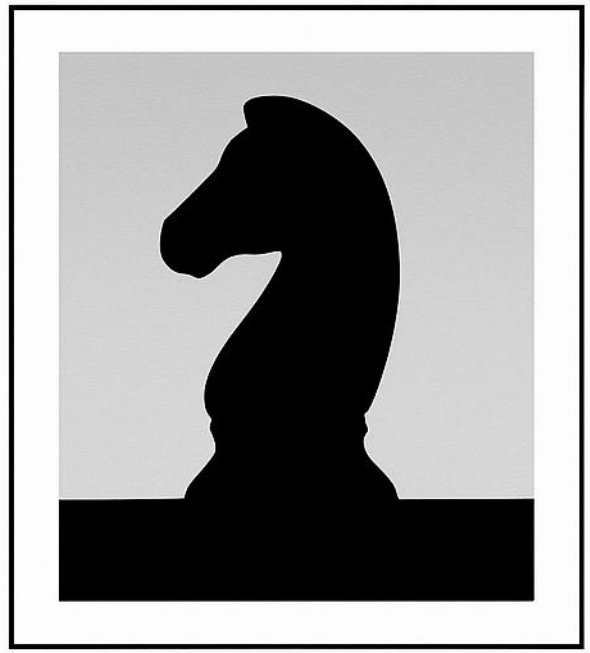
"corner":
significant
change in all
directions

urce: A. Efros

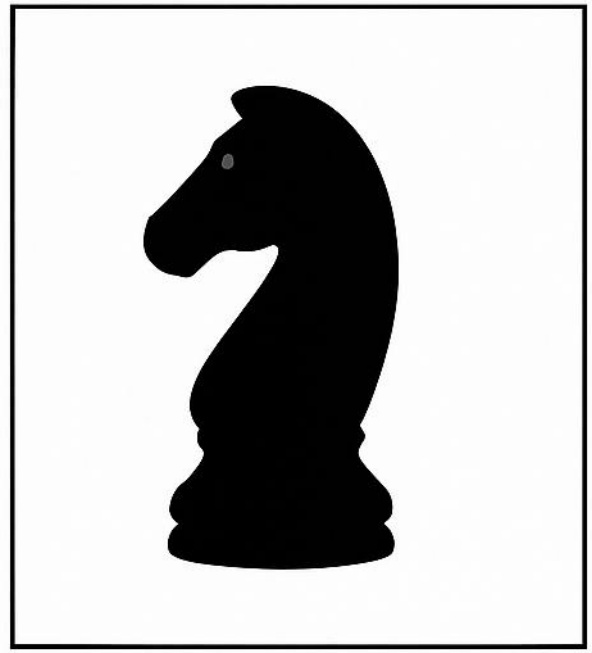
# EDGE DETECTION AND FEATURE EXTRACTION



Gradient Operators (Sobel, Prewitt)



Canny Edge Detector



Corner and Interest Point Detection

**Shi-Tomasi (Good Features to Track)**

•Improvement over Harris.

•Instead of using the Harris response function, it uses the **minimum eigenvalue** of the gradient matrix.

•Better suited for tracking applications.

- **SIFT (Scale-Invariant Feature Transform) – *Advanced method***

•Detects and describes local features in images that are **scale- and rotation-invariant**.

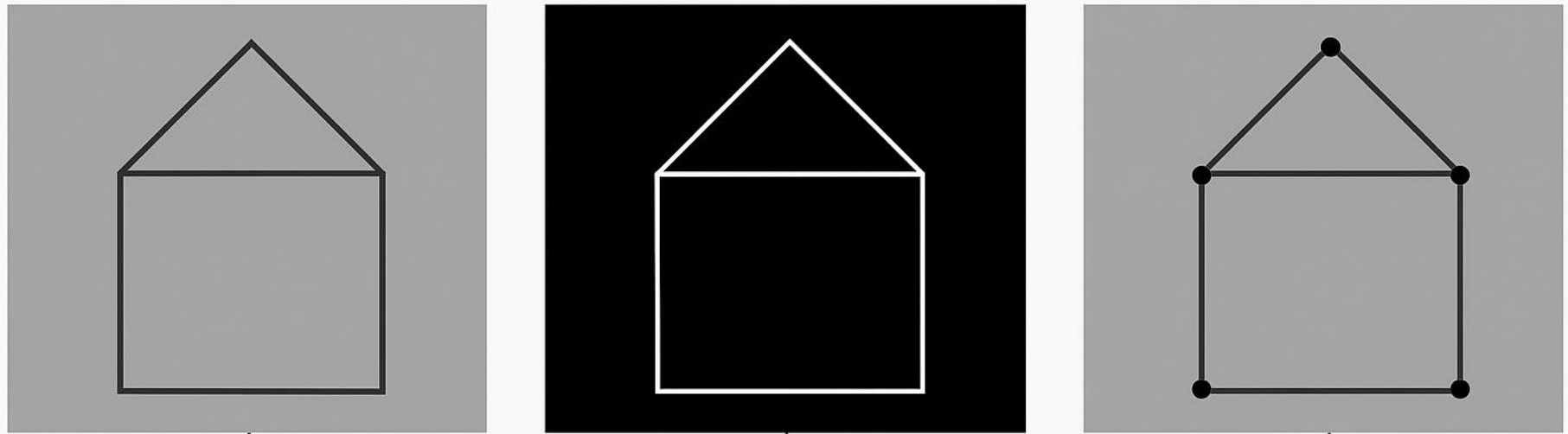
•Involves:

- Scale-space extrema detection
- Keypoint localization
- Orientation assignment
- Keypoint descriptor generation

**Gradient operators (Sobel, Prewitt)**

**Canny edge detector**

**Corner and interest point detection**



Gritdient operators (Sobel, Prewitt)

Canny edge detector

Corner and interest point detection

Edge detection and feature extraction

**Image Segmentation:**

**Image Segmentation** is the process of dividing an image into multiple **segments or regions** to simplify or change the representation of an image into something more meaningful and easier to analyze. The goal is typically to locate objects or boundaries (lines, curves, etc.).

**Thresholding Methods**

**Thresholding** is the simplest method of image segmentation. It converts a grayscale image into a binary image using a threshold value.

**Basic Concept:**

- If the pixel value is greater than the threshold, assign it to the foreground (e.g., white or 1).
- If the pixel value is less than or equal to the threshold, assign it to the background (e.g., black or 0).

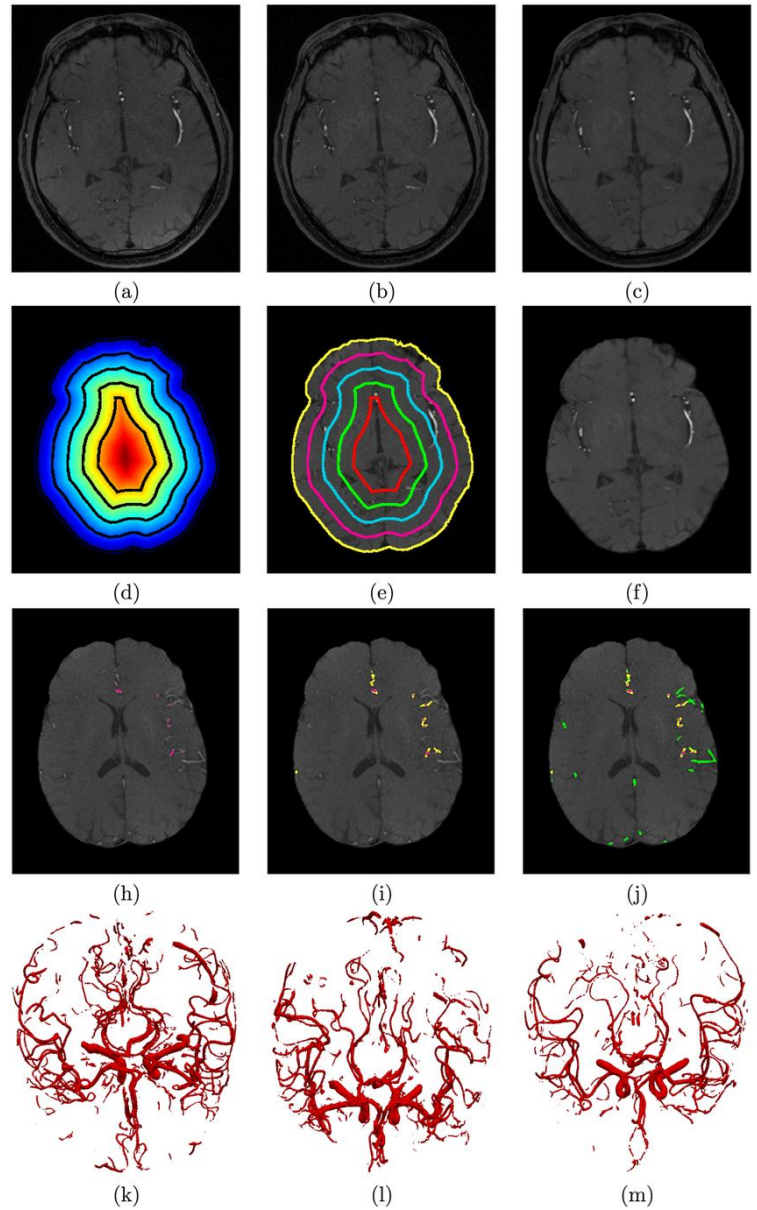
$$f(x,y) = \begin{cases} 1 & \text{if } I(x,y) > T \\ 0 & \text{if } I(x,y) \leq T \end{cases}$$

(**a**) Image

(**b**) Semantic segmentation

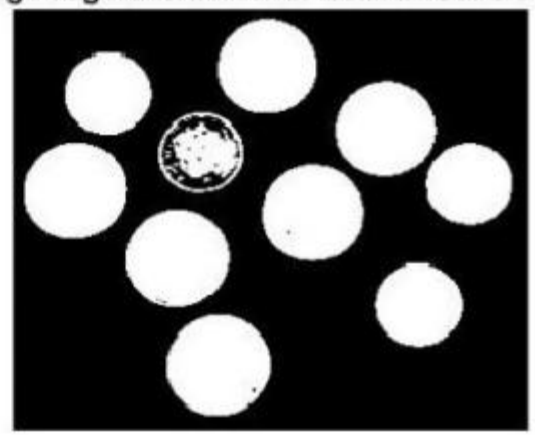
(**c**) Instance segmentation

(**d**) Panoptic segmentation


(a)
(b)
(c)
(d)
(e)
(f)
(h)
(i)
(j)
(k)
(l)
(m)

Original image.

Image segmentation with Otsu thresholding.



$$f(x,y) = \begin{cases} 1 & \text{if } I(x,y) > T \\ 0 & \text{if } I(x,y) \leq T \end{cases}$$

**Types of Thresholding (T):**

- **Global Thresholding**: A single threshold value is used for the entire image.
- **Adaptive Thresholding**: The threshold is computed for small regions, suitable for images with varying lighting.
- **Otsu's Method**: An automatic global thresholding technique that minimizes intra-class variance.

**Advantages:**

- Simple and fast
- Effective for high-contrast images

**Limitations:**

- Not suitable for complex scenes
- Fails when the histogram is not bimodal

**Region-Based Segmentation**

This technique groups pixels into regions based on predefined criteria such as **similarity in intensity or color**.

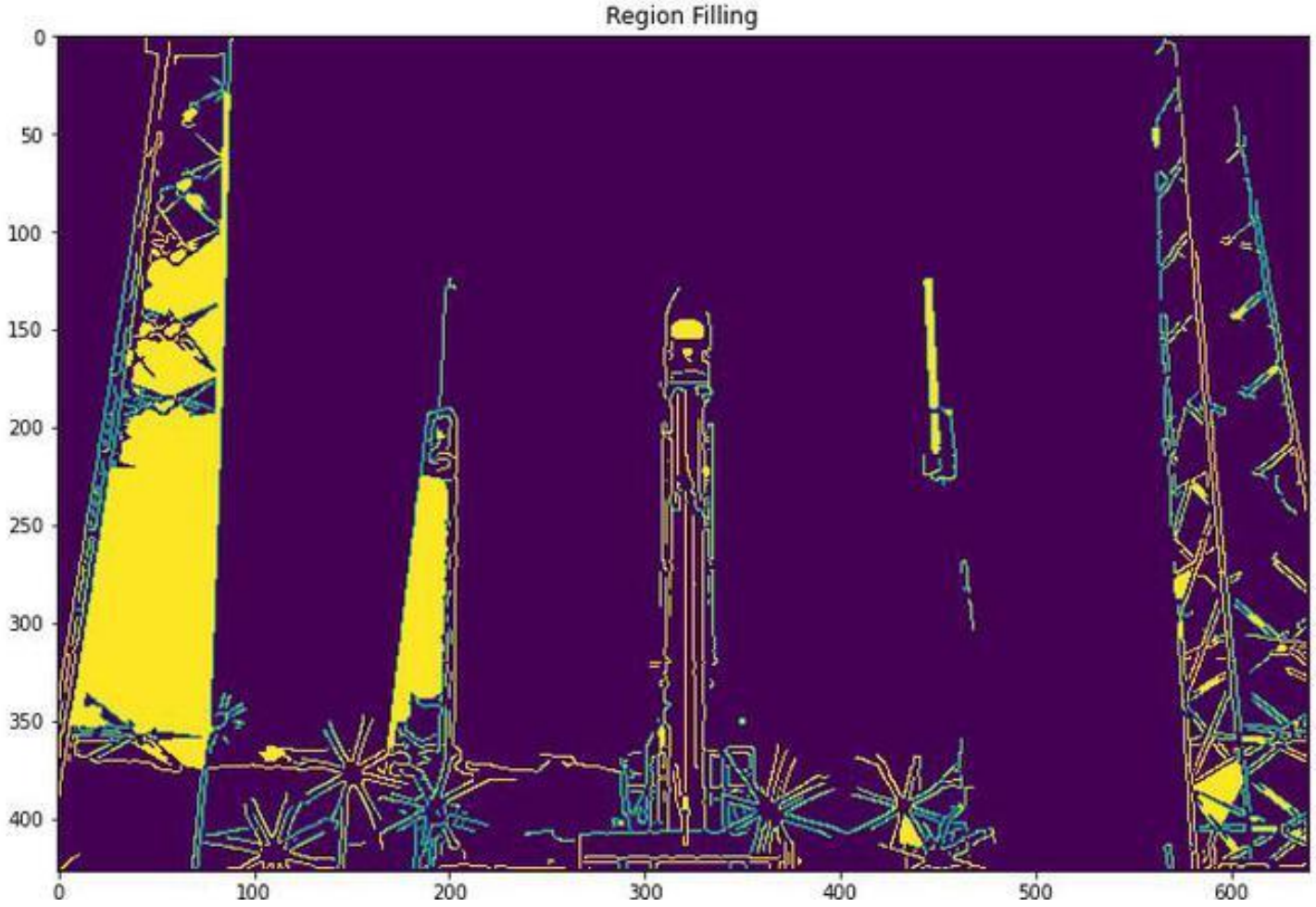
**Key Methods:**

**Region Growing:**
- Start from seed points.
- Grow the region by including neighboring pixels that have similar properties (e.g., intensity, color).
- The process stops when no more pixels satisfy the condition.

**Region Splitting and Merging:**
- The image is initially divided into a set of disjoint regions (e.g., quadtree).
- **Splitting** occurs when a region is too heterogeneous.
- **Merging** occurs when neighboring regions are too similar.

**Advantages:**
- Produces connected and coherent regions
- Can handle noise better than thresholding

**Limitations:**
- Requires selection of good seed points or criteria
- Computationally more expensive




Region based segmentation

## Clustering Techniques

Clustering-based segmentation groups similar pixels into clusters based on feature similarity, such as **intensity, color, or texture**.

### K-Means Clustering

**K-Means** is an unsupervised learning algorithm used to partition the image into **k clusters**.

**Steps:**

Choose the number of clusters (k).
Initialize k cluster centroids randomly.
Assign each pixel to the cluster whose centroid is nearest.
Recalculate the centroids based on the mean of assigned pixels.
Repeat steps 3 and 4 until convergence.

**Advantages:**

Simple and fast
Effective for images with clear color/intensity separation

**Limitations:**

Needs prior knowledge of **k**
Sensitive to initial centroid placement
Assumes clusters are spherical and equally sized

**Mean-Shift Clustering**

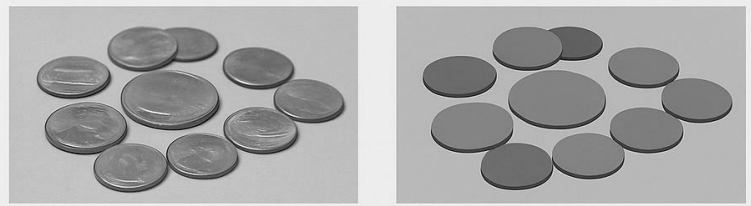
**Mean-Shift** is a non-parametric clustering technique that does **not require specifying the number of clusters** beforehand.

**Steps:**
Define a window (kernel) around each data point.
Compute the **mean shift vector** (mean of all points within the window).
Shift the window toward the mean.
Repeat until convergence.

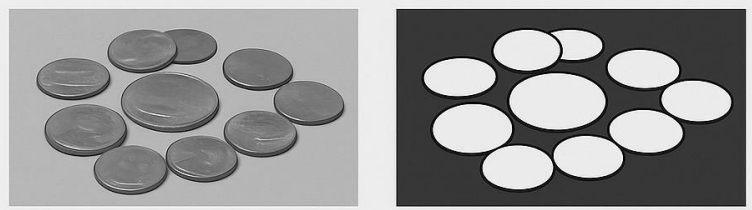
**Advantages:**
Does not require knowing the number of clusters in advance
Can detect **arbitrary-shaped** clusters
Robust to outliers
**Limitations:**
Computationally intensive
Choice of bandwidth (kernel size) is critical

# IMAGE SEGMENTATION

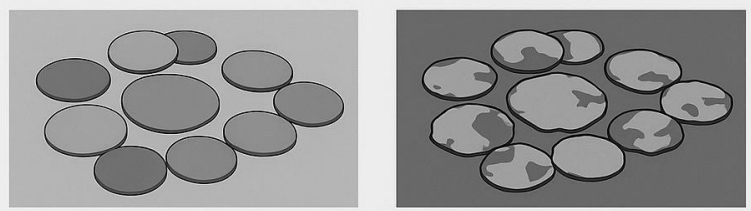
Thresholding



Region-Based Segmentation



Clustering (K–Means)



Clustering (Mean–Shift)

ONACO

| Method | Supervision | Region Shape | Pros | Cons |
|---|---|---|---|---|
| **Thresholding** | Unsupervised | Binary | Simple, fast | Poor with complex or noisy images |
| **Region Growing** | Semi | Connected | Good continuity | Needs good seed points |
| **K-Means Clustering** | Unsupervised | Spherical | Fast, easy to implement | Needs predefined k, sensitive to init |
| **Mean-Shift** | Unsupervised | Arbitrary | Adaptive, no k required | Slow, sensitive to bandwidth |

**Applications of Image Segmentation**

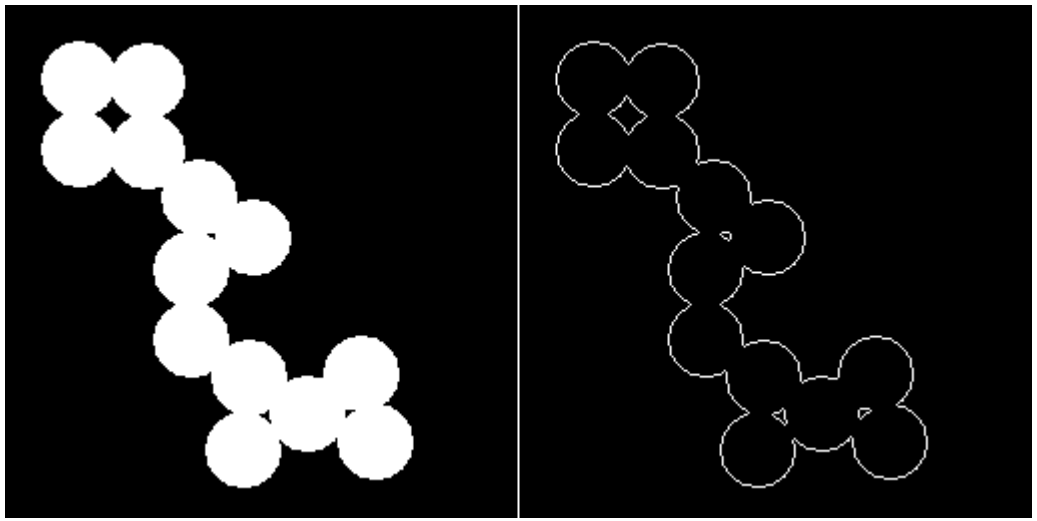
- Medical imaging (tumor detection)
- Satellite imagery analysis
- Face recognition
- Object detection and tracking
- Scene understanding

**Morphological Image Processing**

- Morphological image processing is a set of non-linear operations related to the shape or structure of objects in a digital image.
- It is mainly applied to **binary images**, though extensions to grayscale images also exist.
- The basic idea is to probe and modify an image using a small shape known as a **structuring element (SE)**.
- These operations are widely used in **image preprocessing, segmentation, and shape analysis**.

Image after segmentation

Image after segmentation and morphological processing



Morphological image processing

**Operations:**

- Erosion
- Dilation
- Opening operation
- Closing operation

**1. Erosion** ($\ominus$)

**Definition:**

Erosion removes pixels on object boundaries. It reduces the size of the foreground (white) regions in a binary image.

**Mathematical Concept:**

For a binary image $A$ and structuring element $B$, erosion is defined as:

$$A \ominus B = \{z \mid B_z \subseteq A\}$$

where $B_z$ is the translation of $B$ by vector $z$.

**Effects:**

- Shrinks objects
- Removes small noise
- Disconnects narrow connections between objects

**Example**:

- A white square becomes smaller.
- Thin lines may disappear.

**Dilation ($\oplus$)**

•**Definition**: Dilation expands the boundaries of objects.
•It adds pixels to object boundaries where the structuring element **touches** the object.

**Effect**:

•Grows objects
•Fills small holes and gaps
•Connects broken parts

Mathematical form:

$$A \oplus B = \{z \mid (B_z \cap A) \neq \emptyset\}$$

**Example**:
•A white square becomes larger.
•Small black gaps inside objects are filled.

**Opening Operation (∘)**

•**Definition**: Opening = **Erosion followed by Dilation**

$$A \circ B = (A \ominus B) \oplus B$$

**Effect**:

•Removes small objects/noise

•Smooths the contour of larger objects

•Breaks narrow bridges

**Use case**:

•Removing salt noise (small white spots).

**Closing Operation (•)**

•**Definition**: Closing = **Dilation followed by Erosion**

$$A \bullet B = (A \oplus B) \ominus B$$

**Effect**:

•Fills small holes inside objects

•Connects nearby objects

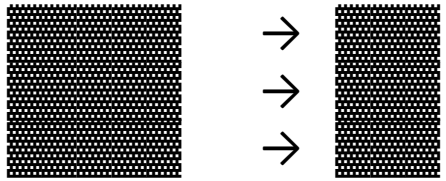
•Smooths object boundaries

**Use case**:

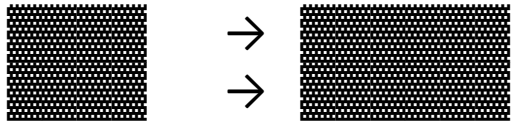
•Removing pepper noise (small black holes).

310,18,21,25,58,62,63,Le3,

Erosion: 👇 (shrinks objects)



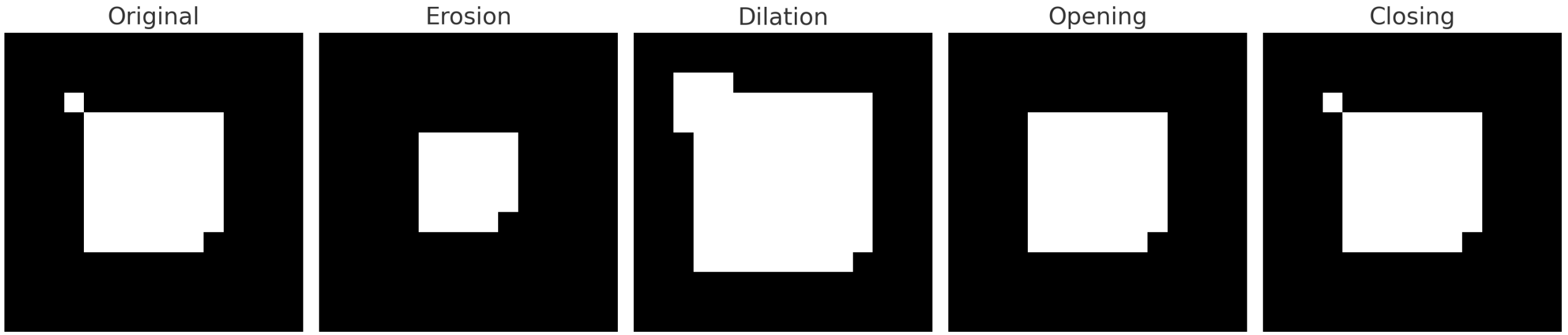
Dilation: 👆 (expands objects)



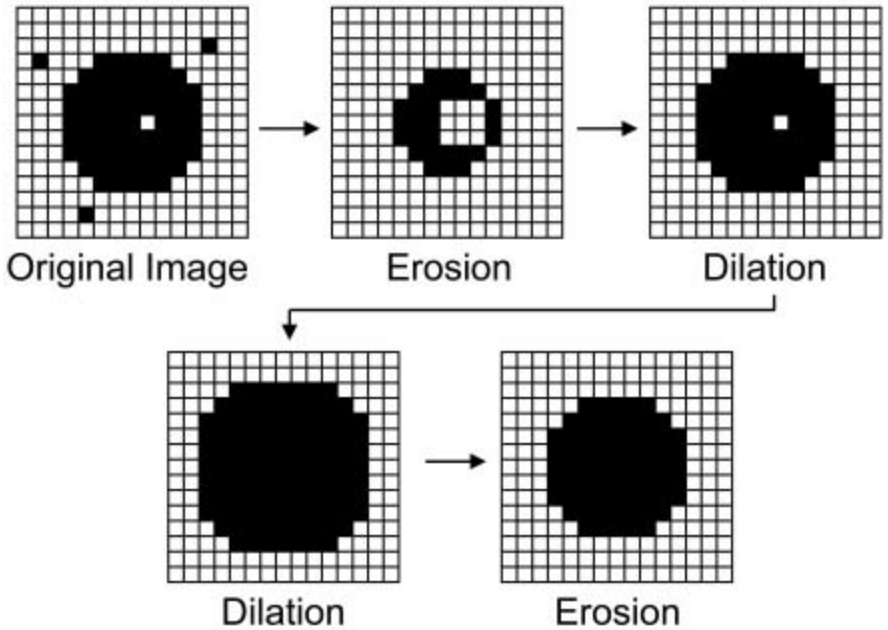
Opening = erosion → dilation
Closing = dilation → erosion

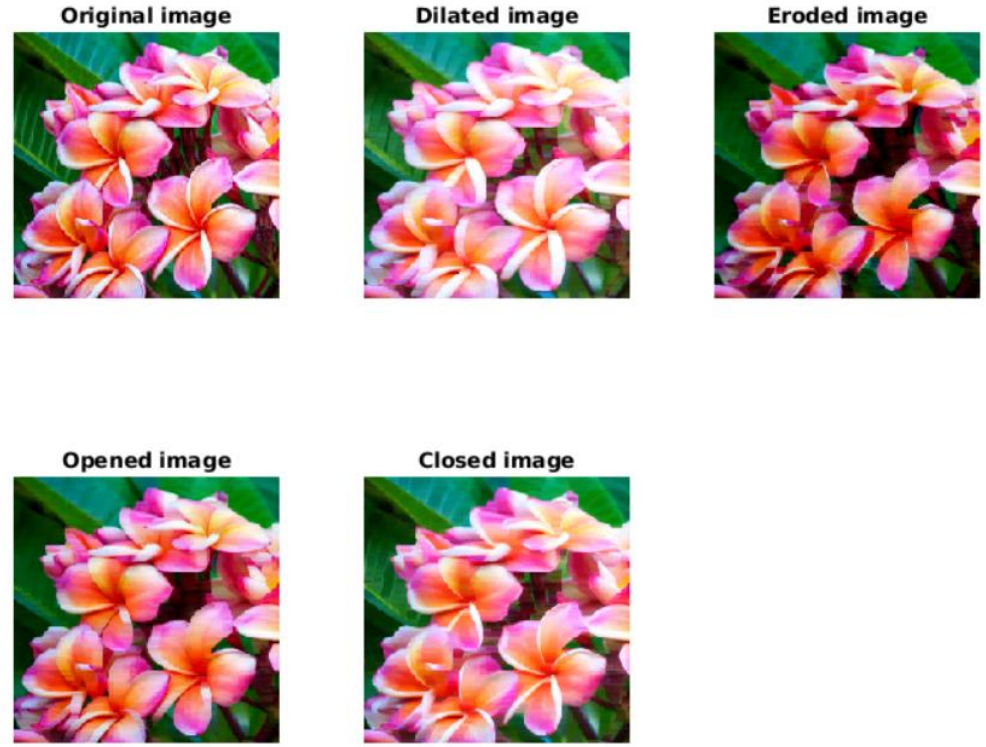
**Applications in Image Processing**

- Noise removal (salt & pepper noise)
- Image preprocessing for object detection
- Extracting shapes (skeletonization, edge detection)
- Medical imaging (removing artifacts, segmenting cells)
- OCR (Optical Character Recognition) preprocessing




- **Erosion** shrinks the object.
- **Dilation** expands it.
- **Opening** removes small noise while keeping the shape.
- **Closing** fills small holes and connects nearby objects.


Original Image
Erosion
Dilation
Dilation
Erosion


Original image
Dilated image
Eroded image
Opened image
Closed image

# Applications

## Shape Analysis:

Shape analysis involves the study and characterization of geometric structures within an image. It uses boundary, contour, and region properties to extract meaningful information about objects and patterns.

**Applications in Image Processing**

**1. Shape Analysis**

Shape analysis involves extracting and analyzing **geometric features** of objects in an image.

**Applications:**

- **Medical Imaging** → Identifying tumors, bone structures, or organ shapes
- **Biometrics** → Fingerprint, face, and iris recognition (shape descriptors)
- **Character Recognition** → Handwriting and OCR (Optical Character Recognition)
- **Object Detection in Robotics** → Shape-based recognition of tools or parts
- **Traffic Monitoring** → Vehicle detection and classification based on shape

## Texture Analysis:

Texture analysis studies the spatial arrangement of pixel intensities in an image. It helps describe surface properties such as smoothness, roughness, and regularity. Methods include statistical, structural, and spectral approaches.

## 2. Texture Analysis

Texture analysis deals with surface properties (smooth, rough, patterned) by studying **pixel intensity variations**.

**Applications:**

- **Medical Diagnosis** → Tumor detection in MRI/CT scans (healthy vs. abnormal tissue textures)
- **Remote Sensing & Agriculture** → Land cover classification (forest, water, crops, soil)
- **Quality Control in Manufacturing** → Detecting surface defects (scratches, cracks, fabric defects)
- **Face Recognition** → Skin texture analysis for liveness detection
- **Security & Forensics** → Analyzing textures in fingerprints, documents, currency notes

# Definition

A Gray-Level Co-occurrence Matrix (GLCM) is a statistical method of examining texture that considers the spatial relationship of pixels. It tabulates how often pairs of pixel with specific values and spatial relationships occur in an image.

# Features Derived from GLCM

Contrast: Measures intensity contrast between a pixel and its neighbor.

Correlation: Measures how correlated a pixel is to its neighbor.

Energy (Angular Second Moment): Measures textural uniformity.

Homogeneity: Measures the closeness of the distribution of elements.

Entropy: Measures randomness in the image.

# Example of Co-occurrence Matrix

Consider a small 4x4 image with gray levels {0,1}:

[0 0 1 1
0 1 1 0
1 0 0 1
1 1 0 0]

The GLCM is constructed by counting how often a pixel with value i is adjacent to a pixel with value j.

# Example of Co-occurrence Matrix

We're looking at **horizontal adjacency (to the right, distance = 1)**.

**Step 2: Extract pixel pairs**

Row 1 → (0,0), (0,1), (1,1)
Row 2 → (0,1), (1,1), (1,0)
Row 3 → (1,0), (0,0), (0,1)
Row 4 → (1,1), (1,0), (0,0)

**Step 3: Count frequencies**

(0,0): **3 times**

(0,1): 3 times

(1,0): 3 times

(1,1): 3 times

Total pairs = **12**

# Example of Co-occurrence Matrix

Step 4: GLCM matrix:

| i → j | 0 | 1 |
|---|---|---|
| **0** | 3 | 3 |
| **1** | 3 | 3 |

# Example of Co-occurrence Matrix

Step 5: Normalize (probabilities)

| i → j | 0 | 1 |
|---|---|---|
| **0** | 0.25 | 0.25 |
| **1** | 0.25 | 0.25 |

## Applications of Co-occurrence Matrices:

Medical Imaging: Characterize tissues (e.g., tumor detection).
Remote Sensing: Classify land cover and vegetation.
Quality Control: Detect defects on surfaces.
Document Analysis: Differentiate text, graphics, and background.
Face and Biometric Recognition: Analyze skin and fingerprint patterns.

# Definition of Gabor Filters

A Gabor filter is a linear filter used for texture and edge detection.

It is formed by multiplying a sinusoidal wave with a Gaussian function.

This makes the filter frequency and orientation selective.

$$g(x, y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cdot \cos\left(2\pi\frac{x'}{\lambda} + \phi\right)$$

$$x' = x\cos\theta + y\sin\theta \qquad y' = -x\sin\theta + y\cos\theta$$

$$g(x, y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cdot \cos\left(2\pi\frac{x'}{\lambda} + \phi\right)$$

$$x' = x\cos\theta + y\sin\theta \qquad y' = -x\sin\theta + y\cos\theta$$

Where:
σ: Standard deviation of Gaussian
λ: Wavelength of sinusoid (frequency)
ϕ: Phase offset
γ: Aspect ratio
θ: Orientation

# Intuition

- Gaussian part: Localizes the filter in space.
- Sinusoidal part: Detects specific frequency and orientation.
- Acts like a band-pass filter for image frequencies.
- Sensitive to edges and textures in a chosen direction.

# Applications of Gabor Filters

Texture analysis (classify fabrics, surfaces).

Medical imaging (tissue and tumor analysis).

Face recognition (extract robust features).

Fingerprint recognition (enhance ridges).

Document analysis (detect text lines, strokes).

Edge detection in noisy images.

# Definition of Pattern Recognition

- Pattern recognition is the process of classifying input data into categories based on key features or regularities.
- In image processing, it involves the identification of objects, textures, or structures from images using statistical, structural, or machine learning methods.

# Applications of Pattern Recognition

- Optical Character Recognition (OCR): Reading printed or handwritten text.
- Face Recognition: Identifying or verifying individuals from facial features.
- Fingerprint & Iris Recognition: Biometric identification.
- Medical Imaging: Detecting anomalies like tumors in scans.
- Remote Sensing: Land use and vegetation classification.
- Industrial Inspection: Detecting product defects.
- Autonomous Vehicles: Traffic sign and pedestrian detection.

# Techniques Used

- Statistical Methods: Decision boundaries using probability distributions.
- Structural Methods: Use of shape and relational descriptions.
- Machine Learning: Neural networks, SVMs, and deep learning.
- Template Matching: Comparing input with stored prototypes.

308,18,21,27,36,37,41,47,52,53,62,LE4,

# UNIT III: 3D Vision and Motion Analysis

Stereo Vision: Epipolar geometry, Disparity mapping, Depth estimation techniques, Structure from Motion (SfM): Feature tracking across frames, 3D reconstruction from motion, Applications in scene understanding, Optical Flow and Motion Analysis: Lucas-Kanade method, Horn-Schunck method, Motion segmentation, Camera Calibration and 3D Reconstruction: Intrinsic and extrinsic parameters, Calibration techniques, 3D point cloud generation.

Course Coding Classroom List:

313, 365, 342, 308,345,303,330,301,335,357,328,359,305,
317,349,325, 302,LE301,326,355,346,347,348,327

Stereo vision (a.k.a. binocular depth estimation) uses two slightly separated cameras to infer scene depth from parallax, similar to human eyes. Below is a compact but thorough walkthrough—from geometry to algorithms, error sources, and practical tips.

- Stereo vision is necessary in computer vision and perception (CVIP) because it enables depth perception, allowing machines to understand the world in three dimensions by calculating the distance and relative position of objects
- which is crucial for tasks like robotics, obstacle detection, and 3D scene reconstruction.
- By using two synchronized cameras to simulate human binocular vision, stereo vision provides highly accurate spatial data that helps AI make more confident decisions and enables applications like **autonomous navigation and advanced object recognition**.

**Depth Perception:**

Stereo vision calculates depth by comparing the slight differences (disparity) between images captured by two cameras. This creates a detailed 3D point cloud of the scene, giving machines the ability to judge how far away objects are.

**Enhanced Object Detection:**

With depth information, systems can more accurately distinguish between real hazards and non-threatening objects, which is vital for reducing false positives in autonomous systems.

**3D Scene Understanding:**

The 3D data generated by stereo vision enables complex tasks such as building 3D occupancy maps, generating 3D models of the environment, and performing semantic segmentation to identify and classify objects.

**Cost-Effectiveness:**

Compared to some other 3D sensing technologies, stereo vision is often more affordable and easier to incorporate into existing machine vision systems, making it a practical choice for many applications.

**Improved Decision-Making:**

The highly accurate spatial data provides AI with the necessary information for more confident decision-making, leading to smoother operation, enhanced passenger comfort, and increased trust in autonomous systems.

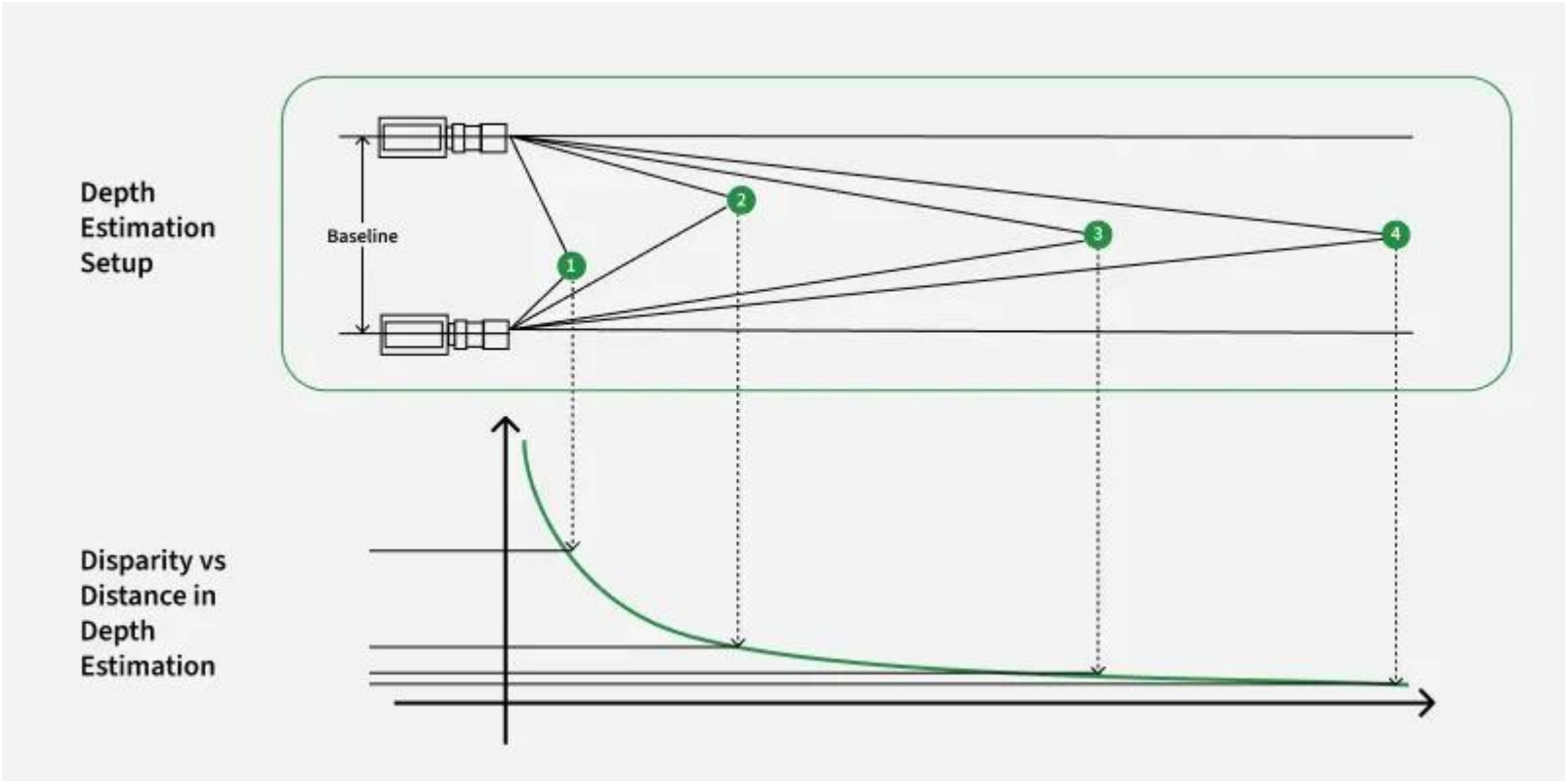
**Wide Range of Applications:**

Stereo vision is used in various domains, including:**Autonomous Vehicles:** For obstacle detection, navigation, and understanding the 3D environment.

**Robotics:** To enable robots to interact with their environment by judging distances and object positions.

**UAVs:** For obstacle detection and generating 3D maps for navigation and surveying.

# Components of a Stereo Vision System



## Components of a Stereo Vision System

- **Stereo Camera Setup** : Two cameras with a known and fixed baseline.
- **Camera Calibration** : Intrinsic and extrinsic parameter estimation for accurate geometry.
- **Image Rectification** : Aligns image planes so epipolar lines are horizontal.
- **Feature Detection / Stereo Matching** : Detects or matches corresponding points (using methods like SGM or block matching).
- **Disparity Estimation** : Measures pixel shifts (disparity) between the two images.
- **Depth Computation** : Calculates depth using triangulation.
- **Depth Map Generation** : Produces a pixel-wise 2D array representing distances.
- **3D Reconstruction (Optional)** : Converts disparity and camera parameters into 3D coordinates or point clouds.

# Components of a Stereo Vision System

**Mathematical Representation 14,25,47,56,59,65,L1,**
To calculate disparity and depth:

Let:

•f = focal length (in pixels)

•B = baseline (distance between cameras)

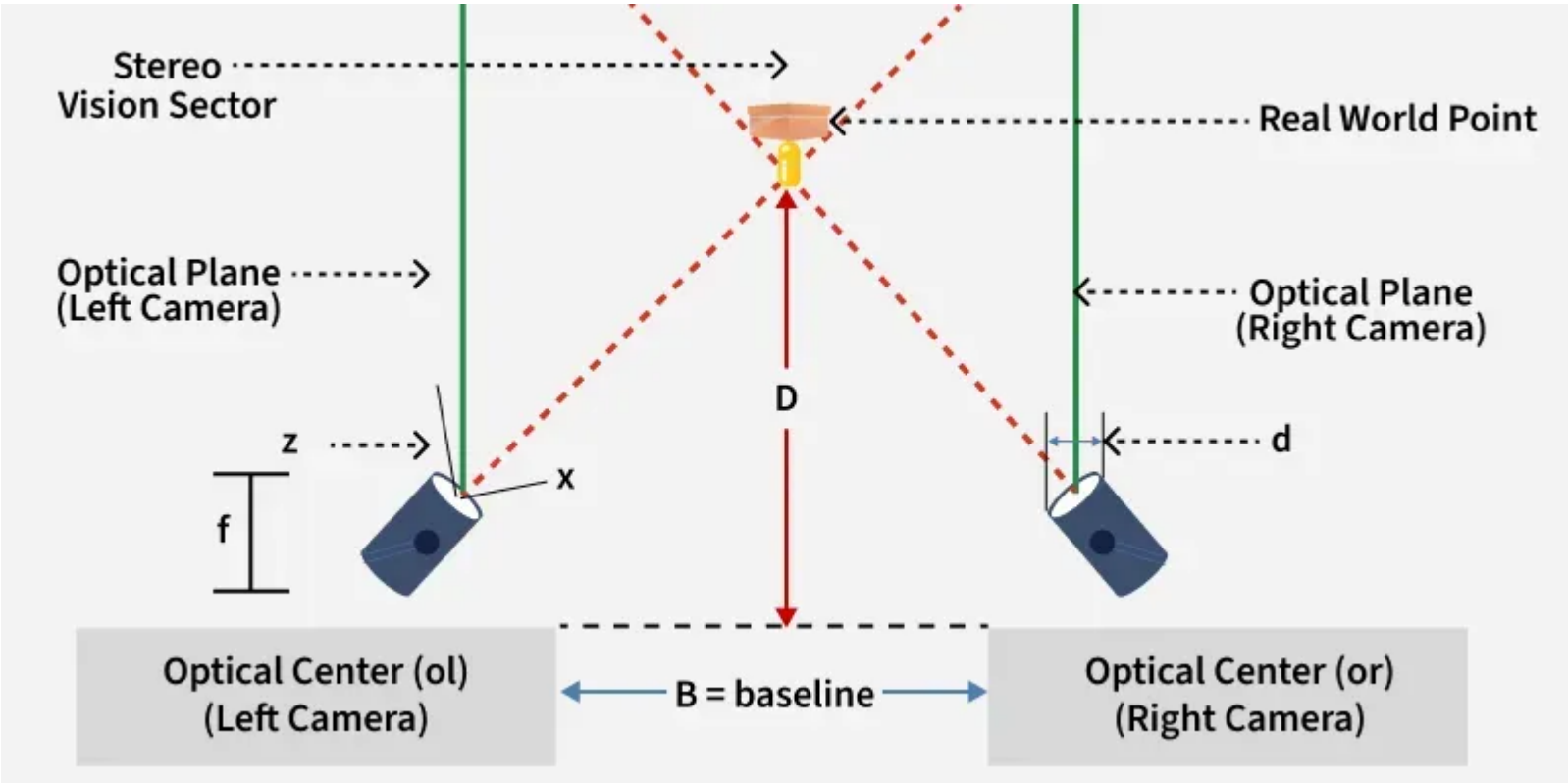
•d = disparity (difference in x-coordinates between matching pixels)

**Depth (Z)** is calculated as:

$$Z = \frac{f \times B}{d}$$

**Disparity**:

$$d = x_{\text{left}} - x_{\text{right}}$$


Stereo
Vision Sector
Real World Point
Optical Plane
(Left Camera)
Optical Plane
(Right Camera)
D
z
x
d
f
Optical Center (ol)
(Left Camera)
B = baseline
Optical Center (or)
(Right Camera)

Stereovision assumes that both cameras are calibrated and aligned. A rectified stereo image pair enables easier disparity computation by aligning corresponding epipolar lines horizontally. By analyzing the slight differences (parallax) between the two images, the system can triangulate depth. It mimics the natural depth perception mechanism in human vision.

**1.Capture Images**: Use two synchronized cameras with a fixed baseline to capture left and right images of a scene.

**2.Camera Calibration**: Estimate **intrinsic parameters** (focal length, principal point) and **extrinsic parameters** (rotation and translation between cameras).

**3.Image Rectification**: Warp both images so that epipolar lines are aligned horizontally. This simplifies correspondence search to one dimension (along scanlines).

**4.Stereo Matching / Disparity Estimation**: Find corresponding pixels in both images using Block Matching, Semi-Global Matching (SGM), Graph Cuts and Deep Learning.

**5.Disparity Map Construction**: Generate a disparity image where each pixel stores the disparity value (in pixels).

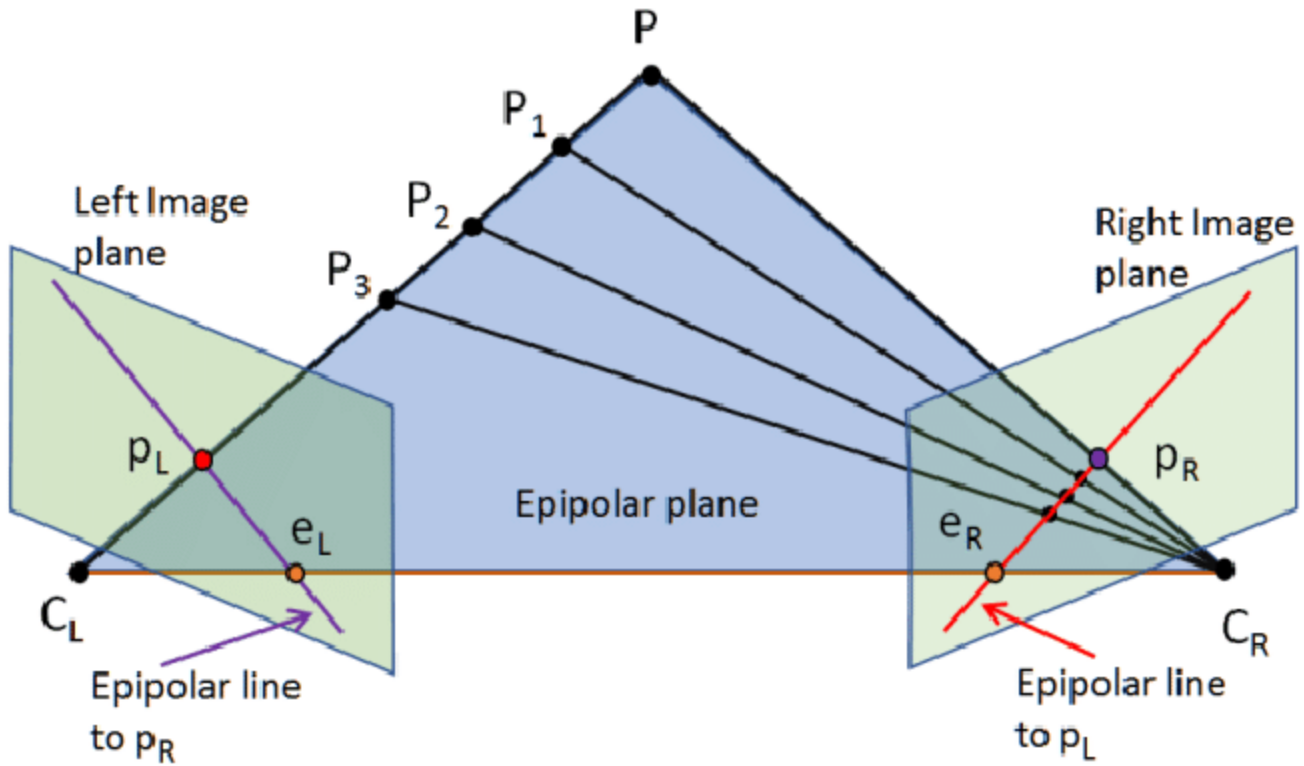
**6.Depth Estimation (Triangulation)**: Compute depth using the formula: $Z = \frac{f \cdot B}{d}$

**7.Depth Map and Visualization**: Generate a 2D depth map or point cloud to represent 3D structure.

## Key components of epipolar geometry

The interaction between two pinhole cameras, an object, and their respective image planes creates the geometric relationships fundamental to stereo vision.

- **Baseline ($B$)**: The line connecting the optical centers of the two cameras.
- **Epipoles ($e, e'$)**: The points where the baseline intersects each of the two image planes. An epipole is the projection of one camera's optical center onto the other camera's image plane.
- **Epipolar plane**: A plane defined by a 3D point in the scene and the optical centers of both cameras.
- **Epipolar lines ($l, l'$)**: The lines formed by the intersection of an epipolar plane with the two image planes. For any 3D point, its corresponding projections in the left and right images must lie on their respective epipolar lines.


P
P1
P2
P3
Left Image plane
Right Image plane
pL
pR
eL
eR
Epipolar plane
CL
CR
Epipolar line to pR
Epipolar line to pL

## The epipolar constraint and stereo matching

The most important consequence of epipolar geometry is the **epipolar constraint**, which simplifies the process of finding corresponding points (stereo matching).

- For any point in the left image, its corresponding point in the right image must lie on the epipolar line defined by the same 3D point.
- Instead of searching the entire second image for a match, the search space is reduced to a single line, making the computation more efficient.
- This constraint can be algebraically represented by the **Fundamental Matrix ($F$)** for uncalibrated cameras, or the **Essential Matrix ($E$)** for calibrated ones.

## Stereo rectification

In practice, a stereo camera rig is not perfectly aligned. The image planes are generally not coplanar, causing epipolar lines to be tilted. **Stereo rectification** is a geometric warping process that transforms both images to a new common image plane, making their epipolar lines perfectly horizontal and aligned. This simplifies the search for correspondences to a 1D problem (along the image rows).

## Triangulation and depth calculation

Once a corresponding pixel pair is found in the rectified images, their 3D position can be determined through triangulation, using the cameras' known intrinsic (focal length, $f$) and extrinsic (baseline, $B$) parameters.

The horizontal difference in pixel coordinates between corresponding points is called **disparity ($d$)**. A simple formula can be used to calculate the depth ($Z$) from the cameras:

$$Z = \frac{f \cdot B}{d}$$

- **Larger disparity ($d$)** indicates a closer object (smaller $Z$).
- **Smaller disparity ($d$)** indicates a farther object (larger $Z$).
- **Zero disparity ($d = 0$)** indicates an object at infinity.

**Triangulation in Stereo Vision**

**Definition**

**Triangulation** is the geometric method used in **stereo vision** to estimate the 3D position (depth) of a point in the scene from its 2D projections in two or more camera images.

It relies on the fact that:

•A 3D point projects onto different locations in the **left and right camera images**.

•Knowing the camera parameters (baseline, focal length, etc.), the depth can be computed by forming a triangle between the **point in 3D space** and the two **camera centers**.

How it works

1. Two cameras are placed a fixed distance apart (baseline **B**).
2. A 3D point $P$ in the scene is projected to:
   - $x_L$ in the left image
   - $x_R$ in the right image
3. The **disparity** is:

$$d = x_L - x_R$$

4. Using **similar triangles**, the depth $Z$ is:

$$Z = \frac{f \cdot B}{d}$$

where

- $f$ = focal length of the cameras
- $B$ = baseline (distance between the cameras)
- $d$ = disparity

## Summary of the geometric workflow

1. **Calibration**: Determine the intrinsic and extrinsic parameters of both cameras to understand their internal and relative geometry.
2. **Rectification**: Geometrically transform the images to make their epipolar lines horizontal. This is done using homographies derived from the calibration parameters.
3. **Correspondence Search**: Find matching pixels in the rectified left and right images. The search is constrained to a single horizontal row for each pixel.
4. **Disparity Map**: Compute the disparity value (the horizontal pixel difference) for every matched pixel.
5. **Depth Estimation**: Use triangulation with the disparity map, focal length, and baseline to create a 3D point cloud or depth map.

## Core Geometry

- **Setup**

  Two pinhole cameras separated by a **baseline** $B$ observe a 3D point $P(X, Y, Z)$. The images contain projections $x = (u, v)$ and $x' = (u', v')$.
- **Epipolar constraint**

  For any point $x$ in the left image, its match $x'$ in the right image lies on a known line:

  $$x'^{\top} F\, x = 0$$

  where $F$ is the **fundamental matrix** (rank 2). If cameras are calibrated, the **essential matrix** is

  $$E = K'^{\top} F K = [t]_{\times} R$$

  with intrinsics $K, K'$, rotation $R$, translation $t$, and $[t]_{\times}$ the skew-symmetric matrix.
- **Rectification**

  Apply homographies so epipolar lines become horizontal; then corresponding points have the **same row**: $v \approx v'$. Search for matches along each scanline only.

# Core Geometry

- **Disparity $\rightarrow$ Depth**

  Disparity $d = u - u'$. For rectified pairs:

  $$Z = \frac{fB}{d}, \qquad X = \frac{(u - c_x)Z}{f}, \qquad Y = \frac{(v - c_y)Z}{f}$$

  where $f$ is focal length in pixels, $(c_x, c_y)$ principal point.

  **Example:** $f = 800$ px, $B = 0.12$ m, $d = 24$ px $\Rightarrow Z = 4$ m.

- **Depth sensitivity**

  Small disparity errors $\delta d$ grow quadratically with distance:

  $$\delta Z \approx \frac{Z^2}{fB}\,\delta d$$

  So a larger baseline or higher resolution improves far-depth accuracy.

**Depth Estimation in Stereo Vision**

Stereo vision is based on the principle of **triangulation**: by observing a scene from two slightly different viewpoints (like human eyes), the depth of objects can be inferred from the difference in their positions in the two images.

**Depth Estimation Techniques**

**1. Block Matching / Local Methods**

- Divide the image into small windows (blocks).
- For each block in the left image, search along the corresponding epipolar line in the right image for the best match.
- Match quality measured by **Sum of Absolute Differences (SAD)**, **Sum of Squared Differences (SSD)**, or **Normalized Cross-Correlation (NCC)**.

- Pros: Simple, fast.
- Cons: Poor in texture less areas, sensitive to noise.

**2. Feature-Based Matching**

- Extract features (e.g., corners, edges, SIFT, SURF).
- Match features between left and right images.
- Use triangulation to compute depth.

- Pros: Works well in textured scenes.
- Cons: Sparse depth map (only at feature points).

**3. Semi-Global Matching (SGM)**

- Combines local costs (like SAD/SSD) with smoothness constraints across multiple scanlines.
- Aggregates disparity costs from multiple directions to minimize energy globally.
- Produces accurate and dense disparity maps.
- Widely used in autonomous driving (e.g., KITTI benchmark).

**4. Graph Cuts and Belief Propagation (Global Optimization)**

•Formulate disparity estimation as an **energy minimization problem**:

$$E(D) = Data\ term + \lambda \cdot Smoothness\ term$$

•Data term = similarity between matched pixels.
•Smoothness term = penalizes disparity changes between neighboring pixels.
•Graph Cuts / Belief Propagation are optimization algorithms to minimize this energy.
•Pros: Accurate and smooth disparity maps.
•Cons: Computationally expensive.

**5. Plane Sweeping**

•Hypothesize a set of depth planes in the 3D space.
•Project image pixels onto each plane and measure photo-consistency.
•The plane with maximum consistency gives the depth.
•Common in multi-view stereo.

**Learning-Based Methods (Deep Stereo Matching)**

- CNN-based models (e.g., GC-Net, PSMNet, GANet) learn to compute disparity from large stereo datasets.
- They combine feature extraction, cost volume construction, and disparity regression.
- Pros: High accuracy, robust to noise.
- Cons: Requires large datasets, computationally heavy.

**Structure from Motion (SfM)**

**What is SfM?**

**Structure from Motion (SfM)** is a computer vision technique used to estimate **3D structure** of a scene and **camera motion** simultaneously from a sequence of 2D images.

•"Structure" = the 3D geometry of the scene.

•"Motion" = the movement of the camera (its position and orientation) while capturing the images.
Unlike stereo vision (which requires multiple cameras in fixed positions), SfM works with a **moving camera** (like a handheld camera, drone, or robot).

**Feature Tracking Across Frames**

The first step in SfM is to detect and track visual features consistently across multiple frames.

•**Feature Detection**:

- Detect interest points (corners, edges, blobs) using algorithms such as **SIFT (Scale-Invariant Feature Transform)**, **SURF (Speeded Up Robust Features)**, **ORB**, or **Harris corners**.
- These features are chosen because they are repeatable (can be found across different views, lighting conditions, and scales).

•**Feature Matching**:

- Match features between consecutive images using **descriptors** that encode local appearance.
- Outlier rejection techniques like **RANSAC (Random Sample Consensus)** are used to remove false matches.

•**Tracking**:

- After matching, features are tracked across frames to build **feature trajectories.**
- These trajectories represent the same 3D point as observed from different camera poses.

**Outcome**: A set of matched key points across multiple images that will be used for triangulation.

**3D Reconstruction from Motion**

Once features are tracked, SfM estimates both the 3D structure of the scene and the motion of the camera.

•**Camera Pose Estimation**:
- Use the **Essential Matrix** (for calibrated cameras) or **Fundamental Matrix** (for uncalibrated cameras) to compute relative orientation and position between frames.
- This gives the camera trajectory.

•**Triangulation**:
- With the known camera poses, matched 2D points are triangulated into 3D coordinates.

•**Bundle Adjustment**:
- A global optimization step that refines both the 3D structure and camera poses simultaneously.
- Minimizes the **reprojection error** (difference between observed feature points and projected 3D points).

•**Dense Reconstruction (Optional)**:
- After initial sparse 3D points are reconstructed, multi-view stereo techniques can densify the reconstruction to create detailed 3D models.

**Outcome**: A 3D point cloud of the scene and an estimated camera path.

**Applications in Scene Understanding**

- **Robotics & Autonomous Vehicles**
  - Navigation and mapping (SLAM: Simultaneous Localization and Mapping).
- **Cultural Heritage**
  - 3D reconstruction of monuments, statues, or archaeological sites.
- **Augmented Reality (AR)**
  - Anchoring virtual objects into real-world 3D scenes.
- **Film and Gaming**
  - Creating realistic 3D models from real environments.
- **Medical Imaging**
  - Reconstructing 3D models from endoscopic video sequences.

**Optical Flow and Motion Analysis**

**Introduction**

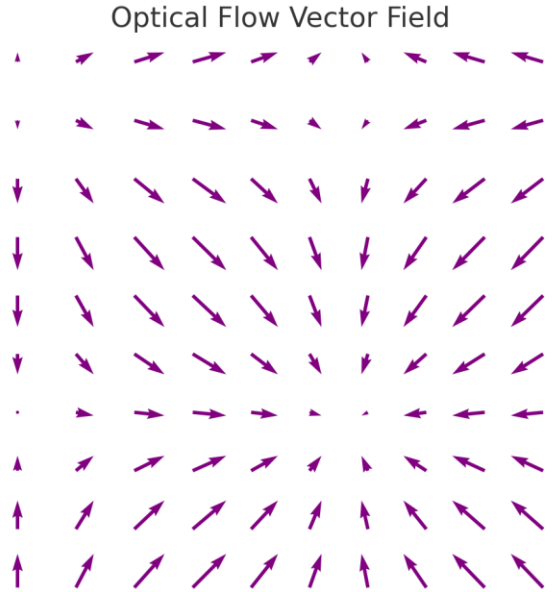
•**Optical Flow**: The pattern of apparent motion of objects, surfaces, or edges in a visual scene, caused by the relative motion between the observer (camera) and the scene.

•Represents **pixel-level motion** between consecutive frames.

## 2. Optical Flow Estimation

- **Differential Methods** (use spatio-temporal derivatives):
  - **Lucas-Kanade Method**:
    - Assumes constant motion within a small neighborhood.
    - Solves optical flow using least squares.
  - **Horn-Schunck Method**:
    - Global method enforcing smoothness constraint.
    - Minimizes energy function combining brightness constancy and smoothness.
- **Feature-based Methods**:
  - Track specific features across frames (faster, but less dense).
- **Deep Learning-based Methods**:
  - Models like FlowNet, PWC-Net directly predict dense flow fields.

3,7,25,28,49,55



## 1. Introduction

- Developed by **Bruce D. Lucas and Takeo Kanade (1981)**.
- Belongs to the **differential methods** for optical flow.
- Based on two key assumptions:
  1. **Brightness constancy** – the intensity of a point/object remains constant between consecutive frames.

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$$

  2. **Small motion** – the displacement of pixels between frames is small enough to be approximated using Taylor series expansion.

## 2. Derivation of the Optical Flow Equation

### Step 1: Brightness Constancy Constraint

From assumption 1:

$$I(x, y, t) = I(x + u, y + v, t + 1)$$

where $(u, v)$ is the displacement (optical flow) at pixel $(x, y)$.

### Step 2: Taylor Expansion

Expand the right-hand side using a first-order Taylor series:

$$I(x + u, y + v, t + 1) \approx I(x, y, t) + I_x u + I_y v + I_t$$

Subtract $I(x, y, t)$:

$$I_x u + I_y v + I_t = 0$$

This is called the **Optical Flow Constraint Equation (OFCE)**.

- $I_x = \frac{\partial I}{\partial x}$: spatial gradient in x
- $I_y = \frac{\partial I}{\partial y}$: spatial gradient in y
- $I_t = \frac{\partial I}{\partial t}$: temporal gradient

**Motion Analysis Using Optical Flow**

- **Motion Segmentation**
  - Separate moving objects from static background.
- **Object Tracking**
  - Track humans, vehicles, or other objects across frames.
- **Activity Recognition**
  - Infer actions (walking, running, gestures) from motion cues.
- **3D Scene Understanding**
  - Combine optical flow with stereo vision for depth and motion reconstruction.
- **Video Compression**
  - Optical flow helps estimate inter-frame changes for efficient encoding.

**Challenges**

- **Aperture Problem**: Motion cannot be determined at edges without additional constraints.
- **Occlusion**: Features disappearing and reappearing across frames.
- **Large Displacements**: Difficult for traditional differential methods, handled better by pyramidal or deep-learning methods.
- **Illumination Changes**: Brightness constancy assumption often violated.

# Camera Calibration and 3D Reconstruction

Camera calibration and 3D reconstruction are fundamental steps in computer vision for extracting reliable geometric and metric information from images. These processes ensure that a system can translate 2D image data into accurate 3D world representations.

## Intrinsic and Extrinsic Parameters

### Intrinsic Parameters

Intrinsic parameters describe the **internal characteristics of the camera** that affect how a 3D point is projected onto the 2D image plane. They form the **camera matrix** (K) and include:

**Focal Length (fx, fy):** Determines the scaling factor in the x and y directions.

**Principal Point (cx, cy):** The point where the optical axis intersects the image plane (usually near the image center).

**Skew Coefficient:** Describes the non-orthogonality of the image axes (often negligible in modern cameras).

**Lens Distortion Coefficients:**

- **Radial distortion** (barrel or pincushion effect).
- **Tangential distortion** (caused by misalignment of lens and sensor).

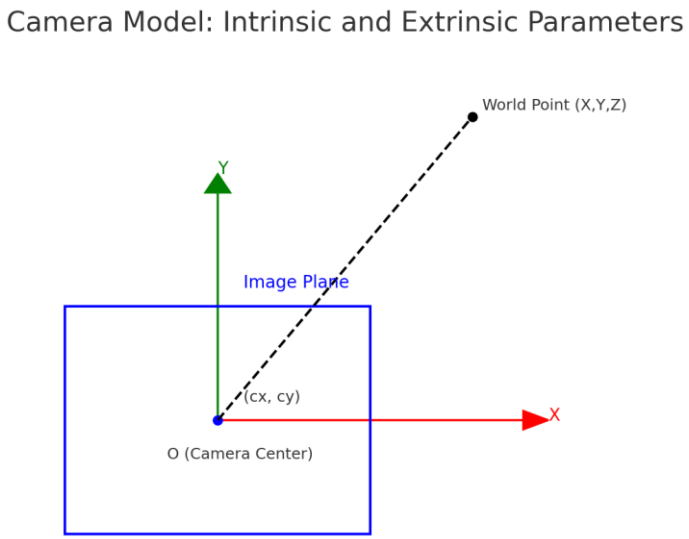
These parameters are essential to correct distorted images and achieve accurate geometric interpretations.

**Extrinsic Parameters:**

Extrinsic parameters describe the **position and orientation of the camera in the world coordinate system**. They define how the camera is placed relative to the scene.

**Rotation Matrix (R):** Orientation of the camera.

**Translation Vector (t):** Position of the camera center in world coordinates.
Together, (R|t) transforms points from world coordinates into the camera coordinate system.



**Calibration Techniques:**

Camera calibration is the process of estimating both intrinsic and extrinsic parameters. Common techniques include:

**Direct Linear Transformation (DLT):**

- Relates 3D world points to 2D image points using a linear mapping.
- Requires multiple correspondences between known world points and their projections.

**Zhang's Method (most widely used):**

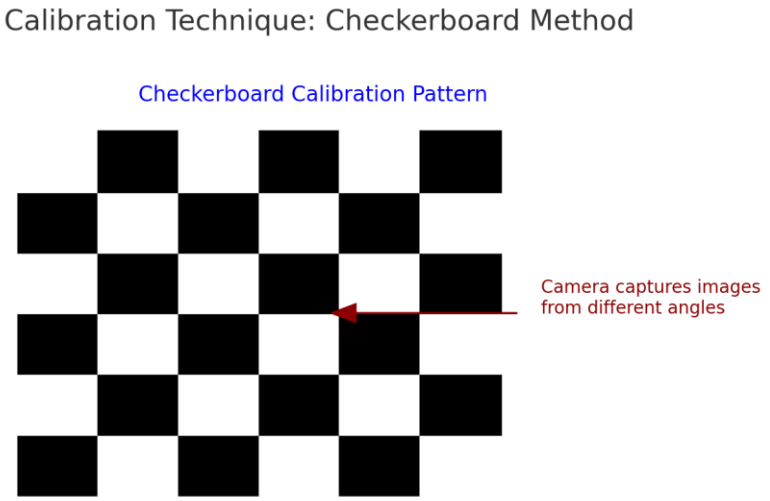
- Uses a planar calibration object (checkerboard pattern).
- Captures multiple images of the pattern at different orientations.
- Estimates intrinsic and extrinsic parameters using nonlinear optimization.
- Corrects lens distortions simultaneously.

**Self-Calibration:**

- No special calibration object is needed.
- Relies on observing natural scene points and camera motion.
- Uses projective geometry constraints across multiple images.

**Photogrammetry-based Methods:**

- Used in surveying and geospatial applications.
- Employs high-precision known world points to calibrate.



**3D Point Cloud Generation**

After calibration, 3D reconstruction techniques are applied to build a point cloud representation of the scene.

**Steps:**

**Stereo Vision:**

Capture images from two or more calibrated cameras.
Use **epipolar geometry** to find correspondences between left and right images.
Compute disparity (difference in image coordinates).
Depth (Z) is estimated as: z = f.B/d

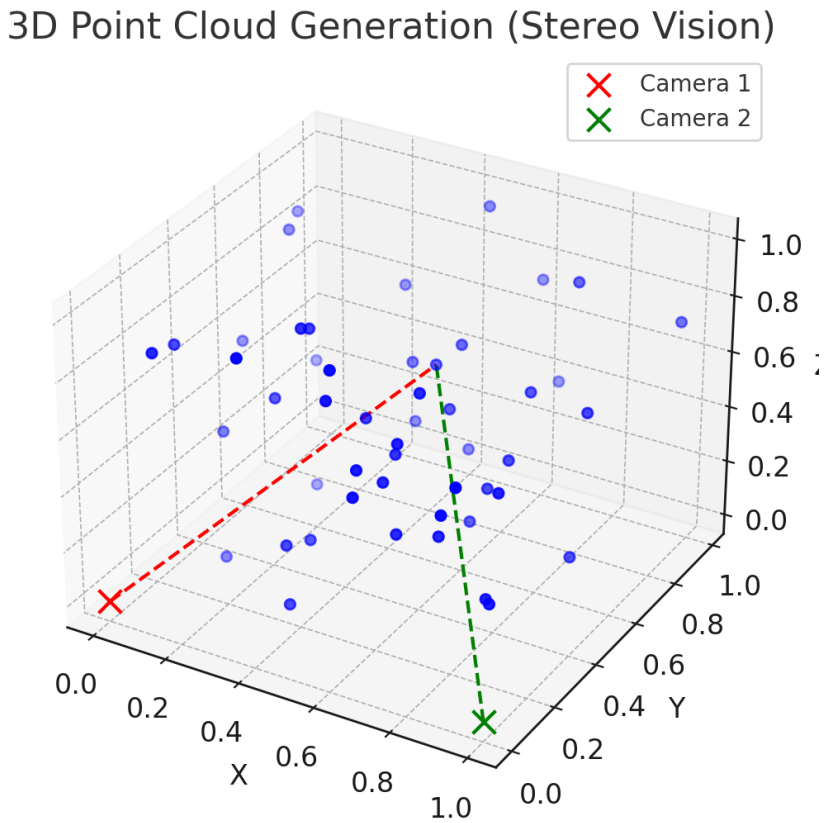
- **Structure from Motion (SfM):**
- Uses multiple views from a moving monocular camera.
- Estimates both 3D structure of the scene and camera motion simultaneously.
- **Multi-View Stereo (MVS):**
- Extends stereo vision to multiple cameras/views.
- Produces dense and detailed 3D reconstructions.
- **Depth Sensors / LIDAR Fusion:**
- Combine RGB images with depth maps (from sensors like Kinect, Intel RealSense).
- Generates accurate colored point clouds. 2,37,40,43,45,64,L3,

**Point Cloud Representation:**

•Each point is represented as $(x, y, z)$.

•May include additional attributes such as color (RGB) and intensity.

•Used in robotics, AR/VR, autonomous driving, and 3D modeling.



UNIT IV:

Object Recognition and Machine Learning in Vision Feature Descriptors and Matching: Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF), Feature matching algorithms, Object Detection and Recognition: Template matching, Deformable part models, Convolutional Neural Networks (CNNs), Introduction to Machine Learning for Vision: Supervised and unsupervised learning, Support Vector Machines (SVMs), Decision trees and random forests, Deep Learning Architectures: Autoencoders, Recurrent Neural Networks (RNNs), Generative Adversarial Networks (GANs)

**Introduction to Object Recognition**

•**Definition**: Object recognition is the process of identifying and categorizing objects within images or videos using computer vision techniques.

•**Applications**:

- Face recognition in security systems.
- Product identification in retail (Amazon Go).
- Autonomous driving (recognizing pedestrians, vehicles, traffic signs).
- Medical imaging (tumor detection, organ segmentation).

Object recognition relies heavily on extracting and describing **features** that uniquely characterize objects, followed by **matching** these features to known models or training data.

**What are Features?**

•**Features** are distinctive image patterns or points that remain consistent under:

- Changes in **scale** (zoom in/out).
- **Rotation** (different orientations).
- **Illumination** (lighting conditions).
- **Viewpoint changes**.

**Types of Features:**

**1.Global features**: Shape, color histograms, texture.

**2.Local features**: Keypoints (corners, blobs, edges).

Local features are particularly powerful in recognition because they are robust to partial occlusion.

**Feature Detection**

•**Goal**: Find "interest points" or keypoints in the image.
**Common Detectors:**

•**Harris Corner Detector** – detects corners (where gradient changes in two directions).

•**Difference of Gaussians (DoG)** – used in SIFT, detects blobs at multiple scales.

•**FAST (Features from Accelerated Segment Test)** – computationally efficient corner detector.

**Feature Descriptors**

After detecting keypoints, we describe them in a way that allows comparison.

**Key Descriptors:**

**1.SIFT (Scale-Invariant Feature Transform)**

1. Captures scale and rotation invariant descriptors.
2. Uses gradient histograms around keypoints.
3. Descriptor length: 128-dimensional vector.

**2.SURF (Speeded-Up Robust Features)**

1. Faster than SIFT (uses integral images).
2. Descriptor length: 64-dimensional vector.

**3.ORB (Oriented FAST and Rotated BRIEF)**

1. Combines FAST keypoint detector + BRIEF descriptor.
2. Binary descriptor, fast and efficient.
3. Used in real-time systems (mobile robotics, AR).

**4.BRIEF (Binary Robust Independent Elementary Features)**

1. Lightweight descriptor using binary tests.
2. Sensitive to rotation, so often combined with orientation adjustment.

**Feature Matching**

•**Definition**: The process of comparing feature descriptors across images to establish correspondences.

**Methods:**

**1.Brute-Force Matching**

1. Compute distance between every descriptor pair.
2. Distance metric: Euclidean (for SIFT/SURF), Hamming (for ORB/BRIEF).
3. Computationally expensive but accurate.

**2.Approximate Nearest Neighbor (ANN)**

1. Uses data structures like k-d trees, FLANN (Fast Library for Approximate Nearest Neighbors).
2. Faster for large-scale matching.

**3.Ratio Test (Lowe's Test in SIFT)**

1. Accept a match if:

$$\frac{d_1}{d_2} < \tau$$

1. where $d_1$ =best match distance, $d_2$ =second-best match distance, and $\tau \approx 0.7$.

**Feature Matching with Machine Learning**

Traditional descriptors are hand-crafted, but modern approaches use **machine learning and deep learning**:

**1.Bag-of-Visual-Words (BoVW)**

1. Treats local features as “visual words.”
2. Creates a histogram representation of objects.
3. Used with SVM classifiers.

**2.Deep Learning Descriptors**

1. CNNs automatically learn hierarchical features.
2. Examples: VGGNet, ResNet, Inception networks.
3. Feature embeddings are extracted and compared using similarity metrics (cosine similarity, Euclidean distance).

**3.Siamese Networks**

1. Neural networks trained on pairs of images.
2. Learn feature embeddings for similarity matching.
3. Widely used in face verification (e.g., FaceNet).

**Challenges in Feature Descriptors and Matching**

- **Illumination variations** – features may look different in shadows.
- **Occlusions** – partial visibility of objects.
- **Cluttered backgrounds** – spurious keypoints.
- **Scale and rotation invariance** – need robust descriptors.
- **Real-time performance** – trade-off between accuracy and speed.

**Challenges in Feature Descriptors and Matching**

- **Illumination variations** – features may look different in shadows.
- **Occlusions** – partial visibility of objects.
- **Cluttered backgrounds** – spurious keypoints.
- **Scale and rotation invariance** – need robust descriptors.
- **Real-time performance** – trade-off between accuracy and speed.