Steps of Digital Image Processing:

1. Image Acquisition

- **Definition:** Capturing an image using a sensor (like a camera) and converting it to a digital form.
- **Steps:**
  - Use a camera or scanner to capture the image.
  - Convert the captured image into a digital format using an analog-to-digital converter.

2. Image Enhancement

- **Definition:** Improving the visual appearance of an image or making it more suitable for a specific task.
- **Techniques:**
  - Adjusting brightness and contrast.
  - Applying filters to sharpen or smooth the image.
  - Removing noise to make details clearer.

3. Image Restoration

- **Definition:** Correcting defects in an image to restore its original appearance.
- **Methods:**
  - Removing blurriness caused by motion or out-of-focus lenses.
  - Fixing damaged parts of the image (like scratches or missing pieces).

4. Color Image Processing

- **Definition:** Handling images in color and manipulating them.
- **Processes:**
  - Adjusting color balance and saturation.
  - Converting between color spaces (e.g., RGB to grayscale).

5. Wavelets and Multi-Resolution Processing

- **Definition:** Breaking down an image into different levels of detail or resolution.
- **Usage:**
  - Compressing images by focusing on important details.
  - Enhancing specific features at various scales.

6. Compression

- **Definition:** Reducing the size of an image file without losing important information.
- **Types:**
  - Lossless compression: No loss of image quality (e.g., PNG).
  - Lossy compression: Some loss of quality but much smaller file size (e.g., JPEG).

7. Morphological Processing

- **Definition:** Analyzing and processing shapes within an image.
- **Operations:**
  - Dilation: Expanding shapes in the image.
  - Erosion: Shrinking shapes in the image.
  - Used to remove noise, fill gaps, and separate objects.

8. Segmentation

- **Definition:** Dividing an image into meaningful parts or regions.
- **Techniques:**
  - Thresholding: Separating objects based on color or intensity.
  - Edge detection: Finding the boundaries of objects.

9. Representation and Description

- **Definition:** Converting segmented regions into a form that a computer can analyze.
- **Steps:**
  - Representation: Choosing how to describe the shapes (e.g., outlines, skeletons).
  - Description: Extracting features like size, shape, and texture.

10. Object Recognition

- **Definition:** Identifying objects or patterns in an image.
- **Methods:**
  - Matching features with known patterns.
  - Using machine learning algorithms to recognize and classify objects.

11. Knowledge Base

- **Definition:** A database of information used to improve image processing tasks.
- **Components:**
  - Storing known patterns, shapes, and models.
  - Using historical data to make better decisions in image analysis.

Gray Level Transformation

All Image Processing Techniques focused on gray level transformation as it operates directly on pixels. The gray level image involves 256 levels of gray and in a histogram, horizontal axis spans from 0 to 255, and the vertical axis depends on the number of pixels in the image.

The simplest formula for image enhancement technique is:

s = T * r

Where T is transformation, r is the value of pixels, s is pixel value before and after processing.

Let,

r = f(x,y)

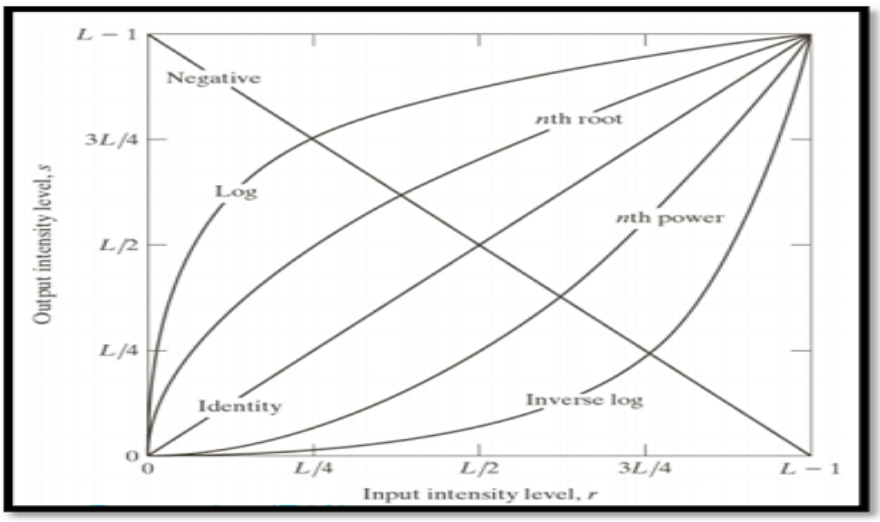
s = g(x,y)

'r' and 's' are used to denote gray levels of f and g at(x,y)

There are three types of transformation:

1. Linear
2. Logarithmic
3. Power - law

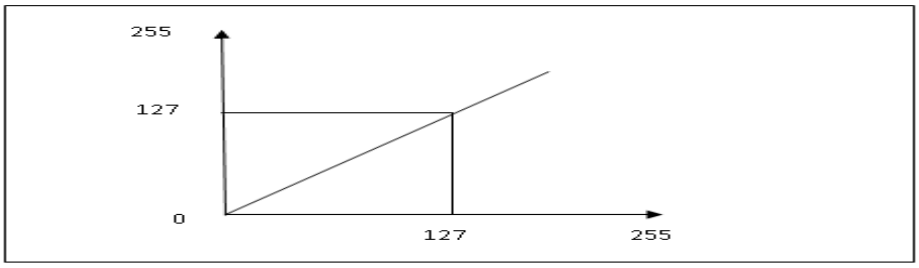
The overall graph is shown below:



Linear Transformation

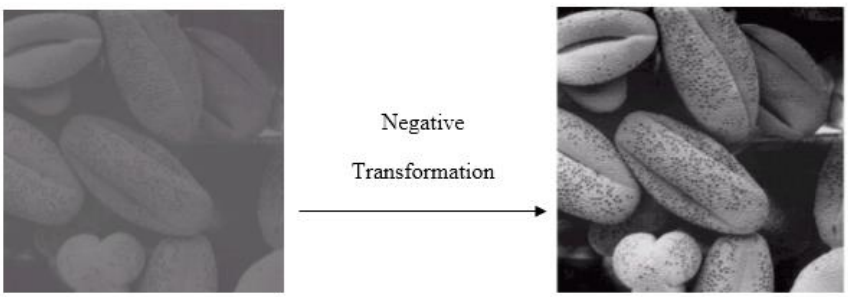
The linear transformation includes identity transformation and negative transformation.

In identity transformation, each value of the image is directly mapped to each other values of the output image.



Identity transformation

Negative transformation is the opposite of identity transformation. Here, each value of the input image is subtracted from L-1 and then it is mapped onto the output image



Input image **Output image**

Logarithmic transformations

Logarithmic transformation is divided into two types:

1. Log transformation
2. Inverse log transformation

The formula for Logarithmic transformation

s = c log(r + 1)

Here, s and r are the pixel values for input and output image. And c is constant. In the formula, we can see that 1 is added to each pixel value this is because if pixel intensity is zero in the image then log(0) is infinity so, to have minimum value one is added.

When log transformation is done dark pixels are expanded as compared to higher pixel values. In log transformation higher pixels are compresses.

Power - Law transformations

Power Law Transformation is of two types of transformation nth power transformation and nth root transformation.

Formula:

s = cr ^ γ

Here, γ is gamma, by which this transformation is known as gamma transformation.

All display devices have their own gamma correction. That is why images are displayed at different intensity.

These transformations are used for enhancing images.

For example:

Gamma of CRT is between 1.8 to 2.5

Components of Image Processing System

- **Image Sensors:**
  Image sensors senses the intensity, amplitude, co-ordinates and other features of the images and passes the result to the image processing hardware. It includes the problem domain.
- **Image Processing Hardware:**
  Image processing hardware is the dedicated hardware that is used to process the instructions obtained from the image sensors. It passes the result to general purpose computer.
  - **Computer:**
    Computer used in the image processing system is the general purpose computer that is used by us in our daily life.
  - **Image Processing Software:**
    Image processing software is the software that includes all the mechanisms and algorithms that are used in image processing system.
  - **Mass Storage:**
    Mass storage stores the pixels of the images during the processing.
  - **Hard Copy Device:**
    Once the image is processed then it is stored in the hard copy device. It can be a pen drive or any external ROM device.
  - **Image Display:**
    It includes the monitor or display screen that displays the processed images.
  - **Network:**
    Network is the connection of all the above elements of the image processing system.

Histogram Equalization

Histogram Equalization is a technique used to enhance the contrast of an image by redistributing the intensity levels. It is particularly useful for improving the visibility of features in images with poor contrast. Here's a breakdown of the process:

Concept

- **Purpose**: To adjust the image contrast so that the intensity distribution is spread more uniformly across the entire range of possible intensity values.
- **Goal**: To improve the overall contrast of an image, making features more distinguishable.

Spatial Filtering and its Types

Spatial Filtering technique is used directly on pixels of an image. Mask is usually considered to be added in size so that it has specific center pixel. This mask is moved on the image such that the center of the mask traverses all image pixels.

Smoothing Spatial Filter

Smoothing filter is used for blurring and noise reduction in the image. Blurring is pre-processing steps for removal of small details and Noise Reduction is accomplished by blurring. They achieve this by averaging pixel values within a defined neighborhood**.**

Types of Smoothing Spatial Filter

1. Linear Filter (Mean Filter)

2. Order Statistics (Non-linear) filter

These are explained as following below.

1. **Mean Filter:** Linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. The idea is replacing the value of every pixel in an image by the average of the grey levels in the neighborhood define by the filter mask. Below are the types of mean filter:
   - **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
   - **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Center pixel is multiplied by a higher value than average filter.
2. **Order Statistics Filter:** It is based on the ordering the pixels contained in the image area encompassed by the filter. It replaces the value of the center pixel with the value determined by the ranking result. Edges are better preserved in this filtering. Below are the types of order statistics filter:
   - **Minimum filter:** 0th percentile filter is the minimum filter. The value of the center is replaced by the smallest value in the window.
   - **Maximum filter:** 100th percentile filter is the maximum filter. The value of the center is replaced by the largest value in the window.
   - **Median filter:** Each pixel in the image is considered. First neighboring pixels are sorted and original values of the pixel is replaced by the median of the list**.**

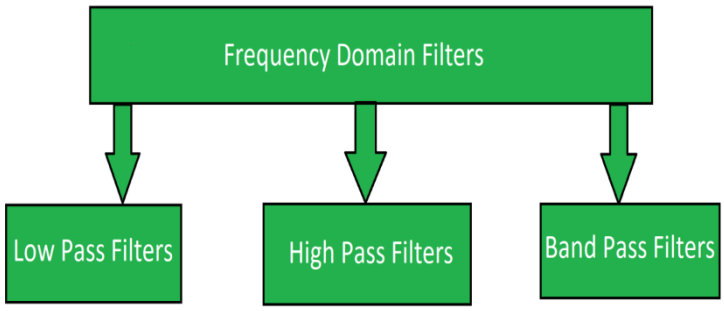
Sharpening Spatial Filter

It is also known as derivative filter. The purpose of the sharpening spatial filter is just the opposite of the smoothing spatial filter. Its main focus in on the removal of blurring and highlight the edges. It is based on the first and second order derivative.

Frequency Domain Filters and its Types

Frequency Domain Filters are used for smoothing and sharpening of image by removal of high or low frequency components. Sometimes it is possible of removal of very high and very low frequency. Frequency domain filters are different from spatial domain filters as it basically focuses on the frequency of

the images. It is basically done for two basic operation i.e., Smoothing and Sharpening.
These are of 3 types:



Classification of Frequency Domain Filters

1. Low pass filter:
Low pass filter removes the high frequency components that means it keeps low frequency components. It is used for smoothing the image. It is used to smoothen the image by attenuating high frequency components and preserving low frequency components.

2. High pass filter:
High pass filter removes the low frequency components that means it keeps high frequency components. It is used for sharpening the image. It is used to sharpen the image by attenuating low frequency components and preserving high frequency components.

3. Band pass filter:
Band pass filter removes the very low frequency and very high frequency components that means it keeps the moderate range band of frequencies. Band pass filtering is used to enhance edges while reducing the noise at the same time.

Sharpening vs. Smoothing Filters

Feature	Sharpening Filters	Smoothing Filters
Purpose	Enhance edges and details	Reduce noise and blur image
Frequency Domain	High-pass filter	Low-pass filter
Effect on Image	Increases image sharpness, may amplify noise	Reduces noise, blurs edges and details
Typical Filters	Sobel, Prewitt, Laplacian, Unsharp Masking	Mean, Gaussian, Median, Bilateral
Application	Edge detection, image enhancement	Noise reduction, image preprocessing

Homomorphic Filtering

Homomorphic Filtering is a technique in image processing that combines both frequency and spatial domain filtering to enhance image quality, particularly in terms of contrast and illumination.

Homomorphic filtering is used to improve the appearance of an image by correcting the non-uniform lighting conditions and enhancing contrast. The idea is to separate the image into its illumination and reflectance components, then process these components separately to improve the overall image quality.

Transfer Function Derivation

a. Image Model: An image I(x,y) can be modeled as:

$$I(x,y)=R(x,y)\times L(x,y)$$

where:

- I(x,y) is the observed image.
- R(x,y) is the reflectance component (details and textures).
- L(x,y) is the illumination component (light intensity).

b. Logarithmic Transformation: Applying a logarithmic transformation to both sides:

$$\log I(x,y)=\log R(x,y)+\log L(x,y)$$

This transformation converts the multiplicative model into an additive model.

c. Frequency Domain Filtering: In the frequency domain, we can apply a filter to the log-transformed image:

$$F\{\log I(x,y)\}=F\{\log R(x,y)\}+F\{\log L(x,y)\}$$

where F denotes the Fourier transform.

d. High-pass Filter: A high-pass filter H(u,v) is used to enhance high-frequency components (details) and suppress low-frequency components (illumination):

$$\text{Filtered Image}=H(u,v)\times F\{\log I(x,y)\}$$

e. Exponential Transformation: Apply the exponential function to revert the image to the spatial domain:

$$I'(x,y)=\exp(F^{-1}\{\text{Filtered Image}\})$$

3. Advantages

- **Enhanced Contrast**: Improves image contrast by separating and processing illumination and reflectance components.
- **Illumination Correction**: Corrects uneven lighting and enhances details that might be obscured by poor illumination.
- **Detail Preservation**: Can enhance details and textures by focusing on high-frequency components.

4. Disadvantages

- **Complexity**: The process involves multiple transformations and can be computationally intensive.
- **Over-enhancement**: Excessive filtering may lead to over-enhancement, introducing artifacts or noise.
- **Parameter Tuning**: Requires careful tuning of filter parameters to achieve the desired enhancement without introducing artifacts.

Inverse Filtering

Inverse filtering is a technique used to recover an image that has been degraded by a known blur. The process assumes that you know the blur function H that caused the degradation. The goal is to reverse this process to retrieve the original image X from the degraded image Y.

Mathematically:

If the degradation process can be described by: $Y=H \cdot X+N$

where Y is the observed image, X is the original image, H is the blur function (or point spread function), and N is noise.

The inverse filter tries to estimate X using: $X=H^{-1} \cdot Y$

Challenges:

- **Noise Amplification:** Inverse filtering can amplify noise, especially when the blur function H is not perfect or if there is noise in the observed image.
- **Instability:** The filter can become unstable if H is close to zero in some frequency ranges, leading to significant artifacts.

Wiener Filtering

Wiener filtering is a more sophisticated technique that aims to balance the trade-off between de-blurring and noise reduction. Unlike inverse filtering, Wiener filtering incorporates statistical information about the noise and the blur function.

Mathematically:

The Wiener filter is designed to minimize the mean square error between the estimated and the original image. The filter can be expressed as:

$H(f) = Sx(f) / [Sx(f) + Sn(f)]$

Where:

- **H(f):** The transfer function of the Wiener filter.
- **Sx(f):** The power spectral density (PSD) of the original signal.
- **Sn(f):** The power spectral density of the noise.

Key Points:

- **Noise Handling:** Wiener filtering is effective in handling noise, as it incorporates information about the noise level.
- **Adaptive Filtering:** It adapts to local variations in the image, which helps in reducing artifacts.

Comparison:

- **Inverse Filtering:** Directly attempts to reverse the degradation but may amplify noise and introduce artifacts.
- **Wiener Filtering:** Balances de-blurring and noise reduction, leading to better results in practical scenarios where noise is present.

Models of Noise in Images

1. **Gaussian Noise:**
    - **Description:** This type of noise is caused by random variations in the image pixel values and follows a Gaussian distribution (bell curve). It's common in many imaging systems and appears as grainy or speckled patterns.
    - **Appearance:** It looks like random, scattered dots or graininess over the image, which can reduce image clarity and detail.
2. **Salt-and-Pepper Noise:**
    - **Description:** This noise is characterized by the presence of random white and black pixels scattered throughout the image. It's like adding "salt" and "pepper" randomly to the image.
    - **Appearance:** It creates small, random spots of black and white on the image, leading to a speckled appearance that can obscure the actual content.

Ways to Assess Image Quality

1. **Peak Signal-to-Noise Ratio (PSNR):**
    - **Description:** PSNR is a measure of how much the original image differs from the processed or restored image. It calculates the ratio between the maximum possible value of a signal and the power of the noise affecting the signal.

    - **How It Works:** A higher PSNR value indicates better image quality with less distortion or noise. It's often measured in decibels (dB).
    - **Use Case:** Useful for comparing the quality of compressed or restored images to the original.
2. **Structural Similarity Index (SSIM):**
    - **Description:** SSIM measures the similarity between two images by comparing their structural information, luminance, and contrast. It assesses how well the structural features of the processed image match those of the original image.
    - **How It Works:** SSIM values range from -1 to 1, where 1 indicates a perfect match with no distortion. Higher SSIM values mean the processed image retains more of the original image's structure and quality.
    - **Use Case:** Provides a more perceptually relevant assessment of image quality than PSNR by considering human visual perception.

Summary:

- **Noise Models:** Gaussian noise introduces random graininess, while salt-and-pepper noise adds random black and white spots.
- **Quality Assessment:** PSNR measures the overall difference between images, while SSIM evaluates how well the image structure is preserved.

Image Compression

Image compression is the process of reducing the size of an image file without significantly compromising its quality.This is achieved by removing redundant data or representing the image data in a more efficient format , which requires less storage space and reduces transmission times over networks.

Why Do We Need Image Compression?

- **Storage Efficiency:** Compressed images occupy less storage space, allowing for more images to be stored on devices with limited capacity.
- **Transmission Efficiency:** Smaller image files can be transmitted faster over networks, improving website loading times and reducing data usage.
- **Bandwidth Reduction:** Smaller file sizes reduce the amount of data that needs to be transmitted over the internet or other networks, leading to faster load times and reduced bandwidth usage.
- **Faster Access:** Users experience quicker access to images when downloading or streaming.
- **Reduced Costs:** Lower storage and bandwidth requirements translate to cost savings for both service providers and users.
- **Faster Processing:** Smaller image files can be processed more quickly by software and hardware, improving overall performance in applications that handle images.

Types of Image Compression

Image compression techniques can be broadly categorized into two types:

1. Lossless Compression

Lossless compression reduces file size without discarding any image data. This means the original image can be perfectly reconstructed from the compressed file. It is ideal for images with critical details, such as medical images or technical diagrams.

- **Examples:**
  - **Run-length encoding (RLE):** Replaces sequences of identical values with a single value and count.
  - **Huffman coding:** Assigns shorter codes to frequently occurring pixel values and longer codes to less frequent ones.
  - **Arithmetic coding:** Represents data as a single number instead of a sequence of characters.

2. Lossy Compression

Lossy compression reduces file size by discarding some image data. This results in a smaller file size but with a loss of image quality. It is commonly used for images where a slight degradation in quality is acceptable, such as photographs and graphics for web use.

- **Examples:**
  - **JPEG (Joint Photographic Experts Group):** Uses discrete cosine transform (DCT) to remove redundant information and quantizes the DCT coefficients.
  - **PNG (Portable Network Graphics):** Supports both lossless and lossy compression, but is primarily used for lossless compression.
  - **GIF (Graphics Interchange Format):** Supports lossless compression and animation.

Lossy vs Lossless compression

Aspect	Lossy Compression	Lossless Compression
Definition	Reduces file size by removing some image data, leading to some quality loss.	Reduces file size without any loss of image quality; the original image can be perfectly reconstructed.
Quality	May result in visible quality loss, especially at high compression levels.	Maintains original image quality, with no loss of details.
File Size	Generally achieves smaller file sizes compared to lossless compression.	Results in larger file sizes compared to lossy compression, as no data is discarded.
Compression Ratio	High compression ratios, meaning more significant reduction in file size.	Lower compression ratios, as it preserves all image data.
Data Recovery	Lossy compression cannot perfectly recover the original image; some data is permanently	Lossless compression allows for perfect recovery of the original image without any loss.

	lost.	
Examples	JPEG, WebP, and some MPEG formats (for video).	PNG, GIF, TIFF (with lossless options), and some parts of JPEG 2000.
Use Cases	Web images, digital photography, and multimedia where file size is a priority over perfect quality.	Archival images, medical imaging, technical drawings, and scenarios where quality is critical.
Compression Methods	Uses techniques like quantization and approximation to reduce data	Uses algorithms like Huffman coding, LZW, and DEFLATE to encode data efficiently without loss.
Visual Artifacts	Can introduce artifacts like blurring or pixelation due to data loss.	No visual artifacts, as all original data is preserved.
Processing Speed	Often faster for encoding due to simpler algorithms, but may require more effort for decoding.	Typically slower for encoding due to the need to preserve all data, but decoding is straightforward.

Run-Length Encoding (RLE)

Run-Length Encoding (RLE) is a simple and efficient compression technique used for reducing the size of images and other types of data where sequences of repeated elements occur. It is particularly effective for images with large areas of uniform color or data with long runs of the same value.

How RLE Works

1. **Identify Runs:** The data is scanned for sequences of identical values, called runs.
2. **Count Occurrences:** The number of consecutive occurrences of the value in the run is counted.
3. **Replace with Count and Value:** The original sequence is replaced with a pair of values: the count and the value itself.

Example

Consider the following sequence of numbers:

111122333345555

Using RLE, this sequence can be compressed as:

41223445

Here, 41 represents four occurrences of the value 1, 22 represents two occurrences of the value 2, and so on.

Limitations of RLE

- **Limited Compression:** RLE is less effective for data with high variability and few repeated values, as it can even increase the size of data if runs are short or non-existent.
- **Loss of Efficiency:** In color images with complex patterns or detailed textures, RLE might not offer significant compression benefits and could even be less efficient compared to other compression methods.

Statistical Compression and **Spatial Compression** are two broad categories of image compression techniques that handle data differently based on the nature of the image data and the desired compression outcomes.

Statistical Compression

Statistical Compression techniques exploit the statistical properties of the data to reduce its size. They are based on the principle that certain data elements (like pixel values) occur more frequently than others. By using this frequency information, these methods encode data more efficiently.

Techniques:

1. **Huffman Coding:**
    - **Description:** Assigns shorter codes to more frequent symbols and longer codes to less frequent ones.
    - **Example:** Compresses text by encoding common letters with shorter codes.
2. **Arithmetic Coding:**
    - **Description:** Encodes entire sequences of symbols into a single number based on symbol probabilities.
    - **Example:** Encodes a string of pixel values as a fractional number.
3. **Run-Length Encoding (RLE):**
    - **Description:** Encodes sequences of repeated values as a count and value pair.
    - **Example:** 5A 3B 2C for a string AAAAABBBCC.

Advantages:

- High compression for repetitive data.
- Simple and effective for certain types of data.

Disadvantages:

- Less effective for non-repetitive or complex data.
- Some methods can be computationally intensive.

Spatial Compression

Spatial Compression techniques focus on exploiting the spatial redundancy in image data. This redundancy occurs due to the correlation between neighboring pixels. By analyzing patterns and relationships between pixels, spatial compression methods can reduce data size.

Techniques:

1. **Transform Coding:**
   - **Description:** Converts data into the frequency domain, compressing less important frequencies more aggressively.
   - **Example:** JPEG uses Discrete Cosine Transform (DCT).
2. **Predictive Coding:**
   - **Description:** Encodes differences between actual pixel values and predicted values based on neighboring pixels.
   - **Example:** Video compression uses motion prediction.
3. **Block-Based Compression:**
   - **Description:** Divides the image into blocks, compresses each block independently.
   - **Example:** JPEG compresses 8x8 pixel blocks using DCT.

Advantages:

- Effective for continuous-tone images with spatial correlations.
- Flexible and widely used in image and video compression.

Disadvantages:

- Can introduce artifacts (e.g., blockiness in JPEG).
- Complex algorithms may be required.

Summary:

- **Statistical Compression** focuses on encoding based on data frequency and statistical properties.
- **Spatial Compression** reduces redundancy by analyzing and transforming spatial relationships between pixels.

Statistical Compression vs Spatial Compression

Characteristics	Statistical Compression	Spatial Compression
Approach	Focuses on probability of occurrence of symbols	Focuses on spatial relationships between pixels
Data Type	Typically used for 1D data (text, audio, binary)	Typically used for 2D data (images, videos)
Compression Ratio	Can achieve high compression ratios for skewed probability	Can achieve high compression ratios for images with uniform color

	distributions	areas
Computational Complexity	Relatively low	Relatively high
Applications	Text compression, audio compression, binary data compression	Image and video compression (JPEG, MPEG)
Lossy vs. Lossless	Typically lossless	Can be either lossless or lossy (e.g. JPEG is lossy)
Techniques	Huffman coding, LZW coding, arithmetic coding	Run-Length Encoding (RLE), Discrete Cosine Transform (DCT)
Advantages	Fast compression and decompression, suitable for real-time applications	Can achieve high compression ratios for images, suitable for storage and transmission
Disadvantages	May not be effective for data with uniform probability distributions	Can be computationally intensive, may not be suitable for real-time applications

Huffman Coding

Huffman coding is a lossless data compression algorithm that assigns variable-length codes to input characters, lengths of which are based on the frequency of corresponding characters. The more frequent a character is, the shorter its code will be.

How It Works

1. **Frequency Analysis:**
   - Calculate the frequency of each character in the input data.
2. **Build a Priority Queue:**
   - Create a priority queue (or min-heap) where each node represents a character and its frequency. Nodes are ordered by frequency.
3. **Build the Huffman Tree:**
   - While there is more than one node in the queue, remove the two nodes with the lowest frequencies.
   - Create a new internal node with these two nodes as children and with a frequency equal to the sum of the two nodes' frequencies.
   - Insert this new node back into the priority queue.
   - Repeat until only one node remains. This node becomes the root of the Huffman tree.
4. **Generate Codes:**
   - Traverse the Huffman tree from the root to each leaf node. Assign '0' for left branches and '1' for right branches.
   - The path from the root to each leaf node represents the Huffman code for the corresponding character.
5. **Encode Data:**

- Replace each character in the input data with its corresponding Huffman code.

6. **Decode Data:**
   - Use the Huffman tree to decode the compressed data by following the path from the root to the leaf nodes according to the encoded bits.

Advantages

- **Efficient Compression:** Generally provides good compression ratios, especially for data with skewed frequency distributions.
- **Lossless:** The original data can be perfectly reconstructed from the compressed data.

Disadvantages

- **Requires Frequency Table:** The frequency table must be stored or transmitted alongside the encoded data.
- **Not Ideal for Small Data:** May not be as efficient for very small data sets or data with nearly uniform frequency distributions.

Color model

- A color model is a system for creating a full range of colors from a small set of primary colors.
- There are two types of color models.
  1. Additive color models which uses light to display color
  2. Subtractive color models which uses printing inks.

Different color model used in image processing are:

A. RGB color model:

- ❖ The RGB color model is an additive color model. In this case red, green and blue light are added together in various combinations to reproduce a wide spectrum of colors.
- ❖ The primary purpose of the RGB color model is for the display of images in electronic systems, such as on television screens and computer monitors and it's also used in digital photography.
- ❖ In order to create a color with RGB, three colored light beams (one red, one green, and one blue) must be superimposed.
- ❖ With no intensity, each of the three colors is perceived as black, while full intensity leads to a perception of seeing white.

B. CMYK (Cyan, Magenta, Yellow, Black)color model:

- ❖ The CMYK color model (four-color process) is a subtractive color model.
- ❖ CMYK works by partially or completely masking colors on a white background.

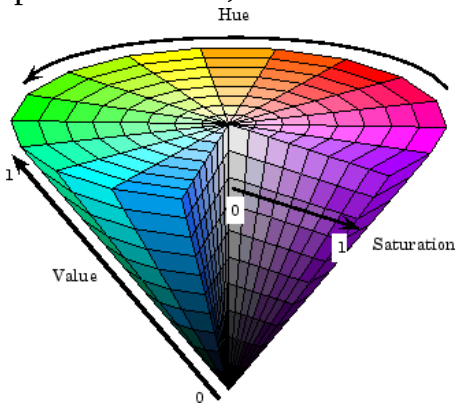
- The printed ink reduces the light that would otherwise be reflected. That's why this model is called subtractive because inks 'subtract' brightness from a white background from four colors: cyan, magenta, yellow and black.
- This enables the human eye to perceive a specific color made from the combination. In order to improve print quality and reduce large-scale interference patterns, the screen for each color is set at a different angle

C. HSV Color Model:

The HSV (Hue, Saturation, Value) color model is a color model that describes colors in terms of their hue, saturation, and value.

Components of HSV:

- **Hue (H):** The hue is the color's actual color, represented as an angle in the range of 0° to 360°. It is the color's position on the color wheel.
- **Saturation (S):** The saturation is the color's purity or intensity, represented as a value in the range of 0 to 1. A saturation of 0 represents a shade of gray, while a saturation of 1 represents a fully saturated color.
- **Value (V):** The value is the color's brightness or lightness, represented as a value in the range of 0 to 1. A value of 0 represents black, while a value of 1 represents white.



D.HSI color model:

- HSI stands for Hue, Saturation and Intensity. When humans view a color object it is described by its hue, saturation and brightness.
- Hue is a color attribute that describes a pure color (yellow, orange or red)
- Saturation gives a measure of the degree to which a pure color is diluted by white light.
- Brightness depends upon color intensity, which is key factor in describing color sensation. The intensity is easily measurable and the results are also easily interpretable
- Thus the model that is used to describe a color object is the HSI model.

Conversions between color models

1. RGB and CMYK

Conversion:

- **RGB to CMY**:

1) C = 1 - R,
2) M = 1 - G,
3) Y = 1 - B

- **CMY to RGB:**

1) R = 1 - C,
2) G = 1 - M,
3) B = 1 – Y

2. RGB to HSI

Conversion:

Normalization:

- Convert RGB values to the range [0, 1]:
- `R = R/255`
- `G = G/255`
- `B = B/255`

Intensity (I):

- Calculate the average of R, G, and B:
- I = (R + G + B)/3

Saturation (S):

- Find the minimum of R, G, and B:
- `min(R, G, B)`
- Calculate S:
- `S = 1 - 3 * min(R, G, B) / (R + G + B)`

Hue (H):

- Calculate intermediate values:
- `C = max(R, G, B) - min(R, G, B)`
- Determine the angle based on R, G, and B values:
- `if C == 0`
- `    H = 0`
- `elseif C != 0 and R == max(R, G, B)`
- `    H = 60 * (G - B) / C`

```
elseif C != 0 and G == max(R, G, B)
    H = 60 * (B - R) / C + 120
else
    H = 60 * (R - G) / C + 240
endif
Adjust H to the range [0, 360]:
if H < 0
    H = H + 360
endif
```

3.RGB to HSV:

a. Normalize RGB values to the range [0, 1]:

```
r' = R / 255

g' = G / 255

b' = B / 255
```

b. Calculate the maximum and minimum values:

```
mx = max(r', g', b')

mn = min(r', g', b')
```

c. Calculate the delta value:

```
df = mx - mn
```

d. Calculate the hue (H):

```
if mx == mn:

    H = 0

elif mx == r':

    H = (60 * ((g' - b') / df) + 360) % 360

elif mx == g':

    H = (60 * ((b' - r') / df) + 120) % 360

else:

    H = (60 * ((r' - g') / df) + 240) % 360
```

e. Calculate the saturation (S):

```
if mx == 0:
```

```
        S = 0
    else:
        S = df / mx
```

f. Calculate the value (V):

```
V = mx
```

Polygonal Approximation

Polygonal approximation is a technique used in image processing and computer vision to approximate a curve or a shape with a polygon. The goal is to find a polygon that closely resembles the original curve or shape, while reducing the number of vertices and simplifying the representation.

Properties of Polygonal Approximation:

Simplification: Polygonal approximation simplifies the representation of a curve or shape, reducing the number of vertices and making it easier to process.

Accuracy: The accuracy of the approximation depends on the algorithm used and the desired level of simplification.

Robustness to Noise: Polygonal approximation can be robust to noise and small variations in the curve or shape.

Regional Descriptor

Regional descriptors in image processing are features that describe the characteristics of specific regions or areas within an image. Unlike boundary descriptors that focus on the shape's contour, regional descriptors capture information about the interior of a shape or region. They are crucial for tasks like object recognition, classification, and image analysis, as they provide insights into the properties and attributes of different regions within an image.

Types of Regional Descriptors

1. **Shape Descriptors:**

- **Area:** Size of the region.
- **Perimeter:** Length of the region's boundary.
- **Compactness:** Measure of how round the region is.
- **Elongation:** Measure of how stretched or elongated the region is.
- **Euler number:** Number of connected components minus the number of holes.

2. **Statistical Descriptors:**

- **Mean:** Average pixel intensity within the region.
- **Standard deviation:** Measure of pixel intensity variation.
- **Histogram:** Distribution of pixel intensities.
- **Texture features:** Statistical measures of texture patterns (e.g., Haralick features).

3. **Spatial Descriptors:**

- **Centroid:** Center of mass of the region.
- **Moments:** Invariant moments that capture shape information.
- **Bounding box:** Rectangular region enclosing the entire region.

4. **Color Descriptors:**

- **Color histograms:** Distribution of color values within the region.
- **Color moments:** Statistical measures of color distribution.

Applications of Regional Descriptors

- **Object Recognition:** Identifying objects based on their regional properties.
- **Image Segmentation:** Grouping pixels into meaningful regions.
- **Image Retrieval:** Searching for images with similar regions.
- **Medical Image Analysis:** Analyzing regions of interest in medical images.

Boundary Descriptor

A boundary descriptor is a feature used in image processing and computer vision to describe the outline or shape of an object within an image. These descriptors help in identifying and analyzing objects by capturing their boundaries. They are crucial for tasks like object recognition, shape matching, and segmentation.

Types of Boundary Descriptors

1. **Simple Descriptors:**
    - **Length:** The total length of the boundary.
    - **Diameter:** The maximum distance between any two points on the boundary.
    - **Curvature:** How much the edge bends or curves at different points.
    - **Centroid:** The average position of all points on the edge, essentially the center of mass of the shape's boundary.

2. **Shape Descriptors:**

    - **Compactness:** A measure of how "round" an object is.
    - **Elongation:** How stretched out or elongated the shape is.
    - **Eccentricity:** How much the shape's boundary deviates from a perfect circle. Higher eccentricity means the shape is more oval-like.
    - **Fourier Descriptors:** Represents the boundary as a Fourier series.

3. **Statistical Descriptors:**
    - **Texture:** Describes patterns and variations in the boundary, often using statistics like how often certain gray levels appear near each other.
    - **Histogram:** Shows the distribution of pixel intensities (brightness levels) along the boundary.

Applications of Boundary Descriptors

- **Object Recognition:** Identifying objects based on their shape.
- **Image Retrieval:** Searching for images with similar shapes.
- **Medical Image Analysis:** Analyzing the shape of organs or tumors.
- **Computer Vision:** Object tracking and detection.

Chain Code

Chain code is a method used to represent the boundary of a shape in a digital image. It is particularly useful for shape analysis and pattern recognition. The technique encodes the path of the boundary as a sequence of directions, making it possible to describe the shape in a compact and efficient manner.

Chain code represents the boundary of a shape by recording the direction of movement from one boundary pixel to the next. The sequence of directions forms a "chain" that describes the shape's boundary.

Types of Chain Codes:

1. **4-Connected Chain Code:**
    - **Description:** Uses a 4-connected neighborhood to describe movements. It considers only horizontal and vertical directions.
    - **Directions:**
        - 0: Right
        - 1: Up
        - 2: Left
        - 3: Down
2. **8-Connected Chain Code:**
    - **Description:** Uses an 8-connected neighborhood to describe movements. It includes diagonal directions as well.
    - **Directions:**
        - 0: Right
        - 1: Up-Right
        - 2: Up
        - 3: Up-Left
        - 4: Left
        - 5: Down-Left
        - 6: Down
        - 7: Down-Right

Generating Chain Codes:

1. **Find Boundary Points:**
    - Extract the boundary points of a shape from a binary image where the shape is represented by pixels with value 1.
2. **Trace the Boundary:**
    - Starting from a known boundary point, move to neighboring boundary pixels and record the direction of movement.
3. **Record Directions:**
    - Use the direction codes to create a chain code sequence that represents the shape's boundary.

Relational Descriptor

Relational descriptors are features or attributes used to describe the spatial and contextual relationships between objects within an image or dataset. Unlike boundary descriptors, which focus on the shape and outline of individual objects, relational descriptors emphasize how objects are positioned and related to one another.

Types of Relational Descriptors

1. Spatial Relationships:

- **Distance Metrics:** Measures the distance between objects (e.g., Euclidean distance).
- **Relative Position:** Describes the location of one object relative to another.
- **Proximity:** Indicates how close objects are to each other.

2. Geometric Relationships:

- **Overlap:** Measures how much two objects intersect (e.g., Intersection Over Union).
- **Alignment:** Describes if objects are aligned (parallel, perpendicular).
- **Adjacency:** Indicates if objects are next to each other or share a boundary.

3. Topological Relationships:

- **Connectivity:** Shows if objects are connected or separate.
- **Containment:** Describes if one object is inside another.
- **Enclosure:** Indicates if one object is completely enclosed by another.

4. Structural Relationships:

- **Hierarchy:** Represents parent-child or part-whole relationships between objects.

5. Statistical Relationships:

- **Correlation:** Measures the relationship between changes in features.
- **Co-occurrence:** Describes how often features or objects appear together.

PCA

Principal Component Analysis (PCA) is a statistical technique used to reduce the dimensionality of a dataset while preserving as much information as possible. It achieves this by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

DCT

Discrete Cosine Transform is used in lossy image compression because it has very strong energy compaction, i.e., its large amount of information is stored in very low frequency component of a signal and rest other frequency having very small data which can be stored by using very less number of bits (usually, at most 2 or 3 bit).
To perform DCT Transformation on an image, first we have to fetch image file information (pixel value in term of integer having range 0 – 255) which we divides in block of 8 X 8 matrix and then we apply discrete cosine transform on that block of data.

The DCT is defined as:

$X[k] = \sum[n=0 \text{ to } N-1] \; x[n] * \cos(\pi * (2n+1) * k / 2N)$

where:

X[k] is the k-th DCT coefficient

x[n] is the n-th sample of the input signal or image

N is the number of samples in the input signal or image

k is the frequency index, ranging from 0 to N-1

Dilation

Dilation expands the boundaries of objects in a binary image. It adds pixels to the edges of objects, making them larger and filling in small holes or gaps.

How Dilation Works:

- **Structuring Element:** A small binary matrix (e.g., 3x3 or 5x5) used to determine the shape of the dilation operation. The structuring element slides over the image and applies the dilation.
- **Operation:** For each pixel in the image, if any pixel under the structuring element is set to 1, the pixel at the center is set to 1.

Applications of Dilation:

- **Filling Gaps:** Fills small holes or gaps within objects.
- **Connecting Disjoint Parts:** Connects broken or disconnected parts of objects.
- **Object Enhancement:** Makes objects larger and more visible.

Erosion

Erosion shrinks the boundaries of objects in a binary image. It removes pixels from the edges of objects, making them smaller and erasing small noise or unwanted details.

How Erosion Works:

- **Structuring Element:** Similar to dilation, a small binary matrix is used to define the erosion operation.
- **Operation:** For each pixel in the image, if all pixels under the structuring element are 1, the pixel at the center remains 1; otherwise, it is set to 0.

Applications of Erosion:

- **Noise Removal:** Removes small noise or unwanted pixels from objects.
- **Object Separation:** Separates connected objects or structures.
- **Shape Reduction:** Makes objects smaller and less prominent.

Opening and Closing

Opening and **Closing** are fundamental morphological operations used in image processing to modify the shape and structure of objects in binary or grayscale images. They are particularly useful for preprocessing and cleaning up images.

Opening

Opening is a sequence of two operations: **erosion** followed by **dilation**.

How Opening Works:

1. **Erosion:** Removes pixels from the edges of objects. This step shrinks the objects and removes small noise or details.
2. **Dilation:** Expands the edges of the shrunk objects back to their original size, but with noise removed.

Effect of Opening:

- **Smooths Object Boundaries:** Removes small noise and smooths contours.
- **Separates Connected Objects:** Helps in separating objects that are connected or very close to each other.

- **Removes Small Objects:** Eliminates small objects that are smaller than the structuring element.

Closing

Closing is a sequence of two operations: **dilation** followed by **erosion**.

How Closing Works:

1. **Dilation:** Expands the edges of objects. This step fills small holes and gaps within the objects.
2. **Erosion:** Shrinks the expanded objects back to their original size, but with small holes filled.

Effect of Closing:

- **Fills Small Holes:** Fills small gaps and holes within objects.
- **Smooths Object Boundaries:** Closes small gaps between objects.
- **Connects Disconnected Objects:** Helps in connecting nearby objects or segments.

Applications:

- **Opening:** Noise reduction, separating connected objects, smoothing boundaries.
- **Closing:** Filling small gaps, connecting nearby objects, removing small holes.

Hit or Miss transform

Hit-or-Miss Transform is a morphological operation used in image processing to detect specific patterns or shapes within a binary image. It is particularly useful for pattern recognition and template matching.

The Hit-or-Miss Transform checks if a specific pattern (defined by a structuring element) is present in a binary image. The structuring element is a template that defines the pattern you are looking for.

How It Works

1. **Define a Structuring Element:** This is a binary matrix (template) that represents the pattern you want to detect. The structuring element contains both 1s (foreground) and 0s (background).
2. **Complement the Structuring Element:** Compute the complement of the structuring element, where the 1s become 0s and the 0s become 1s.
3. **Apply the Transform:**
    - **Hit Condition:** For each position in the image, check if the structuring element fits perfectly when overlaid on the image at that position. This means that all the pixels under the 1s of the structuring element match the corresponding pixels in the image, and the pixels under the 0s in the structuring element correspond to 0s in the image.

    - **Miss Condition:** If the pattern matches, mark that position as a hit.
4. **Result:** The output is a binary image where the detected patterns are marked. The output image has 1s at positions where the pattern is detected and 0s elsewhere.

Applications:

- **Pattern Recognition:** Detecting specific shapes or features in an image.
- **Template Matching:** Finding occurrences of predefined patterns in images.
- **Feature Detection:** Identifying features or structures that match a given template.

Boundary Extraction

Boundary Extraction is a morphological operation used to identify and outline the boundaries or edges of objects in a binary image. It is crucial for tasks such as object detection, shape analysis, and image segmentation.

Boundary Extraction isolates the contour or outline of objects in a binary image by highlighting the edges. This operation is typically performed using morphological operations like erosion and subtraction.

How Boundary Extraction Works

1. **Perform Erosion:**
    - Apply erosion to the binary image using a structuring element. Erosion removes pixels from the edges of objects, effectively shrinking them.
2. **Subtract Eroded Image from Original:**
    - Subtract the eroded image from the original binary image. This subtraction highlights the boundary pixels, as these are the pixels that were removed by erosion.

Steps for Boundary Extraction

1. **Original Image:**
    - Start with a binary image where objects are represented by 1s and the background by 0s.
2. **Erosion Operation:**
    - Apply erosion with a structuring element (e.g., a 3x3 square) to the binary image. This shrinks the objects and creates a version of the image with reduced object size.
3. **Subtract Eroded Image:**
    - Subtract the eroded image from the original image. The result will be an image that shows only the boundary or edges of the objects.

Applications

- **Object Detection:** Identifying and outlining objects within an image.
- **Shape Analysis:** Extracting contours for shape analysis and recognition.

- **Image Segmentation:** Defining object boundaries for segmentation purposes.
- **Feature Extraction:** Capturing the outlines of features for further analysis or processing.

Gradient Operators

Gradient Operators are essential tools in image processing and computer vision for detecting edges and changes in intensity within an image. They calculate the gradient of image intensity to find regions of high spatial frequency, which correspond to edges and transitions in the image. Here are some key gradient operators:

1. Roberts Operator

Roberts Operator is a simple gradient operator that uses small 2x2 kernels to compute the gradient of the image. It is designed to detect edges along diagonals.

Usage:

- Detects diagonal edges.
- Simple and computationally efficient.
- Sensitive to noise due to its small kernel size.

2. Prewitt Operator

Prewitt Operator calculates gradients using larger 3x3 kernels. It detects edges by measuring the rate of change in both horizontal and vertical directions.

Usage:

- Detects edges in both horizontal and vertical directions.
- More robust to noise compared to Roberts due to larger kernel size.

3. Sobel Operator

Sobel Operator combines gradient calculation with Gaussian smoothing to detect edges. It uses 3x3 kernels to approximate the gradient of the image intensity.

Usage:

- Detects edges in both horizontal and vertical directions.
- Provides a smoother gradient approximation with less noise compared to Roberts and Prewitt due to the larger kernel size and smoothing effect.

Image Segmentation

Image segmentation is the process of partitioning an image into multiple regions or segments, each corresponding to a specific object or feature of interest. The goal is to simplify the representation of the image into meaningful regions, making it easier to analyze and understand.

Why Image Segmentation?

- **Object Detection:** Identifying and locating objects within an image.
- **Image Analysis:** Understanding the content and structure of an image.
- **Image Compression:** Reducing image size while preserving essential information.
- **Medical Image Analysis:** Analyzing medical images for diagnosis and treatment planning.

Types of image segmentation:

1. Semantic Segmentation:

Semantic segmentation involves assigning a class label to each pixel in the image, such as "car," "road," "sky," etc. This type of segmentation is useful for applications such as object recognition, scene understanding, and autonomous driving.

2. Instance Segmentation:

Instance segmentation involves identifying and separating individual objects within a class, such as multiple cars in an image. This type of segmentation is useful for applications such as object detection, tracking, and robotics.

3. Edge-Based Segmentation:

Edge-based segmentation involves detecting edges in the image and using them to separate objects. This type of segmentation is useful for applications such as object recognition, image compression, and image enhancement.

4. Region-Based Segmentation:

Region-based segmentation involves grouping pixels into regions based on their similarity in terms of color, texture, or other features. This type of segmentation is useful for applications such as image compression, image retrieval, and medical imaging.

5. Thresholding-Based Segmentation:

Thresholding-based segmentation involves converting the image into a binary image by setting a threshold value, where pixels above the threshold are considered part of the object, and pixels below are considered part of the background. This type of segmentation is useful for applications such as object recognition, image enhancement, and quality control.

Challenges in Image Segmentation

- **Noise:** Image noise can significantly impact segmentation accuracy.
- **Illumation Variations:** Changes in lighting conditions can affect segmentation results.
- **Object Overlap:** Objects that overlap can be difficult to segment accurately.
- **Computational Cost:** Some segmentation methods can be computationally expensive.

Edge Detection

Edge detection is a fundamental step in image processing and computer vision, which involves identifying the boundaries or edges between different objects or regions in an image. Edges are crucial features that help in object recognition, segmentation, and understanding the structure of an image.

Purpose

- **Identify Boundaries:** Detect edges to outline objects.
- **Feature Extraction:** Extract key features for further analysis.
- **Image Enhancement:** Improve contrast and visibility of edges.

Common Edge Detection Techniques

1. **Sobel Operator**
2. **Prewitt Operator**
3. **Roberts Operator**
4. **Canny Edge Detector**
    - **Description:** A multi-stage algorithm that includes smoothing, finding gradients, non-maximum suppression, and edge tracking by hysteresis.
    - **Steps:**
        - **Smoothing:** Applies Gaussian filter to reduce noise.
        - **Gradient Calculation:** Uses Sobel operators to find edges.
        - **Non-Maximum Suppression:** Thin out edges by removing non-maximum pixels.
        - **Edge Tracking:** Uses hysteresis to finalize edges based on threshold values.
    - **Usage:** Provides precise and continuous edge detection with reduced noise.
5. **Laplacian of Gaussian (LoG)**
    - **Description:** First applies Gaussian smoothing and then the Laplacian operator to find edges.
    - **Steps:**
        - **Gaussian Smoothing:** Reduces noise.
        - **Laplacian Calculation:** Finds regions with rapid intensity changes.
    - **Usage:** Detects edges by identifying zero-crossings in the Laplacian.