Image Processing Functions and Filters

December 24, 2023

1 Introduction

This document covers various image processing functions and filters, providing insights into their purposes, parameters, and expected outcomes. It includes sections on Noise Filters, Smoothing Filters, Sharpening Filters, Edge Detection, and Histograms.

2 Noise Functions

This Section Describes Several Noise Image Processing Functions.

2.1 Salt and Pepper Function

Introduces salt or pepper noise to an image.

2.1.1 Parameters

- `image_path`: Path to the input image file.
- `noise_type`: Type of noise ('PS' for salt, 'PP' for pepper).
- `noise_density`: Density of the noise (between 0 and 1).

2.1.2 Expected Outcome

Returns an image with added salt or pepper noise.

2.2 Uniform Function

Adds uniform noise within a specified range to an image.

2.2.1 Parameters

- `image_path`: Path to the input image file.
- `noise_start`: Start value of the uniform noise range.
- `noise_end`: End value of the uniform noise range.
- `noise_percentage`: Percentage of pixels affected (between 0 and 1).

2.2.2 Expected Outcome

Returns an image with added uniform noise.

3 Point Processing Functions

This section describes the various point processing functions

3.1 Add a Value to an Image Function

Adds an integer value to each pixel in an image, adjusting overall brightness.

3.1.1 Parameters

- `image`: The input image.
- `value`: Integer value to be added to each pixel.

3.1.2 Expected Outcome

Returns an image with increased pixel values.

3.2 Subtract a Value From an Image Function

Subtracts an integer value from each pixel, adjusting overall darkness.

3.2.1 Parameters

- `image`: The input image.
- `value`: Integer value to be subtracted from each pixel.

3.2.2 Expected Outcome

Returns an image with decreased pixel values.

3.3 Multiply a Value to an Image Function

Multiplies each pixel by an integer value, adjusting contrast.

3.3.1 Parameters

- `image`: The input image.
- `value`: Integer value to be multiplied with each pixel.

3.3.2 Expected Outcome

Returns an image with adjusted pixel values.

3.4 Divide Image by a Value Function

Divides each pixel by an integer value, normalizing intensity levels.

3.4.1 Parameters

- `image`: The input image.
- `value`: Integer value to divide each pixel.

3.4.2 Expected Outcome

Returns an image with normalized intensity levels.

4 Smoothing Filters

Smoothing filters are utilized to reduce noise and create a smoother version of an image.

4.1 Average Filter

The average filter replaces each pixel value with the average value of its neighborhood, reducing noise and producing a blurred version of the original image. The average filter equation is given by:

$$\bar{I}(x, y) = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} I(x+i, y+j) \tag{1}$$

4.2 Max and Min Filters

The max filter replaces each pixel value with the maximum value within its neighborhood, while the min filter replaces it with the minimum value. These filters are used in morphological operations to enhance or suppress certain image features.

$$\max(I)(x, y) = \max_{i=0}^{m-1} \max_{j=0}^{n-1} I(x+i, y+j) \tag{2}$$

4.3 Median Filter

The median filter replaces each pixel value with the median value of its neighborhood, effective in reducing impulse noise while preserving edges.

$$\text{median}(I)(x, y) = \text{median}\left(I(x+i, y+j) \mid i = 0, \ldots, m-1,\, j = 0, \ldots, n-1\right) \tag{3}$$

4.4 Ideal Low-Pass Filter (LPF)

The ideal LPF theoretically eliminates high-frequency components in the frequency domain, allowing only low-frequency components to pass.

$$H(u, v) = \begin{cases} 1, & \text{if } D(u, v) \leq D_0 \\ 0, & \text{if } D(u, v) > D_0 \end{cases}$$

where $D(u, v)$ is the distance from the center in the frequency domain, and D_0 is the cutoff frequency.

4.5 Butterworth Low-Pass Filter

The Butterworth LPF provides a smoother transition between the passband and stopband compared to the ideal LPF, characterized by its order and cutoff frequency.

$$H(u, v) = \frac{1}{1 + \left(\frac{D(u,v)}{D_0}\right)^{2n}}$$

where $D(u, v)$ is the distance from the center in the frequency domain, D_0 is the cutoff frequency, and n is the filter order.

4.6 Gaussian Low-Pass Filter

The Gaussian LPF uses a Gaussian function in the frequency domain to attenuate higher frequencies, effective for noise reduction while preserving image details.

$$H(u, v) = \exp\left(-\frac{D(u, v)^2}{2D_0^2}\right)$$

where $D(u, v)$ is the distance from the center in the frequency domain, and D_0 is the cutoff frequency.

5 Sharpening Filters

Sharpening filters aim to enhance edges and details in an image, accentuating high-frequency components.

5.1 Ideal High-Pass Filter

The ideal high-pass filter eliminates low-frequency components and allows high-frequency components to pass, enhancing edges.

$$H(u,v) = \begin{cases} 0, & \text{if } D(u,v) \leq D_0 \\ 1, & \text{if } D(u,v) > D_0 \end{cases}$$

where $D(u,v)$ is the distance from the center in the frequency domain, and D_0 is the cutoff frequency.

5.2 Butterworth High-Pass Filter

The Butterworth high-pass filter provides a smoother transition between the passband and stopband compared to the ideal high-pass filter.

$$H(u,v) = \frac{1}{1 + \left(\frac{D(u,v)}{D_0}\right)^{2n}}$$

where $D(u,v)$ is the distance from the center in the frequency domain, D_0 is the cutoff frequency, and n is the filter order.

5.3 Gaussian High-Pass Filter

The Gaussian high-pass filter uses a Gaussian function in the frequency domain to attenuate lower frequencies, emphasizing high-frequency components.

$$H(u,v) = 1 - \exp\left(-\frac{D(u,v)^2}{2D_0^2}\right)$$

where $D(u,v)$ is the distance from the center in the frequency domain, and D_0 is the cutoff frequency.

6 Edge Detection Filters

The edge detection module identifies and enhances boundaries within an image. It utilizes Sobel operators and custom convolution matrices to compute gradients, revealing intensity changes and edges.

6.1 Point Processing for Edge Detection

Converts the input image to grayscale, applies Sobel operators to compute gradients in both horizontal and vertical directions, and then combines them to obtain the gradient magnitude.

6.2 Vertical Edge Detection

Focuses on vertical edge detection using the vertical Sobel operator.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

6.3 Horizontal Edge Detection

Responsible for horizontal edge detection using the horizontal Sobel operator.

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

6.4 Left-Diagonal Edge Detection

Implements left-diagonal edge detection using a custom Sobel operator.

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

6.5 Right-Diagonal Edge Detection

Performs right-diagonal edge detection using a custom Sobel operator.

$$\begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}$$

7 Logarithmic Filters

log transformation and its inverse are used for enhancing the visibility of details in images that have a wide range of intensity levels, particularly those with predominantly low-intensity values. The log transformation is a type of gray-level transformation that is applied to the pixel values of an image.

7.1 Log Transformation

The purpose of the log transformation is to compress the higher intensity values and expand the lower intensity values, making it useful for images with a wide range of intensity levels. It is particularly effective in bringing out details in the darker regions of an image.

$$s = c \log(1 + r) \tag{4}$$

Where c is a constant, and r, the original image pixel, ≥ 0, s is the output pixel

7.2 Inverse-Log Transformation

Applying the inverse log transformation is important if you want to recover the original intensity values after performing log transformation.

$$r = \exp\left(\frac{c}{s}\right) - 1 \tag{5}$$

Where c is a constant, and r is the original pixel intensity, s, the transformed pixel intensity, $\neq 0$ Histograms provide a visual representation of the distribution of pixel intensities in an image.

8 Power-Law (Gamma) Transformation

The gamma transformation is a non-linear operation applied to pixel intensities in an image. It is used to adjust the image's contrast and brightness, particularly in the low-intensity region. The transformation is defined as:

$$s = cr^{\gamma} \tag{6}$$

Where s is the new pixel value, c is a constant, r is the old pixel γ is the power, c and γ are positive constants.

- $\gamma < 1$ maps a narrow range of dark input values into a wider range of output values, (more bright image).
- $\gamma > 1$ maps a narrow range of bright input values into a wider range of output values (more dark image).

9 Histograms Filters

a histogram is a graphical representation of the distribution of pixel intensities in an image. It provides a visual summary of the frequency or count of each intensity level within the image. The x-axis of the histogram represents the intensity values, and the y-axis represents the frequency of occurrence of each intensity level.

9.1 Histogram Equalization

Histogram equalization is a technique used in image processing to enhance the contrast of an image by redistributing the pixel intensities. The basic idea is to spread out the intensity values over the entire range of possible values. The transformation function for histogram equalization is often denoted as:

$$\mathrm{HE}(I)(x,y) = \frac{L-1}{MN} \sum_{k=0}^{I(x,y)} n_k \tag{7}$$

9.2 Histogram Stretching

Improves the contrast in an image by stretching the range of intensity values to span a desired range of values Histogram Stretching is often denoted by:

$$I_{new} = (I - Min)\frac{NewMax - NewMin}{Max - Min} + NewMin \tag{8}$$

10 Conclusion

This comprehensive report covers various image processing functions, smoothing filters, sharpening filters, edge detection techniques, and histograms. Each section provides detailed explanations, parameters, and expected outcomes, contributing to a better understanding of image processing concepts.