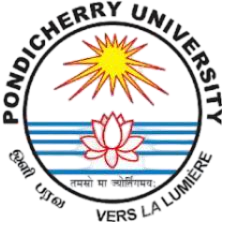
**DEPARTMENT OF ELECTRONICS ENGINEERING**

**SCHOOL OF ENGINEERING AND TECHNOLOGY**

PONDICHERRY UNIVERSITY

KALAPET, PUDUCHERRY



**ADSP Assignment**

**ON**

**MATLAB GUI for Real-Time Image Denoising with Haar Transform and Resolution Control**

Submitted By Submitted To

Name: Vishnu S Dr. K. ANUSUDHA

Register Number: 24MTECEPY0003 Pondicherry University

Programme: M. Tech (ECE) – Intelligent

Communication Systems

Year: 2024 - 2025

# CONTENTS

S. No. Topic Page No.

1. Objective 4
2. Introduction 5
3. Block Diagram 9
4. Results 12
5. Discussion 14
6. Conclusion 15
7. Reference 16

# Objective: -

To develop a real-time image denoising application including Haar wavelet transform, providing users with an intuitive GUI for loading, processing, and visualizing images. The application allows dynamic adjustments of noise level, filter strength, and resolution in real time, enabling effective noise reduction. Users can choose from multiple denoising algorithms and save processed images for further use. This project aims to enhance understanding of wavelet transforms and improve digital image quality.

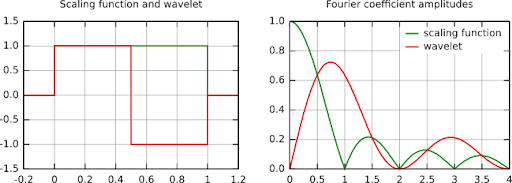
# INTRODUCTION

The continuous wavelet transform (WT) is a mathematical technique that has gained prominence in signal analysis, particularly for its ability to decompose signals into various frequency components. This technique is recognized for its effectiveness in revealing the spatial distribution of singularities in multifractal measures. One of the key advantages of the wavelet transform is its capability to analyze singular functions, making it a versatile tool in image processing and analysis.

In this project, we explore the application of the Haar wavelet transform for advanced real-time image denoising. The Haar transform allows for effective noise reduction by breaking down images into their constituent frequency components, facilitating the identification and removal of noise while preserving critical image features. This capability is essential for enhancing image clarity and improving the quality of visual data.

Our application provides users with a user-friendly graphical interface that supports the loading, processing, and visualization of images. Users can select from multiple denoising algorithms, including Haar wavelet, Gaussian blur, median filter, bilateral filter, and Wiener filter. The application allows for real-time adjustments of parameters such as noise level, filter strength, color balance, and resolution, ensuring immediate feedback on the denoising process.

1. *HAAR Wavelets:*





The Haar wavelet transform is a mathematical operation that decomposes a signal into Haar wavelets to capture local features and discontinuities. It's the simplest type of wavelet transform and is named after mathematician Alfréd Haar.

Haar wavelets are a foundational type of wavelet in signal and image processing, known for their simplicity and efficiency. Introduced as the first wavelet by Alfréd Haar, the Haar wavelet transform decomposes a signal or image into multiple levels of detail, making it highly effective for applications like image compression, edge detection, and noise reduction.

The Haar wavelet transform operates by breaking down a signal into low-frequency (approximate) and high-frequency (detail) components at various scales. This decomposition is achieved through scaling and wavelet functions, where each level captures increasingly fine details of the signal. This allows for a compact representation of the signal, retaining essential features while discarding noise or unnecessary information. Haar wavelets are especially suited for real-time applications due to their fast computation and low memory requirements.

In image denoising, Haar wavelets can isolate and remove high-frequency noise while preserving the underlying structure of the image. By thresholding the high-frequency components, noise can be significantly reduced, leading to a cleaner and clearer image. Haar wavelets are widely used because of their effectiveness in identifying sharp changes in an image, making them particularly valuable in applications that require detail preservation, such as medical imaging, remote sensing, and video processing.

**Components:**

**Graphical User Interface (GUI) Design**

* **GUI Window and Panels**: The code creates the main window for the application, along with two main panels—one for controls and another for displaying images.
* **Image Display**: The display panel contains four quadrants to show different versions of the image: the original, noisy, processed (color adjusted), and denoised versions. This layout allows users to visually compare the effects of different adjustments and algorithms.

**2. Image Loading and Saving**

* **Load Image Function**: The loadImage function allows the user to load an image file, which is then displayed in the "Original Image" quadrant. It’s the starting point of the image processing workflow.
* **Save Image Function**: After processing, users can save the final denoised image using the saveImage function. This saves the output in a chosen file location.

**3. Denoising Algorithms**

* **Algorithm Selection**: The code provides a dropdown menu for choosing different denoising algorithms, each with distinct characteristics:
  + **Haar Wavelet**: This applies the Haar wavelet transform for denoising, which decomposes the image into low and high-frequency components. It removes high-frequency noise by thresholding and reconstructs the image with reduced noise.
  + **Gaussian Blur**: A smoothing filter that reduces noise by averaging pixels in a specified neighborhood. It's often used to reduce high-frequency noise.
  + **Median Filter**: This filter replaces each pixel’s value with the median of neighboring pixels, which helps remove salt-and-pepper noise.
  + **Bilateral Filter**: This filter smooths the image while preserving edges, making it suitable for denoising without blurring important details.
  + **Wiener Filter**: An adaptive filter that reduces noise based on local statistics, often effective for images with Gaussian noise.

**4. Noise Addition and Slider Controls**

* **Noise Level Slider**: Users can adjust the amount of noise added to the image using the noise level slider. This simulates different noise conditions, allowing users to see how various algorithms handle them.
* **Filter Strength Slider**: Controls the strength of the selected denoising algorithm, giving flexibility to make the denoising effect stronger or weaker.

**5. Color and Brightness Adjustment**

* **RGB Sliders**: These sliders allow users to adjust the red, green, and blue levels separately, providing control over the color intensity of the image. This feature can enhance or alter the color balance before applying denoising.
* **Brightness Slider**: This control adjusts the image's brightness, allowing for further refinement. It is particularly useful for images that might appear too dark or too bright after processing.

**6. Resolution Adjustment**

* **Resolution Selection Menu**: The dropdown menu lets users choose different output resolutions (e.g., 640x480, 800x600), which can help in scaling images for specific uses or optimizing for display on devices with different resolutions.

**7. Image Processing Workflow**

* **Generate Button**: After setting all adjustments and choosing an algorithm, the generateImage function is triggered to apply all selected settings, such as noise level, color adjustments, and denoising algorithm. The processed and denoised images are then displayed in their respective quadrants.
* **Step-by-Step Processing**:
  1. **Adding Noise**: Based on the noise level slider, the code adds salt-and-pepper noise to simulate image degradation.
  2. **Color and Brightness Adjustments**: RGB and brightness values are adjusted to modify the image’s color tone and intensity.
  3. **Applying Denoising Algorithm**: The chosen algorithm is applied to reduce the noise, with adjustments based on filter strength and selected algorithm parameters.
  4. **Resolution Scaling**: Finally, the processed image is resized based on the selected resolution, allowing the output to match specific resolution requirements.

**8. Wavelet Transform and Haar Wavelets**

* **Haar Wavelet Denoising**: When selected, the Haar wavelet transform decomposes the image into multiple frequency bands, isolating high-frequency noise and then thresholding it. The image is then reconstructed without high-frequency noise. Haar wavelets are simple yet effective for detecting and removing noise due to their straightforward implementation and efficiency.

**Working Principle:**

Key principles include:

1. **Denoising with Wavelet Transforms**: The project utilizes the Haar wavelet transform to decompose the image into frequency components, isolating noise from significant details. This separation allows for targeted noise reduction without significant blurring or loss of critical image features.
2. **Interactive Parameter Control**: Through sliders and dropdown options, users can adjust noise levels, color intensity, brightness, and resolution in real-time. This enables users to visually assess the effect of each parameter and tailor the output based on image quality and clarity.
3. **Multi-Resolution and Comparison View**: The project’s GUI divides the workspace into multiple panels, enabling simultaneous comparison between the original, noisy, processed, and denoised images. This design helps users evaluate the effectiveness of denoising and color adjustments at various resolutions.
4. **Versatile Denoising Algorithms**: By integrating various filtering techniques (e.g., Gaussian blur, median filtering, and bilateral filtering), the application accommodates a range of noise types and intensities, providing users with flexible solutions for different imaging scenarios.

## BLOCK Diagram

A diagram of a computer

Description automatically generated

* **Input (Image)**: This is the starting point where the user loads an image into the application. The loaded image is then ready for further processing.
* **Algorithm Type**: This block allows the user to select the specific algorithm or type of wavelet transformation they wish to apply to the image. Different algorithms may be used depending on the nature of the noise or the specific enhancement requirements.
* **Noise Level**: This option lets the user adjust the level of noise to be handled in the image. Setting an appropriate noise level is essential for efficient denoising, as it helps the algorithm distinguish between noise and image details.
* **Filtering Strength**: This block provides a control to set the intensity of the filtering applied during the processing. Higher filtering strength will result in more aggressive noise reduction, potentially at the cost of finer details in the image.
* **RGB Levels**: This allows adjustment of individual RGB (Red, Green, Blue) color levels to fine-tune the image’s color balance and contrast. It is particularly useful for enhancing visual quality or correcting color distortions.
* **Resolution**: This control allows the user to adjust the image resolution, either increasing or decreasing it. Higher resolution provides more detail, while lower resolution may reduce processing time and file size.
* **Processing**: This central block represents the actual processing that occurs in the application, where all the selected parameters (algorithm, noise level, filtering strength, RGB levels, and resolution) are applied to modify the image.
* **Output (Processed Image)**: The processed image is the final result of the GUI application, displayed to the user after applying the selected adjustments and enhancements.

## Results

A screenshot of a computer

Description automatically generated

Fig: Matlab GUI Interface for WAVELET Transform

A screenshot of a computer

Description automatically generated

Fig: Morse WAVELET Transform

A screenshot of a computer

Description automatically generated

Fig: Haar WAVELET Transform

A screenshot of a computer

Description automatically generated

Fig: Amor WAVELET Transform

A screenshot of a computer

Description automatically generated

Fig: MATLAB GUI for WAVELET Transform for image

Input image = 2.1 mb

A screenshot of a computer

Description automatically generated

Fig: Median WAVELET Transform Denoised Image (241kb)

A screenshot of a computer

Description automatically generated

Fig: Weiner Filter WAVELET Transform Denoised Image (600kb)

## CONCLUSION

This project demonstrates the effectiveness of wavelet-based real-time image denoising and enhancement using an interactive MATLAB GUI. By integrating Haar wavelet transforms and other denoising techniques, the application provides users with precise control over noise reduction, color adjustments, and resolution scaling. The project highlights the adaptability of wavelet transforms in managing various noise types without sacrificing critical image features, making it suitable for diverse applications in fields like medical imaging, satellite sensing, and digital media. This tool exemplifies how advanced signal processing techniques can be translated into user-friendly applications, providing a powerful resource for enhancing image quality in real-world scenarios.

## References

1. Schepers HE, van Beek JH, Bassingthwaighte JB. Four Methods to Estimate the Fractal Dimension from Self-Affine Signals. IEEE Eng Med Biol Mag. 2002 Aug 6;11(2):57-64. doi: 10.1109/51.139038. PMID: 23024449; PMCID: PMC3459993.
2. <http://www.scholarpedia.org/article/Wavelet-based_multifractal_analysis>