Image Informatics - 240150459

Varsha Venkataraman

2025-01-08

Task 01 – Image Reading and Displaying

Brief Introduction

As highlighted in [11], understanding the fundamentals of image file formats [1] and implementing efficient file readers significantly enhances performance in image manipulation tasks. By applying similar principles [12], [13], this algorithm outlines the process of reading and displaying a PGM (Portable Gray Map) image using custom functions, supporting future applications in digital image processing and computer vision.

Algorithm for Image Reading and Displaying

Step 1: Call `process_image()` and validate `.pgm` extension with `validate_pgm_file()`.
Step 2: If valid, call `image_read()` to read header info (magic number, width, height, max value). If any line exceeds 70 characters, throw an error.
Step 2.1: If magic number is `'P5'`, call `process_pgm_noisyImage()` for binary PGM, else process ASCII PGM.
Step 3a: For ASCII PGM, initialize `warning_count = 0`. Decode lines, strip whitespace, skip comments.
Step 3.1: If lines exceed 70 characters, increment `warning_count` and throw a warning.
Step 3.2: Call `read_pgm_header()` for width, height, `max_value`, and `pixel_lines`.
Step 3.3: Validate pixel data, apply gamma correction, reshape pixels, and return the image.
Step 3b: For `'P5'`, call `process_pgm_noisyImage()` and read binary data.
Step 3.1: Call `read_binary_lines()` to extract header and pixel data, validate, apply gamma correction, reshape, and return the image.
Step 4: If header exceeds 70 characters, throw an error: "Header exceeds 70 characters."
Step 5: Return and display the processed image.

Key Findings

- **Format Validation:** Successfully differentiated between P2 (ASCII) and P5 (Binary) PGM formats, handled the header information ensuring robust input handling for both types.
- **Custom Implementation:** Implemented a detailed algorithm for reading and processing PGM files, demonstrating an understanding of file handling and image processing concepts.
- **Gamma Correction:** Applied gamma correction to adjust pixel intensity levels, improving image representation for visualization.[1]

Results and Conclusion

- Successfully processed and displayed a the provided PGM image file. The pixel intensities were accurately read, reshaped, and visualized.
- Validated file formats and handled invalid formats with appropriate errors. Algorithm handled images of varying sizes and formats efficiently. Displayed with clear intensity differentiation and enhanced visibility through gamma correction.

- **Robust Workflow and learning outcome:** The algorithm ensures correctness and efficiency in processing PGM images.Gained a deeper understanding of image file formats, header parsing, data validation, and visualization techniques.

Flow Chart

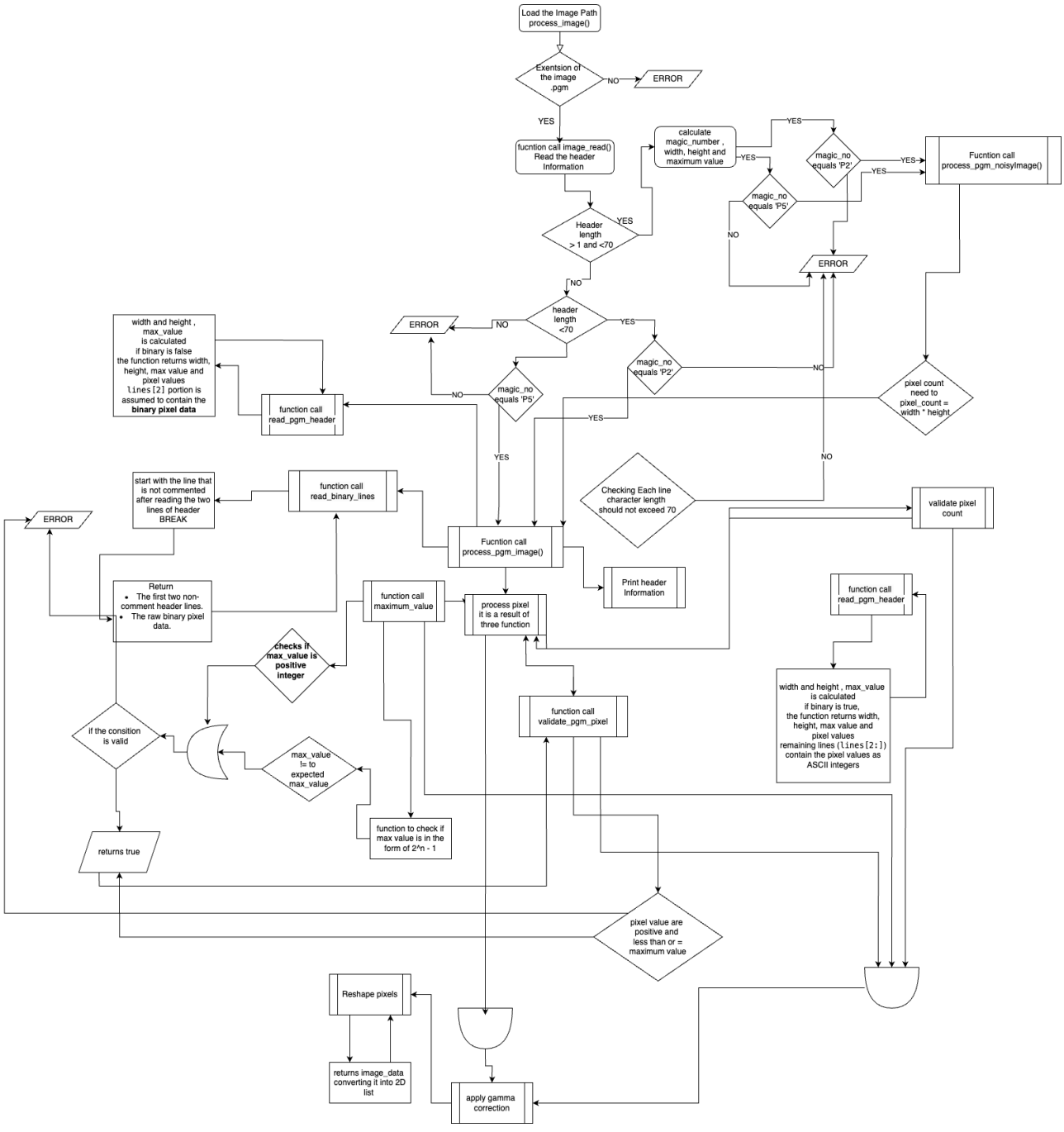


Figure 1: Flowchart illustrating the PGM processing algorithm.

Output

Figure 2: Output of Image by read function.

Task 02 – Wavelet decomposition

Brief Introduction

Wavelet transformation is a powerful technique for analyzing and processing images, providing multi-resolution decomposition of data. It allows the separation of an image into approximation and detail coefficients, enabling efficient compression, denoising, and feature extraction[14]. This algorithm explores the implementation of wavelet decomposition and reconstruction for image processing.

Input: A 2D numpy array representing the grayscale image. with number of levels for the Forward Discrete Wavelet Transform (FDWT). [15]

Output: Approximation and detail coefficients (horizontal, vertical, diagonal).[15] and The original image reconstructed from the decomposed coefficients using the Inverse Discrete Wavelet Transform (IDWT).

Algorithm for Haar Wavelet Transformation and Image Reconstruction

Step 1: Pad the Image Data as Haar deals with even dimension [15], Check if the dimensions (rows and columns) of the image are even. If the dimensions are odd, pad the array with zeros to make both dimensions even.

Step 2: Perform Forward Discrete Wavelet Transform (FDWT)
-Step 2.1: For each level of decomposition: Apply the 1D Haar wavelet filter: [14]

$$\text{Low-pass filter output} = \frac{x_{2n} + x_{2n+1}}{sqrt(2)}, \quad \text{High-pass filter output} = \frac{x_{2n} - x_{2n+1}}{sqrt(2)}$$

For each row: Compute low-pass and high-pass filtered values, For each column: Compute low-pass and high-pass filtered values.

-Step 2.2: Combine results into four components, Approximation ,Horizontal Details, Vertical Details, Diagonal Details

-Step 2.3: Store the detail coefficients (horizontal, vertical, diagonal) and update the approximation for the next level.

-Step 3: Perform Inverse Discrete Wavelet Transform (IDWT)
-Step 3.1: For each level in reverse: Combine the approximation and detail coefficients into a single matrix.
-Step 3.1.1: Apply the inverse Haar wavelet filter: For each column: Reconstruct the original column from low-pass and high-pass components. For each row: Reconstruct the original row from low-pass and high-pass components.

-**Step 3.2:** Update the approximation until the original image dimensions are restored.
-**Step 4:** Perform IDWT on the decomposed coefficients to reconstruct the image.
The Inverse Discrete Wavelet Transform (IDWT) reconstructs the original image by combining the approximation (low-pass) and detail (high-pass) coefficients.

-**Step 5:** Display the reconstructed image using a grayscale colormap (`cmap='gray'`).

-**Step 6:** Calculate the Mean Squared Error (MSE) between the original and reconstructed images.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_{\text{original},i} - x_{\text{reconstructed},i})^2$$

Evaluate the wavelet decomposition's quality based on the MSE value.

Key Findings and results

The results of the forward and inverse discrete wavelet transform (FDWT and IDWT) show that the Haar wavelet filter effectively decomposes the image into approximation and detail coefficients across multiple levels.

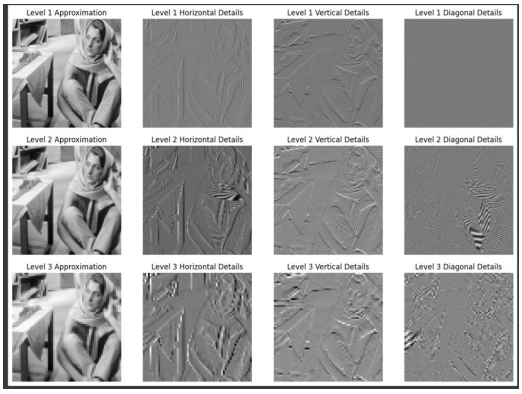


Figure 3: Output of Image by read function.

After applying the inverse transform (IDWT), the image is successfully reconstructed, with minimal loss of information. The computed Mean Squared Error (MSE) between the original and reconstructed image confirms the efficiency of the wavelet transform, with very low values indicating accurate reconstruction.The computed Mean Squared Error (MSE) between the original and reconstructed image confirms the efficiency of the wavelet transform, with very low values indicating accurate reconstruction.

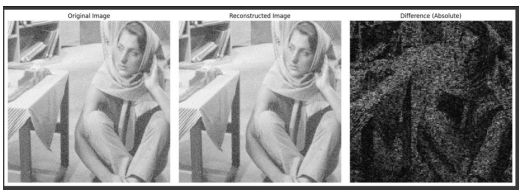


Figure 4: Output of Image by read function.

Flow Chart

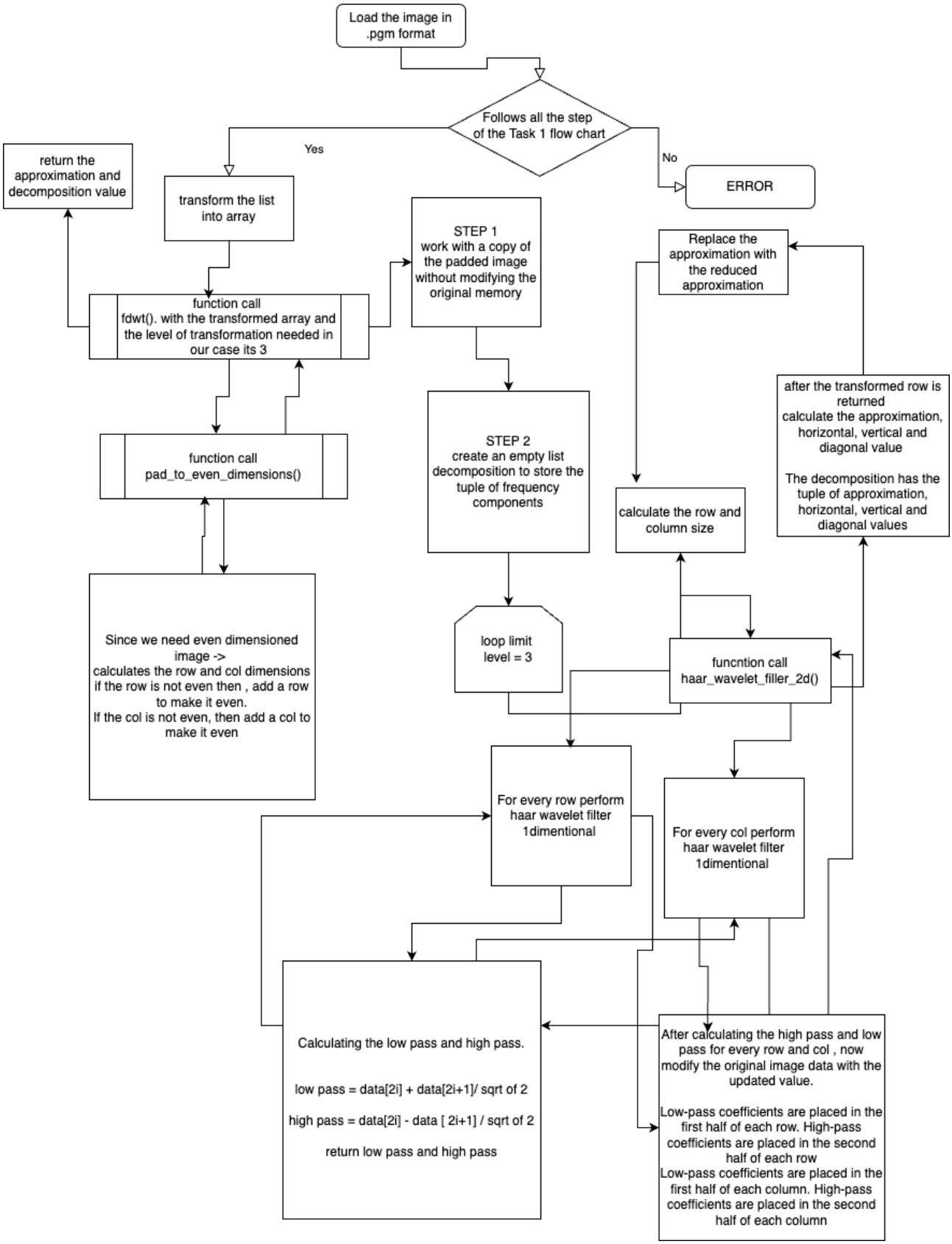


Figure 5: Flowchart illustrating forward wavelet decomposition algorithm.

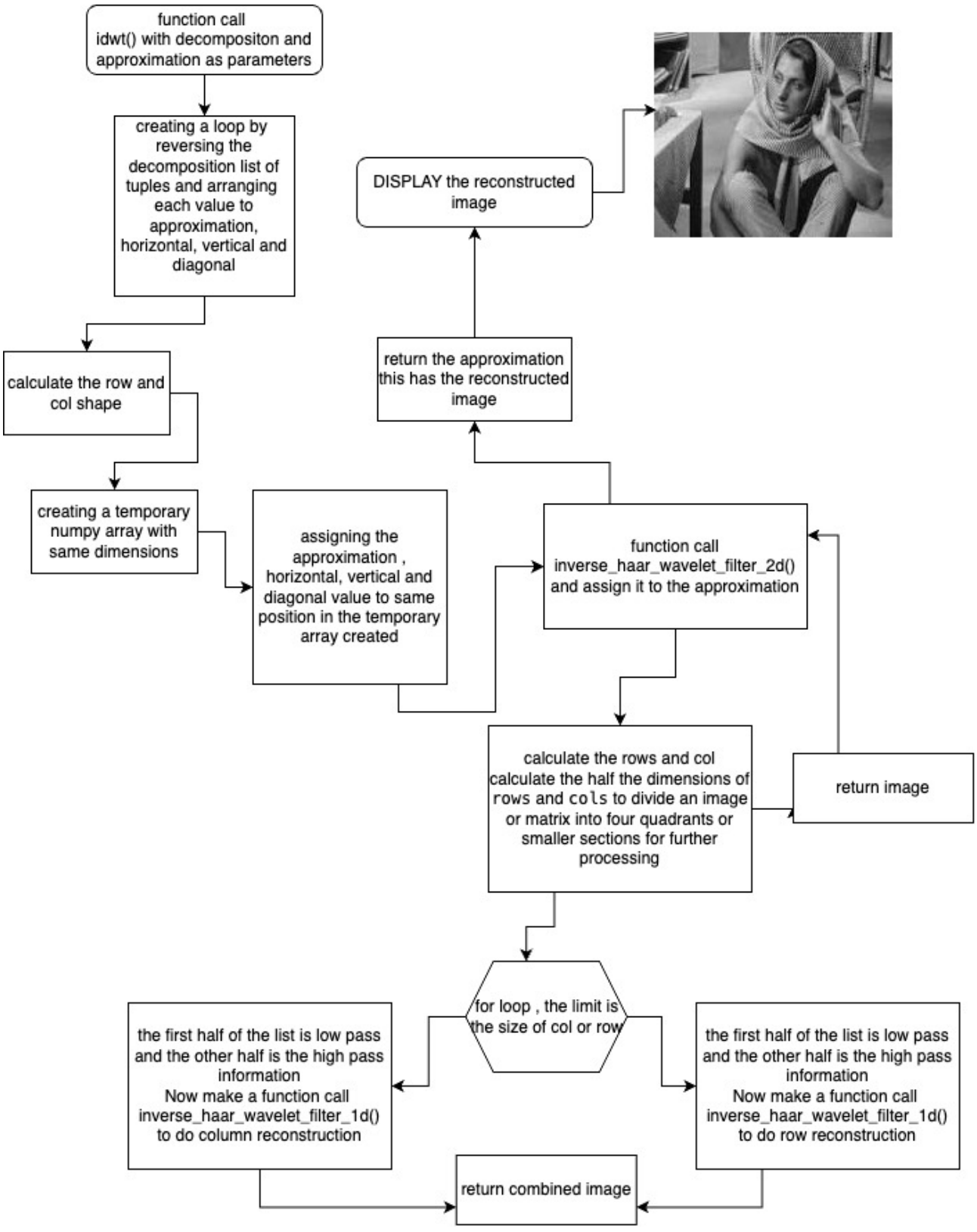


Figure 6: Flowchart illustrating inverse wavelet decomposition algorithm.

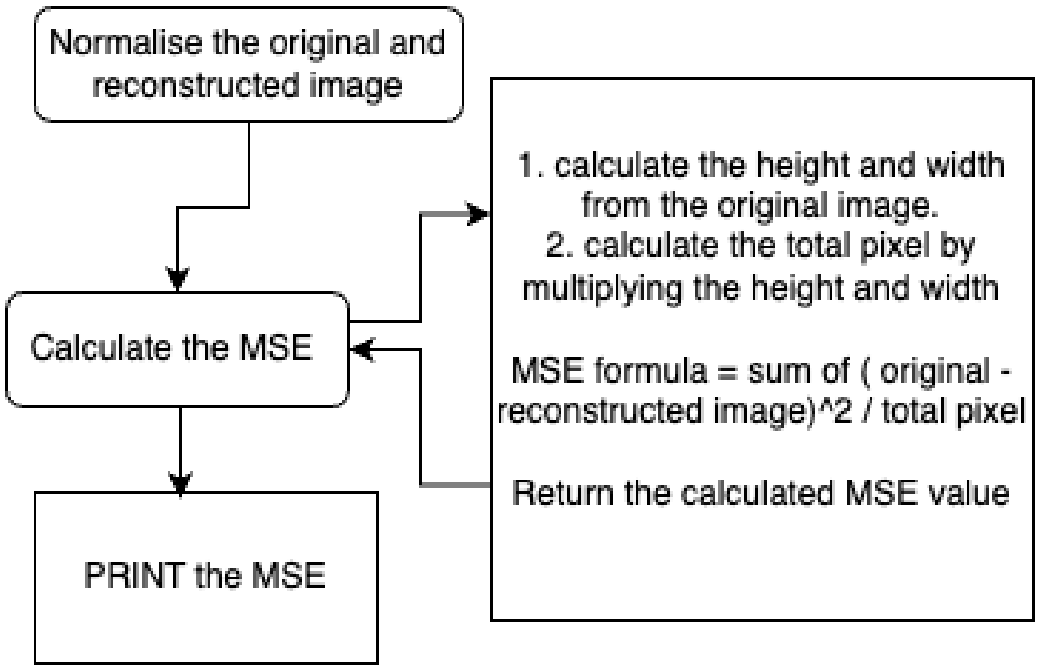


Figure 7: Flowchart illustrating MSE calculation algorithm.

Discussions and Conclusions

The Haar wavelet decomposition provides a simple yet powerful technique for multi-level image compression and analysis. The low MSE values demonstrate that the FDWT and IDWT are working as expected, preserving the key features of the original image. This method can be further optimized and applied in various image processing tasks, such as denoising, compression, and feature extraction, while maintaining computational efficiency.

Task 03 – Image Denoising

Brief Introduction

The primary goal of denoising is to recover a clean image from a noisy one while preserving essential details. Various types of noise, such as Gaussian, salt-and-pepper, and impulse noise, necessitate diverse denoising strategies [4]. The Wavelet Transform (WT) is a powerful tool for image denoising due to its multi-scale analysis capabilities. Unlike spatial domain methods, WT operates in the frequency domain, allowing effective noise separation and feature preservation. Classic methods like VisuShrink use WT to compute adaptive noise suppression thresholds, significantly improving denoising performance [3]. Recent hybrid strategies combine wavelet-based denoising with median filtering, Gaussian filtering, and total variation minimization to address limitations of individual methods, enhancing overall denoising effectiveness.

Algorithm: Image Denoising using Wavelet Transform

-**Step 1:** Initialize Parameters, Set wavelet level to 3, filter strength `nlm_h` to 0.4, and TV denoising weight `tv_weight` to 0.1.

-**Step 2:** Classify image noise as impulse or Gaussian.

-**Step 3:** Apply median filter to remove impulse noise by replacing each pixel with the median value from its neighborhood.

-**Step 4:** Apply Gaussian filter to smooth image and remove Gaussian noise by averaging pixel values with a Gaussian kernel.

-**Step 5:** Perform wavelet decomposition and separate noise at multiple scales.
- Classify noise (‘salt and pepper’ or ‘Gaussian’).
- Apply median filtering for ‘salt and pepper’ and Gaussian filtering for ‘Gaussian’ noise.
- Perform thresholding using VisuShrink on wavelet coefficients and reconstruct the image.
- Apply Non-Local Means Denoising by averaging similar patches with filter strength `nlm_h`.
- Apply Total Variation Denoising for edge-preserving smoothing using `tv_weight`.

-**Step 6:** Predefined Wavelet Denoising Methods ,Use VisuShrink with a universal threshold based on noise levels.

-**Step 7:** Return the denoised image close to the original clean image.

-**Step 8:** Apply adaptive weight fusion [18], [19] using wavelet denoising, mean filtering, and median filtering outputs to get a fused denoised image.

-**Step 9:** Calculate MSE between original and denoised images.

-**Step 10:** Compute Structural Similarity Index:

$$\text{SSIM}(I, K) = \frac{(2\mu_I\mu_K + c_1)(2\sigma_{IK} + c_2)}{(\mu_I^2 + \mu_K^2 + c_1)(\sigma_I^2 + \sigma_K^2 + c_2)}$$

Key findings:

The proposed algorithm effectively denoises images, preserving critical structures while removing noise. Unlike the mean and median filters, which introduce blurring and detail loss, the fusion method retains edges and fine textures more effectively. The custom wavelet denoising, which incorporates dynamic thresholding and adaptive fusion, outperforms the predefined wavelet methods.

This combined approach shows excellent noise reduction for both Gaussian and impulse noise, common in medical imaging. It excels at preserving edges, textures, and key anatomical structures, especially in boundary regions, confirming its effectiveness for medical image denoising [4], [5], [6].

Result:

With respect to the attached below figure (8), the fusion image acheived the lowest MSE value and the highest SSIM value when compared with the traditional filters and standalone wavelet denoising algorithm. The method preserves the fine details in the medical images, particularly the bone stuctures, while significantly suppresing noise without over-smoothing when compared with the mean and median filters.The results confirm that the adaptive wavelet-based fusion outperforms the individual methods in both qualitative and visual metrics and quality.

Method	MSE	SSIM
Combined Denoising	0.0009213849325072198	0.8843780259974102
Wavelet Denoising	0.0014756743536076 51	0.8683068749318255
Mean Filter	0.0010888744705791946	0.7067134803353814
Median Filter	0.0010550637186100599	0.6670486239959674
VisuShrink	0.0011717210342295273	0.7660917705985782

Figure 8: Comparison metrics

Image Denoising Flowchart

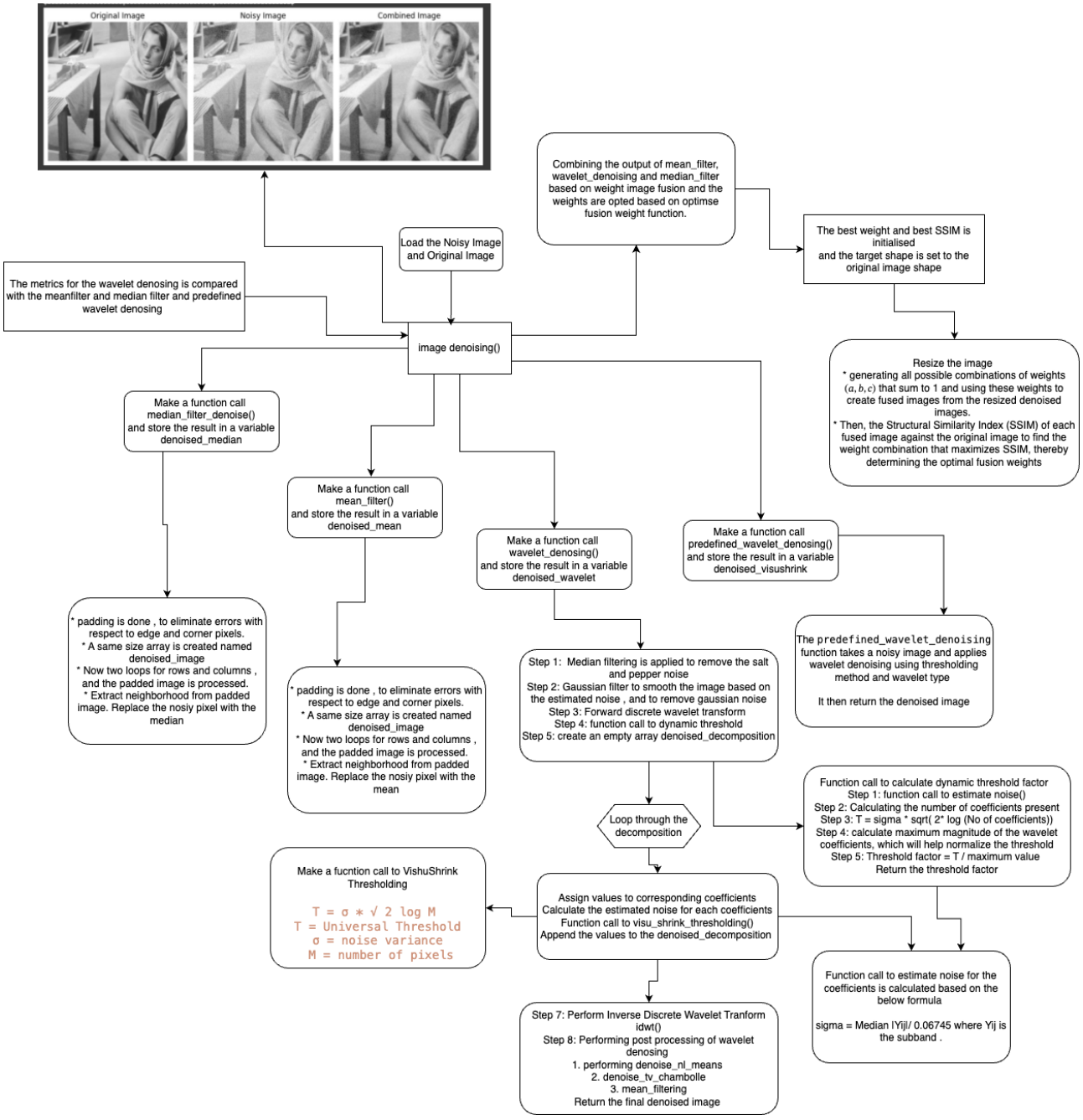
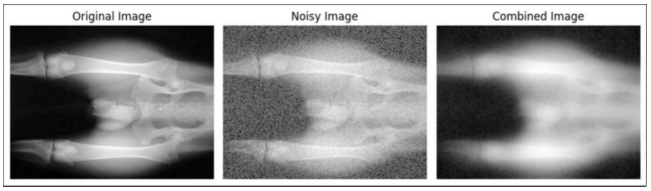


Figure 9: Flowchart illustrating Image Denoising algorithm.

Conclusion:



The combined denoising approach maintains a balance between the noise supression and detail preservation. The weighted averaging fusion (or Weighted image fusion) effectively leverages the strengths of each denoising method, resulting in a denosied image that retains the important features such as edges, textures and other fine

details. Future work could focus on further optimizing the fusion process by experimenting with more advanced optimization techniques, such as machine learning-based weight selection and deep learnig approach [7].

References

[1] Netpbm Documentation, ”PGM (Portable Gray Map) image format specification,” 2024. [Online]. Available: https://netpbm.sourceforge.net/doc/pgm.html. [Accessed: Dec. 28, 2024].

[2] L. Hongqiao and W. Shengqian, ”A new image denoising method using wavelet transform,” in *Proc. 2009 Int. Forum Inf. Technol. Appl.*, 2009, pp. 111–114, doi: 10.1109/IFITA.2009.47.

[3] B. K. S. Kumar, ”Image denoising based on non-local means filter and its method noise thresholding,” *Signal, Image Video Process.*, vol. 7, pp. 1211–1227, Nov. 2013, doi: 10.1007/s11760-012-0389-y.

[4] G. Xi, W. Sun, and L. Ma, ”Mixed wavelet and median filter for image denoising,” in *Proc. 2009 Int. Conf. Comput. Intell. Softw. Eng.*, 2009, pp. 1–4, doi: 10.1109/CISE.2009.5364696.

[5] A. Boyat and B. K. Joshi, ”Image denoising using wavelet transform and median filtering,” in *Proc. 2013 Nirma Univ. Int. Conf. Eng. (NUiCONE)*, 2013, pp. 1–6, doi: 10.1109/NUiCONE.2013.6780128.

[6] Q. Song, L. Ma, J. Cao, and X. Han, ”Image denoising based on mean filter and wavelet transform,” in *Proc. 2015 4th Int. Conf. Adv. Inf. Technol. Sens. Appl. (AITS)*, 2015, pp. 39–42, doi: 10.1109/AITS.2015.17.

[7] C. Liu and L. Zhang, ”A novel denoising algorithm based on wavelet and non-local moment mean filtering,” *Electronics*, vol. 12, no. 6, p. 1461, 2023, doi: 10.3390/electronics12061461.

[8] P. L. Parmar, ”Image denoising using VisuShrink algorithm,” *Vishwakarma Gov. Eng. Coll. J.*, vol. 2, no. 2, pp. 762–764, 2013.

[9] A. Buades, B. Coll, and J.-M. Morel, ”A non-local algorithm for image denoising,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, 2005, pp. 60–65, doi: 10.1109/CVPR.2005.38.

[10] M. Mittermary, S. G. Nikolov, H. Hutter, and M. Grasserbauer, ”Wavelet de-noising of Gaussian peaks: A comparative study,” *Chemometrics and Intelligent Laboratory Systems*, vol. 34, pp. 187–202, 1996.

[11] S. Pawar, P. Halgaonkar, J. W. Bakal, and V. M. Wadhai, ”Implementation of PPM image processing and median filtering,” *International Journal of Computer Applications*, vol. 14, 2011. doi: 10.5120/1843-2489.

[12] vailable: https://ilkerbayram.github.io/ehb372e/HaarDWT/

[13] vailable: https://github.com/Tanu-N-Prabhu/Python/blob/master/Reading_An_Image_In_Python_(Without_Using_Special_Libraries).ipynb

[14] C. S. Leung and S. Kulkarni, ”Implementationofwaveletdecompositionandreconstructionforanimage usingTMS320C6701,” *Electrical Engineering Department, Texas A&M University-Kingsville*, Kingsville, Texas 78363.

[15] Available: https://uk.mathworks.com/help/wavelet/ref/haart.html

[16] Available: https://stackoverflow.com/questions/57439509/implementing-haar-wavelet-in-python-without-packages

[17] S. Kavitha and H. Inbarani, “COVID-19 and MRI Image Denoising Using Wavelet Transform and Basic Filtering,” 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)

[18] L. Song, Y. Lin, W. Feng, and M. Zhao, “A Novel Automatic Weighted Image Fusion Algorithm,” State Key Laboratory of Precision Measuring Technology and Instruments, Tianjin University.

[19] G. Qi, G. Hu, N. Mazur, H. Liang, and M. Haner, “A Novel Multi-Modality Image Simultaneous Denoising and Fusion Method Based on Sparse Representation,”