2D DWT Based Segmentation for Noisy Samples

Jared Thomas

September 12, 2024

1 Introduction/Objective

This week, I aimed to create a Python GUI application that allows users to apply a series of 2D Discrete Wavelet Transforms (DWTs) to segment areas of interest on a noisy sample. The application is designed to process an image, perform DWT to highlight various features, and provide visual outputs for analysis.

2 Code Explanation

2.1 Importing Libraries

The following libraries are imported:

```
import cv2
import numpy as np
import pywt
import tkinter as tk
from tkinter import filedialog
import matplotlib.pyplot as plt
```

These libraries provide image processing (OpenCV), numerical computing (NumPy), wavelet transforms (PyWavelets), and a file dialog GUI (Tkinter).

2.2 File Upload Functionality

The function below allows the user to upload an image using a file dialog: This function opens a file dialog

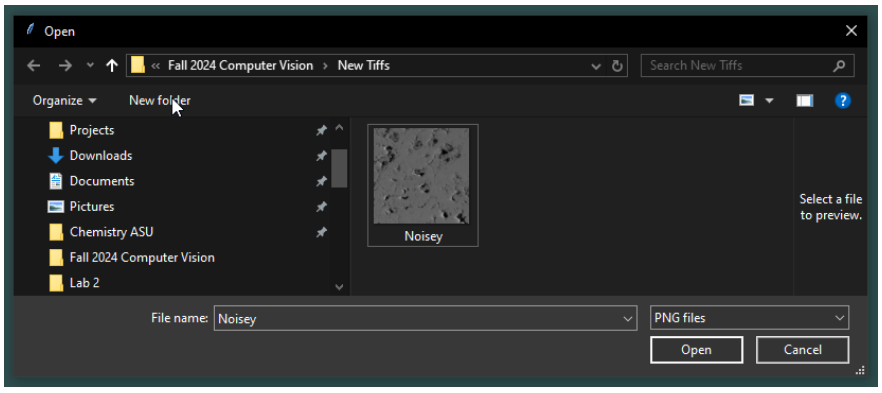

Figure 1: Example of file upload functionality

for the user to select a PNG image. If a file is selected, it is processed further.

```
def upload_image():
    root = tk.Tk()
    root.withdraw()
```

```
    file_path = filedialog.askopenfilename(filetypes=[("PNG files", "*.png")])
    if file_path:
        process_image(file_path)
    else:
        print("No file selected.")
```

2.3 Image Processing and Wavelet Transform

The image is loaded, normalized, and a 2D Discrete Wavelet Transform (DWT) is applied:

```
def process_image(file_path):
    original_image = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
    image = original_image / 255.0
    wavelet = "haar"
    coeffs2 = pywt.dwt2(image, wavelet)
    cA, (cH, cV, cD) = coeffs2
```

The image is first converted to grayscale and normalized to a range of [0,1]. The 2D DWT decomposes the image into approximation (cA) and detail coefficients (cH, cV, cD). This is given by the equation:

$$DWT2(f(x,y)) = \sum_{m,n} f(x,y)\psi(x-m, y-n) \tag{1}$$

Ref. the bounding box below as an example of the DWTs stepped-through.

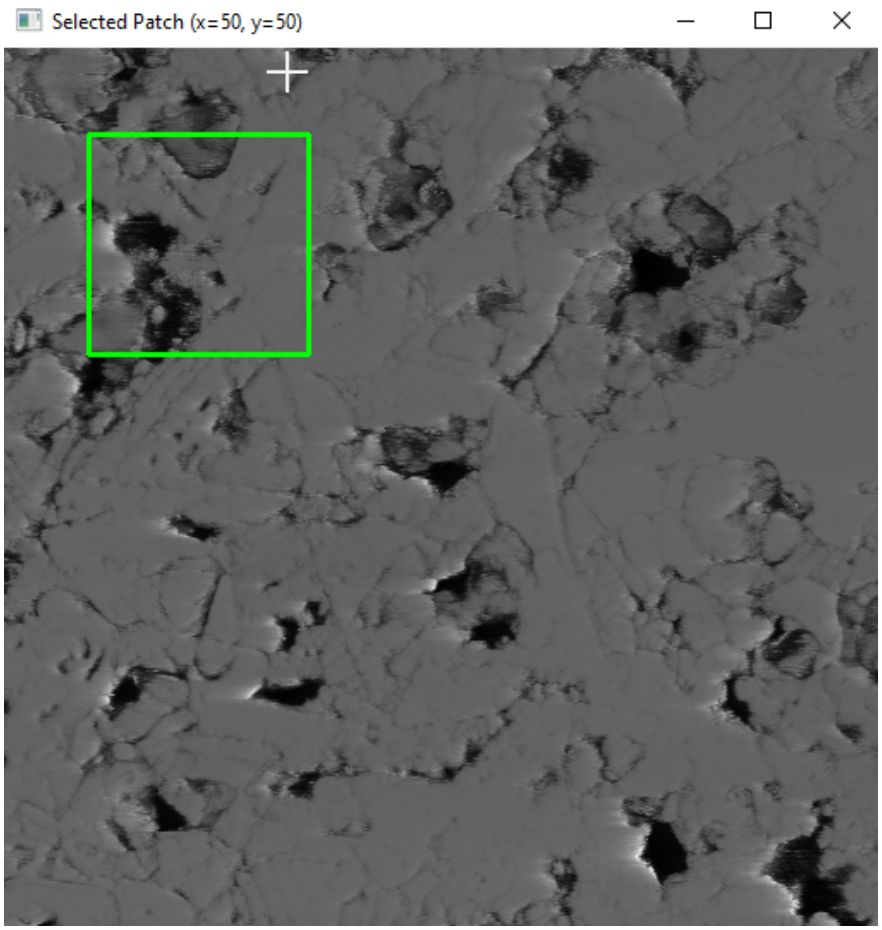

Figure 2: Example of the selected patch and its pixel values

2.3.1 Approximation Coefficients (cA)

The approximation coefficients represent the low-frequency content of the image:

$$cA(x, y) = \sum_{m,n} f(x - m, y - n)\psi_{LL}(m, n) \tag{2}$$

2.3.2 Horizontal Detail Coefficients (cH)

The horizontal detail coefficients capture horizontal edges:

$$cH(x, y) = \sum_{m,n} f(x - m, y - n)\psi_{LH}(m, n) \tag{3}$$

2.3.3 Vertical Detail Coefficients (cV)

The vertical detail coefficients capture vertical edges:

$$cV(x, y) = \sum_{m,n} f(x - m, y - n)\psi_{HL}(m, n) \tag{4}$$

2.3.4 Diagonal Detail Coefficients (cD)

The diagonal detail coefficients capture diagonal edges:

$$cD(x, y) = \sum_{m,n} f(x - m, y - n)\psi_{HH}(m, n) \tag{5}$$

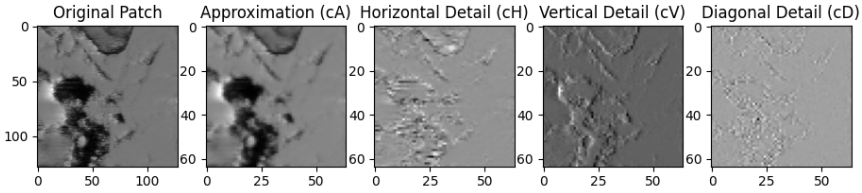

Figure 3: Example outputs for cA, cH, cV, and cD

3 Final Segmented Image

The following image shows the result of the segmentation process:

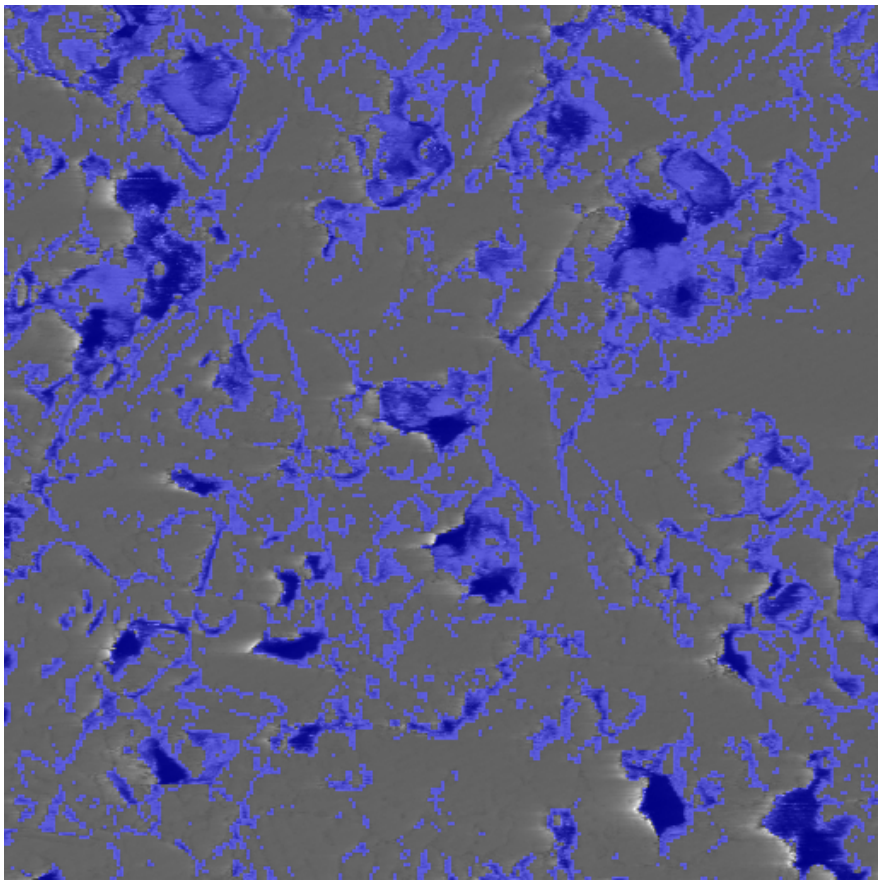

Figure 4: Final segmented image showing the result of the processing