CCE3015 - Assignment 1 – Problem Research and Planning

Graham Pellegrini

November 6, 2024

1 Question 1

The Multi-level 3D Discrete Wavelet Transform (DWT) is an effective technique in signal processing, particularly useful for data compression and denoising applications. This transform decomposes a 3D dataset into sub-bands, capturing frequency components across multiple resolutions. Each decomposition level separates the data into approximation and detail coefficients, progressively breaking down the data into low and high frequency componenets.

1.1 Problem Overview

In this problem, we will be working with the CHOAS dataset [1], which contains large 3D medical images saved as slices in DICOM (.dcm) files. Given the dataset's size and format, these slices are first pre-processed in Python to convert them into a binary (.bin) file. The preprocessing pipeline involves reading DICOM files, transforming them into NumPy arrays, and saving the arrays as binary files. This binary format will allow for efficient loading and processing in the C++ implementation of the 3D DWT.

In the preprocessing step, to achieve a 3D volume from 2D slices. The slices are stacked along the depth axis to form a 3D volume, as shown in 1above. This 3D volume is saved as a numpy array in Python and then converted to a binary file for input into the C++ program. Like this the dicom slices have been converted into their respective 3D image as a volume.

1.2 Algorithm Overview

To implement the 3D DWT in C++, we will follow these key steps:

Input/Output Handling: Two utility header files will be included, one to load the binary data into a 3D volume array and another to save the processed data back into a binary file. These functions will facilitate data handling

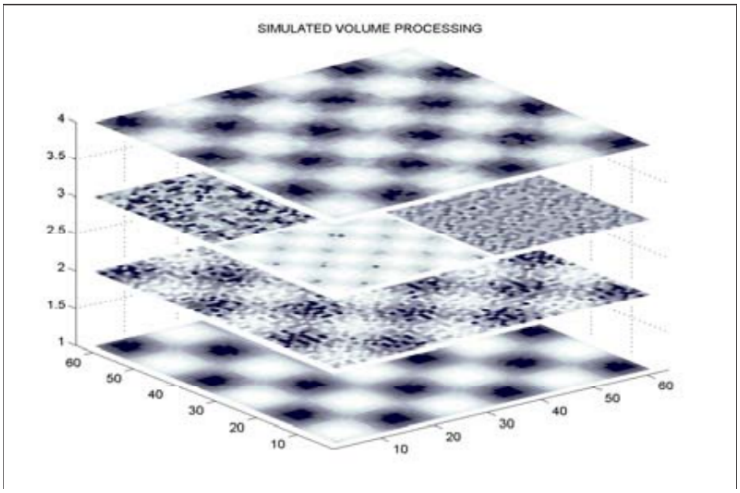

Figure 1: 3D Volume from 2D Slices [2]

between Python and C++.

Wavelet Coefficients: Daubechies wavelet filters will be used, taking from db1 till db4. These filter coefficients will directly be sampled from pywavelets, for comparison purposes.These coefficients will be hardcoded as vectors with seperation between low-pass and high-pass filters. The filters will be applied to the 3D data in the convolution step. Having the low filters and high filters seperated will allow for easy implementation of the convolution step. The low emphasise the approximation coefficients and the high emphasise the detail coefficients of the signal.

Discrete Wavelet Transform (DWT):
The pipeline shown in 2 shows how the 3D volume can be spilt into 8 subbands. However first a function to perform the 1D DWT on a signal will be defined. This function will take the signal and convolve it with the low-pass and high-pass filters. The two seperated convolved otuputs will then be downsampled by selecting every second value. The downsampling technique is there as a way to reduce the data size by half along each axis as well as a data redundancy reduction technique.

This 1D DWT function will then be called three times in the 3D DWT function, as it will be applied to each axis of the 3D volume. The 3D DWT function will take the 3D volume and apply the 1D DWT function to each axis. To keep convention the dwt is first applied to the column axis and the respective L and H subbands will be stored back to the original volume. Then the dwt is applied to the row axis and now we get the LL LH HL HH subbands. Finally the dwt is applied to the depth axis and we get the 8 subbands (LLL LLH LHL LHH

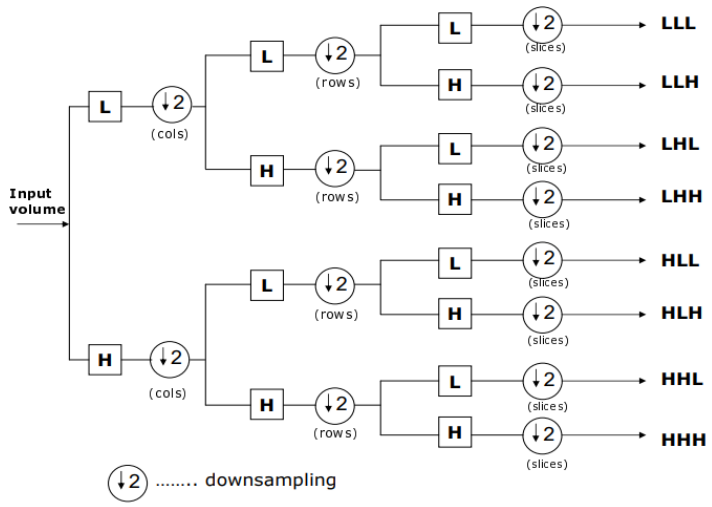

Figure 2: Discrete Wavelet Transform [2]

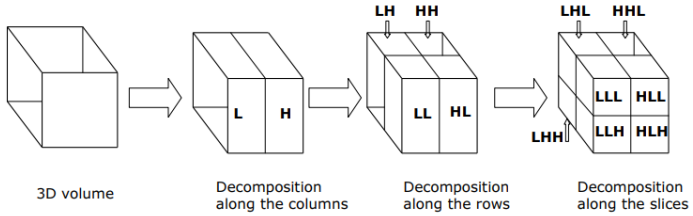

Figure 3: 3D DWT Subbands [2]

HLL HLH HHL HHH). So the order 3D DWT fucntion will include 3 nested for loops to apply the respective column-wise, row-wise and depth-wise DWT.

The importance of always saving back the L and H subbands to the original volume is so we utalise memory efficiently, especially since we are working with 3d vectors of a large dataset.

Multi-level Transform: Our implementation aims at achieving Multi-level 3D DWT. This means we can take the approximation coefficients (LLL) from the first level and apply the 3D DWT again to get the approximation coefficients of the next level. This process can be repeated for as many levels as desired. However, it is important to note that while there is no theoretical limit to the number of levels, practical constraints are imposed by the image dimensions. In our case since the slices have dimensions of 78x256x256, a 4-level decomposition results in dimensions of 10x64x64, beyond which further decomposition would

degrade image quality excessively.

A multi-level function will be defined that will call the 3D DWT function for the number of levels specified. If the level is not the final one then the volume dimensions will be halved along each axis, taking the approximation coefficients (LLL) and applying the 3D DWT function to them. In this process the higher level detail coefficients are discarded but they are not needed.

To be able to handle odd dimensions, either the dimention division must handle odd numbers or the volume must be padded with zeros to make the dimensions even.

1.3 Expected Results

Each level of the 3D DWT will yield a progressively compressed representation of the original image data, maintaining the approximation coefficients (LLL) across each axis and discarding the higher-frequency details. This approach allows us to retain essential low-frequency components, suitable for compression and noise reduction in medical imaging applications.

References

[1] Choas Medical Dataset, available at: `https://medicaldatasetlink.org`

// Note there is no limit on the number of levels that can be performed theoretically. but in practice since the image dimensions are not infinite, the number of levels that can be performed is limited by the image dimensions. In our case at the 4 level the image dimensions will be 10*64*64. If the level is increased beyond 4 the image approximation will be too degraded to be useful.

References

[1] Ali Emre Kavur, M. Alper Selver, Oğuz Dicle, Mustafa Barış, and N. Sinem Gezer. CHAOS - Combined (CT-MR) Healthy Abdominal Organ Segmentation Challenge Data, April 2019.

[2] Jan Procházka. Integration of an adaptive filtering algorithm into a low-cost smart sensor. In *Proceedings of the IASTED International Conference on Intelligent Systems and Control*, Cambridge, UK, 2011.

https://chaos.grand-challenge.org/Download/