IST 562 Programming 3

Xusheng Li, Qian Sun

September 27, 2018

Overview

This program is able to do multi-level forward and inverse 2D D4 wavelet transformation of an image. Also, it can work as an image compressor and decompressor. It treats boundary values as mirroring and can handle different input size of images, (e.g., even and odd numbers of pixels).

The code is organized into several functions:

- D4_1D(): Calculates the 1D forward D4 wavelet transformation of a 1D vector.
- D4_1D_inv(): Calculates the 1D inverse D4 wavelet transformation of a 1D vector.
- test_1D(): Given an arbitrary list of numbers, calculates the forward and inverse 1D D4 transformation, and test if the results agree with the input. Also, this program compares the results with that from an existing package, i.e., PyWavelet.
- D4_2D_one_level(): Do one level of 2D D4 forward wavelet transformation. For every row of the image, the programs uses D4_1D() to calculates its D4 wavelets. Then for every column of the resulting image, it uses the D4_1D() to calculate its D4 wavelets. The resulting image is returned.
- D4_2D(): Do several level of forward D4 to an image. It calls D4_2D_one_level() to do the job. It also maintains the shape of the image correctly.
- D4_2D_inv_one_level(): Do one level of 2D D4 inverse wavelet transformation. It is similar to the one level forward function, except it does the inverse transformation.
- D4_2D(): Do several level of inverse D4 to an image. It calls D4_2D_inv_one_level() to do the job. It also maintains the shape of the image correctly.
- compress_image(): Compress an image. It calls D4_2D() on an image, and retains only the result of the high pass information. Thus, every level of transformation compresses the image by a factor of 4 in size. Two level of compression results in a factor of 16.

- decompress_image(): Decompress an image. It calls D4_inv_2D() on an image. The information discarded in the compression process is treated as zeros.

Results

Our input image is a 512 * 512 grayscale image (Figure 1).

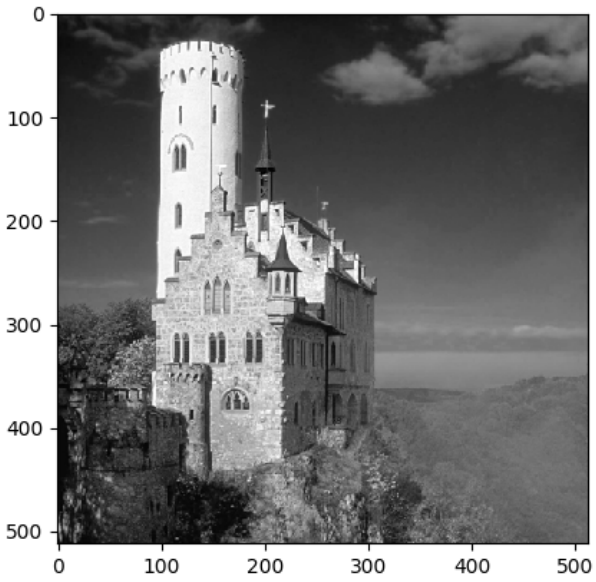

Figure 1: Input image

The result of the first level of 2D D4 wavelet transformation is Figure 2:

For the 256 * 256 image in the upper-left corner, we do the a second level of D4 transformation, and the output is Figure 3:

For the 128 * 128 image in the upper-left corner, we do the a third level of D4 transformation again, and the output is Figure 4:

The size of the image is marked with ticks. Note the 64 * 64 output is already quite blurred, so we only do 3 levels of forward transformations.

To inverse the transformation, we start from the third level and get an 128 * 128 output shown in Figure 5:

Then, we reverse the 128 * 128 image and get an output of size 256 * 256 in Figure 6.

Finally, we reverse the 256 * 256 image and get an output of size 512 * 512 in Figure 7.

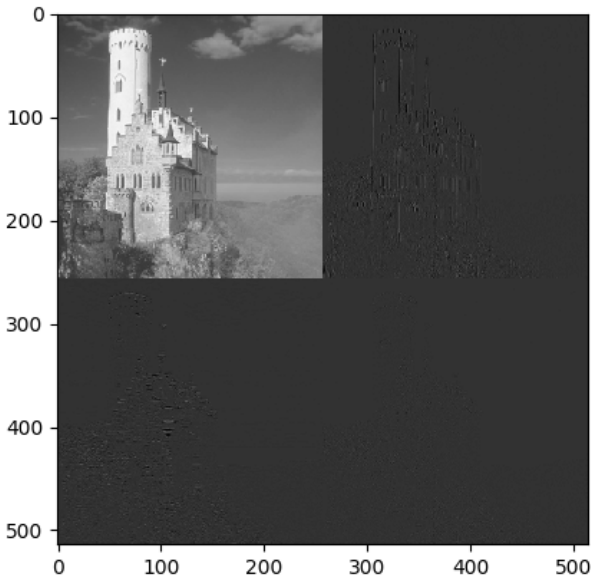

Figure 2: Forward output level 1

Note we do not discard any information in the process (the low pass information is retained and correctly passed by the program), the last resulting image is exactly the same as our very input in Figure 1.

To do an image compression, we only retain the high pass and discard the low pass information.

A one level D4 transformation results in a 4x file size reduction (theoretically), and the resulting image (shown in Figure 8) is still of good quality.

A two level of D4 transformation results in a 16x reduction in file size, and the resulting decompressed image is shown in Figure 8. Although the image is blurred to a certain degree, it is worthwhile if we want the 16x size reduction.

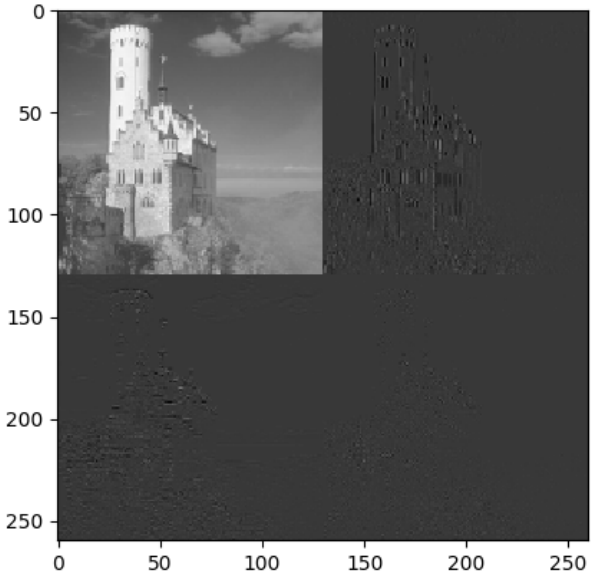

Figure 3: Forward output level 2

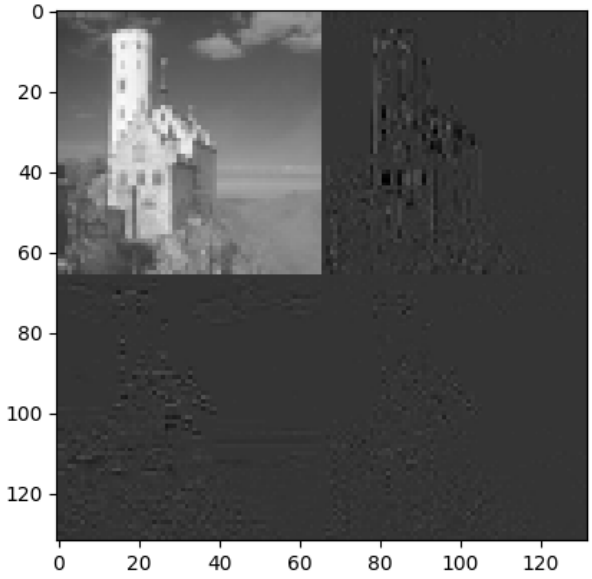

Figure 4: Forward output level 3

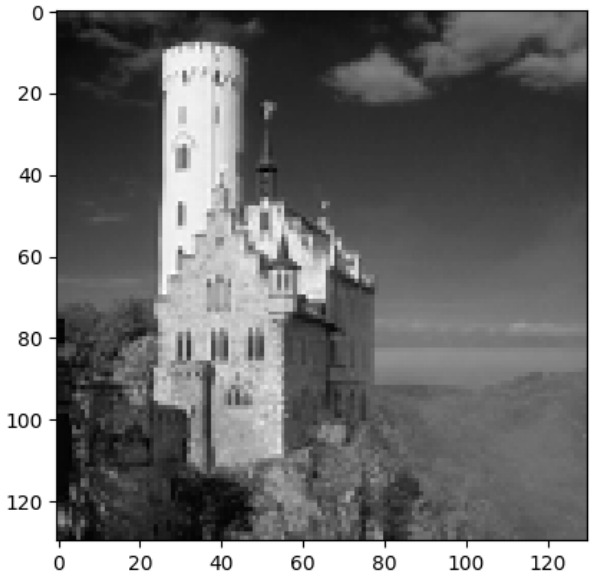

Figure 5: Inverse output level 1

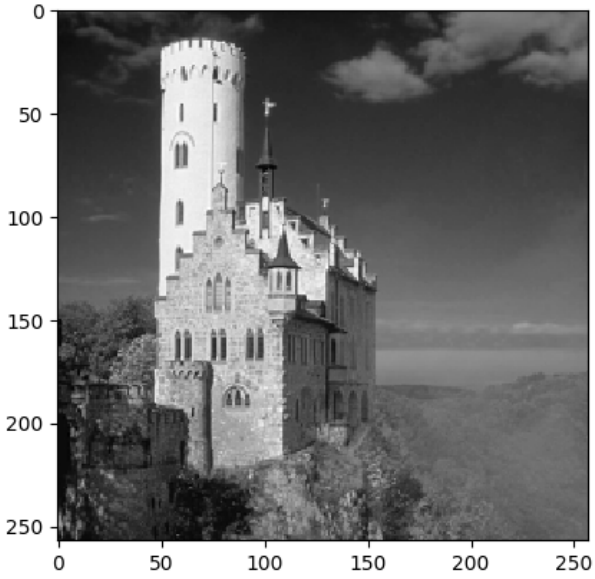

Figure 6: Inverse output level 2

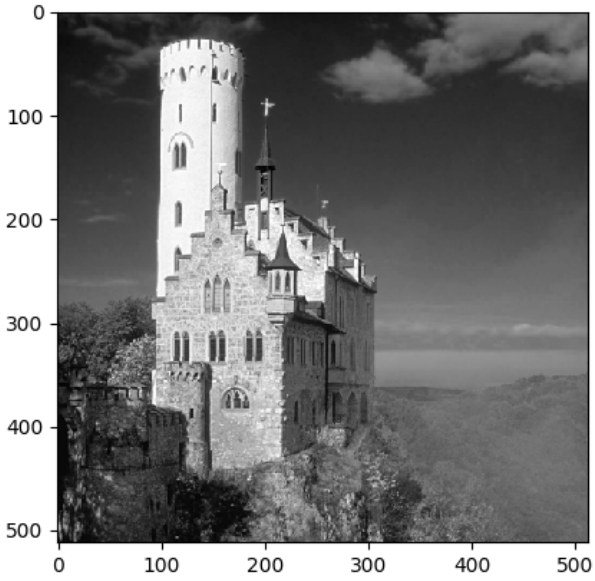

Figure 7: Inverse output level 3

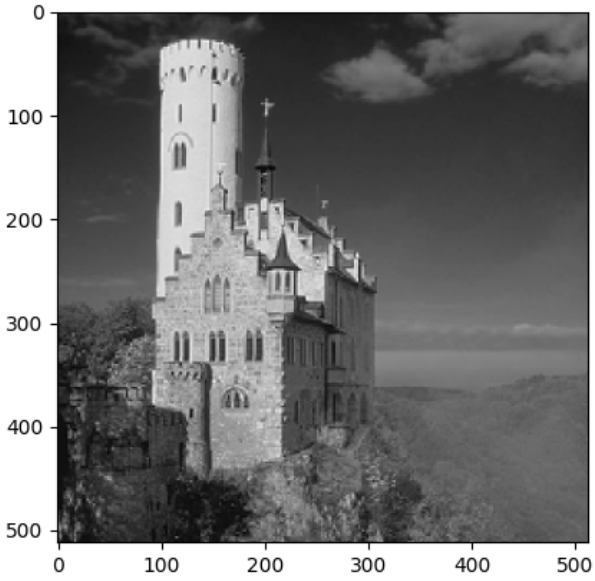

Figure 8: Decompressed image; 1 level D4 wavelets

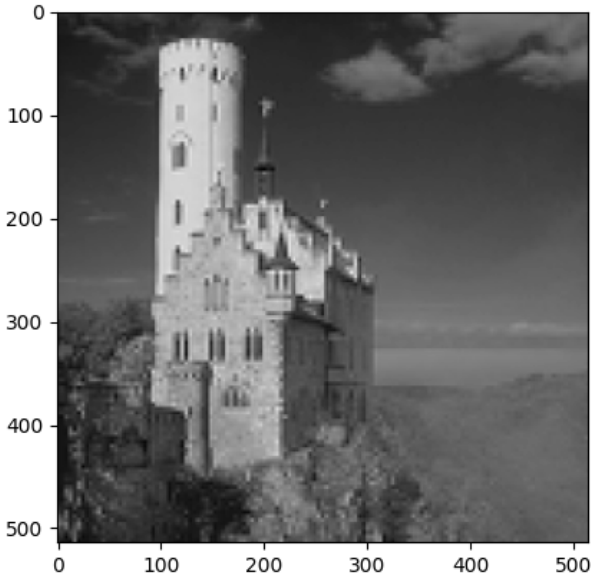

Figure 9: Decompressed image; 2 level D4 wavelets