COL783 Assignment 3

Due date: 25 October 2024

This assignment deals with image transforms and compression. As in Assignment 2, you are advised to use single-component (intensity) images for all problems. You may use smaller images in this assignment; anything with $\max(M, N) \geq 500$ is OK.

Part 1

1. In this problem, we will experiment with the cosine basis for representing images. Similar to Assignment 2 problem 3, choose an image f that has large smooth regions as well as regions with strong edges and fine detail. Make sure its length and width are multiples of 16, otherwise crop it to such a size.

   a. Implement functions to compute the discrete cosine transform and inverse discrete cosine transform of a 1D sequence of arbitrary length. Use these to perform the DCT of f in a separable manner, and verify that applying the IDCT to the resulting coefficient array t recovers the original image. If you have access to a built-in DCT, also verify that your implementation gives identical results; after that, you may use the built-in DCT for subsequent parts.

   b. Plot the histograms of f and t.

   c. Suppose we retain only the 1/16th largest (in absolute value) coefficients in t and zero out the rest. Show the resulting approximation of f, and report its MSE. In your report, prove that for any orthogonal transform, such a procedure results in an MSE equal to $1/(MN)$ times the sum of squares of the deleted coefficients, and verify this numerically for this example.

   d. Divide the image into blocks of size 16×16 and take the DCT of each block. Consider two approximation strategies: (i) in each block, keeping only the top 1/16 coefficients; (ii) keeping the top 1/16th of all coefficients across all blocks. For both, show the resulting image and the MSE.

2. The DCT is good for compression but not so useful for denoising. As we will see, wavelet transforms do a good job for the latter as well. You may use the same image f as in problem 1, or a different one.

   a. Implement the 2D discrete wavelet transform for a user-specified number of levels (not necessarily going all the way to a 1×1 image). You may restrict attention to Haar wavelets. Create a “zone plate” image similar to the one in lecture 6, and verify that you get detail coefficients corresponding to the horizontal, vertical, and diagonal features at different scales in the image.

b. Add Gaussian noise η of some chosen variance to the test image f and take the DWT. Verify that the noise in the transform domain, $\mathrm{DWT}\{f+\eta\} - \mathrm{DWT}\{f\}$, remains Gaussian with the same variance, by plotting its histogram and computing its statistics.

c. Denoising can be performed by zeroing out detail coefficients of small magnitude. For a given threshold $t > 0$, one can perform hard thresholding or soft thresholding:

$$d_{\text{hard}} = \begin{cases} 0 & -t \le d \le t \\ d & \text{otherwise,} \end{cases} \qquad d_{\text{soft}} = \begin{cases} d+t & d < -t \\ 0 & -t \le d \le t \\ d-t & d > t. \end{cases}$$

For both cases, try different values of t, plot the MSE, and show the best reconstructed image.

d. **(Optional, no extra marks)** Extend your DWT implementation to support the Daubechies db2 wavelets, recalling that all you need are the scaling function coefficients h_ϕ (given as g_0 in lecture 17 and in the textbook) and the wavelet function coefficients h_ψ (which can be obtained from h_ϕ). Repeat task (c) with the db2 wavelets and discuss the results.

Part 2

3. Now that we have a block DCT, we can implement a toy JPEG-like codec (encoder/decoder). We will not exactly match the JPEG file interchange format, but just get all the main stages of the compression process working. For each stage, also implement a decoder and verify that it can successfully decode the encoder output. Use an 8×8 DCT here instead of 16×16.

a. After computing the block DCT coefficients, the next step is to quantize them using a quantization matrix Z. Implement this stage for a user-specified Z and verify that the decoder given the same Z can reconstruct an approximation of the original coefficients. Test this process using four different choices of Z: two different multiples of the JPEG quantization matrix that give similar visual quality as the examples in lecture 19 (slide 18), and two constant matrices that give a similar number of zeroed coefficients. In each case, show the reconstructed image after block IDCT, report the MSE and the percentage of zeroed coefficients, and comment on whether the MSE correlates with the subjective visual quality. In the remaining parts of the assignment, use just the first choice of Z, i.e. the medium-quality JPEG quantization matrix.

b. Perform zigzag ordering of the quantized DCT coefficients in each block. JPEG uses RLE only for the zeroes in the AC coefficients, i.e. the coefficients are stored as follows: value of DC coeff, length of run of zeroes (can be 0), value of next nonzero AC coeff, …, length of run of zeroes, value of next nonzero AC coeff, “end of block” symbol for last run of zeroes. For example, the block on slide 16 would be stored as [−26, 0, −3, 0, 1, …, 0, 2, 5, −1, 0, −1, EOB]. Implement this process and its inverse. Pick an interesting block of your image, show its original intensities as an 8×8 image, and report the quantized coefficients before and after this process. Also report the compression ratio of the entire image, in terms of the total number of symbols before and after.

c. Implement a Huffman encoder that takes a set of symbols and their corresponding probabilities, and builds a code table that associates each symbol with a list of bits. Also implement a Huffman decoder that takes the code table and a list of bits, and returns the first decoded symbol and the list of remaining undecoded bits. (This may be more efficient to implement via indices into a fixed list rather than always taking sublists). Build a single Huffman code for all the symbols coming out of the zigzag+RLE stage, and use it to encode the entire image as a list of bits. Report the entropy of the symbols’ probability distribution and compare it with the average length of your Huffman code. (*P.S:* JPEG actually uses separate Huffman codes for the DC coefficient differences, the nonzero AC coefficients, and the run lengths. If you want, you can implement this slightly more complex process.)

d. Now, instead of storing the DC coefficient of each block, only store the difference with the DC coefficient of the previous block. Build a new Huffman code, and again report the entropy and the average code length.

e. Implement an encoder that puts all the stages together, taking an image f of size $M \times N$ and a quantization table Z as input, and producing a tuple containing the Huffman codes H and the list of encoded bits $\mathbf{b}$. Also implement a decoder that takes M, N, H, Z, and $\mathbf{b}$ as input and reconstruct an approximate image $\hat{f}$. If you were verifying the decoders and encoders in each previous part, this should just be a matter of chaining them together. Report the final compression ratio as the number of bits in uncompressed image, $8MN$, divided by the number of bits in $\mathbf{b}$. (This is an overly optimistic estimate, do you see why?)

Submission

Submit a zip file that contains (i) all your code for the assignment, and (ii) a PDF report that includes your results for each of the assignment problems. The report must include the name of all teammates who worked on the project. Each output image in the report must be accompanied with a brief description of the procedure used to create it, and the values of any relevant parameters.

All images should ideally be saved in PNG format, which is lossless and so does not cause any information loss (other than quantization if the intensities are not already in 8-bit format). JPEG is permitted if your submission PDF is becoming too big for the upload limit.

Your assignment should be submitted on Moodle before midnight on the due date. Only one person in a group needs to submit. Late days are counted with a quantization of 0.5 days: if you cannot finish the assignment by midnight, get some sleep and submit by noon the following day.

Separately, each of you will individually submit a short response in a separate form regarding how much you and your partner contributed to the work in this assignment.