RRY025-- Image Processing

Exercises 8 – Lecture 10 – Compression II

EX 1. Data types during wavelet compression.

One general issue with transform decoders is that transforms are usually composed of decimal numbers instead of integers between 1-256. For example, Fourier transform results in a complex number and Wavelet transform in decimal coefficients.

Explore by experimenting what are the pros and cons of pre-compressing the cameraman using wavelet transform ('Haar', lvl 5) and then treating the coefficients as short/long integers (*uint8()* and *uint16()*) instead of decimal numbers. In particular, try to find out: Can you increase the compression in comparison to using the original decimal number coefficients, while still retaining acceptable image quality? Note that once you have integer numbers in hand, you can also use Huffman coding…

ANSWER: Using both uint8() and uint16() leads to a greatly decreased quality of the reconstruction. This is the case regardless of the amount of pre-compression.

The conclusion here is that it seems necessary to treat the wavelet transform as decimal numbers – using integers does not give enough accuracy.

Reconstructed image after precompression + rounding to uint16() integers:

(Unfortunately, the last note about Huffman coding was a bit misleading. Since the quality of the image is bad when it is transformed to integers, it is not really sensible to code it further in any other way…)

EX 2. Predictive compression

Implement in Matlab predictive coding on cameraman using the second order linear (=gradient based) predictor (cf., Lecture notes). Does this predictor work well in the case of cameraman? (The key here is to think about what "working well" means…)

ANSWER: One example solution given as a .mlx file. One can consider that the predictor "works well" if the error is highly compressible, i.e., has low entropy. The entropy is about 1.0 bits/px , which is smaller than the entropy of the original (~7 bits/px) and smaller than the error using a first order linear predictor (~2 bits/px). Thus, it seems that the second order predictor works very well in this case.