RRY025-- Image Processing

Exercises 6 – Lecture 6 – Wavelet II

EX 1: Wavelet background in Matlab

Read the Matlab help for choosing a wavelet:

https://se.mathworks.com/help/wavelet/gs/choose-a-wavelet.html

(from 'Multiresolution Analysis' downwards)

EX 2. Noise removal using Wavelet Analyzer

Experiment how removing noise works using the Wavelet Analyzer.

First, impose different noise patterns on an image (suggestion: cameraman).

1. A Gaussian pixel-to-pixel noise (recommendation: cameraman). Note: Gaussian refers to the probability of the noise value within an individual pixel.
2. The same as 1, but with a larger spatial scale. The spatial scale is an input parameter given in pixel units. Hint: If you want to create noise at the scale *x*, create an empty image that is of size 1/x compared to the original, create the pixel-to-pixel noise there, and resize the noise image to the original size (see *imresize()*). This way, you have created random fluctuations at scale *x*. Note that having x a multiple of 2 is a good idea for an even-sized image…
3. A systematic wavy pattern that is imposed on the image. Create the wavy pattern using a trigonometric function.

Experiment with the Wavelet Analyzer to denoise the image in the case of the above noise patterns. Try with both the 'compression' utility and the 'denoising' utility. Try also with different wavelet families.

Control Questions:

Can you understand the basic operations in the compression and denoising functions of the Wavelet Analyzer?

What approach gives you the most pleasing result? Can you describe why that approach works the best? Can you relate some of the properties you seen in the compressed images with the properties of the wavelet you use? (There are some clear cases, but some are difficult to disentangle...)

ANSWER: The type 1 noise appears dominantly in lvl 1 decomposition (perhaps lvl 2). Using level-wise thresholding at these levels leads to a good noise reduction.

The type 2 noise should appear dominantly in the level that corresponds to the size scale of the noise that was created. Thresholding at these levels leads to a reasonable noise reduction.

It is possible to lessen the type 3 noise by thresholding at the scale that corresponds to half the wavelength of the wave. However, this is quite imperfect, because the wave gets confused with the real features in the image.

Can you “break” the image using the noise 2? In other words, can you always remove this kind of noise adequately with Wavelets?

ANSWER: When the noise has the scale similar to the details/features in the image, denoising starts smoothing over those details/features. Also, when the noise has a large spatial scale, the square-like Haar wavelet does a poor job in removing it (because the noise has a rounder appearance in the image), leaving box-like artifacts in the image.

Can you describe (and try if you have time) other fruitful approaches in case of noise 3?

ANSWER: Since noise 3 is a wave, it would seem possible to remove it efficiently using Fourier filtering. One could identify the fourier component corresponding to the wave and filter it out from the Fourier transform.

EX 3. Pre-compression using Matlab functions (not Wavelet Analyzer)

Find out via experiments how much a photograph can be pre-compressed by nullifying wavelet coefficients. In other words, find out what fraction of the data in the wavelet transform you can nullify, before the quality of the image becomes unacceptably bad.

This exercise is similar to the classroom demo, only now you have to write a Matlab script – and learn use of *dwt2()* etc. – while getting it done.

One possible roadmap:

Perform wavelet transform with Haar down to fourth level.

At each level, nullify *x* % of the smallest coefficients in each detail image.

Reconstruct the image using the reduced amount of coefficients.

Compare the reconstructed image with the initial one. Vary *x* and decide, by eye, when the quality differs too much from the original.

Finally, repeat the experiment for a modern jpg image (e.g., Mikki1_crop_close.jpg). The result is likely a bit different from one you got for the cameraman above. Can you figure out why that is the case?

ANSWER: Example given as .mlx file.

If using cameraman, at around 85% of zeros the image starts to get boxy artifacts. Until that the noise reduction works pretty well. At larger values, the quality quickly degrades.

In case of Mikki, one starts to loose quality at around 60% of zeros. This is because the Mikki image has clearly higher amount of detail than the cameraman; reducing the information in the detail coefficients impacts the appearance of the image more.