Task 4

Localization Properties and 2-D Denoising

Christian Riess (christian.riess@fau.de)

Haar wavelets are particularly useful for analysis and processing of piecewise constant signals. However, many applications require to process smooth signals, for which a number of different wavelets with varying properties have been developed. For example, we discussed the Daubechies wavelets in the lecture.
In this exercise, we extend our 2-D multiresolution analysis to (almost) arbitrary wavelets. As an application, we look into image denoising using different wavelets.

1 2-D Multiresolution Analysis

Extend the implementation of 2-D Haar-based multiresolution analysis, such that other wavelet (high pass and low pass) filters can be used for analysis and synthesis.

1. The high pass and low pass filters of choice are now passed as arguments to the functions `mra2` and `mrs2` for analysis and synthesis.
2. In each iteration of analysis and synthesis, the respective filters are passed on to the functions `mydwt2` and `myidwt2`.
3. The length of filters may vary. However, you may assume that the high pass and low pass filters have identical length L, and that L is an even number. Hence, the periodic supplement of the boundary regions has to have length $L - 1$.
4. For testing your implementation, you can use the matlab function `wfilters` to obtain filter coefficients for one particular wavelet, such as `'haar'` or `'db2'`.

2 2-D Denoising

Typically, image acquisition and sensor readout introduce noise to the final image. We assume that this noise is additive and normally distributed. Thus, let $f \in \mathbb{R}^{N \times M}$ be the ideal noise-free image, and $n \in \mathbb{R}^{N \times M}$ the normally distributed noise per pixel with standard deviation σ. Then, the observed image $x \in \mathbb{R}^{N \times M}$ is assumed to be

$$x = f + n \quad . \tag{1}$$

Use this model to add noise to synthetically created images in matlab.

Use wavelet thresholding for denoising the images, and discuss your results. Follow the steps listed below:

1. The threshold can be determined from the standard deviation σ of the noise. However, σ is typically not known, and hence has to be estimated. Implement a function `estimateNoise` that estimates σ on w_1^{HH}, denoting the detail coefficients from the first level of decomposition. The estimate is calculated as

$$\sigma = \frac{\text{median}(\left|w_1^{HH}\right|)}{0.6745}. \tag{2}$$

2. Implement in the function `denoise_UT` the denoising algorithm. Within this method, estimate the wavelet coefficient threshold (called *Universal Threshold*) from σ by calculating

$$\tau^{\text{univ}} = \sigma \cdot \sqrt{2 \log_e(N \cdot M)}. \tag{3}$$

   Additionally, the user shall be able to multiplicatively increase or decrease the denoising threshold. This multiplicative factor is passed as parameter `thrweight`.

3. Thresholding can be performed as *hard thresholding* and *soft thresholding*. *Hard thresholding* performs the operation

$$y_{\text{hard}}(t) = \begin{cases} x(t), & |x(t)| > \tau, \\ 0, & |x(t)| \leq \tau, \end{cases} , \tag{4}$$

   and *soft thresholding* performs the operation

$$y_{\text{soft}}(t) = \begin{cases} \text{sign}(x(t)) \cdot (|x(t)| - \tau), & |x(t)| > \tau, \\ 0, & |x(t)| \leq \tau, \end{cases} . \tag{5}$$

   Implement both methods and visually compare the results. What do you observe?

4. Extend the method `denoise_UT` for the possibility to compute the Universal Threshold for each level of decomposition individually, using

$$\tau_l^{\text{univ}} = \sigma \cdot \sqrt{2 \log_e(N/2^l \cdot M/2^l)}, \quad 1 \leq l \leq \text{maxlevel} \ . \tag{6}$$

5. Another approach to denoising is the so-called *Interscale Correlation*. Implement this approach in the function `denoise_IC`. This method computes the thresholds by jointly using levels l and $l+1$ for computing the threshold of level l.

   First, use `interp2` to interpolate the wavelet coefficients of level $l+1$ onto the finer grid of level l. Then, convolve the interpolated coefficients with a 2×2 box filter, and denote the resulting coefficients as m_{l+1}.

   Now compute the product with the coefficients at level l, dented as w_l, as

$$n_l = \sqrt{2} m_{l+1} \cdot w_l \ , \tag{7}$$

   Apply the level-adaptive threshold of Eqn. 6 in a slightly modified form: instead of σ, use σ^2. Other than that, the analogous thresholding is applied.

6. Compute the mean squared error (MSE) and the peak signal to noise ratio (PSNR) between the ideal noise-free image, the noisy image, and your denoising results. PSNR is a commonly used metric for denoising, and is defined as

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{I_{max}^2}{\mathrm{MSE}}\right) \ , \tag{8}$$

where I_{max} is the maximum possible intensity of the image (i.e., 255 for 8-bit images). Experiment with the denoising methods. Use different wavelets and test images. What do you observe? Which wavelets are particularly good for which images?