Problem Set 02

1. **(Variations of Histogram Equalization)** The book [R. Klette and P. Zamperoni: *Handbook of Image Processing Operators*. Wiley, Chichester, 1996] discusses variations of histogram transforms, in particular variations of histogram equalization

$$g_{equal}^{(r)}(u) = \frac{G_{\max}}{Q} \sum_{w=0}^{u} h_I(w)^r \quad \text{with } Q = \sum_{w=0}^{G_{\max}} h_I(w)^r$$

Use noisy (scalar) input pictures (of your choice) and apply the sigma filter prior to histogram equalization. Verify by your own the following statements:

a) A stronger or weaker equalization can be obtained by adjusting the exponent $r \geq 0$;
b) The resultant histogram is uniformly (as good as possible) distributed for $r = 1$;
c) For $r > 1$, sparse grey values of the original picture will occur more often than in the equalized picture;
d) For $r = 0$, we have about the identical transform;
e) A weaker equalization in comparison to $r = 1$ is obtained for $r < 1$.

Visualize results by using 2D histograms where one axis is defined by *r* and the other axis, as usual, by grey levels. Show those 2D histograms either by means of a 2D grey-level image or as a 3D surface plot.

2. **(Developing an Edge Detector by Combining Different Strategies)** Within an edge detector we can apply one or several of the following strategies:

a) An edge pixel should define a local maximum when applying an operator (such as the Sobel operator) that approximates the magnitude of the gradient ∇I;
b) After applying the LoG filter, the resulting arrays of positive and negative values need to be analyzed with respect to zero-crossings (i.e. pixel locations *p* where the LoG result is about zero, and there are both positive and negative LoG values at locations adjacent to *p*);
c) The discussed operators are modeled with respect to derivatives in *x*- or *y*- directions only. The consideration of directional derivatives is a further option; for example, derivatives in directions of multiple of 45°;
d) More heuristics can be applied for edge detection: an edge pixel should be adjacent to other edge pixels;
e) Finally, when having a sequence of edge pixels, then we are interested in extracting “thin arcs” rather than having “thick edges”.

The task in this programming exercise is to design your own edge detector that combines at least two different strategies as listed above. For example, verify the presence of edge pixels by tests using both first-order and second-order derivatives. As a second example, apply a first-order derivative operator together with a test for adjacent edge pixels. As a third example, extend a first-order derivative operator by directional derivatives in more than just two directions. Go for one of those three examples or design your own combination of strategies.

3. **(Amplitudes and Phases of Local Fourier Transforms)** Define two $(2k + 1) \times (2k + 1)$ local operators, one for amplitudes and one for phases, mapping an input image I into the *amplitude image* M and *phase image* P as follows.

Perform the 2D DFT on the current $(2k + 1) \times (2k + 1)$ input window, centered at pixel location p.

Visualize M and $P(p)$ as grey-level images and compare with edges in the input image I. For doing so, select an edge operator, thresholds for edge map, amplitude image, and phase image and quantify the numbers of pixels being in the thresholded edge and amplitude image versus numbers of pixels being in the thresholded edge and phase image.