Assignment 2

Submission Deadline: 21.05.25, 11:00 pm

Topics:

- Image filtering with Gradient of Gaussian (GoG)
- Interest point operator (Förstner)

You are free to use the provided image (*ampelmaennchen.png*) or own photos.
Always use **grayscale images** and scale your input images to be represented in **double-precision floating-point** format with values in the **range** $[0.0,1.0]$.

Tasks:

A) **Gradient of Gaussian (GoG) filtering**

a. Compute continuous GoG-filter kernels for convolution in x- and y-direction.

Example: for standard deviation $\sigma = 0.5$ the two 2D kernels are

$$G_x = \begin{bmatrix} 0.0000 & 0.0001 & 0.0000 & -0.0001 & -0.0000 \\ 0.0002 & 0.0466 & 0.0000 & -0.0466 & -0.0002 \\ 0.0017 & 0.3446 & 0.0000 & -0.3446 & -0.0017 \\ 0.0002 & 0.0466 & 0.0000 & -0.0466 & -0.0002 \\ 0.0000 & 0.0001 & 0.0000 & -0.0001 & -0.0000 \end{bmatrix}, G_y = G_x^T$$

b. Apply these filters to your input image I to derive two **gradient images**: I_x and I_y (one in x- and one in y-direction). Write a function for the convolution of the image with the kernel and ignore the boundaries of the image for simplicity, i.e. no padding needed (you may use built-in convolution functions such as `convolve in` NumPy or `convolve2d` in SciPy).

c. Compute and visualize the **gradient magnitude** image G

$$G = \sqrt{{I_x}^2 + {I_y}^2}$$

B) **Förstner interest operator:**

Use the gradient images to identify Förstner interest points in your input image.

a. Compute the autocorrelation matrix M for each pixel using a moving window w of 5×5 pixels. Perform convolution based on this window to include the local neighborhood around each pixel (use I_x, I_y and ignore the boundaries of the images).

$$M = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

b. Compute the **cornerness** w and **roundness** q from M for each pixel and store the values in two matrices W and Q. Plot the values in W and Q with an appropriate colormap (`imshow, cmap='jet'`).

c. Derive a binary mask of potential interest points by simultaneously applying the thresholds $t_w = 0.004$ and $t_q = 0.5$ on W and Q, respectively.

d. Plot an overlay of the initial input image with the detected points (`plt.plot`).

Computer Vision in Engineering – Prof. Dr. Rodehorst
M.Sc. Mariya Kaisheva
mariya.kaisheva@uni-weimar.de