COM-SGN.110 Introduction to Image and Video Processing

EXERCISE 10

29.11.2023 – 01.12.2023

The tasks should be completed and presented to TA during the lab session. **Do not forget to upload your solutions to Moodle!** Questions about exercises should be addressed to the TA personally, through Moodle messages or via email, which can be found on the Moodle page of the course.

1. Color distances and slicing

Implement 2 MATLAB functions `sliceCube` and `sliceSphere`, which allow the user to choose a color from an image and suppress all the colors that are sufficiently different from the chosen one.

a) The functions should take an image I and a distance $dist$ as inputs ($dist \geq 0$).
b) Use `impixel` to allow the user to choose a color C. Note that many pixels can be chosen with one call to `impixel`, in which case it will return a matrix instead of an RGB vector. This case can be handled in different ways, e.g. use the first chosen pixel, or the last one, or average the colors.
c) Construct a distance-based image mask. `sliceCube` should use Manhattan distance, while `sliceSphere` uses Euclidean distance:

$$Cmask = \begin{cases} 1 & \text{where } |I_R - C_R| + |I_G - C_G| + |I_B - C_B| \leq dist \\ 0 & \text{otherwise} \end{cases}$$

$$Smask = \begin{cases} 1 & \text{where } (I_R - C_R)^2 + (I_G - C_G)^2 + (I_B - C_B)^2 \leq dist^2 \\ 0 & \text{otherwise} \end{cases}$$

NB: the variable types are of major importance here, as the range of `uint8` cannot fit the squared values from the above expressions. Perform calculations in `double`.

d) Use the masks to suppress dissimilar colors. Suppressed regions can be either left black or get assigned a certain color, e.g. gray (127, 127, 127).

Verify the functions on the images *cheetah.jpg* and *chameleon.jpg*. You can try your own images as well.

2. Noise and filtering across different colorspaces

a) Load the RGB image *lena.tiff* and convert it to HSI color space with the provided function *rgb2hsi.m*. Display every component of both color spaces in a 2x3 subplot (R, G, B, H, S, I).
b) Introduce Gaussian noise (`imnoise`) to the green component alone. Create a new copy of the image with noisy green channel (other channels intact) and display it. Convert the copy to HSI colorspace and display all the components in a subplot. Which of them are affected by the noise and which (if any) are not? Explain the reason.
c) Repeat step b), but now add Gaussian noise to all three color components. Are there any differences between the results of this step and step b)? Why/why not?
d) Try applying arithmetic mean filtering (`imfilter, fspecial`) to remove the noise and use the provided function *hsi2rgb.m* to convert the HSI image back to the RGB space. Which component(s) should be filtered for best results? Which will damage the image further when filtered? Demonstrate your conclusions for both RGB and HSI colorspaces.