Step-by-step instructions for hands on lab

1 Problem Statement

Read digital images, and understand image processing using Matlab and USB camera device.

2 Tasks

- Read an image: **A_read_img.m**
  - Red/Green/Blue channels, and Gray-scaled image
  - Image thresholding
- Reduce noise in an image: **B_noise_reduction.m**
  - Compare noise reduction filters between Blur(linear) and Median(non-linear)
- Apply linear filters to images: **C_linear_filter.m**
  - Identity, Blur(average), Gaussian Blur(weighted average), Outline, and Sharpen filters
  - Edge operators (prewitt, sobel)
- Understand a procedure of traditional edge detection: **D_edge_detection.m**
  - Smoothing, Differentiation, Non-Maximal Suppression, and Thresholding
- Understand a procedure of traditional shape detection: **E_circle_detection.m**
  - Edge Detection, Hough Transform, Voting (Collection), and Thresholding
- Retrieve your own image from real world: **F_test_camera.m**
  - Take a picture using USB camera, and perform your preference among A to E.

3 Notes

Steps to perform each task:

- Double-click a source file to launch Matlab. (If Matlab is already running, you can skip this step.)
- Change your current folder by executing the following command in Command Window pane.

  >> **cd** D:\GirlsCamp\Day1\srcs

- Prepare for images at *Images* folder
- Double-click a Matlab code in Current folder pane, which you want to launch.
- Click Run button & Press *enter* key to continue
- You can stop program execution by pressing *Ctrl* and C keys simultaneously in Command Window pane.

3.1 A_read_img.m

Try to change *threshold* variable ranging from 0 to 255 for figuring out the impact of thresholding.

3.2 B_noise_reduction.m

Try to change *filter_size(default: 3)* variable, which indicates the number of rows and columns in blur and median filters. The larger filter size you want to perform, the more time you have to wait for completing operations.

3.3 C_linear_filter

.m Try to assign a preferred filter to *filter* variable. There are 11 pre-defined filters: *identity, average, average2, weighted_average, weighted_average2, outline, sharpen, prewitt_vertical, prewitt_horizontal, sobel_vertical*, and *sobel_horizontal*. You can also create your own filter. Please ask your mentor how to write a matrix filter.

3.4 D_edge_detection.m

Try to change *sigma(default: 1.05), threshold_alpha(default: 1.7),* and *threshold_beta(default: 0.5)*. *sigma* determines how much smoothing you are going to perform before starting gradient differentiation. *threshold_alpha* sets the lowest valid magnitude of edge pixels in an image, and *threshold_beta* restricts the highest negligible magnitude of edge pixels in an image. Between *threshold_alpha* and *threshold_beta*, the intermediate pixels are valid only if they are connected to edge pixels.

3.5 E_circle_detection.m

Try to change *sigma(default: 2.5), min_radius(default: 15), max_radius(default: 100),* and *alpha(default: 0.9)*. Like Section 3.4, *sigma* adjusts the impact of smoothing to edge detection. *min_radius* and *max_radius* deal with the range of circle's radius which you would like to find. *alpha* will determine how much pixels of a circle have to be caught in voting. Currently, *alpha* is set to 90%.

3.6 F_test_camera.m

This code will take a picture by connecting to USB camera equipped with your computer. When pressing a *enter* key, the taken image will be stored as *original.jpg* at *Day1/imgs* folder. You can copy and paste program codes of Sections 3.1 to 3.5 here to reproduce image processing with your own image. If it takes time for you to complete this practice, just execute your preferred code among Sections 3.1 to 3.5. You can get the result immediately!