ISS Project 2009 / 10

Tomáš Mikolov, Jiří Kopecký and Honza Černocký, DCGM FIT BUT

Goal and submission

The aim of the project is to make use of some fundamental functions for digital image processing. As the input use the individually assigned image `xlogin00.bmp`, where "xlogin00" is your login. The project can be solved in Matlab, C or any other programming or scripting language.

Project submission will be done by means of the information system WIS as a single zip-file names as `xlogin00.zip`, where "xlogin00" is your login. The archive should contain:

- text file `reseni.txt`, that containes numerical results. The file should not contain any header, only rows with results (no empty rows nor comments).
- files `*.bmp`, containing the output images. The output images have to be in the same format as the input image – uncompressed BMP 512x512 with 8-bit color depth (will be automatically checked).
- directory `src/`, containing the sourse code of the solution. The project is comprehended as an **individual work**, thus the sourse code will be compared among students.

Please pay attention to the format of the submitted work. In case the results cannot be processed automatically but should be processed by hand, penalty will be applied (-2 points).

Task

Image unblur [1 point]

Load the input image (`xlogin00.bmp`) and apply unbluring by using a linear filter:

$$\mathbf{H} = \begin{bmatrix} -0.5 & -0.5 & -0.5 \\ -0.5 & 5.0 & -0.5 \\ -0.5 & -0.5 & -0.5 \end{bmatrix}$$

The result must be an output file `step1.bmp`

Rotating of an image [2 points]

Perform a vertical flip of the unblured image from the previous step.

The result must be an output file `step2.bmp`.

Median filter [1 point]

On the input image, apply a median filter with a window size of 5 pixels (matrix 5x5). Median filter is comonly used for noise reduction in images. It sorts the values fitting the window and outputs the median value of the sorted sequence (number 13. in our case). If using Matlab, you can use a built-in function `medfilt2`.

The result must be an output file `step3.bmp`

Image blur [2 points]

Here, use the following filter:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 3 & 3 & 1 \\ 1 & 3 & 9 & 3 & 1 \\ 1 & 3 & 3 & 3 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} /49$$

The result must be an output file `step4.bmp`

Error in the image [1 point]

In the previous steps the image was gradually modified several times. Compute the average error per pixel comparing to the original image (`xlogin00.bmp` vs. `step4.bmp`). **Compare only images with the same rotation**. Note: prior to error computing you must convert image format `unit8` to the standard Matlab format `double`.

The result must be submited as a row
`chyba=xx.yyy`
in the file `reseni.txt`. xx.yyy is the computed error.

Histogram correction [2 points]

Vizualize the histogram of the input image. As you can see, there are not all values used – expand the diagram so it contained the values from the range 0-255. Hint: find the minimum to maximum values in the original image and consequently map linearly this interval into the interval of the range 0-255. Generate an image with the mapped values. Here again you have to convert image format `unit8` to the standard Matlab format `double`, the resulting image convert back to `unit8`. Note: the task is to treach the histogram, not equalize it (!) thus function `histeq` cannot be used.

The result must be an output file `step5.bmp`

Mean value and standard deviation [2 points]

Compute the mean value and standard deviation of the image before and after histogram streching. Again dont forget to convert the image format to `double`.

The results must be submited as a rows
`mean_no_hist=xx.yyy`
`std_no_hist=xx.yyy`
`mean_hist=xx.yyy`
`std_hist=xx.yyy`
in the file `reseni.txt`. xx.yyy are the computed values. **please use decimal dots, not commas !**

Image quantization [2 points]

Perform quantization of the image obtained after histogram stretching. Consider quantization on 2 bits – the resulting image should contain 4 colors. Again dont forget to convert the image format to `double`. If using the quantization function from the lab "Sampling, quantization", correct an error in the 5. example – use rounding, not flooring:

```
round(((2^N)-1)*(double(I5)-a)/(b-a))*(b-a)/((2^N)-1) + a;
```

The result must be an output file `step6.bmp`