School of Computer Science and Applied Mathematics
UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

APPM4058A&COMS7238A: Digital Image Processing
Exercise 2

2019-2-13

1 Objectives

- Perform gray level transformation for an image
- Perform histogram processing including histogram equalization, histogram matching, and adaptive histogram equalization.

2 Problems

1. Explain why the discrete histogram equalization technique does not in general yield a flat histogram.
2. Take each of the examples in Table 1, with 16 pixels and 3 bit dynamic range monochrome values, and apply a contrast stretch and histogram equalization modification. Comment on the results. What would you say are the key differences between a contrast stretch and a histogram equalization?

0	1	1	1
0	2	1	6
1	1	6	5
6	7	7	7

7	6	5	4
6	7	5	5
5	5	6	7
4	5	7	6

2	3	2	3
3	2	3	5
3	6	5	6
6	5	6	5

Table 1: Examples

3. (a) What effect would setting to zero the lower-order bit planes have on the histogram of an image in general?
   (b) What would be the effect on the histogram if we set to zero the higher-order bit planes instead?
4. Suppose that a digital image is subjected to histogram equalization. Show that a second pass of histogram equalization will produce exactly the same result as the first pass.
5. Given a 3-bit 8×8 image with its histogram shown in Table 2, perform a histogram matching using the desired histogram shown in Table 3.

r_k	0	1	2	3	4	5	6	7
n_k	8	10	10	2	12	14	6	2

Table 2: Original histogram

z_k	0	1	2	3	4	5	6	7
n_k	0	0	0	0	20	20	16	8

Table 3: Desired histogram

6. A 3-bit image `T` is given below. Perform an adaptive histogram equalization on the pixels of `T` (excluding the border pixels) using a 3×3 tile.

   `T =`

   ```
   3   3   3   3   3   3
   3   3   4   2   3   3
   3   3   3   3   3   3
   3   4   3   2   3   3
   4   3   3   3   3   3
   3   3   4   3   3   3
   ```

3 Lab exercise

1. Implement a function `img_out = histeq_DIP(img_in)` which performs histogram equalization on `img_in`, where `img_out` is the output image. Test your implementation on image 'fractured_spine.tif'. For this exercise, you should attempt to put the results in a very short report that includes the original image, a plot of its histogram, a plot of the histogram-equalization transformation function, the enhanced image, and a plot of its histogram. Use this information to explain why the resulting image was enhanced as it was.
2. (a) Implement a histogram matching function `img_out = histogram_matching_DIP(img_in, p)`, where `img_in` is an input image, `p` is a specified histogram, and `img_out` is the output image.
   (b) Apply your implementation to image 'aerial_washedout.png' using the histograms of images 'pollen_1_he.png', and 'mars_moon.png'.
   (c) Apply your implementation to image 'mars_moon.png' using the histograms of image 'pollen_1_he.png'.
3. * Implement an adaptive histogram equalization function. To do this, you need to divide an image into small blocks (or tiles). Then each of these blocks are histogram equalized as usual. In adaptive histogram equalization, how would you deal with the issues caused by the following? Test your program for images 'mars_moon.png' and 'car.png'.
   (a) The resulting image might have blocking effects, i.e., the borders of the tiles are clearly visible.?
   (b) Noise or outlier pixels.
4. Perform adaptive histogram equalization for images 'mars_moon.png' and 'car.png' using matlab `adapthisteq` or python scikit-image package. (See the example code `plot_equalize.py`).

4 Further readings

- Adaptive histogram equalization [Pizer et al., 1987]

References

Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., Romeny, B. T. H., and Zimmerman, J. B. (1987). Adaptive histogram equalization and its variations. *Comput. Vision Graph. Image Process.*, 39(3):355–368.