EE6310 Image and Video Processing, Spring 2024

Indian Institute of Technology Hyderabad
Homework 1, Assigned 21.01.2024, Due **11:59 pm on 28.01.2024**

The highest education is that which does not merely give us information, but makes our life in harmony with all existence – Rabindranath Tagore

Instructions:

- This homework is intended to get you upto speed on working with images. It is **strongly recommended** that you work on your homework on an *individual* basis. This is to make sure all of you become equipped with the basic tools required for the remaining assignments. If you have any questions or concerns, feel free to talk to the instructor.
- In this assignment, you will be working with grayscale images. Download images from University of Southern California's image database at `http://sipi.usc.edu/database/database.php?volume=misc`.
- Do not use built-in functions for the binary image processing question. Use *matplotlib* to read and plot images - `https://matplotlib.org/tutorials/introductory/images.html`.
- Please turn in Python Notebooks with the following notation for the file name: `your-roll-number-hw1.ipynb`.
- Do not turn in images. Please use the same names for images in your code as in the database (and as mentioned in the problem statement below). The TAs will use these images to test your code.

1 The Human Eye (5)

1. In this problem you will implement the contrast sensitivity function (CSF) based *living room design*. Write a Python script that accepts the TV specifications (size, resolution), the available distance for the couch placement (from the TV), and outputs the "optimal" distance. Recall from class that this distance corresponds to 6 cycles/degree being intended at the visual axis. Assume that the couch is placed directly in front of the TV. You are expected to handle corner cases gracefully. (5)

2 Gray Scale Images (5)

For this part, work with the gray scale images *aerial, airplane, APC*. Write a program to do the following:

1. Read the contents of an image into an array I (recall from class that digital images are represented as an array of numbers). Use *matplotlib* to read and display images - `https://matplotlib.org/tutorials/introductory/images.html`. Display the image I (1)
2. Print the maximum and minimum pixel values of I. Based on these values, how many bits are needed (used) per pixel? What is the resolution (i.e., width and height) of I? Print your answer. (1)
3. What is the size of the compressed image you downloaded? Print your answer. Based on the previous answers and assuming 256 gray levels, how efficienctly (compressed image size versus 8 bits per pixel size) is the image compressed for storage? Print your answer. (1)
4. Write a function that accepts as input an image I and a bitplane index i.e., a number b such that $0 \leq b \leq B-1$. The function must display the bitplane b of the image I as a binary image. Use the convention that $b = 0$ corresponds to the least significant bitplane and $b = B-1$ corresponds to the most significant bitplane. (2)

3 Binary Image Processing (20)

For this part, work with the gray scale images *airplane, truck, airport, APC*.

1. Write a function to compute the histogram of an image and plot the same. What is the modality of the histogram for the images mentioned above? (2)

2. Binarize I using the above histogram following the modal thresholding approach discussed in class. (2)

3. Implement Otsu's algorithm in two ways.

   (a) Construct the weighted intra-class variance array $\sigma_w^2(t) = P_0(t)\sigma_0^2(t) + P_1(t)\sigma_1^2(t)$ where t is the threshold, $P_0(t) = \sum_{i=0}^{t-1} p(i)$ is the probability of gray level 0, $P_1(t) = 1 - P_0(t)$ is the probability of gray level 1, $\sigma_0^2(t)$ is the variance of the pixels assigned to gray level 0, $\sigma_1^2(t)$ is the variance of the pixels assigned to gray level 1. Note that $p(t)$ corresponds to the probability that the image intensity is t (i.e., the normalized histogram or the probability mass function). Sweep over all possible values of t and pick the threshold where $\sigma_w^2(t)$ takes its minimum value. (4)

   (b) A more efficient implementation is based on the observation that the total image variance $\sigma^2 = \sigma_b^2(t) + \sigma_w^2(t)$ where $\sigma^2, \sigma_b^2(t), \sigma_w^2(t)$ are the image variance, between class (or inter-class) variance and the within class (or intra-class variance) respectively. Further, it can be shown that $\sigma_b^2(t) = P_0(t)(1 - P_0(t))[\mu_0(t) - \mu_1(t)]^2$, where $P_0(t), P_1(t)$ are as defined in the previous question, and $\mu_0(t), \mu_1(t)$ are the means of gray level 0 pixels and gray level 1 pixels respectively. *Show these results yourself but no need to turn in your work.* Importantly, the between class variance $\sigma_b^2(t)$ can be computed recursively as follows. Note that μ is the image mean. (6)

   - Initialize $P_0(0) = p(0), \mu_0(0) = 0$
   - Sweep over all possible values of t and do the following:
     - $P_0(t+1) = P_0(t) + p(t+1)$
     - $\mu_0(t+1) = \frac{\mu_0(t)P_0(t) + (t+1)p(t+1)}{P_0(t+1)}$
     - $\mu_1(t+1) = \frac{\mu - \mu_0(t+1)P_0(t+1)}{1 - P_0(t+1)}$
     - Update $\sigma_b^2(t+1)$
   - Find the threshold at which $\sigma_b^2(t)$ takes its maximum value.

   For each method, your program should output the "optimal" threshold t^* and display the corresponding binary image. Demonstrate your solution using any of the gray scale images mentioned above.

4. Implement the connected components algorithm and use it to label the binarized version of I. Demonstrate your result using any of the gray scale images mentioned above. (5)

5. Implement minor blob removal to get rid of minor blobs. Demonstrate your result using the gray scale image you used in the previous question. (1)

4 Binary Morphology (10)

1. Implement the following filters that accept a binary image I and window B as inputs: DILATE, ERODE, MEDIAN. Filter the above binary image using the following windows B = CROSS(5), B = SQUARE(3) (meaning the side of the square window is 3 pixels). Use the binary images generated in the previous question as input to this function. (3)

2. Verify that DILATE and ERODE operators are duals of each other with respect to complementation. Also verify that the MEDIAN operator is its own dual with respect to complementation. Show that the difference between the images from the direct and complement paths is indeed zero. (2)

3. Implement the compound operators OPEN and CLOSE using the basic filters implemented above. Use the same windows as in the previous problem. (2)

4. Finally, implement OPEN-CLOS and CLOS-OPEN. Filter the binary image using the same windows as above. (2)

5. For the *APC* image, count the number of pixels in the object of interest - i.e., the military vehicle. Use the blob counting code from the previous problem. (1)