Topics on High-Dimensional Data Analytics
ISYE 8803
Homework 2

Problem 1. Image Segmentation: Otsu's Method
Suppose you have an $m \times n$ gray image, with x_{ij} the intensity of pixel (i,j), $i = 1, \cdots, m$ and $j = 1, \cdots, n$. We can define the total variance of the image as:

$$\sigma_T^2 = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} - \mu_T)^2$$

where μ_T is the average intensity of the image.

Now, consider thresholding the histogram of the image in order to get a black and white image. Let t be the threshold. The goal of Otsu's method is to find an optimal value of t, by minimizing the intra(within)-class variance, defined as:

$$\sigma_w^2(t) = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t)$$

where

- $\omega_i(t)$ is the proportion of pixels in class i, $i = 1, 2$
- σ_i^2 is the variance of class i, $i = 1, 2$

It is easy to see that minimizing the intra-class variance is equivalent to maximizing the inter(between)-class variance:

$$\sigma_b^2(t) = \sigma_T^2 - \sigma_w^2(t)$$

Show that:

$$\sigma_b^2(t) = \omega_1(t)\omega_2(t) \left(\mu_1(t) - \mu_2(t)\right)^2 .$$

where $\mu_i(t)$ is the average intensity of class i, $i = 1, 2$.

Problem 2. Image Processing
In this problem you will practice the techniques learned in the image analysis module. The image you will study can be found as FlowerN.jpg. Please, follow the instructions and submit your code and the images you obtain in each step.

a. Open the image.

b. Transform the image to a gray image, and to a black and white image (using a level of 0.6).

c. Resize the original image to a quarter of its size. What is the size of the new image?

d. For the gray image, show the histogram. Additionally, apply the following transformations. Report the images obtained and the corresponding histograms.

- Thresholding: use a threshold of 150.
- Histogram shift: use a lower bound of 25, an upper bound of 225 and a shift of 50.
- Histogram stretch: use a lambda value of 205.
- Log-transformation: use $c = 40$.
- Power-law transformation: use $c = 0.1$ and $\gamma = 1.4$.

What is the purpose of each of these transformations?

e. For the coloured image, use the following convolution mask to blur the image:

$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

f. For the coloured image, use the following convolution mask to sharpen the image:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

g. For the coloured image, use the following convolution mask to emboss the image:

$$\begin{bmatrix} -1 & -1 & -1 & -1 & 0 \\ -1 & -1 & -1 & 0 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

h. For the coloured image, use the following convolution mask to detect the edges of the image:

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

i. For the gray image, use Otsu's method to determine the optimal threshold for a black and white image. Report the threshold value you obtained.

j. For the colored image, use K-means clustering algorithm to segment the image, use 2, 3, 4 and 5 clusters.

k. Use the Sobel operator to detect the edges in the image. Use different cutoff values. What happens when the cutoff increases?

Problem 3. Canny's edge detection algorithm

Gradient operators, such as those introduced by Sobel, may be used to identify the border of an image. However, this method broadens the edge of the image and can be easily corrupted by noise. Canny proposed a multiple detection method that avoids these two shortcomings.

In this problem you should code your own version of Canny's edge detection algorithm. Use your code to detect the edges of the figure given in problem 2. You need to submit your code and the image obtained.

Given the original image $f(i,j)$, $0 \leq i \leq m-1, 0 \leq j \leq n-1$, and using the notation for neighbouring pixels shown in Figure 1, you should:

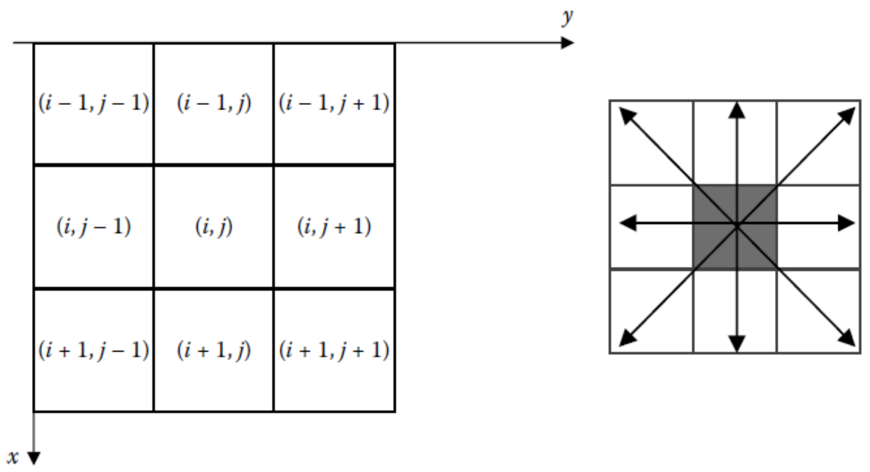

Figure 1: Neighbour of the pixel (i,j) and gradient directions.

a. Use a Gaussian filter h to smooth the image f leading to the smoothed result $S = h * f$. In this problem we will use the following Gaussian filter:

$$\frac{1}{1115}\begin{bmatrix} 1 & 4 & 7 & 10 & 7 & 4 & 1 \\ 4 & 12 & 26 & 33 & 26 & 12 & 4 \\ 7 & 26 & 55 & 71 & 55 & 26 & 7 \\ 10 & 33 & 71 & 91 & 71 & 33 & 10 \\ 7 & 26 & 55 & 71 & 55 & 26 & 7 \\ 4 & 12 & 26 & 33 & 26 & 12 & 4 \\ 1 & 4 & 7 & 10 & 7 & 4 & 1 \end{bmatrix}$$

b. Compute the gradient magnitude $G(i,j)$ and the gradient direction $\theta(i,j)$ of the pixel at point (i,j) of the image function S by using the formula:

$$G(i,j) = \sqrt{\left(\frac{\partial S}{\partial x}(i,j)\right)^2 + \left(\frac{\partial S}{\partial y}(i,j)\right)^2}$$

$$\theta(i,j) = \arctan\left(\frac{\frac{\partial S}{\partial y}(i,j)}{\frac{\partial S}{\partial x}(i,j)}\right)$$

where the approximate discrete formulas of the to partial derivatives are given by:

$$\frac{\partial S}{\partial x}(i,j) = \frac{1}{2}\left[S(i+1,j) - S(i,j) + S(i+1,j+1) - S(i,j+1)\right]$$

$$\frac{\partial S}{\partial y}(i,j) = \frac{1}{2}\left[S(i,j+1) - S(i,j) + S(i+1,j+1) - S(i+1,j)\right]$$

c. Determine the edge pixel using nonmaximal suppression. The characteristic of an edge pixel is that its gradient magnitude is the local maximal in the gradient direction. To determine whether the gradient magnitude of the pixel at (i,j) is a local maximal or not, one needs to locate the two neighbouring pixels p_1 and p_2 of the pixel at point (i,j) and calculate the gradient magnitudes of the three pixels. Suppose pixels p_1 and p_2 are located at positions (i_1,j_1) and (i_2,j_2), respectively. If the gradient magnitude of the pixel at position (i,j) is maximum, it is an edge point, and the gradient magnitude is used as its intensity; otherwise, the pixel is not an edge point, and its intensity is set to 0. The resulting image ϕ can be described as follows:

$$\phi(i,j) = \begin{cases} G(i,j) & \text{if } G(i,j) \geq G(i_1,j_1) \text{ and } G(i,j) \geq G(i_2,j_2) \\ 0 & \text{otherwise} \end{cases}$$

Because the locations of pixels are discrete, gradient directions also need to be quantized. Take the 8-neighbouring domain, as shown in Figure 1, with the pixel at position (i,j) as an example. The positions (i_1,j_1) and (i_2,j_2) of the neighbouring pixels p_1 and p_2 in the gradient direction can be computed as follows:

- If $-\frac{1}{8}\pi \leq \theta(i,j) \leq \frac{1}{8}\pi$, $\theta(i,j)$ is quantized as 0, and
$$(i_1,j_1) = (i,j-1),\ (i_2,j_2) = (i,j+1)$$
- If $\frac{1}{8}\pi \leq \theta(i,j) \leq \frac{3}{8}\pi$, $\theta(i,j)$ is quantized as $\frac{1}{4}\pi$, and
$$(i_1,j_1) = (i+1,j-1),\ (i_2,j_2) = (i-1,j+1)$$
- If $-\frac{3}{8}\pi \leq \theta(i,j) \leq -\frac{1}{8}\pi$, $\theta(i,j)$ is quantized as $-\frac{1}{4}\pi$, and
$$(i_1,j_1) = (i-1,j-1),\ (i_2,j_2) = (i+1,j+1)$$

- If $\frac{3}{8}\pi \leq \theta(i,j) \leq \frac{1}{2}\pi$ or $-\frac{1}{2}\pi \leq \theta(i,j) \leq -\frac{3}{8}\pi$, $\theta(i,j)$ is quantized as $\frac{1}{2}\pi$, and

$$(i_1, j_1) = (i-1, j),\ (i_2, j_2) = (i+1, j)$$

d. Threshol with hysteresis. Non-maximal suppression reduces the border of an object to the width of just one pixel. Due to the existence of noise and thin texture, this process may result in spurious responses, which lead to streaking problem. Streaking means the breaking up of an edge contour caused by the operator fluctuating above and below the threshold. Hysteresis using two thresholds $\tau_1 < \tau_2$ can eliminate streaking. If the value $\phi(i,j)$ of the pixel at position (i,j) in the resulting image is larger than τ_2, the pixel is definitely an edge pixel, and all such edge pixels constitute the edge output. Any pixel connected to this edge pixel that has its value larger than τ_1 is selected as an edge pixel. The following algorithm details the thresholding with hysteresis:

```
Let $\Omega(i,j)$ denote the 8-neighbourhood of the pixel at $(i,j)$.
For the given image $\phi(i,j) : 0 \leq i \leq m-1, 0 \leq j \leq n-1$
Set two thresholds: $\tau_1 \leq \tau_2$
Initialize the resulting edge image $E(i,j) = 0, 0 \leq i \leq m-1, 0 \leq j \leq n-1$
while $count \neq 0$ do
    $count = 0$
    for $i = 0$ to $m-1$ do
        for $j = 0$ to $n-1$ do
            if $\phi(i,j) \geq \tau_2$ and $E(i,j) = 0$ then
                $E(i,j) = 1$ and $count = count + 1$
            else if $\phi(i,j) \geq \tau_1$ and $E(i,j) = 0$ then
                For each $(k,l) \in \Omega(i,j)$
                if $E(k,l) = 1$ then
                    $E(i,j) = 1$ and $count = count + 1$
        end
    end
end
Output the resulting edge image: $E(i,j)$
```