Topics on High-Dimensional Data Analytics
ISYE 8803 - Homework 2

Problem 1. Image Segmentation: K-Means Method (15 points)

Consider an $m \times n$ gray image with x_{pq} being the intensity of pixel (p, q), $p = 1, \ldots, m$ and $q = 1, \ldots, n$. We want to segment the image into k non-overlapping clusters by using the K-means clustering method.
We first reshape the image pixels such that the resulting matrix is an $m * n = t$ vector. The total variation of the reshaped image is defined as:

$$\sum_{i=1}^{t}(x_i - \mu_T)^2$$

where μ_T is the average intensity of the image.

Given k a priori, the K-means algorithm aims to minimize the total within-cluster variation, denoted as:

$$\sum_{j=1}^{K}\sum_{i=1}^{t} w_{ij}(x_i - \mu_j)^2$$

where μ_j is the mean intensity of cluster C_j, and $w_{ij} = \begin{cases} 1 & x_i \in C_j \\ 0 & x_i \notin C_j \end{cases}$

However, according to the Huygens Theorem, minimizing the within-cluster variation is equivalent to maximizing the between-cluster variation. Show that the total between-cluster variation can be defined as:

$$\sum_{j=1}^{K} t_j(\mu_j - \mu_T)^2$$

where t_j is the number of elements in the cluster C_j

Problem 2. Image Processing (45 points)

In this problem you will practice the techniques learned in the Image Analysis module. The image you will use can be found as horse1.jpg. Please, follow the instructions and submit your code and the images you obtain in each step.

a) Read and show the image.

b) What is the size of the original image? Resize the original image to a third of its size. What is the size of the new image?

c) Convert the original image to a gray image and to a black and white image (using a level of 0.5).

d) Show the histogram of the gray image. What observations can be made from this histogram?

e) For the gray image, apply the following transformations. Report the images obtained and the corresponding histograms. Also, explain the purpose of each of these transformations.

   1) Linear transformation
   2) Log-transformation: use c = 20 and c = 40.
   3) Thresholding: use thresholds of 100 and 150.
   4) Histogram shift: use a lower bound of 25, an upper bound of 225 and a shift of 50.
   5) Histogram stretch: use a lambda value of 200. What's the contrast of the original gray image and transformed image?

f) Create a noisy image, by adding ramdom noise with mean zero and variance of 100 to the gray image. Use the following convolution mask to denoise the image. Show noisy vs. denoised image.

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

g) For the original colour image, use the following convolution mask to sharpen the image:

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

h) For the gray image, use Otsu's method to segment the image into 2, 3, 4, and 5 levels. Report the threshold values you obtained for each instance.

i) For the original colour image, use K-means clustering algorithm to segment the image into 2,3,4 and 5 clusters.

j) Use the Prewitt and Sobel methods to detect the edges in the gray image. Use different threshold values. What happens when the threshold increases?

Problem 3. Image Edge Detection- Canny's algorithm (40 points)

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. The image you will use can be found as horse1.jpg. You need to submit your code and the image obtained.

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

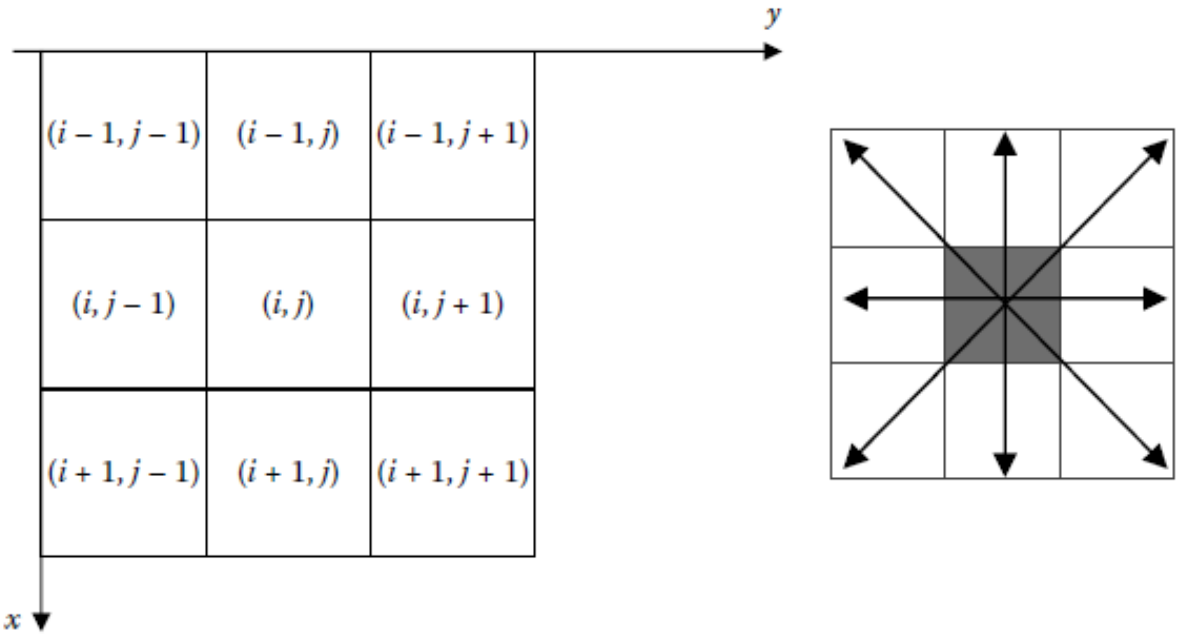

Figure 1: Neighbour of the pixel (i,j) and gradient directions.

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 formulas:

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

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

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

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

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

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 $-\frac{1}{8}\pi < \theta(i,j) \leqslant \frac{1}{8}\pi, \theta(i,j)$ is quantized as 0, and $(i_1, j_1) = (i, j-1), (i_2, j_2) = (i, j+1)$
- If $\frac{1}{8}\pi < \theta(i,j) \leqslant \frac{3}{8}\pi, \theta(i,j)$ 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 $-\frac{3}{8}\pi < \theta(i,j) \leqslant -\frac{1}{8}\pi, \theta(i,j)$ 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 $\frac{3}{8}\pi < \theta(i,j) \leqslant \frac{1}{2}\pi$ or $-\frac{1}{2}\pi \leqslant \theta(i,j) \leqslant -\frac{3}{8}\pi, \theta(i,j)$ is quantized as $\frac{1}{2}\pi$, and $(i_1, j_1) = (i-1, j), (i_2, j_2) = (i+1, j)$

d) Threshold 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 $\phi(i,j) : 0 \leqslant i \leqslant m-1, 0 \leqslant j \leqslant n-1$
Set two thresholds: $\tau_1 \leqslant \tau_2$
Initialize the resulting edge image $E(i,j) = 0, 0 \leqslant i \leqslant m-1, 0 \leqslant j \leqslant n-1$
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
            end
            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
    end
end
Output the resulting edge image: $E(i,j)$