**2110431 Introduction to Digital Imaging**

**2147329 Digital Image Processing and Vision Systems**

**Homework #1**

**Deadline: September 5th, 2023 @23:59**

**Submissions: (1) PDF version of this file**

**(2) .ipynb file; template in this link:**

[**https://colab.research.google.com/drive/1jub1fLi4a16kmD5JB2mhJdCPJKGOSkxw?usp=sharing**](https://colab.research.google.com/drive/1jub1fLi4a16kmD5JB2mhJdCPJKGOSkxw?usp=sharing)

1. Write a program in python into **homework1\_1()** function in **homework1.ipynb** file to implement contrast stretching follow the transformation in the graph below.

Chart, diagram

Description automatically generated

Test your program using kitty.jpg and your own image and display your results in the blank below.

Results of the processed images:

|  |  |
| --- | --- |
| **Original** | **After filtering** |
| Kitty.jpg |  |
| Your image: |  |

2. Design your own filter on an RGB image. Write your code in **homework1\_2()** function in **homework1.ipynb** file Provide motivation behind the designed filter. Display it in terms of an RGB image.

Idea / Motivation:

|  |
| --- |
|  |

Your filter design (at least two equations and/or conditions):

|  |
| --- |
|  |

Examples of filtered image:

|  |  |
| --- | --- |
| **Original** | **After filtering** |
|  |  |
|  |  |

3. Two images, f(x,y) and g(x,y), have histograms hf and hg. Write a program to display the histograms hf and hg. Then implement the operations below and display the new histogram of the output of each operation. Determine the new histograms in terms of hf and hg and explain how to obtain the histogram in each case (Optional).

(a) f(x,y) + g(x,y)

(b) f(x,y) - g(x,y)

(c) f(x,y) x g(x,y)

(d) f(x,y) g(x,y)

**Hint**: design one of the images very simple.

4. Assume you work in the field of image processing. Your boss has assigned you a task to detect malignant tumors (assuming in this case, in bright intensity) from CT-SCAN images. The pain point is that the doctors saved images from the CT-SCAN, but the output images are incomplete and have Salt and Pepper noise. Please help the doctor remove the noise.

4.1 Apply a filter to remove the noise and select the appropriate size of the kernel. Provide your filtered image into the blank box below. Hint cv2.medianBlur

|  |  |
| --- | --- |
| Original | After Filtering |
| A close-up of a body  Description automatically generated |  |

4.2 Apply Region of Interest (ROI) with width=380 and height=435 start at x=300, y=275 as shown in the orange rectangle below and provide the ROI image in the blank box below. Hint cv2.rectangle

|  |  |
| --- | --- |
| Hint | Mark ROI Image |
| An x-ray of a person's body  Description automatically generated |  |

4.3 Apply the transformation function shown in the graph below on the ROI image. This transformation function is used for segmenting malignant tumors (assuming, in this case the higher intensity) and show in a white mask. Provide the final segmented tumors in the blank box below.

|  |
| --- |
| Transformation Function |
| L-1 is the highest intensity level.  You can estimate the stepping point from the graph. |

|  |  |
| --- | --- |
| Original (Mark ROI) | Segmentation with Transformation |
|  |  |

**Download ipynb template in this link and submit your own ipynb in MCV (Not accept a link):**

[**https://colab.research.google.com/drive/1jub1fLi4a16kmD5JB2mhJdCPJKGOSkxw?usp=sharing**](https://colab.research.google.com/drive/1jub1fLi4a16kmD5JB2mhJdCPJKGOSkxw?usp=sharing)

|  |
| --- |
| Your code (in **homework1.ipynb**):  # import libraries here  def homework1\_1(image\_grayscale):  # input -> image\_grayscale - type -> np.ndarray, size of - (height, width)  # output -> image\_grayscale - type -> np.ndarray, size of - (height, width)    # TO DO - Implement transformation based on the contrast stretching graph    return filtered\_image  def homework1\_2(rgbimage):  # input -> rgbimage - type -> np.ndarray, size of - (height, width, 3)  # output -> filtered\_image - type -> np.ndarray, size of - (height, width, 3)    # TO DO - Design your own filter    return filtered\_image |

|  |
| --- |
| Your code (in **homework1.ipynb**):  # import libraries here  def Home\_work1\_4 (rgbImage):  # Convert the image to grayscale  grayImg = cv2.{Fill}(rgbImage,{Fill})    # 4.1 Use the median filter to smooth the image  smoothed\_img = cv2.{Fill your filter}(grayImg, {Fill num of filter})  # 4.2 Make ROI with  # Create an interesting area  mark = np.ones((smoothed\_img.shape[0], smoothed\_img.shape[1]), dtype={Fill}) \* {Fill}  # Set the coordinates of the rectangle  start\_point = (300, 275)  end\_point = ({Fill}, {Fill})  # Draw a black square at the center  cv2.{Fill}(mark ,start\_point, end\_point, (0), -1)  # Mark the area in the image  mark\_ROI = {Fill condition}    # 4.3 Use Gray Level slicing  Gray\_Level\_img = mark\_ROI.copy()  Gray\_Level\_img[(mark\_ROI >= {Fill Threshold})] = 255  Gray\_Level\_img[(mark\_ROI < {Fill Threshold})] = 0    # Picture show Row1  plt.figure(figsize=(20, 20))  plt.subplot(1, 2, 1)  plt.imshow(rgbImage[:, :, ::-1])  plt.title('Original')  plt.axis('off')    plt.subplot(1, 2, 2)  plt.imshow(smoothed\_img, cmap='gray')  plt.title('Image after smoothing')  plt.axis('off')    # Picture show Row2  plt.figure(figsize=(10, 8))  plt.subplot(2, 2, 1)  plt.imshow(mark\_ROI, cmap='gray')  plt.title('Mark ROI Image')  plt.axis('off')    plt.subplot(2, 2, 2)  plt.imshow(Gray\_Level\_img, cmap='gray')  plt.title('Segmentation with transformation img')  plt.axis('off')  plt.show() |