CIM 2023 Lab Assignment - Visual Part - assignment 1

Assignment Name: **introduction to visual signal processing in Matlab**
Due Date: **March 20th, 2023.** Submit in Moodle your report file in PDF and all the Matlab files with the code you have developed as a single compressed archive.

Purpose

This work intends to apply and consolidate knowledge about the representation of visual signals in the digital domain; to get acquainted with basic image processing techniques in Matlab - color spaces; spatial sampling; convolution and filtering; contour detection.

Why is it important that multimedia students gain the listed skills and knowledge? Because it provides them the foundations for the manipulation of visual signals. Because in visual signals compression many times such type of operations are performed as preparatory steps.

Skills

In this assignment, you'll be learning how to:

1) develop simple practical algorithms using Matlab to manipulate characteristics of visual signals (still images);
2) how to apply filters to still visual signal to extract some features, namely object counters, and
3) evaluate the efficiency of the selected techniques.

Knowledge

In this assignment, you'll be learning about:

1) different color spaces for representing visual signals;
2) the effects of spatial sub-sampling in visual signals;
3) the effects of applying different types of spatial filters to still images.

Task overview

The objective of this work is to allow students to acquire a better understanding of the fundamentals of representing visual signals in the digital domain and of processing techniques that allow for improving image quality, extracting low-level features and preparing signals for compression.
To carry out these experiments, whenever necessary, files with images available on the UC's Moodle will be used as input signals for the algorithms/processes implemented by Matlab programs or scripts also available on the UC's Moodle page. Most of these images are in BMP (bitmap) format, which means that each pixel is represented by three 8-bit RGB values, that is, a total of 24 bits. Comparing results obtained by applying different algorithms, it is intended that the student acquires a better understanding of the role played by different techniques and formats.

Note: the ✍ symbol means that you must include in your report graphics or images that resulted from the processing carried out or even code that you have developed. The 🕵 symbol indicates that you should include a brief analysis of the results you obtained in the report.

Implementation work to be carried out

1. **Experiments with color spaces**

There are several color spaces to represent visual signals, each with its own coordinate system, having different purposes or areas of application. In this part of the work we will use three color spaces: RGB (Red, Green, Blue), HSV (Hue, Saturation, Value, where V represents brightness) and YUV (luminance and color difference signals).

1.1. Write a Matlab script ✍ that:

1.1.1. import an image in bitmap format (RGB color space) and display this image on the screen;

1.1.2. separate each RGB component into a different matrix and display each one of them on the screen ✍;

1.1.3. convert that image to HSV color space and display that image on the screen;

1.1.4. separate each HSV component into a different matrix and display each one of them on the screen ✍;

Run the script with various images such as “peppers.png”, “lighthouse.png” and others available in Matlab, or “floresVermelhas.bmp”, “folhasVerdes.bmp”, “praia.bmp” and “elephant.bmp”, available on Moodle. Compare components of each image 🕵 with each other.

1.2. Develop a similar script but instead of converting to HSV convert to YCbCr.

Run the script with the same images. Compare components of each image 🕵 with each other. Compare with the results obtained with the previous script.

1.3. Use now the script “rgb2yuv.m” and check if there are differences in relation to the results obtained in experiment 1.2. 🕵

2. Variation of image spatial dimensions using or not filters with the “imzoneplate” test image

In this part, you must develop the “ampliaReduz.m” script in Matlab, which must use the “imzoneplate.m” script that is available in the UC's Moodle (obtained from the MathWorks website).

Before starting the work, analyze the code of the programs provided in order to understand the operations carried out.

The program to be developed “ampliaReduz.m” must allow enlarging/reducing the spatial dimensions of a “zone plate” test image created during its execution using the “imzoneplate.m” function. It should allow enlarging/reducing by simply repeating/eliminating pixels and enlarging/reducing using Matlab's built-in function “imresize.m”. You may have the option of using an averaging or gaussian filter before zooming in/out. The program must display the original and processed images on the screen, as well as the respective graphs of spectral density and variation of the signal in space.

Start Matlab and change to your own working directory. Copy all necessary files (programs and images).

2.1. develop your program “ampliaReduz.m” and run it with different dimensions for the test zoneplate image and using different interpolation methods of the built-in function “imresize.m”. ✍ Review the built-in “resample.m” function Run this function using the zoneplate test image. Compare and interpret the results. 🕵

3. Filtering experiences

In these experiments, you must develop two programs that allow you to perform different types of image filtering and you will apply an edge detector using the built-in Matlab function “edge.m”.

3.1. The first program must be developed using the Matlab built-in functions “fspecial” and “imfilter”. ✍

3.1.1. Run the program using different images as input signal and testing the various possible filter types. Check the effects of each filter ✍.

3.1.2. Run the program with different filter dimension values for the mean and gaussian filter case. Analyze the results for the different values. 🕵

3.2. The second program to be developed must perform the explicit convolution between an image and the kernel of a filter specified in the program ✍. The kernels to be specified must be the following:

3.2.1. Mean filter (3X3 and 5x5)

3.2.2. median filter

Use the mean and median filters to remove noise from “noise1.jpg” and “noise2.jpg” images. Comment the results. ✍ 🕵

3.2.3. Sobel Operator. $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

3.2.4. Prewitt Operators. $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Apply the Sobel and Prewitt operators to the images “casa1.jpg”, “casa2.jpg” and “cameraman.tif” and comment the results. ✍ 🕵

3.3. Apply the Canny's contour detector to the “cameraman.tif” image using Matlab's “edge” function. Test different “sigma” and “threshold” values and interpret the results. ✍ 🕵

Criteria for Success

It is important that you understand the information conveyed in each color channel of the different color spaces and that when visualising them, you are able to correctly indicate each one. Also that you understand how Matlab manipulates images. It is also important that you understand and are able to explain the effects of resizing images and adequately relate this operation with sampling. You should also be able to describe the role of filters in this process. Finally, you should understand the practical aspects of performing spatial filtering and explain the different results that different kernels (filters) obtain applied to the same images.

The report should be delivered in Moodle by March 20th.

Note: The template used to describe this assignment is adapted from the University of Las Vegas, Nevada, and the Transparency in Learning and Teaching (TILT) Project.