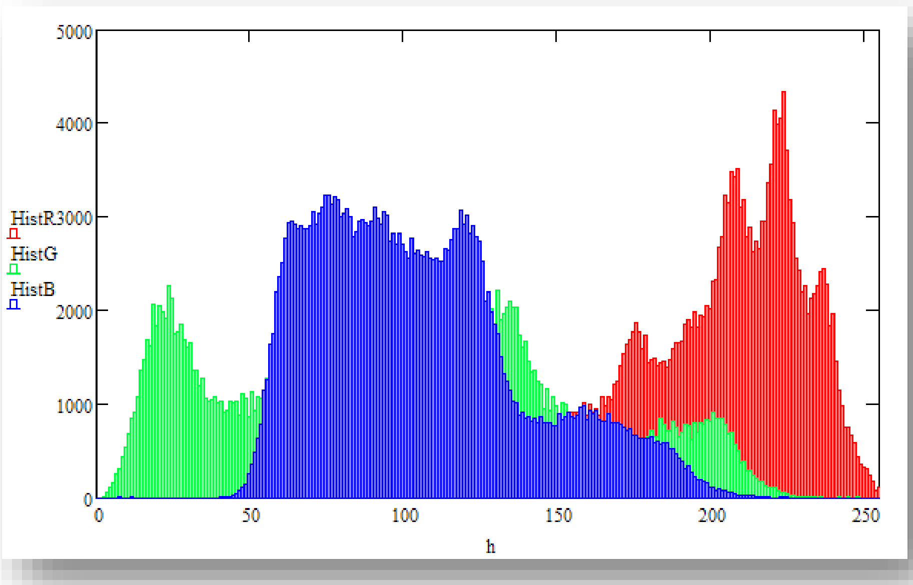
Introduction to Audio-visual Processing

Lab 1: Statistical Signal Modelling I

Memoryless processes



Course 19-20/Q2

Contents

Introduction
Pre Lab Activities
Lab Activities
1. Memoryless processes of image signals
2. Memoryless processes of audio signals

Introduction

This first lab session focuses on elementary memoryless operations over processes. The aim is, first, to understand the information that histograms provide; second, to consolidate the concept of quantization (uniform, non-uniform and optimal quantization) including its quality measures (distortion, SNR, PSNR); and finally, to implement simple operations at pixel-level (or sample level): mappings, histograms and equalization. Image and audio signals are used to illustrate these topics. Complementary activities can be found in Atenea processing data signals.

Pre Lab activities:

Review the concepts.

The aim of the lab is to review, consolidate and extend the basic concepts studied in class. You should review Units 1.1 and 1.2 before the lab session.

Install MATLAB.

Some MATLAB functions are provided to perform the different activities, so you do not have to code. Before the Lab session, you have to install MATLAB in your laptops. In Atenea you can find the link and the instructions to install MATLAB using the UPC student MATLAB license. Be sure that you install not only MATLAB but also the Image processing toolbox.

Open Images.

The first part of the Lab session uses different pictures to understand image histograms and implement some image processing algorithms (mapping and quantization) that operate at pixel level. With the manual lab you can find five images whose characteristics will help us to understand the main concepts of this practice.

Using the function `Image_Read()` you can open the different images. This function allows to select by default any `*.png` image that you have in the workspace folder (other image formats could also be opened if you select "*All files (*.*)*" on the emerging window). In case you select a color image this is converted into a gray level image. The functions maps the gray level values in the [0,1] range (where 0 represents the black level and 1 represents the white level). The syntax is:

```
ima=Image_Read;
```

To display images you can use the function `Image_Display()`, which allows you to display up till five simultaneous images:

```
Image_Display(ima1,ima2,ima3,ima4,ima5);
```

Open the five images and display them.

Load and listen audio signals

The second part of the Lab session aims to understand the quantization of audio signals. Here, the information given by the histograms is also relevant to understand the behavior of the different quantization strategies. With this manual lab you can find two different audio files (*Audio1.wav* and *Audio2.wav*) that have been quantized with 16 bits and recorded at a sampling rate of Fs=48KHz. This audio sequences will allow us to understand some audio quantization concepts.

Using the function `Audio_Read()`you can load the audio files. This function generates a mono signal normalized in a dynamic range between -1 and 1. The function also plots the shape of an audio interval and reproduces the entire audio piece. The syntax is:

```
x= Audio_Read(FileName)
```

Load the two audio files and listen to them.

1. Memoryless processes of image signals

Open the five images using `Image_Read()` function. If you read the fives images using five different names (e.g., ima1, ima2 ...) it will not be necessary to read them anymore during the Lab session.

1.1. Image histogram

The objective of this part is to generate image histograms, understand the information that they provide, and characterize the images by means of their histograms.

The histogram computes the relative frequency of the different gray levels of a given image. Use the function `Image_histogram()` to show the histograms of the five images;

```
Image_histogram(ima,ima2,ima3,ima4,ima5);
```

Compare the different histograms and relate its information to the characteristics of the image. Copy and paste on the report document the original image, its histogram and comment the results. 1.1

1.2. Image contrast (linear and non-linear)

Once you have plotted and analyzed the histograms of the five images, let us process those images (at pixel level) in order to extract some additional information.

The function `J=Image_Contrast(I,[low high],[min max])` maps the luminance values of image `I` between the darkest value (`low`: close to 0.0) and the brightest value (`high`: close to 1.0) between `min` and `max` in the output image `J`. Values below `low` (resp. above `high`) map to `min` (resp. `max`).

In order to see how this function works, let us increase the contrast of the image *Im1_Sea.png* using this function:

```
Ima1_c = Image_Contrast(ima1,[0.25 0.75],[0.0 1.0]);
```

Display the contrasted image and also plot its histogram to understand the result. Notice that the values above 0.75 on the original image are clipped and mapped to the maximum brightness and, similarly, that the values below 0.25 are mapped to the minimum brightness.

Once you know this function, let us modify the input range [low high] and output range [min max] in the function in order to get the mappings requested below. You must execute the command in MATLAB choosing [low high] and [min max] adequately. Write in the report document the command, copy and paste the resulting images and comment the observed effects.

a. Enhance the contrast.
   Analyze the histogram of *Im5_Boy.png* and increase its contrast making the blacks darker and the whites brighter. Have you lost information doing this transformation? Is it possible to reverse the transformation recovering the original image? 1.2a

b. Before counting drops.
Counting objects is one of the activities that can be done automatically implementing image processing routines. Enhancing the contrast between the objects and the background is often performed and it can be done binarizing the image. *Im3_Drops.png* represents an image full of drops that we want to count. Open the image and binarize it selecting the appropriate threshold trying to clip the image with two levels, one for the drops and another one for the background. Notice that it is impossible to get an image with black *rounds* that represent the drops over a white background? Try to do your best. 1.2b

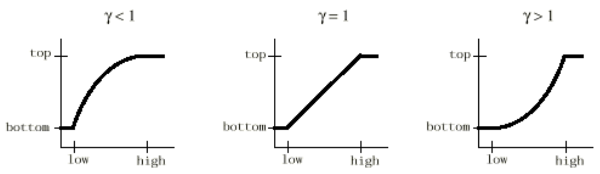
Note: This result evidences that image processing algorithms (in general signal processing algorithms) that operate at pixel (sample) level are not enough to process data. As we will study in class, exploding neighborhood information will be in most cases advantageous.

c. A white image?
Apparently *Im4_White.png* image seems to be a white image. However an analysis of its histogram evidences that there is some information. Select a range of values from x1 to x2 where you consider that something meaningful appears in your image but that are difficult to observe and expand this range of values so that they occupy the full range (from 0.0 to 1.0). How much did the *Lasagne Bolognaise* cost? 1.2c

d. The image was manipulated.
Apparently *Im2_Clock.png* shows a clock marking four minutes past four. However a forensic study of this picture evidenced that the picture had been manipulated. The image contains some relevant hidden information that was neglected when the picture was edited. How did the analyst discover that the image had been manipulated? Could you say what time it was when the picture was taken? 1.2d

e. Negative image
Analyze the histogram of *Im1_Sea.png* and invert the range of values of the input image to create a negative. 1.2e

Up till now, the input-output transformation between low and high levels has been formed by segments of lines. The function `Image_Contrast()` allows adding a last argument, `gamma`, in order to obtain other types of mappings as shown in the following figure:



The complete syntax of the contrast function is:

```
ima3_c=Image_Contrast(ima3,[low high],[min max],gamma);
```

Use this command with the appropriate [low high], [min max] to edit *Im5_Boy.png* image. Choose one value of gamma below 1.0 and one above 1.0. Cut and paste the images with the two gamma values and describe the visual effects. 1.2f

1.3. Image histogram equalization

The objective of this part is to study the histogram equalization.

The function `Image_EqHistogram()` equalizes the histogram implementing a pixel-based transform aiming at producing a (near to) flat histogram output image. The syntax of the function is:

```
ima_eq = Image_EqHistogram(ima);
```

Use this function to equalize *Im1_Sea.png* and *Im2_Clock.png* images. Compare the original and the equalized images using the function `Image_Display()`. Compare, also, the original and equalized histograms.

Discuss the following aspects:

- Comment the main differences between the original and equalized images. 1.3
- Equalization appears to be much more perceptible in one of the images. Why? When is it interesting to equalize an image?
- Is it possible to perfectly equalize a digital image? Why?

1.4. Uniform and optimum quantizer

In this section, we will design a uniform quantizer and an optimum quantizer adapted to the values of the image to be quantized.

The way to quantize an image (in general a set of data) is by truncating the matrix (or vector) to integer values (`round()` function can be used). Quantization implies loss of information and quality. The quality of a quantizer can be measured with the mean squared error and the peak-signal-to-noise ratio defined in next table.

		Quality measures
Error	$e[i,j] = \hat{I}[i,j] - I[i,j]$	$SNR(dB) = 10\log_{10}\dfrac{\sigma_x^2}{\sigma_e^2}$
Estimated MSE	$MSE = \sigma_e^2 = \dfrac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\lvert e[i,j]\rvert^2$	$PSNR(dB) = 10\log_{10}\dfrac{1}{\sigma_e^2}$

The function `Image_UniformQ()` implements a uniform quantizer with N quantization levels. This function returns the quantized image, and the quality measures `mse` and `psnr`. If N is the number of levels, the syntaxes is:

```
[ima_quantN, mse, psnr]= Image_UniformQ(ima,N);
```

Quantize the *Im1_Sea.png* image for N={256,50,10} levels. Use `Image_Display()` to simultaneously visualize the original image and the three quantized images. Answer the following questions:

- Observe and comment the effect of "*false contours*" in the quantized image that appears for low N values. 1.4a

- Determine the number of gray levels N necessary to achieve "*transparent quantization*" (unnoticeable quantization effects). Maximize the window to better see the result. For all the different N values that you assess, report also the mse and psnr values and comment the results. How is the drop in perceptual quality related to the decrease of these measures (`mse` and `psnr`)? 1.4b

Let us now compare uniform quantization with the **optimum Max Lloyd algorithm** for a given number of levels N. The `Image_MaxLloydQ()` function implements the optimum quantizer and also reports its quality (`mse` and `psnr`)

```
[ima_uniform, mse, psnr]  = Image_UniformQ(ima,N);

[ima_maxlloyd, mse, psnr] = Image_MaxLloydQ(ima,N);
```

Quantize the *Im1_Sea.png* and *Im5_Boy.png* images using the uniform and optimum quantizers. Analyze the results answering the following questions:

- Visualize the results and report their quality measures: `mse` and `psnr`. 1.4c
- Comment, for both images, the differences that you perceive between the uniform and optimum quantizer (perceptual measure).
- Discuss also the differences in terms of the `mse` and `psnr` (objective measures).
- Compare the `psnr` gain between the optimum and the uniform quantizers for both images. Which image presents a larger gain? Can you explain this behavior? Use the information given by the histograms of the original images to justify the answer.

2. Memoryless processes of audio signals

2.1. Uniform quantization of audio signals

Next, using the mid-tread uniform quantization scheme, we are going to quantize the audio signals with different numbers of bits.

To quantize the audio signal with N_b bits we use the function `Audio_UniformQ()`. This function returns the quantized signal, the SNR in dB and the quantization error. The function also plots an interval of the quantized signal, reproduces all the audio signal, and displays the quantization error. The syntax is:

```
[x_quantN, SNR, Qerr] = Audio_UniformQ(x,Nbits,Fs)
```

Load the two audio signals using the function `Audio_Read()`. For N_b=5 quantize both audio files, listen to the quantized sounds and analyze the reported figures answering the following questions:

2.1a

- How many quantization levels are used to represent the interval [-1, 1]?
- Measure the dynamic range of the quantization error and compare it with the expected theoretical value.
- What is the representation level of the samples that are close to 0?

Although the number of bits used to quantize both audio signals was the same, you have evidenced that the perceived quality is different.

2.1b

- Describe the distortion you perceive listening both quantized audio signals.
- Check the SNR in both cases. How do you explain this quality difference? Use the information given by the histograms to justify this answer.

Next, we will verify, experimentally, a theoretical result given in class: "*a signal that is quantized with a uniform, scalar quantizer increases its quality (SNR) in 6 dBs with every additional bit used in the quantizer*".

For both audio files calculate the SNR as a function of the number of bits (N_b) in the range from 1 to 16. Plot the experimental SNR and overlap the theoretical result:

$$SNR_t(N_b) = 6 \cdot N_b + 4.77 - 20\log_{10}(1/\sigma_x)$$

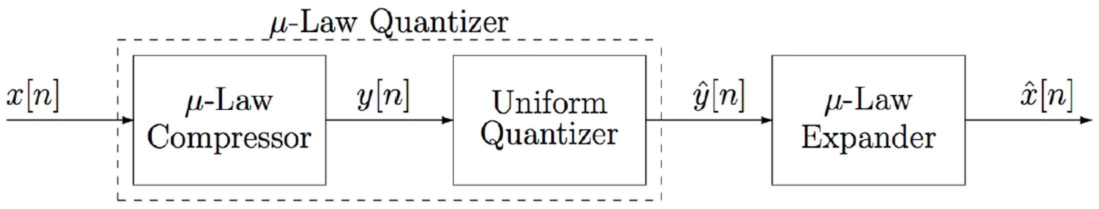
where `std2(x)` is the MATLAB function that calculates σ_x (i.e., the standard deviation of the values in vector x) and must be calculated for each audio signal. Analyze the results:

- Describe the difference between the theoretical and the experimental results. In order to gain some insight on the difference between both SNR plots, consider the histogram of the quantization error (study the histogram of the quantization error for N_b=5 and N_b=14). 2.1c
  Note: The theoretical expression assumes that the pdf of the quantization error is uniform.
- Give the number of quantization bits and the corresponding SNR value at which you start to perceive the distortion. Indicate, also, the number of bits and the corresponding SNR value at which you would say that the distortion is unacceptable. 2.1d

2.2. μ-law quantization of speech signals

To assess non-uniform quantization, we analyze the μ-law quantization comparing its performance with the uniform quantizer in terms of SNR and perceptual quality.

As we have seen in class, next figure illustrates a convenient way of implementing μ-law quantization. In this representation, a μ-law compressor precedes a uniform quantizer.



The μ-law compressor and μ-law expander are defined by the equations (both equations apply when the dynamic range of the signal is -1 to 1):

$$\begin{cases} y[n] = sgn(x[n])\,\dfrac{\ln(1+\mu|x[n]|)}{\ln(1+\mu)} \\ \hat{x}[n] = sgn(\hat{y}[n])\,\dfrac{1}{\mu}\left((1+\mu)^{|\hat{y}[n]|}-1\right) \end{cases}$$

The syntax of the MATLAB function that implements the μ-law quantizer for μ=255 (a standard value used in telephony) is:

```
[x_quantN, SNR, Qerr] = Audio_MuLawQ (x,Nbits,Fs)
```

Let us analyze this quantizer:

- Quantize both audio signals using the μ-law quantizer in the range from 1 to 16 bits and plot the experimental SNR as a function of the number of bits (N_b). Overlap in the same figure the SNR obtained with a uniform quantizer and compare the potential gain of the μ-law quantizer with respect to the uniform quantizer. Use the information given by the histograms to justify when the μ-law quantizer presents a relevant gain. 2.2a

- Listen to the quantized signals for N_b=5 bits (comparing the perceptual distortion with respect to the uniform quantizer) in order to report the perceived quality and its gain.

- Compare the differences on the quantization noise between the uniform and the μ-law quantizers. Plot the quantization error histograms for both quantizers with N_b=5 and comment the results. How do the dynamic range and the power of the quantization noise change? 2.2b