Principles of Biosignals and Biomedical Imaging

Master on Biomedical Engineering

P_3, 2nd Semester 2022/2023

Filtering

Lab Session 2

João Sanches
Bioengineering Department (DBE)
Instituto Superior Técnico / Universidade de Lisboa

In this work two types of filters, one linear and other non linear, will be used to attenuate or, if possible, to remove two types of noise corrupting audio and image data; *Additive white Gaussian Noise* (AWGN) and *salt and pepper Noise* (SPN).

In the first part the SPN is used to simulate the traditional noise present in the old vinyl long play records. This type of noise, non additive nor Gaussian, is characterized by maximum and minimum saturated samples at random instants. Removing of this type of noise using linear filtering is usually not efficient. Conversely, non linear filtering is usually quite efficient when dealing with this type of non-linear degradation process, e.g., the **median** filter.

The output of a basic 1D median filter is the median value of the set of values composed by the intensities of the pixels in the neighborhood of the current sample plus the current sample, that is, the median of the pixel set contained in the $2M+1$ dimension neighboring window. In this strategy the noisy pixels tend to accumulate at the extrema of the window which leads to its elimination.

I. 1D Filtering

1) Read the file $filtro.m$ that implements the following equation of differences

$$y(n) = \sum_{r=0}^{q} b_r x(n-r) + \sum_{k=1}^{p} a_k y(n-k). \tag{1}$$

The routine should be called with the following command: $y = filtro(x, A, B)$; where $A = [a_1, ..., a_N]^T$, $B = [b_0, b_1, ..., b_M]^T$ are vectors of coefficients and x and y are the input and output signals (1 D column vectors) respectively. Explain how it works.

2) Generate a square wave of $T = 10$ seconds with a period of $T_p = 1$ second and a sampling frequency of 50 Hz. Filter the signal with the following coefficients
   a) $A = [0], B = ones(10, 1)/10$
   b) $A = [0.75], B = [0.25]$
   c) $A = [0], B = [1, -1]$

   Comment the results.

3) For each audio file provided, *fungeesAWGN.wav* corrupted by AWGN and *fuggesSaltPeper.wav* corrupted by SPN, make the following
   a) Read the audio file, listen it and represent it graphically using the MatLab function $plot()$. Can you identify, in the plot, the noise corrupting the music (you can read the original file, *fugges.wav* for comparison purposes)?

   b) Use the routine implemented in question 1) to filter the signal/music with a linear low-pass filter with the following coefficients $A = [0.75], B = [0.25]$.
   Comment the results, namely concerning the quality of the audio output signal.

c) Filter now the audio signal with the median filter, by using the following command: $y = medfilt(x, N)$; where $N = 3$ is the window dimension. Listen the result and compare it with the one obtained with the linear filter.

d) Comment the results.

II. 2D Filtering

1) Load the image *len_gray.jpg* and corrupt it with salt and pepper noise for different amounts of noise (use the MatLab command *imnoise*). Visualize the noisy image with the *imagesc* and *mesh* functions. Comment.

2) Filter the original image *len_gray.jpg*, using the MatLab funtion $conv2(x, H)$ with the following masks for different values of N, e.g. $N = 5, 10, 50$
   a) $H = ones(1, N)$
   b) $H = ones(N, 1)$
   c) $H = ones(N)$

   Comment.

3) Filter the image by convolving it (use the MatLab function *conv2*) with a Gaussian mask (use the MatLab function *window* to generate the mask) for different values of the parameters of the window. Comment the result with respect to the effectiveness of noise removal.

4) Filter the image using a 2D median filter (use the MatLab function *medfilt2*) and compare the results with the one obtained in the previous item. Test with several values of the parameters. Comment.

5) Convolve the original image *len_gray.jpg*, $\mathbf{f} = \{f_{i,j}\}$, with the following 2D masks (use the *MatLab* function *conv2*)

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{2}$$

   and compute and visualize the image

$$g_{i,j} = \sqrt{(G_x * \mathbf{f})^2_{i,j} + (G_y * \mathbf{f})^2_{i,j}}. \tag{3}$$

6) The previous filter is called *Sobel* operator. What type is this filter? Why? Comment?

7) Use the *MatLab* function *edge* with parameters *'sobel'* and *'canny'* with the same image. Comment and compare the results and correlate them with the ones obtained in the 4) and 6).