Project Group 3

Grant Brown, Thomas Warren, Rick
Digital Signal Processing
University of Utah

INTRODUCTION

- Project Group 3 focused on design of FIR filters and their applications for performing deconvolution and image processing
- The project consisted of two labs
  - Lab 9 - Deconvolution with Audio and Image Processing
  - Lab 10 – Image Classification
- Lab 9 focused on deconvolution and echoing using FIR filters, the results are demonstrated on audio and image processing fronts
- Lab 10 utilizes FIR filters to perform discrete derivates which can be used to perform edge detection in images

FIR Filter Background

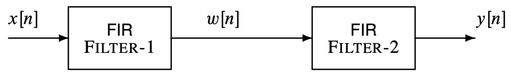
$$y[n] = \sum_{k=-\infty}^{\infty} b_k[w[n-k]$$

- Ideal filters require non-casual infinite impulse response (IIR) systems, which are not physically realizable
- Finite impulse response (FIR) systems are created by windowing IIR systems
- To ensure the output signal is still identical to the desired input components after filtering we need linear phase:

$$Y[n] = X(w)H(w) \rightarrow H(w) = \begin{cases} Ce^{-jwn_0}, & w_1 < w < w_2 \\ 0, & otherwise \end{cases}$$
$$y[n] = Cx[n - n_0]$$

- If an FIR filter has linear phase its unit sample response, $\{b_k\}$, satisfies the symmetry property:

$$h[n] = \pm h[M-1-n], \qquad n = 0,1,\ldots\ldots,M-1$$



Deconvolution

$$w[n] = \sum_{k=-\infty}^{\infty} h_1[k][x[n-k]$$
$$y[n] = \sum_{k=-\infty}^{\infty} h_2[k][w[n-k]$$
$$h_{total} = h_1[n] * h_2[n] = \delta[n]$$

Discrete Derivative

- Two difference operators:

$\Delta f = f[n+1] - f[n]$ or $\Delta f = f[n] - f[n-1]$

Contribution

- Lab 9 – Grant and Thomas
- Lab 10 – Rick
- Poster – Grant
- GitHub – Rick, Grant, and Thomas
- Final Submission – Rick, Grant, and Thomas

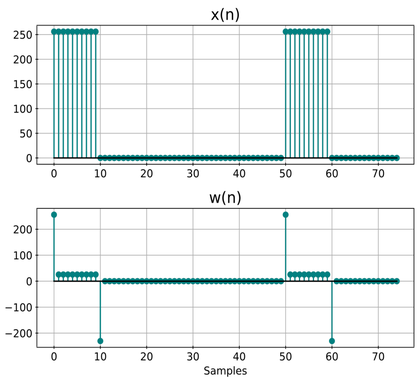
https://github.com/GrantBrown1994/DSP_Project

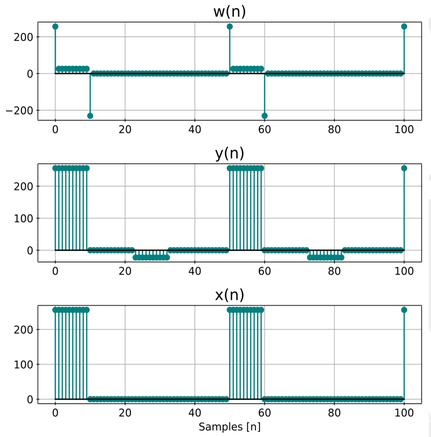
LAB 9

Echo Deconvolution Image Processing – Transmission mediums can cause unwanted effects, such additive WGN. To overcome this FIR filters can be used to perform deconvolution, giving the input signal at the output, which is very important in communication systems such as software radios. This lab focuses on the deconvolution process and how it can be applied to images.

Destructive Filter

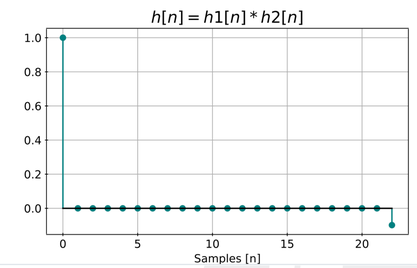
$$w[n] = x[n] - 0.9x[n-1]$$
$$w[n] = \sum_{k=-\infty}^{\infty} h[k][x[n-k] \rightarrow h[k] = [1, -0.9]$$
$$\boldsymbol{len}(y[n]) = \boldsymbol{len}(h[n]) + \boldsymbol{len}(x[n]) - 1$$





Restoration Filter

$$y[n] = \sum_{l=0}^{M} r^l w[n-l] \rightarrow r = 0.9, M = 22$$

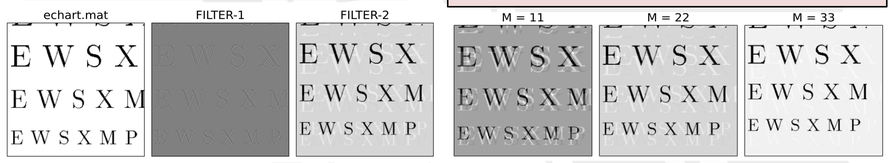


Echo Filter

$$y[n] = x[n] + rx[n-P]$$
$$\tau = 0.2\,s, f_s = 8\,KHz,\ Echo = 90\% \rightarrow$$
$$y[n] = x[n] - 0.9x[n-1600]$$
$$b_k = [1; zeros(1599,1), 0.9]$$

Total Deconvolution Response

$$W[z] = X[z](1 - 0.9z^{-1})$$
$$H_1(z) = \frac{W[z]}{X[z]} = (1 - 0.9z^{-1})$$
$$r[n] = r^n(u[n] - u[n-22])$$
$$H_2[z] = \sum_{n=0}^{\infty}\left(\frac{0.9}{z}\right)^n - 0.9^{22}z^{-22}\sum_{n=0}^{\infty}\left(\frac{0.9}{z}\right)^n$$
$$H_2[z] = \frac{1}{1-0.9z^{-1}} + \frac{0.9^{22}z^{-22}}{1-0.9z^{-1}}$$
$$H_{1,2}[z] = H_1(z) * H_2[z] = 1 + 0.9^{22}z^{-22}$$
$$h_{1,2}[n] = \delta[n] + 0.9^{22}\delta[n]$$



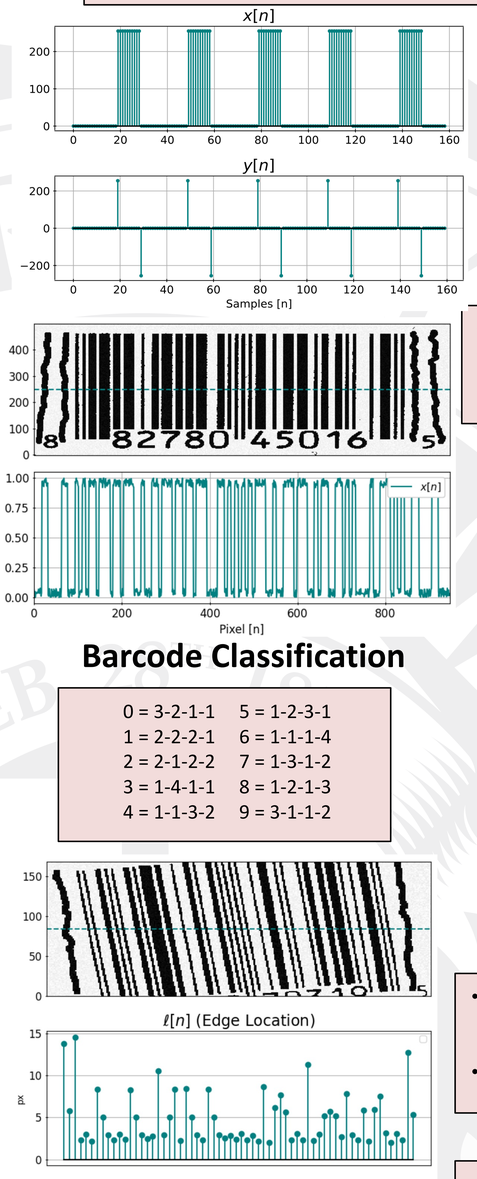
- Increasing M causes a further delayed echo, however at a further level of attenuation
- The gray-scale display uses values between 0 and 255 to differentiate between different shades of white, gray, and black
- Performing error analysis of the images before and after the deconvolution process:
  - M = 11 → Maximum Error = 160.671
  - M = 22 → Maximum Error = 50.420
  - M = 33 → Maximum Error = 15.822

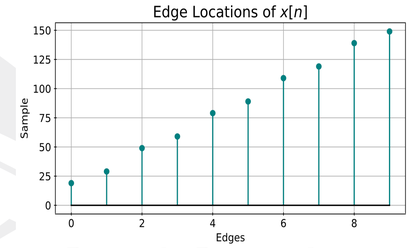
LAB 10

Image Detection Lab – Use first order derivate FIR filter to find edge transitions, from the edge transitions we can determine bar length and the associated message

Difference Filter

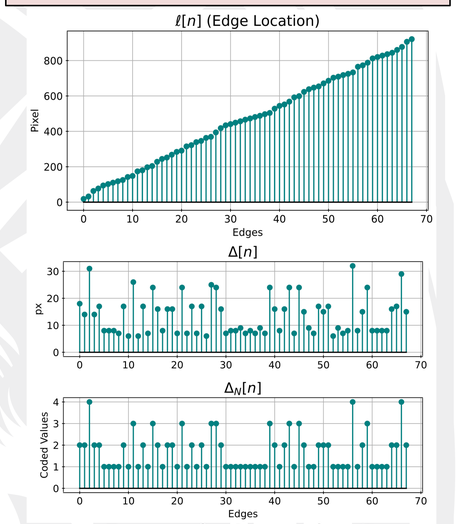
$$y[n] = x[n] - x - x[n-1] \rightarrow y[n] = \sum_{k=-\infty}^{\infty} h[k][x[n-k] \rightarrow h[k] = [1, -1]$$





Barcode Processing

- Barcode pixels show 1 for black, 0 for white
- Applying the difference filter and using a threshold for the derivate one can find black and white stripe pixel locations



Barcode Classification

0 = 3-2-1-1	5 = 1-2-3-1
1 = 2-2-2-1	6 = 1-1-1-4
2 = 2-1-2-2	7 = 1-3-1-2
3 = 1-4-1-1	8 = 1-2-1-3
4 = 1-1-3-2	9 = 3-1-1-2

Barcode Length

- Each Barcode is 12 numbers, and the classification gives $7\theta_1$ per number, where θ_1 is smallest bar pixel width
- Total Barcode Width = $84\theta_1 + 6\theta_1 + 5\theta_1 = 95\theta_1$

Barcode Length Averaging

- The image is slanted, therefore if image is deciphered using one row the results can be incorrect due to edge elongation
- Grabbing multiple rows and averaging gives an approximation of the row length which can be used to decipher the message