Lab 5

Digital Filtering II

Authors:

Jonas Lussi, Tarik Rifai, Naveen Shamsudhin, Franziska Ullrich, Kathrin E. Peyer, Dimitris Felekis, Bradley E. Kratochvil, Chauncey F. Graetzel, and Prof. Bradley J. Nelson

Institute of Robotics and Intelligent Systems

Date: 2019

Version: 2.0

Summary: These weeks you will design and implement two different first order FIR (Finite Impulse Response) low pass filters. You will learn how to determine the experimental frequency response of each filter. During the lab session, you will measure the frequency response of the filters and compare the results with the analytical solution. You will learn:

- Offline and online FIR filtering in **C**
- Digital filter design and analysis with MATLAB

5.1 Background

5.1.1 Digital Filtering

The background information for this lab is given in the lecture entitled **Digital Filtering**. This week you will continue to work in Digital Filtering by applying your filters (Moving Average and Blackman filter from last lab) to given signals and writing code to filter a signal online.

5.2 Prelab Procedure

Note: Pelab assignments must be done before reporting to the lab and must be turned in to the lab instructor at the beginning of your lab session. Additionally, you also have to upload your solution as a single PDF-file to moodle. Make sure to upload the file before your lab session starts, late submissions will not be corrected. The prelab needs to be handed in as a group. All the prelab tasks are marked with **PreLab Qx**

For the two algorithms we need in this lab, we would like to be able to sample the signals indefinitely and in real time. This makes the programming a bit more challenging due to having to keep track of the last M samples or develop a recursive implementation. In the lab procedure, you will first run the program over a finite series of pre-sampled measurements. This is useful to test if you understood the Blackman filter and have chosen sensible parameters. Afterwards, you will program your algorithms to filter indefinitely in real time.

1. Use the following Matlab code to analyze the frequency response of the filters.

```
%%
clear;
 close all;
 newplot;
 % How long is the sample kernel?
 M = 100;
 % What is the cutoff frequency?
 fc = .2;
 % This function builds a Blackman filter kernel
 % with cutoff frequency fc.
 blk = fir1(M,fc, blackman(M+1));
 % We also want a moving average filter
 % It can be of different lenghts than the Blackman
 N = 20;
 % This will not be as long as the Blackman filter
 % So we have to pad the rest of the vector with 0s
 avg = [zeros(M/2-N+1,1) ; ones(N,1)./N ; zeros(M/2,1)];
 % We'd like to plot multiple times on the same graph to show filter
 % kernels:
 hold on;
 plot(blk,'-b');
 xlabel('sample number')
 ylabel('Amplitude')
 plot(avg,'.-r');
 legend('Blackman','Moving Average');
 % This is the filter visualization tool: go on
 % Analysis->Analysisparemters and change Magnitude Display from
 % "Magnitude(dB)" to "Magnitude" to have a linear scale.
 fvtool(blk,1,avg,1);
 legend('Blackman','Moving Average');
```

   With the Filter Visualization tool, plot the Magnitude Response at different frequencies for your previously designed Blackman window (from prelab question 2, lab04). Then double the cut-off frequency and plot again the response. Do the same with the kernel length M. Finally, also plot the smoothing filter with N = 10 and 100. On your report, briefly describe the influence of these parameters (cut-off frequency, M, N) on the magnitude response. When you look at the magnitude response (linear scale), where do you see the two biggest disadvantages of the moving average filter (for your Prelab Report you don't have to hand in any plots)? **(PreLab Q1)**

2. State the sampling theorem and explain why it is important. **(PreLab Q2)**

3. Study the files of this week's lab from the course website

4. Print out your Prelab solutions and turn them in to the lab instructor at the beginning of your lab session. Also upload your prelab as a single pdf on the moodle platform.

5.3 Lab procedure

The skeleton code is available on moodle. Download the files on your Udoo and copy them into a folder called `irm/lab05`. If you already have code written from your last lab (e.g. `digital_filter.c` you can use this code again in this lab.

5.3.1 Digital Filtering

This laboratory assignment involves implementing a first order digital filter in **C** and comparing the results to the analytical frequency response that you determined in the prelab.

1. You are given 4 signals, a ramp and a sinus without (`ramp.txt`, `sinus.txt`) and with noise (`ramp_noise.txt`, `sinus_noise.txt`). All signals were sampled at 1 KHz.

- Write a C function `lab05_Task1.c` (provided as skeleton) that implements your filters on the noise free signals and saves the filtered data in a file. Call this function in your main file `lab05_test.c`, then plot for example in Matlab and comment on the differences between them. (Note: Account for the delays in the filters to be able to easily compare the output to the raw data) **(Postlab Q1)**

  Hint: To be able to read a file in C you must define a variable of type `FILE` which the code uses to access the file. For example, your file is called `myfile.txt` and you want the variable that access it to be named `mydata`. What you need to do is `FILE *mydata` in the variables declaration, `mydata=fopen("myfile.txt","r");` to open the file for reading and `fscanf(mydata,"%f%*c%f",&variable1,&variable2);` to read its contents. Do not forget to close the file by `fclose(mydata);`!

```
FILE *mydata;
double variable1[100], variable2[100];
mydata = fopen("myfile.txt","r");
for (j=0;j<length_array;j++)
    {
        fscanf(mydata,"%f%*c%f\n",&variable1[j], &variable2[j]);
    }
fclose(mydata);
```

- Vary the parameters of your filters to optimize the output for both of the noised signals at the same time (verify at what frequency the noise occurs and then use filter tool in Matlab to design adequate filter). Plot your results for one set of optimal filter parameters to compare the outputs for the ramp and sinus signal in terms of the rejected noise and distortion of original signal. What do you observe? Which filter worked better? Justify and elaborate your answer. **(Postlab Q2)**

2. Write functions to implement both the moving average and Blackman window as online filters in Arduino without passing data to the µP. **(Postlab Q3)**

3. Gather data indefinitely from your phone at around 1 kHz (use delayMicroseconds() function to achieve sampling frequency), filter it online, and output it to the oscilloscope. For the Blackman windowed sinc filter, use a cutoff frequency of 100 Hz and a Kernel Lenght of M = 200. Show the running system to the assistant. **(Postlab Q4)**

5.4 Postlab and lab report

Please hand in an organized report before your next lab. Upload a single PDF-file with your solutions to the moodle platform that includes:

- The plots from task 1 (Postlab Q1 and Q2). Which filter is more useful to filter the noise and why?
- All the source code you wrote from tasks 1 and 2 (Postlab Q1, Q2 and Q3)
- After you upload the PDF-file on moodle, print it out and hand it in to the assistant at the beginning of the next lab session.
- Make sure you have time to show your working system in your next lab session(Task 3, Postlab Q4) to your assistant.

5.5 Literature Reference

[1] Smith, Steven W., "The Scientist and Engineers Guide to Digital Signal Processing." California Technical Publishing, Sand Diego, CA 1997-1999.