GSOC'12: Extend and improve filter design components

Sreeraj Rajendran

April 13, 2012

1 Introduction

Digital filter is one of the inevitable signal processing blocks in any DSP system. GNU Radio provides a true free and open platform for designing practical filters which can be used in real time. The proposal addresses ideas for improving filter design components as suggested by Martin Braun (CEL, KIT). Section 2 covers basic details about digital filter designs and the filters currently available in GNU Radio. A small review of the feature additions suggested by the mentor is included in section 3 for a better understanding. Proposed improvements are detailed in section 4 of the document. A rough project schedule is given in section 5. The proposal is concluded in section 6.

2 Digital filters and GNU Radio filters

Digital filters have a long design history from 1960s onwards. The major filter design techniques include Butterworth filter design, Chebyschev approximation, Remez exchange algorithm, Parks-McClellan algorithm (based on remez exchange algorithm) etc. Most of the algorithm classification is based on the nature of the passband and stopband (equiripple, monotonic etc). A brief explanation of some of these filter designs is covered in the implementation section.

GNU Radio provides two ways for designing digital filters. First one by making use of 'gr_firdes' which returns taps for windowed FIR filters. The second method uses 'optfir' module which calls 'gr_remez' function which implements Parks-McClellan algorithm for finding optimum filter taps. The filter taps thus obtained are given to the filter blocks (gr_fft_filter_ccc etc) which does basic convolution in time to complete the filter functionality. The program 'gr_filter_design.py' is a QT frontend which makes use of these modules to design the filter.

3 Suggested feature improvements

A summary of the feature improvements suggested by the mentor are listed below

- **Estimation of number of taps with the Parks-McClellen algorithm:** The estimation of the order of the filter for given requirements (passband, stopband, magnitude, allowable magnitude deviations etc) is always crucial.
- **Quality Assurance codes to test design algorithms:** Test codes need to be updated for all filter designs to make sure that the design algorithms are robust.
- **Design tools for IIR filters:** Currently there are no design tools available in gnuradio to support design of IIR filters, half band filters etc.
- **Pole-zero plots:** Pole zero plots will be very useful in understanding the rationality and stability of the final digital filter.
- **Improved GUI look and feel and an interactive GUI:** This is the pre-final stage of project and this includes addition of filter responses and other features to the graphical user interface once the filter design is complete. Filter responses gives us an idea about how well the designed filter performs.
- **GRC integration for gr_filter_design.py:** In the final stage the designing tool should be integrated to GRC so that the filter taps and other values can be imported to filter blocks directly.

4 Proposed work during GSOC'12 period

Proposed improvements for GNU Radio filters are listed below on priority basis.

- **QA test codes for gr_remez design module:** In GNU Radio Parks-McClellan algorithm is implemented in 'gr_remez.cc' which is used for optimum filter designs. This module lacks quality assurance tests. QA tests are available for FIR design module (gr_firdes) which do basic coefficient checks and tap symmetry checks. The 'optfir' module makes use of 'gr_remez' to design different filters. QA tests will be added for 'gr_remez' and associated 'optfir' module.

- **Design tools for IIR filters:** FIR filters over-perform IIR filters when control of phase and instability due to numerical errors are considered. IIR filters usually requires fewer filter coefficients for the same filtering action when compared to FIR filters and hence it is faster and requires only less memory space. IIR filters are useful in designs where linear phase response characteristics are not required (e.g signal amplitude monitoring applications). Figure 1 gives a general design procedure for IIR filter design. Sample code for IIR and FIR filters, their pole zero plots and documentation can be found in the repository (git://github.com/zeroXzero/dsp_filter_design.git). IIR filter design module along with QA tests will be added and improvements will be made in gr_filter_design.py for designing IIR filters, displaying their pole-zero plots etc.

- **Improve GUI for filter response comparisons:** Filter responses can be plotted to analyze designed filters. A feature shall be added to the current gr_filter_design.py tool to compare different filter responses (e.g one using firdes module and one using optfir module). It is desirable to add filter time responses like impulse and step response for the designed filter taps. These two features will be added to the GUI designing tool.

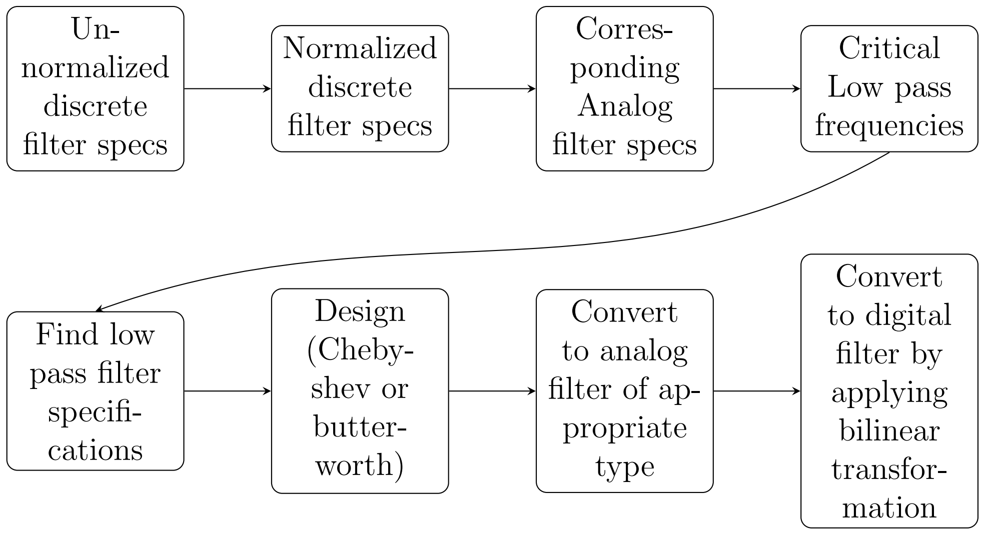
Un-normalized discrete filter specs
Normalized discrete filter specs
Corres-ponding Analog filter specs
Critical Low pass frequencies
Find low pass filter specifi-cations
Design (Cheby-shev or butter-worth)
Convert to analog filter of ap-propriate type
Convert to digital filter by applying bilinear transfor-mation

Figure 1: General filter design procedure

5 Timeline

As per the GSOC schedule there are 18 weeks including midterm evaluation, final evaluation, documentation and code clean-up period. This is just a rough estimate which is very likely to change after discussions with the mentor and the coding time may vary depending on the coding complexity.

- **April 25 - May 4** – Initial discussions with the mentor
- **May 5 - May 11** – Learning period- Parks-McClellan algorithm and basic GNU Radio QA tests.
- **May 12 - May 18** – Coding - QA tests for gr_remez module.
- **May 19 - May 25** – Coding - QA tests for gr_remez module.
- **May 26 - June 1** – Coding - QA tests for optfir module.
- **June 2 - June 8** – Coding - IIR filter design module.
- **June 9 - June 15** – Midterm evaluation submission includes QA tests and initial IIR filter design code

- **June 16 - June 22** – Coding - IIR filter design module.
- **June 23 - June 29** – Coding - IIR filter design module.
- **June 30 - July 6** – Coding - QA tests for IIR filter module.
- **July 7 - July 13** – Coding - QA tests for IIR filter module.
- **July 14 - July 20** – Integrate IIR filter design code to GUI (gr_filter_design.py).
- **July 21 - July 27** – Add pole-zero plots for IIR filters to GUI.
- **July 27 - Aug 3** – Add filter time responses to GUI.
- **Aug 4 - Aug 10** – Improve gr_filter_design.py which will help to compare different filter designs.
- **Aug 11 - Aug 20** – Integration, code clean-up, adding documentation, and final submission.

6 Conclusion

A general overview of the Filter design enhancement project is given in the previous sections. A sample codeset for FIR and IIR filter designs are also provided in the repository. The project is divided into proper subsections so that the mentor and community can easily track the progress of the project.

References

[1] Theory and Application of Digital Signal Processing, *Rabiner L R.* 1975.

[2] GNU Radio Documentation *http://gnuradio.org/doc/doxygen/modules.html*

[3] Digital Signal Processing and its Applications (EE-603) lecture notes by Prof.Vikram M. Gadre *http://www.ee.iitb.ac.in/wiki/faculty/vmgadre*