GNU Radio - Radar toolbox
GSoC 2014 proposal

Stefan Wunsch
stefan.wunsch@student.kit.edu

March 21, 2014

Contents

1 Introduction

Radar is a highly diverse radio technology for object detection, ranging and velocimetry and is fundamental in a wide range of everyday-applications such as automotive systems, weather forecasts or traffic monitoring and control. Radar signal processing and waveforms go hand-in-hand and can only deliver maximum performance when developed together. Thus, SDRs (Software Defined Radios) are the ideal development platform for radar design.

The purpose of this project is to develop an OOT (Out Of Tree) module for the GNU Radio project that provides a generic environment in GNU Radio to experiment with various radar types. Furthermore some of the most commonly used radar processing algorithms shall be implemented. Viable algorithms include FMCW (Frequency Modulated Continuous Wave), FSK (Frequency Shift Keying) and OFDM (Orthogonal Frequency Division Multiplexing) radar. USRPs (Universal Software Radio Peripherals) from Ettus Research with UHD (USRP Hardware Driver) interface and adequate daughterboards are intended to be used for transmission and reception of real signals.

The release of the new USRP X series [1] makes this project even more promising due to 120 MHz of baseband bandwidth which provides improved range resolution up to 1.25 m [2] and frequencies up to 6 GHz, which allow for small beamwidths and operation e.g. in the 5.7 GHz ISM Band.

Figure 1 shows the basic idea of the radar toolbox. In general there are three main parts. First of all there is a need for suitable combinations of signal generators and estimators. These blocks provide the transmitted signal for the hardware and combine it with the received signal in order to extract range and velocity information. This information is displayed in the GUI (Graphical User Interface), which is part of the toolbox. An intended visualisation form is a range-velocity diagram. The third part is the implementation of USRP support. In addition the replacement of the hardware with a simulator block for testing and evaluation purposes will be provided.

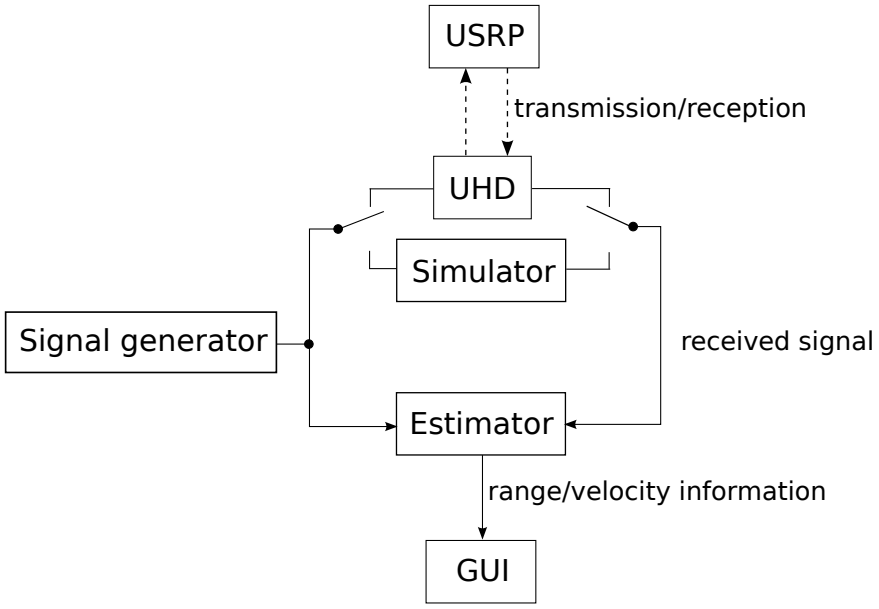

Figure 1: Generic flowgraph of the radar signal processing

2 Benefits

The GNU Radio project benefits from this project in various ways. First of all, as mentioned before, radar is an excellent technology to make use of SDR. Software defined signal processing allows to unify many kinds of radar types in one device. This provides on the fly switching capabilities and can be adapted to the given situation in no time. Doubtless, having such a powerful technology easy accessible as OOT module is desirable.

The toolbox will be as generic as possible. Not only the capability to deal with different modulation types is important but also easy extensibility for MIMO (Multiple Input Multiple Output) Radar, which both will be considered. So there are numerous possibilities to use the toolbox for other projects.

A large group of engineers, students and scientists work and study in the field of radar. This project can give them easy access to the GNU Radio project through a comprehensible signal flow, ready-to-use GUIs and plug&play hardware. The toolbox should provide a simple way to get a working demonstrator of an exciting technology with real world interaction.

3 Deliverables

Easily expandable radar toolbox The main objective is the development of a generic, comprehensible and easily extensible radar toolbox for GNU Radio. This project is intented to be the basis for further development after GSoC has ended. The different components of the toolbox are listed below.

Signal generator and estimator blocks These are responsible for the radar signal processing with respect to a selected modulation type. The emphases are on realtime capability and the development of highly reusable blocks. Possible estimator algorithms besides range, velocity and angle estimation are moving target indication, ghost target reduction and radar cross-section estimation. Especially for estimators hierarchical blocks are a proper choice to allow the maximum reusability of the underlying blocks. Furthermore high modularity is a design goal as it eases modification of the estimator algorithms. E.g. OS-CFAR [3] would be a reliable algorithm for multi peak detection.

USRP support This block transmits the given signal from the signal generator via USRP and delivers received samples to the estimator. Special care has to be taken to ensure proper synchronization between transmit and receive path. For this purpose tagged streams are a suitable choice. All USRP series will be supported. Additionally it will be possible to connect multiple USRPs for phase analysis and angle resolution.

Simulator The simulator can be used alternatively to the hardware interface and will have the same ports to ensure compatibility. Its function is to emulate propagation effects such as time and frequency shifts. Multiple target simulation will be supported. This allows performance tests of the implemented algorithms under defined circumstances and radar signal processing development without the need of hardware.

Visualisation A comprehensible and clearly represented visualisation is mandatory for every successful demonstrator. The presentation of the estimator data with range-velocity diagrams is intended. Another interesting visualisation would be a microdoppler representation with waterfall diagrams, e.g. used in object classification. Preferred GUI implementation is Qt with Python [4]. As communication mechanism between estimator and GUIs the message passing API will be used. Importance will be attached to a consistent interface across estimators so that GUIs are not dependent on particular estimators.

Documentation For sure the availability of a useful documentation is one of the most important parts of a software project. Therefore a comprehensive documentation will be provided with the code. Furthermore the official coding style [5] will be used for easier reading of the source.

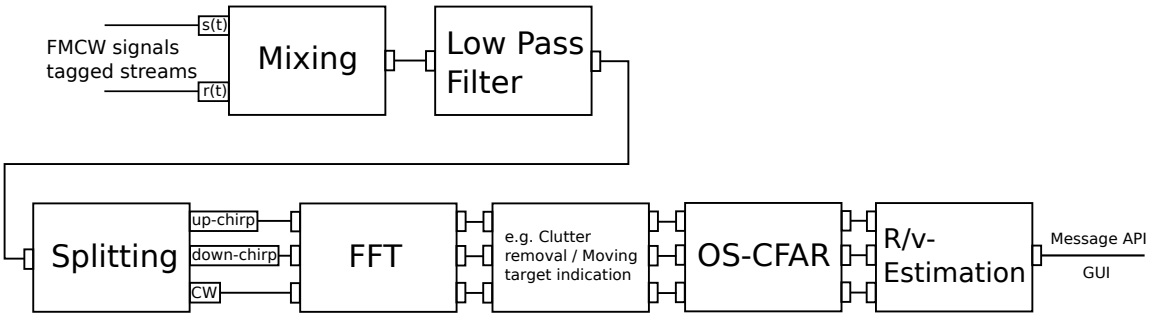

Figure 2: FMCW estimator flowgraph

4 Timeline

Below a preliminary project timeline is shown. It is oriented at the official GSoC 2014 timeline [6]. Changes after initial discussions are likely.

Python will be used for the rapid prototyping of blocks. After successful validation the timing-critical parts will be ported to C++ in an iterative manner. Every block will be delivered with QA code.

April 22 - May 6 Discussion of project details with my mentor. Define final signalflow including block interface specifications and conceptual GRC (GNU Radio Companion) flowgraph. Create a finer timeline with milestones and get familiar with so far unused features of GNU Radio.

May 7 - May 15 Start with CW signal generator and estimator to get a working interface. OS-CFAR will be used for peak detection block. Add simulator with frequency-shift capability for testing.

May 16 - May 29 Going on with FMCW signal generator and estimator. Intended FMCW modulation is split up in up-chirp, down-chirp and CW. The waveform parameters such as rise time and sweep bandwidth will be fully configurable. CW estimator is reusable in large parts. The FMCW estimator flowgraph is shown in figure 2. Extend simulator with time-propagation effects for testing. Time-shifts of fractional samples will be fully supported. Get simulator-based system working.

May 30 - June 20 Add USRP support. Accurate synchronization of multiple devices is fundamental. Get demonstrator with USRP X300 working. Ranges up to 50 m are intended in first test scenarios. Additional testing is likely.

June 21 - July 3 Implement visualisations with Qt and Python. Start with range-velocity diagrams and possibly add waterfall plot for microdoppler representation.

July 4 - July 18 Implementation of second modulation scheme as usability proof of the framework. OFDM radar is a proper choice due to the availability of numerous blocks, e.g. modulation and demodulation of an OFDM stream. Thereby the integration of the toolbox in the GNU Radio ecosystem is demonstrated.

July 19 - August 5 Write the documentation. Additional testing and probably implementation of new minor features, e.g. data logging in CSV files.

August 6 - August 21 Clean up the code and improve documentation. Submit code samples to Google before pencil-down date.

5 Qualification

I am a fifth semester student in Physics with minor field of study in Computer Science at the Karlsruhe Institute of Technology (KIT). Additionally I have been working as a student research assistant at the CEL (Communications Engineering Lab) since June 2013. My focus is on FMCW and FSK radar signal processing, clutter detection and object classification with radar. Main development environment has been MATLAB [7]. Due to my work at the CEL I am always in close contact to our local experts who are involved in the GNU Radio and radar development.

Based on the excellent reputation of the GNU Radio project at the CEL I have made my first contact with GNU Radio early on and acquired the needed knowledge for creating new modules and blocks. First simple radar simulators have already been written. Code samples are provided in my Github repository [8].

Besides I have experience with C/C++ and Python. For example I have made use of the particle physics data analysis library ROOT [9] developed by CERN frequently. Furthermore I am looking forward to enlarge my knowledge about data analysis in my favored master program.

My personal motivation for this proposal is the opportunity to develop and be part of an exciting application for software defined radio with GNU Radio. Especially the combination of software signal processing with configurable hardware makes this approach to radar very powerful and interesting. I have not contributed to an open-source software so far and this would be my biggest

development project yet. But I am looking forward to enhance my coding skills, development strategy and understanding of radar steadily.

During GSoC I have to take a few classes but these are not graded and I do not have to write exams after this semester. Also there are no times I will be on vacation or otherwise unavailable. GSoC would be my major task during this period. Therefore I think GSoC would fit well into my schedule.

I would appreciate to be part of the GNU Radio community and to maintain this project even after the pencil-down date of GSoC 2014.

6 Conclusion

I think radar for GNU Radio is a great project with an extraordinary wide range and number of applications, which can be realized in short time with an adequate toolbox.

Participating in the GNU Radio project with this OOT module would be an exciting experience for me. I hope you were able to get an impression of the project. In case you have any questions or suggestions please feel free to contact me.

References

[1] https://www.ettus.com/product/category/USRP-X-Series
March 15, 2014

[2] M. Richards, Principles of modern radar: basic principles. SciTech Pub., 2010.

[3] Blake, S., OS-CFAR theory for multiple targets and nonuniform clutter, Aerospace and Electronic Systems, IEEE Transactions on , vol.24, no.6, pp.785,790, Nov 1988

[4] http://www.qt-project.org/
March 19, 2014

[5] http://gnuradio.org/redmine/projects/gnuradio/wiki/BlocksCodingGuide
March 19, 2014

[6] https://www.google-melange.com/gsoc/events/google/gsoc2014
March 15, 2014

[7] http://www.mathworks.de/products/matlab/index.html
March 15, 2014

[8] https://github.com/stwunsch/gsoc-proposal

[9] http://root.cern.ch/drupal
March 15, 2014