Google Summer of Code - 2018

Adding passive radar capability to gr-radar toolbox

Suraj Hanchinal

Introduction

Radar toolbox which was developed by Stefan Wunsch in GSoc 2014 was a great success and included a variety of tools such as target simulator as well as OFDM signal support. This toolbox was a great addition to the GNU Radio OOT modules. The toolbox mainly provided support for active radar.

Even though the initial toolbox is impressive, it is lacking functionality such as passive radar support and support for devices other than USRPs. Adding support for devices like RTL-SDR and WiFi routers might increase the functionality of the toolbox.
Passive radars are generally easier to setup and use as you do not need a powerful transmitter or a permit and you just need a device to receive the radio waves and process them. This ease of use can translate into better usability and better applications.

My proposal consists of implementing passive radar capabilities along with their simulation to the gr-radar toolbox and adding support for devices other than USRP such as RTL-SDR for passive radar. It will also allow you to use WiFi signals for indoor target detection with a specific algorithm called CLEAN for the same [1].

Section 1 describes the overall structure of the project and the flow of the passive radar. Section 2 describes all the individual components in detail. Section 3 describes the deliverables to be expected at the end of GSoc ‘18. Section 4 contains the timeline for the entire project and Section 5 lays out my qualifications.

Section 1

Structure of the project

Signal Recovery and Synchronization block: This is the first step of the flow of passive radar. The signal source can be anything from a virtual signal source to a RTL-SDR to a proper software defined radio. The signal may be single antenna or multiple depending on the type of algorithm being used. This block also synchronises the signals and corrects the time shift due to hardware delays.

Passive radar Algorithm block: These blocks take in the acquired signals above and process them. The algorithms for this are described in the next section.

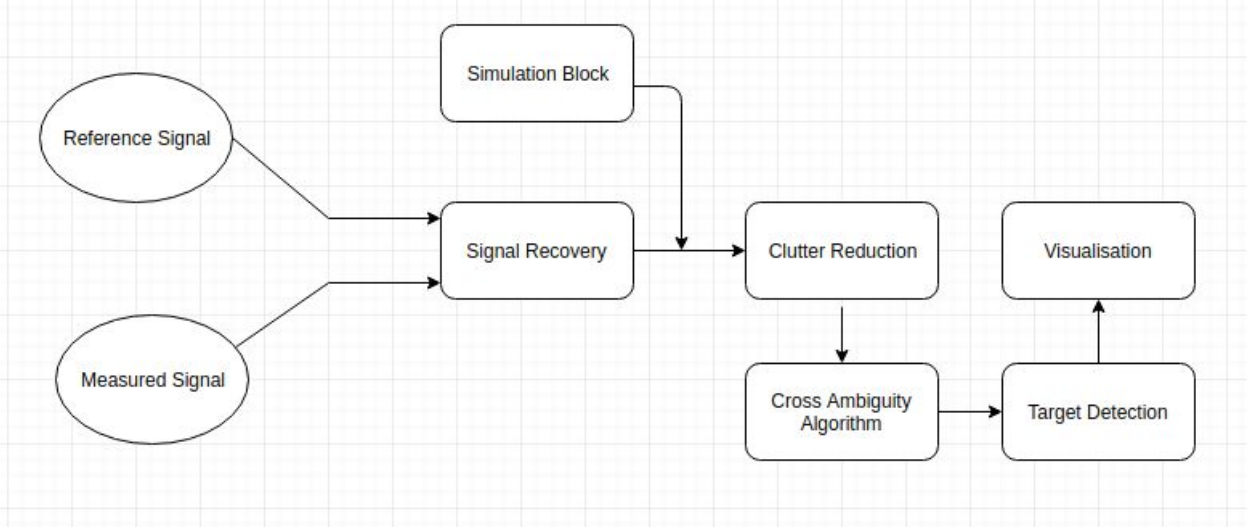
Clutter Reduction block: This algorithm takes the reference signals and measured signals and removes direct signal interference as well static target clutter and outputs a cross-ambiguity surface with minimal clutter.

Target Detection block : Once the clutter has been removed, this block applies a 2-D CFAR filter which isolates the moving targets.

Visualisation: The data from the target detection block with isolated targets is plotted using the visualisation tool. This may include target paths or just doppler diagrams.

Simulation: A very important feature of the original toolbox was the target simulator block which used the point-target radar model to simulate the RX signals. A simulator will be built for the passive radar with features like multiple receiver antennas and multiple TX “illuminators”. For example, you could simulate a triangulation using 3 fm stations with a receiver antenna using the simulator.

Flowchart for Passive radars



Section 2

Signal Recovery

This is the first step of the entire passive radar tool. Since for a passive radar system, you usually need two input signals, a reference signal and a measured signal. The signals are acquired from 2 different devices for example, 2 different RTL-SDRs. Due to various reasons such as hardware delay, the signals are not always in sync.

In the research paper by JM Friedt [1] , he observed that for RTL-SDRs there is a constant delay between the two signals as long as the datastream isn't stopped or restarted. A similar trend was observed by Stefan Wunsch when he implemented the "Synchronisation echotimer" where the time delay was constant.

With this in mind, one method to address is to use the disadvantage that is the direct signal leaking into the measured signal to our advantage to measure this delay. Since there is a very strong component of the the direct signal in the measured signal and the antenna are generally very nearby, we can use the cross-correlation function to determine the time delay between these two signals while ignoring the doppler shift etc.

$$\gamma(\tau_{delay}) = \int_{-\infty}^{+\infty} ref(t + \tau_{delay}) \times meas^{*}(t)dt$$

$\gamma(\tau)$ is the cross-correlation function. By searching for a τ_{delay} for which $\gamma(\tau)$ is above a certain threshold, the time delay can be found.

Algorithm for Passive Radar (Cross Ambiguity)

Passive radar unlike an active radar deals with a lot of incomplete information like where is the transmission source? how far is it? What modulation is it using ? To counter these problems, there are usually two antennas, one of which is pointed towards the transmitter. The received signal on this antenna is called the **reference signal**. The other antenna is pointed towards the object(s) of interest. The signal acquired by this antenna is called the **measured signal**.

After the acquiring of these two signals, We compute the cross-ambiguity function of the signals.

$$\gamma(\tau, f) = \int_{-\infty}^{+\infty} ref(t+\tau) \times meas^*(t) \times exp^{(2\pi f t)} dt$$

The cross ambiguity is a two-variable function which is a product of conjugated multiplication of the measured signal with the time shifted and frequency shifted copies of the reference signal. The frequency shift is caused by doppler effect of moving targets and the time shift is caused by the radio waves travelling from the "illuminator" and back.
Thus higher value of the cross-ambiguity function means that the shifted reference signal and the measured signals are more alike.
From the values of the cross ambiguity function, a plot of distance vs. frequency is plotted. This is the Doppler Diagram.

So how do we get the distance as well as the velocity?

$$d = c \times \tau \qquad\qquad f = 2 \times (\tfrac{V}{c}) f_o$$

d = distance c = speed of light τ = time shift V = velocity of body

f = frequency shift f_0 = Frequency of transmission.

Clutter Removal Algorithms

The performance of passive radar systems is limited by the strong interference of the direct signal in the measured signal. This can be quite clearly seen at the center of the doppler diagram at 0 Hz. This interference is known as Direct Signal Interference(DSI). The other reason for limited performance is the presence of large static targets like buildings and mountains which might lead to weaker targets being invisible in comparison. There are different algorithms for the removal of these two interferences.

1. Extensive Cancellation Algorithm (ECA)

This algorithm is used to mainly remove the DSI from the measured signal. This algorithm is an optimization algorithm that searches for the signal with the minimum power from the reference signal as possible.

$$meas_{corr}[t] = meas[t] - \sum_{k=0}^{K-1} \alpha_k \times ref[t-k]$$

The clutter removed signal $meas_{corr}(t)$ is obtained by subtracting a linear combination of k time delayed signals of reference signals.

matrix B is defined as the column matrix of the K time delayed reference signals.

The solution of the optimization can be obtained by basic linear algebra.

$$meas_{corr}[t] = (I - B(B^tB)^{-1}B^t)meas[t]$$

As you can see the algorithm is quite computationally expensive. Also this algorithm is not that effective in removing the static clutter. The reason for this is that it only minimizes power of the reference signal, but never takes into consideration the measured signal and how that is changing. The static target contribution in the

measured signal is very static in the sense that it changes very less. For this we will explore different algorithms.

2. Sequential Cancellation Algorithm (SCA)

The previous algorithm had very high computational complexity. This can be avoided by iterative calculation of the cross-ambiguity function. This is similar to the Extensive Cancellation Algorithm but runs much faster.

The matrix B in the previous algorithm is split in the following way

$B = B_0 = [b_0\ B_1]$ where b_0 is the first column of the matrix B.

And P_1 is defined as $P_1 = (I_N - B_1(B^H{}_1 B_1)^{-1} B^H{}_1)$

And $P_o = P_1 - \frac{P_1 b_0 b^H{}_0 P_1}{b^H{}_0 P_1 b_0}$

Now $meas_{corr}[t] = P_0 \times meas[t]$

We define some helping variables.

$m_1[t] = P_1 \times meas[t] \qquad b'_1 = P_1 \times b_1 \qquad Q_1 = I - \frac{b'_0 b'^H{}_0}{b'^H{}_0 b'_0}$

With the help of the variables we can write

$meas_{corr}[t] = Q_1 \times m_1[t]$

And further this can be written as $meas_{corr}[t] = Q_1 Q_2 Q_3 \ldots . Q_k \times meas[t]$

This is obtained by stepwise decomposition of P_1 and therefore stepwise decomposition of B_1 into $B_1 = [b_1\ B_2]$ and so on.

3. CLEAN Algorithm

CLEAN algorithm is an iterative algorithm generally used in WiFi based passive radar. This algorithm is better suited for signals with high doppler range resolution and with large multipath interferences such as indoor environments. It was first used by radio astronomers to identify point sources of light from images of space. It has been modified to work with passive radar. It works on the following principle. This algorithm is better than ECA in the sense that it operates on the static targets which cause the multi-path interference as well as Direct Signal Interference.

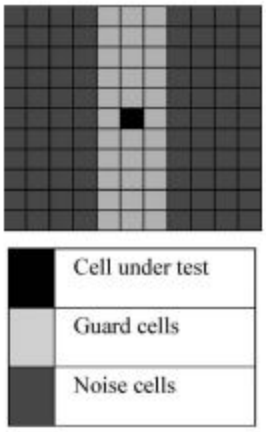
First, the cross-ambiguity function is calculated.
1. The peak of the cross-ambiguity function is chosen.
2. This peak or the strongest signal is subtracted from the cross-ambiguity function and the cross-ambiguity is calculated again.
3. Again the next peak is chosen and step 2 is repeated.
4. This is done until the value or the magnitude of the peak is below a certain threshold which is a parameter that we can set.

The subtraction of the peak and the subsequent recalculation of the cross-ambiguity function is where different methods are used. This can be done in a number of ways. This algorithm generally clears up the entire zero doppler interference in the Cross-ambiguity function. The paper followed for this particular implementation will be [2].
This algorithm is fairly popular for WiFi based passive radar applications.

Target Detection Algorithm

The target detection will be based on a basic 2-d constant false alarm rate detector. This involves the three following steps.

1. The dynamic range of the cross-ambiguity surface is reduced by thresholding the value of the surface.
2. A moving average filter is applied over the entire surface to smoothen the noise.
3. A point on the surface is chosen and the points around it are chosen in the way shown in the below image.
4. The average magnitude of noise cells is taken and a target is said to be present at the point if its power or magnitude is greater than the average magnitude of the noise cells.



Once the targets have been detected, their locus will be sent to the visualisation tool to be plotted.

Deliverables during GSoc 2018

The main objective will be implementing passive radar methods to the gr-radar toolbox. At the end of the coding period the deliverables will be:

1. Add passive radar support to gr-radar toolbox.
2. Add multiple device support such as RTL-SDR and USRP.
3. Add support for WiFi signals as signals of opportunity for passive radar and specific algorithms to make this possible.
4. Signal recovery and syncing block.
5. Cross-ambiguity calculation block.
6. A clutter reduction block with three implemented algorithms described above to remove clutter and isolate and detect moving objects.
7. A target detector with a CFAR filter to seperate moving targets from noise
8. A visualisation tool for viewing the results from the clutter reduction block such as the locus of the object path.
9. A simulator block which lets you add multiple illuminators as well as antennas to simulate passive radar as an extension to the current target simulator.
10. Documentation of the entire coding process as well as all of the tools.

The deliverables may be extended to add more algorithms or other blocks after further discussion with the mentor.

Timeline

The following is the timeline planned for the project and I will try very hard to follow all the set deadlines here. Documentation will be written alongside the development. Since there are a lot of modules in the toolbox, pull requests will be sent regularly and efforts will be made to merge them as the work on them is completed.

April 23rd to May 14th: I will familiarise myself with the GNU-Radio community as well as discuss the minute details of the project with my mentor. Since I have already gone through the codebase of gr-radar, I will also spend this time setting up a

testbench for experiments such as RTL-SDR setup since some of the features implemented need to be tested in real life.

May 14th to May 21st: The first week will consist of setting up a code structure and work is started on the signal recovery block as well as the cross-ambiguity block. Since this does not involve a lot of work, This will be finished with the same week.

May 22st to May 29th: The signal recovery block as well as cross-ambiguity block will be merged. Work on the ECA clutter reduction algorithm is started.

May 30th to June 5th: Since ECA algorithm has already been implemented many times, the work on it finished this week and is tested rigorously.

June 6th to June 13th: Pull request for ECA algorithm is submitted and is merged. Work will start on the target detection algorithm which is quite basic and will be finished in the same week. A GUI plotter for the isolated targets is also made for the same. Work will start on the simulator.

June 14th to June 21st: The multiple "illuminator" part of simulator is finished. Work is started on the multiple receiver antennas.

June 22nd to June 29th: Several example scenarios such as FM triangulation are added for people getting started with the passive radar. The simulator is rigorously tested. Work on simulator in completed and pull request is submitted for the same.

June 30th to July 6th: Work is started on the iterative SCA clutter reduction algorithm.

July 7th to July 14th: SCA algorithm is completed and pull request is sent. Work begins on WiFi based passive radar with testing the already implemented algorithms with WiFi routers as "illuminator" and USRP as receivers instead of RTL-SDR.

July 15th to July 22nd: Work begin on CLEAN algorithm for clutter reduction in WiFi based passive radar.

July 22nd to July 29th: CLEAN algorithm is implemented by the end of the week and pull request is sent.

July 30th to August 5th: Every component is rigorously tested and code is formatted and refactored. Also the remaining documentation and a user guide is written. Project is complete.

Personal Information and Qualifications

I am a 2nd year undergraduate Electronics and Communication engineering student studying in **Indian Institute of Technology, Kanpur.** I am **not** getting credit for the GSoC project. I am proficient in 4 languages including English. I will have internet access over the entire course of the GSoC period.

I am proficient in C++, Python, VHDL, C# and javascript. I have read up and studied the **entire codebase** of gr-radar toolbox.

Below are some of the projects I have worked on:

1.I have extensively worked on Kalman filters and real-time data processing in a project on Localization using Gait Analysis using Inertial Measuring Units(IMUs)[3]. This project involved converting a **research paper** [4] into code since the code was not made available by the author. This project is written in Python.

2. I have also worked on RTL-SDRs for the project “Encrypted Wireless Video transmission” . Video link is here [5] . I used Raspberry Pi to transmit DVB-S modulated video to an RTL-SDR and demodulated it using leandvb. This project is written in Python. I cannot share the code as this was done as a University Project.

3. I am also working on a **open-source** project “Openage”, which is a free and improved remake of the Age of Empires game engine where I am currently working on the OpenGL renderer. This project is based on C++ as well as GLSL. The renderer is a core part of the engine which is being reworked, so it still hasn’t been merged to the main branch. Here [6] is my fork on which I am working.

My coursework includes signal processing and I have read up on various research papers on passive radar to familiarize myself with it. I also believe that “**cyberspectrum is the best spectrum**”.

License

Acknowledgements

In the end, I promise to adhere to all the guidelines set by GNU-Radio Organisation as well GSoC . I accept the three-strikes rule and the other rules in the rules and conducts page. I would also like to thank GNU-Radio for providing me this opportunity to contribute to the great open-source community. I would also like to thank **Martin Braun** and **Marcus Muller** for providing valuable guidance while developing this proposal.

Contact Details

Name: Suraj Hanchinal
Email: surajhanchinal@gmail.com
College email: surajvh@iitk.ac.in
Github: https://www.github.com/surajhanchinal

References

[1] http://jmfriedt.free.fr/URSI.pdf

[2]:https://pdfs.semanticscholar.org/0a3c/9b13412fc3929f679217d1d4f11497a50e53.pdf

[3]: https://www.github.com/surajhanchinal/imu_localisation

[4]:https://github.com/surajhanchinal/imu_localisation/blob/master/IEEE_Trans-on-HMS-2016-1.pdf

[5]: https://photos.app.goo.gl/cR6rCZAerIIDGe963

[6]: www.github.com/surajhanchinal/openage/tree/new-renderer