Engineering Proceedings

MDPI

Proceedings

A Synthetic Wide-Bandwidth Radar System Using Software Defined Radios †

Onur Toker [1],* and Ozgur Ozdemir [2]

[1] Department of ECE, Florida Polytechnic University, Lakeland, FL 33805, USA
[2] Department of ECE, North Carolina State University, Raleigh, NC 27695, USA
* Correspondence: otoker@floridapoly.edu

† Presented at the 7th International Electronic Conference on Sensors and Applications, 15–30 November 2020; Available online: https://ecsa-7.sciforum.net/.

Published: 15 November 2020

Abstract: In this paper, we present a synthetic wide-bandwidth radar system using software defined radios (SDR), and demonstrate the proposed approach using a Universal Software Defined Radio Peripheral (USRP) device. Normally, USRP devices have tens of MHz bandwidth, and cannot generate large bandwidth sweeps to achieve cm level range resolution. By using a synthetic wide-bandwidth approach, we can generate frequency sweeps up to 5 GHz bandwidth and obtain high-resolution range profiles. We will first summarize the mathematical details of the proposed approach, then present a pure Python based solution using the UHD library, and a GNU radio + Octave based implementation, and finally present experimental results for two different test cases. The developed code is available on a public GitHub repo. Compared to the FMCW radars with a voltage controlled oscillator, the sweep time or the experiment duration is longer, but very large bandwidth sweeps can be realized easily by using low-cost USRP devices, and sweeps are more accurate. All of our experimental results indicate the effectiveness of the proposed low-cost software defined radar system.

Keywords: radar sensors; software defined radar; USRP

1. Introduction

Software defined radio (SDR) devices, and in particular Universal Software Radio Peripheral (USRP) devices can be used to build radar systems in a variety of different ways. In this paper, we present a synthetic wide-bandwidth approach to achieve a couple of GHz sweep bandwidth, and achieve cm level range resolution.

In [1], a frequency modulated continuous wave (FMCW) radar is built using two NI-USRP 2920 units, and because of the the bandwidth limitations of USRP devices, achieved range resolution was approximately 10 m. In [2], a completely different USRP based radar system is presented using two N210 devices. Basically, an OFDM signal which is normally used to carry digital information from a transmitter to a receiver is used to build a radar sensor. The transmitter system has a receive antenna as well, and by processing the reflected OFDM waveform, range and velocity estimates are obtained. Both FMCW and OFDM radars need high bandwidth USRPs to achievable better range resolution. Basically, $c/(2B)$ is the theoretical range resolution where c is the speed of light, and B is the bandwith of the chirp signal or of the OFDM frame. In [3], a different USRP based OFDM radar is presented. This is a two stage converter system operating at mm-wave frequencies, and as in the previous two cases, bandwidth limitations of USRP devices resulted lower range resolution.

In this paper, we address this bandwidth limitation of USRP devices by using a synthetic wide-bandwidth design approach. Basically, in a single experiment, USRP devices can generate only narrow band signals. But by changing the local oscillator frequency and repeating the same narrow band measurement multiple times, it is possible to emulate a wide-bandwidth radar system. In [4,5], a frequency stacking approach is presented using pseudo random signals as the transmit signal. This is basically a frequency domain synthetic wide-bandwidth design to improve the range resolution. In [6], a completely different time domain synthetic wide-bandwidth design is presented using an E312 model USRP. The mathematical theory behind this time domain approach is based on [7].

In this paper, we present a simple frequency domain synthetic wide-bandwidth (SWB) design to achieve GHz level bandwidths and improve the range resolution. The mathematical theory is based on modelling the physical medium as a multiple delay system. Consider the USRP based radar and/or communication system shown in Figure 1.

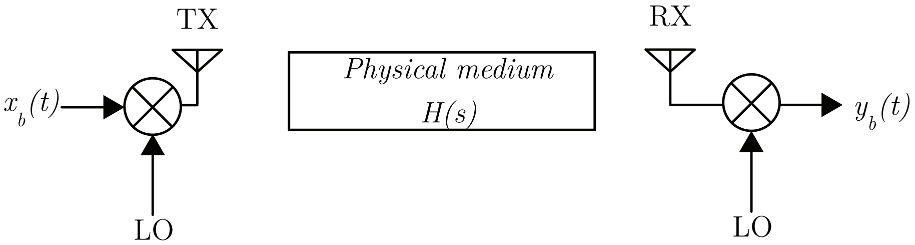


Figure 1. The physical medium has transfer function $H(s)$, and the local oscillator (LO) frequency is f_e.

The physical medium between the TX and RX antennas has a transfer function, $H(s)$, which can be modelled as

$$H(s) = L(s) \sum_q a_q e^{-h_q s},$$

where a_q's are complex scalars, and $L(s)$ is band-pass centered around f_e. Basically, the RF subsystem, and the antennas define the filter $L(s)$. To simplify the derivations, we assume that the system is calibrated and $L(s) \approx 1$ in the frequency band $[f_e - f_B, f_e + f_B]$. We define the radar problem as a measurement process to estimate h_q's and a_q's.

This is basically a pure software defined radar approach, and has various advantages. First of all, it is made of standard USRP devices, and is highly reconfigurable. FMCW radars built using voltage controlled oscillators (VCO) exhibit nonlinearity in range measurements, mainly due to VCO nonlinearity. Although there are various techniques to mitigate this effect, see [8] and the references therein, software defined radars will have a competitive advantage from this perspective.

This paper is organized as follows: In Section 2, we present mathematical preliminaries, and in Section 3, we present the proposed SWB algorithm. In Section 4, a pure Python based implementation built on the UHD driver is presented, and all the developed code is shared on a publicly accessible GitHub repo. Using the developed software, we generated a synthetic sweep from 1.9 GHz to 4.2 GHz using a B205 mini, and a B210 USRP device, and used this system for a real measurement. In Section 5, we present an alternative system based on GNU radio graphs and Octave based programming. In Section 6, we present another experimental result using a delay line with two propagation paths, and finally in Section 7, we make some concluding remarks.

2. Mathematical Preliminaries

In this section, we will summarize our mathematical notation and review some preliminaries from [9]. We will frequently use the complex baseband representation of passband signals, and reserve the subscript, $_b$, for baseband signals. Basically, if $s(t)$ is the real passband signal, $s_b(t)$ will be the complex baseband version. In this paper we describe a measurement procedure based on changing the local oscillator frequency multiple times and repeating a basic measurement process. Therefore, the local oscillator frequency is not fixed throughout the paper, and whenever the local oscillator frequency is clear from the context, its value will not be explicitly mentioned in the discussion. This applies for the bandwidth, and the sampling period as well.

For a SWB experiment, we will consider the system defined in Figure 1 with local oscillator frequency f_e, and complex baseband signals in the frequency range $[-f_B, f_B]$. This will be a hypothetical wide-band system which exists only in the mathematical domain. For actual measurements, we will operate the same system with local oscillator frequencies $f_c \in [f_e - f_B, f_e + f_B]$, and bandwidth $B \ll f_B$. By combining multiple narrow-band measurements, we will emulate a wide-band measurement, which is the main idea behind the SWB approach. All of our actual narrow-band measurements are done by using USRP's as the transmitter and the receiver. To simplify the mathematical analysis, we will first assume shared local oscillator configuration, and address other configurations later in the paper.

Consider a complex baseband signal, $s_b(t)$, band limited to $[-B, B]$, with $B < f_c$. The passband version, $s(t)$, will be

$$s(t) = \sqrt{2}\mathrm{Re}\left(s_b(t)e^{j2\pi f_c t}\right),$$

see [9]. We use capital letters to denote the Fourier transform of continuous time signals,

$$S(f) = \int_{-\infty}^{+\infty} s(t)e^{-j2\pi ft}dt, \quad s(t) = \int_{-\infty}^{+\infty} S(f)e^{-j2\pi ft}df.$$

In the frequency domain, we have the following equations for the baseband and passband signals,

$$S_b(f) = \sqrt{2}S^+(f + f_c), \quad S(f) = \frac{S_b(f - f_c) + S_b^*(-f - f_c)}{\sqrt{2}},$$

where $S^+(f) = S(f)I_{\mathbb{R}^+}$, and $I_{\mathbb{R}^+}$ is the characteristic function of the set $\mathbb{R}^+$, see [9]. For the system shown in Figure 1, with $x_b(t)$ as the input and $y_b(t)$ as the output, we have

$$Y(f) = H(f)X(f), \quad Y_b(f) = \frac{1}{\sqrt{2}}H_b(f)X_b(f).$$

For a given complex baseband signal, $s_b(t)$, we will also define a (complex) discrete time signal, $s_d[n] = s_b(nT)$, where T is the sampling period equal to $1/(2B)$, or $1/(2f_B)$.

2.1. Channel Frequency Response Measurement

In this subsection, we focus on the frequency response of the channel, and describe a measurement procedure. Consider the system shown in Figure 1 with local oscillator frequency set to f_c. Assume that the output $y(t)$ is sampled as $y_d[n] = y_b(nT)$, and the input is driven by

$$x_b(t) = \sum_n x_d[n]g(t - nT),$$

where $g(t) = \mathrm{sinc}(t/T)$, and $T = 1/(2f_B)$. If $x_d[n]$ is a complex exponential, i.e. $x_d[n] = e^{j\Omega n}$, with $\Omega \in [-\pi, \pi]$, then

$$x_b(t) = e^{j\Omega t/T}, \quad y_b(t) = H\left(f_e + \frac{1}{2\pi}\frac{\Omega}{T}\right)e^{j\Omega t/T},$$

and

$$y_d[n] = H\left(f_e + \frac{1}{2\pi}\frac{\Omega}{T}\right)e^{j\Omega n}, \quad H_d(e^{j\Omega}) = H\left(f_e + \frac{1}{2\pi}\frac{\Omega}{T}\right),$$

where H_d is the transfer function from x_d to y_d.

The frequency response can be measured by using a relatively simple narrow-band experiment. Basically, to compute $H(f_c)$, one can change the local oscillator frequency to f_c, set $x_d[n] = 1$, use zero-order hold on the transmitter side, and observe the steady state value of $y_d[n]$. Below, we present an improved algorithm which does not use DC values.

Algorithm 1: Narrow-band CFR measurement

```
Result: $H(f_c)$
Set $T = 1/(2B)$ for a narrow-band experiment, and use zero-order hold on the
    transmitter USRP,
On the transmitter USRP, set the local oscillator frequency to $f_c$, set $x_d[n] = 1$,
On the receiver USRP, choose an integer $M > 2$, set the local oscillator frequency to
    $f_c - (B/M)$, and estimate the amplitude of complex exponential $e^{j\pi n/M}$ in the steady state
    response of $y_d[n]$,
This amplitude can be computed by doing a simple FFT analysis, and the result will be equal
    to $H(f_c)$.
```

2.2. *Channel Impulse Respone Based Radar*

In this subsection, we study the relationship between the channel impulse respone and the original radar problem. If the input $x_d[n]$ is a discrete-time impulse, then $x_b(t) = \text{sinc}(t/T)$, where $T = 1/(2f_B)$. For a calibrated system with $L(s) = 1$, we have

$$H_b(s) = \sum_q e^{-j2\pi f_e h_q} \sqrt{2}\, a_q\, e^{-h_q s}$$

and

$$y_b(t) = \sum_q e^{-j2\pi f_e h_q} a_q \, \text{sinc}((t - h_q)/T).$$

The main question is the following: If we simply look at the sampled version, $y_d[n] = y_b(nT)$, can we detect all reflectors, and estimate their radar cross section simply from the channel impulse response?

To understand this problem in a better way, let's consider the case of a single reflector. If $H(s) = e^{-h_q s}$, we expect $y_d[n] \approx \delta[n - n_o]$ where $n_o \approx h_q/T$. However, depending on the exact value of the ratio, h_q/T, this is not guaranteed, and $y_d[n]$ may have a very small main lobe, much smaller side lobes, and the presence of the reflector may not be detected when we simply look at $y_d[n]$. However, if we sample

$$r_b(t) = \frac{2\pi}{4+\pi}\Big(y_b(t - T/2) + y_b(t + T/2)\Big),$$

the sampled version will have a single main lobe with peak value close 1, smaller side lobes, and the presence of the reflector will be clearer when we simply look at $r_d[n]$. The only appearant disadvantage is the main lobe may have two samples (taps) rather than one.

3. Proposed SWB Radar Algorithm

In this section, we will summarize our main SWB algorithm. Consider the transfer function, $H_d(e^{j\Omega})$, from $x_b[n]$ to $y_b[n]$, and the transfer function $G_d(e^{j\Omega})$, from $x_b[n]$ to $r_b[n]$. We have $H_d(e^{j\Omega}) = H\left(f_e + \frac{1}{2\pi}\frac{\Omega}{T}\right)$,

$$G_d(e^{j\Omega}) = \frac{2\pi}{4+\pi}(e^{-j\Omega/2} + e^{+j\Omega/2}) H\left(f_e + \frac{1}{2\pi}\frac{\Omega}{T}\right),$$

and the function, $\frac{4\pi}{4+\pi}\cos(\Omega/2)$, can be viewed as a "windowing" function.

Below, we present an algorithm to estimate $g_d[n]$. The N is chosen large enough so that the most of the energy of $g_d[n]$ is concentrated in $[0, N)$. This algorithm is based on the narrow-band measurement algorithm described earlier.

```
Algorithm 2: SWB radar
  Result: Reflector locations.
  Define a purely mathematical wide-bandwidth radar system with $T = 1/(2f_B)$,
  Compute $H_d(e^{j2\pi k/N}) = H\left(f_e + \frac{k}{N} f_B\right)$, using the narrow-band CFR algorithm on a real
    narrow-band USRP device,
  Apply the windowing function $\frac{4\pi}{4+\pi}\cos(\pi k/N)$ to $H_d(e^{j2\pi k/N})$,
  Compute the inverse FFT of this result to compute $g_d[n]$,
  A peak at index $n_o$ in the impulse response, $g_d[n]$, is interpreted as a reflector at
    distance $n_o Tc/2$.
```

Performance Limitations

All round-trip time delay's are estimated with error bound $T = 1/(2f_B)$, and reflector positions are estimated with error bound $cT/2$. Therefore,

$$\text{Range resolution} = \frac{c}{4f_B},$$

and the maximum range will be

$$\text{Maximum range} = N\frac{cT}{2} = N\frac{c}{4f_B}.$$

Therefore, for better radar performance, the hypothetical system should have large bandwidth, $2f_B$, and N should be selected as a large positive integer.

4. Python Based Implementation

In this section, we summarize a pure Python based implementation of the SWB radar algorithm using USRPs. The developed code is available at https://github.com/onurtoker/swb_radar, and has been tested using using a USRP B200 mini as a transmittter, and USRP B210 as a dual channel receiver. The theory presented in the previous section was based on local oscillator sharing, but instead we used a loopback signal from the transmitter to the receiver. Even with local oscillator sharing, or with a common 10 MHz reference input, USRPs will have different A/D sampling clocks, and use of a loopback signal solves all of these synchronization problems [10].

In Table 1, we summarize the version numbers of the major software components used in this work.

Table 1. Software versions.

Ubuntu	18.04.1
PyCharm	2020.1
Python	3.6.9
numpy	1.16.3
scipy	1.2.1
UHD driver	3.15.LTS

The Python code is easy to read, and mostly self-explanatory. It is a multi-threaded design using the UHD API, and there is an object called `myUSRP` which provides a layer of abstraction. The main program, `usrp_radar`, is built on this abstraction layer and has the loop for changing the local oscillator frequency, reading the received data, and writing these to a .mat file. Although it is possible to do all DSP processing in Python using `numpy, scipy,matplotlib`, we use the Python code for only coordinating the real-time experiment, and recording the results. All post-processing is done in MATLAB, see the `rangePlot.m` in the GitHub repo.

We consider two test configurations shown in Figure 4. The reflector shown in the left configuration is about 30 cm closer compared to the one shown on the right. When we look at the normalized CIR plots shown in Figure 3, we see a stationary major lobe, which corresponds to a strong line of sight coupling, and but also see a smaller lobe shifted about 30 cm between two configurations. For measurements, we used 200 uniformly seperated frequencies from 1.9 GHz to 4.2 GHz, and used zero-padding to obtain smoother FFT plots. The GitHub repo has the MATLAB code with windowing, and inverse FFT to compute the CIR.

5. GNU Radio + Octave Implementation

In this section, we summarize a GNU radio + Octave based implementation of the SWB algorithm. The solution presented here is based on XLMRPC and ZMQ blocks of the GNU radio. The developed sample GNU radio graph and the Octave script are available at https://github.com/onurtoker/swb_radar. Basically, we use the same USRP devices, and the same setup shown in Figure 2. The GNU radio graph has an XLMRPC block which allows external programs, like Octave, to change variables defined within the graph. This feature is used to change the local oscillator frequency of the USRP devices from an Octave script. To get XMLRPC functionality in Octave, we are using a simple script available in the GitHub repo. However, the same functionality can be achieved by using other tools, for example the Apache's XMLRPC library. Our script based solution may not be elegant, but it does not require any extra software installation. In the GitHub repo, `swb_radar.grc`, is the GNU radio graph file that we used for the experiments, see Figure 5, and `swb_radar.m` is the Octave script which acts as the master program. To transfer captured data from the GNU radio graph to an external program, we are using a ZMQ block. Basically, the Octave script changes the local oscillator frequency using XMLRPC, reads the captured data using ZMQ, and then processes the captured raw data to generate radar range profiles. All of these details are available in the sample code `swb_radar.m` Octave script available in the GitHub repo.

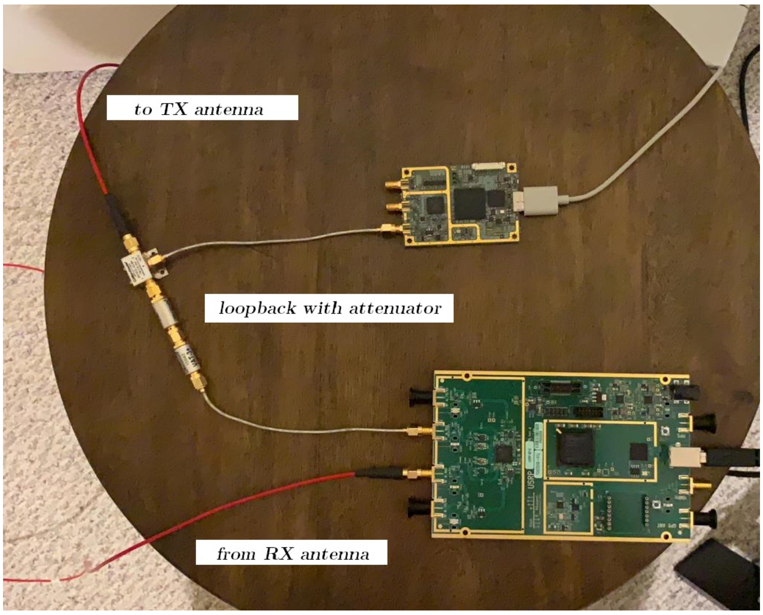


Figure 2. Test setup with two USRPs (B200 mini and B210), a splitter, and a loopback connection.

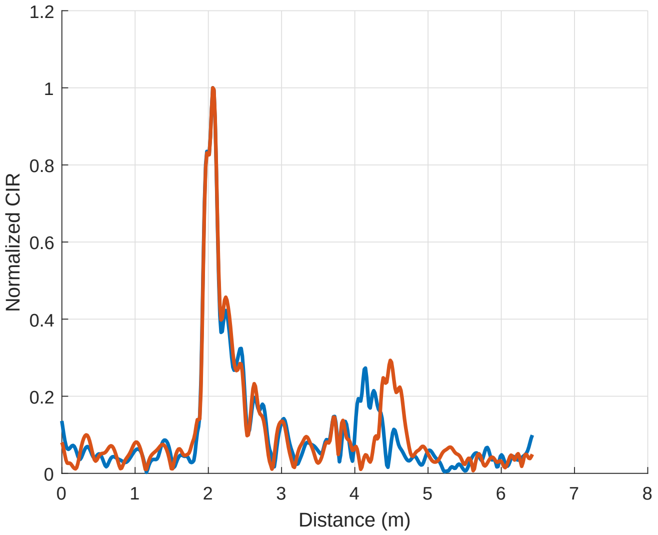


Figure 3. Normalized CIR in absolute value. Blue and orange plots correspond to the left and right configurations shown in Figure 4.

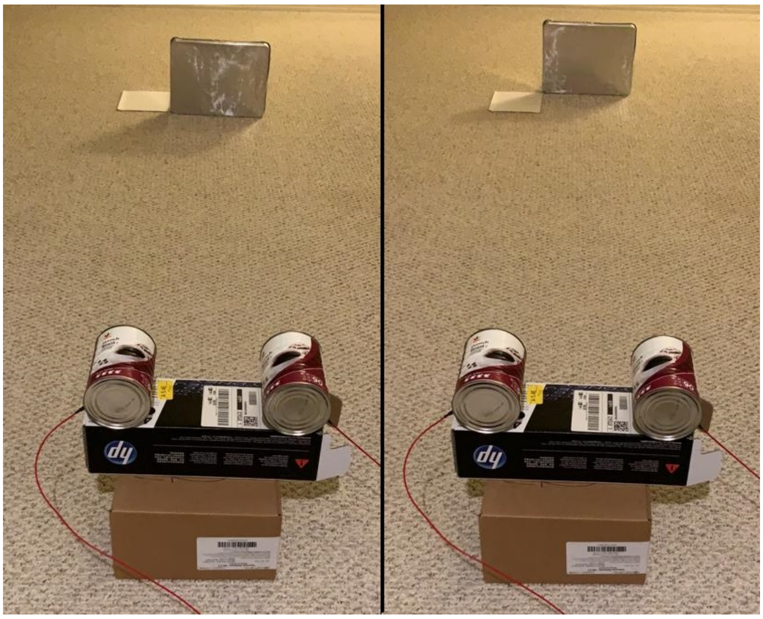
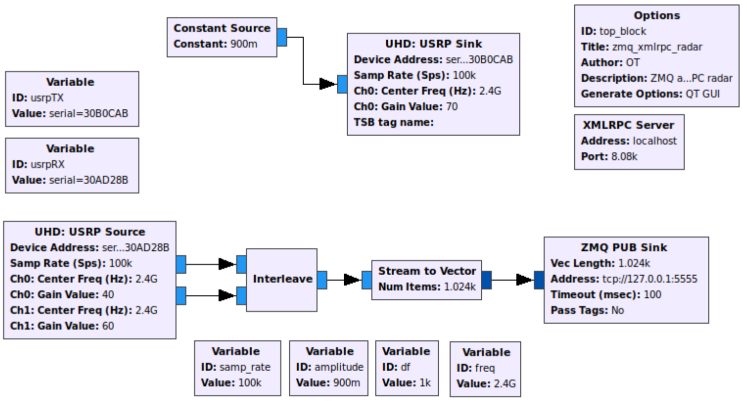


Figure 4. Test configurations. On the left, the reflector is closer compared to the configuration shown on the right.



Constant Source
Constant: 900m

UHD: USRP Sink
Device Address: ser...30B0CAB
Samp Rate (Sps): 100k
Ch0: Center Freq (Hz): 2.4G
Ch0: Gain Value: 70
TSB tag name:

Options
ID: top_block
Title: zmq_xmlrpc_radar
Author: OT
Description: ZMQ a...PC radar
Generate Options: QT GUI

Variable
ID: usrpTX
Value: serial=30B0CAB

Variable
ID: usrpRX
Value: serial=30AD28B

XMLRPC Server
Address: localhost
Port: 8.08k

UHD: USRP Source
Device Address: ser...30AD28B
Samp Rate (Sps): 100k
Ch0: Center Freq (Hz): 2.4G
Ch0: Gain Value: 40
Ch1: Center Freq (Hz): 2.4G
Ch1: Gain Value: 60

Interleave

Stream to Vector
Num Items: 1.024k

ZMQ PUB Sink
Vec Length: 1.024k
Address: tcp://127.0.0.1:5555
Timeout (msec): 100
Pass Tags: No

Variable
ID: samp_rate
Value: 100k

Variable
ID: amplitude
Value: 900m

Variable
ID: df
Value: 1k

Variable
ID: freq
Value: 2.4G

Figure 5. GNU radio graph used for the SWB radar experiment.

6. Delay Line Experiment

In this section, we present the results of a delay line experiment with two propagation paths. The experimental setup is shown in Figure 6, and there is a short loopback from the transmitter to the receiver, similar to the previous experiments. The transmitter signal also goes through another longer channel which has 3 identical coaxial cables, a splitter, and a combiner. Basically, this longer channel has two propagation paths of length 2 h and 3 h respectively. The normalized CIR presented in Figure 7 has a peak around the tap 23 and another one at 35, and their ratio is 1.52, which is very close to the theoretically expected value of 3/2. For this experiment, we generated a synthetic sweep of bandwidth 2 GHz, and obtained approximately 10 cm resolution in cable length. The speed of light in the coaxiable cable is assumed to be two thirds of the speed of light in free-space.

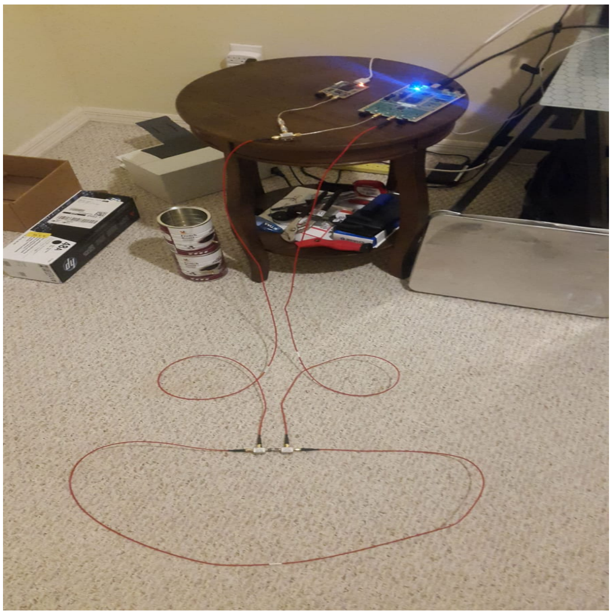


Figure 6. Delay line experiment with two propagation paths.

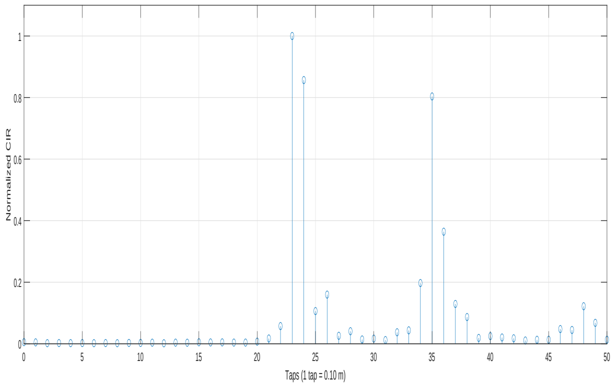


Figure 7. Normalized CIR of the channel shown in Figure 6.

7. Conclusions

In this paper, we have presented a synthetic wide-bandwidth radar design using USRPs. It is possible to do narrow band measurements using USRPs, and by changing the local oscillator frequency several times and repeating the same narrow band experiment, it is possible to achieve synthetic wide bandwidth. We have developed and shared two different software implementations of the proposed approach, and presented two different GHz level bandwidth experimental measurements using USRP devices.

Author Contributions: Both authors contributed to all parts equally. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported in part by Florida Polytechnic University grant GR-1900022, Advanced Mobility Institute grants GR-2000028, GR2000029, and the NSF grant 1919855. Both authors acknowledge their individual startup funds.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Alawsh, S.A.; Basheer, O.O.A.; Sirag, A.O.; Muqaibel, A.H. Range and Angle Measurements based on Software Defined Radio Platform. In Proceedings of the 2019 IEEE 10th GCC Conference Exhibition (GCC), Kuwait City, Kuwait, 19–23 April 2019; pp. 1–5.
2. Braun, M. OFDM Radar Algorithms in Mobile Communication Networks. Ph.D. Thesis, Karlsruher Instituts für Technologie, 2014. Available online: https://publikationen.bibliothek.kit.edu/1000038892/2987095 (accessed on).
3. Ozkaptan, C.D.; Ekici, E.; Altintas, O. Demo: A Software-Defined OFDM Radar for Joint Automotive Radar and Communication Systems. In Proceedings of the 2019 IEEE Vehicular Networking Conference, Los Angeles, CA, USA, 4–6 December 2019.
4. Friedt, J.M.; Feng, W. Software defined radio based Synthetic Aperture noise and OFDM (WiFi) RADAR mapping. In Proceedings of the GNU Radio Conference, Charlotte, NC, USA, 14–20 September 2020. Available online: http://jmfriedt.free.fr/grcon2020_radar.pdf (accessed on).
5. Friedt, J.M.; Feng, W. Noise RADAR Implementation Using Software Defined Radio Hardware. *GNU/Linux Mag. Fr.* **2020**. Available online: http://jmfriedt.free.fr/sdra_radar.pdf (accessed on).
6. Prager, S.; Thrivikraman, T.; Haynes, M.S.; Stang, J.; Hawkins, D.; Moghaddam, M. Ultrawideband Synthesis for High-Range-Resolution Software-Defined Radar. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 3789–3803.
7. Papoulis, A. Generalized sampling expansion. *IEEE Trans. Circuits Syst.* **1977**, *24*, 652–654.
8. Toker, O.; Brinkmann, M. A Novel Nonlinearity Correction Algorithm for FMCW Radar Systems for Optimal Range Accuracy and Improved Multitarget Detection Capability. *Electronics* **2019**, *8*, 1290, doi:10.3390/electronics8111290.
9. Madhow, U. *Fundamentals of Digital Communication*; Cambridge University Press: Cambridge, UK, 2008, doi:10.1017/CBO9780511807046.
10. Pagadarai, S. Enabling Precise Timing Control in SDRs. In Proceedings of the GNU Radio Conference, Huntsville, AL, USA, 16–20 September 2019. Available online: https://www.gnuradio.org/grcon/grcon19/presentations/enabling_precise_timing_control_in_sdrs/ (accessed on).

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.