Software defined radio based Synthetic Aperture noise and OFDM (Wi-Fi) RADAR mapping

J.-M Friedt JMFRIEDT@FEMTO-ST.FR
FEMTO-ST/Time & Frequency department, Besançon, France

W. Feng FENGWEIKE007@163.COM
National Laboratory of Radar Signal Processing, Xidian University, Xi'an, China

Abstract

We demonstrate the use of commercial, off the shelf, emitter and receiver Software Defined Radio hardware for implementing various RADAR schemes allowing for stacking signals with adjacent carrier frequencies and hence improve range resolution well above the intrinsic hardware bandwidth. Sources include the PlutoSDR and Wi-Fi dongles. The receiver is a dual channel Ettus Research B210, with one channel collecting the emitted signal and the other channel the received signal, allowing for emission/reception synchronization. We demonstrate synthetic aperture RADAR (SAR) and interferometric SAR (InSAR) with such a setup well suited for short range (<150 m) target mapping in an urban environment.

1. Introduction

RADAR measurements provide a fascinating field for experimenting with electromagnetic wave propagation characteristics otherwise called, in digital communication, multipath fading (Charvat, 2014). This field has however remained confined to the few lucky enough to access broadband emitters and receivers, and even (Charvat, 2014) leaves the hobbyist level when addressing azimuth compression with professional hardware. We aim at demonstrating that current Software Defined Radio (SDR) hardware allows for addressing most signal processing techniques used in RADAR measurements.

The range resolution ΔR of a RADAR is solely determined by the signal bandwith B and related to the speed of light by $\Delta R = c/(2B)$ independent on the carrier frequency. SDR hardware is plagued with a bandwidth limited by communication speed and processing power to a few hundred megahertz for high-end hardware, and a few megahertz for consumer grade electronics. A 2-MHz bandwidth, as provided by low cost Digital Video Broadcast-Terrestrial (DVB-T) hardware or USB-2 communication bandwidth when collecting two-channel complex 16-bit samples, will only allow for 75 m range resolution. Our objective in this work is to monitor the range to a house located at an estimated range of 50 m from aerial photography, with a meter resolution, using commercial, off the shelf SDR frontends.

In this work, we separate emission and reception in order to ease synchronization issues (Fig. 1). Indeed, general purpose SDR is unable to synchronize with sub-microsecond (150 m two-way trip) emission and reception, and we wish here to avoid modifying Field Programmable Gate Array (FPGA) firmware to achieve controlled latency between emission and reception (Prager et al., 2019). Thus, the propose approach is an emitter illuminating the scene with as broad a spectral characteristics as possible, sampling on a dual coherent receiver (Ettus Research B210) this emitted signal on one channel, and on the second channel collect the signal detected by the receiving antenna. Since both channels are clocked with the same local oscillator, any offset between emitter and receiver local oscillators during acquisition will be cancelled, and synchronization is achieved thanks to the common analog to digital converter clock.

This basic scheme allows for experimenting with various sources, in our case a noise RADAR emitted from a PlutoSDR generating a pseudo-random phase modulated signal, and a Wi-Fi emitter allowing for both digital communication and covert active RADAR measurement. The core processing on the receiver side will be correlation, implemented as a product of Fourier transform of the reference signal with the complex conjugate of the Fourier transform of the measurement signal, or its closely related inverse filtering considering the ratio of the Fourier transform of the reference signal with the Fourier transform of the measurement signal. In both cases the difference of the argument is computed, but in the former the product of the magnitude magnifies any divergence from the flat spectrum, while in the latter such magnitude fluctuations on both reference

Proceedings of the 1st GNU Radio Conference, Copyright 2016 by the author(s).

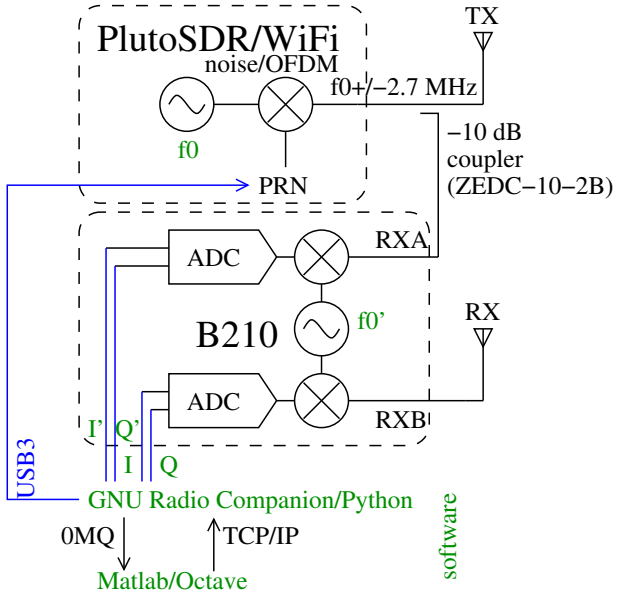

Figure 1. Schematic of the experiment, with separate emission and reception with the coupled emission recorded by the reference channel of the receiver clocked with a common oscillator shared with the measurement channel. Software (green) asynchronously controls hardware and saves the collected and broadcast datastream only when the measurement conditions are known to be stable.

and measurement channel are cancelled, as needed for Orthogonal Frequency Division Multiplexed (OFDM) Wi-Fi signals.

While range resolution is independent on the carrier frequency, beam aperture is determined by the ratio of the electromagnetic wave to the antenna width and hence to the inverse of the signal frequency. While large antennas allow for narrow beams and hence sharp azimuth resolution, they are not convenient to setup in an urban environment. An alternative to wide antenna is Synthetic Aperture RADAR (SAR) in which antennas are moved along a path matching the equivalent antenna width. If the motion step is linear and equidistant, the Uniform Linear Array (ULA) leads to a convenient processing formalism for azimuth compression with no free parameters once the wavelength, antenna motion step and trajectory length have been defined. The flexibility of SDR RADAR is emphasized by our transition from a UHF RADAR operating in the 650–750 MHz band as described at (Friedt & Feng, 2020) to the current 2.45 GHz setup with *no hardware update* other than the antennas.

The outline of the discussion is as follows: at first we demonstrate range compression, only dependent on signal bandwidth, and improved range resolution thanks to frequency stacking. Then SAR is demonstrated by moving the receiving antenna and keeping the emitting antenna fixed. Finally, SAR interferometry for sub-wavelength displacement detection along the line of sight is demonstrated by analyzing the phase of the detected signal.

2. Range compression

Radiofrequency ranging requires timing with as high as possible a resolution the time of flight of electromagnetic waves from a source to a target and back to a receiver. When the emitter and receiver are separate entities, synchronization is needed for time of flight measurement. In the proposed scheme, the emitted signal is recorded using analog to digital converters clocked with the same oscillator than the measurement channel connected to the receiving antenna. Thanks to this scheme, a common time is available between emission and reception, so that cross-correlating the received signal with the reference signal will accumulate energy whenever delayed copies of the emitted signal are found in the received signal. Spreading the spectrum aims at increasing B so that the range resolution is increased, limited by the sampling rate of the SDR hardware. The most suitable waveform in which the emitted pattern never repeats is noise, leading to the noise RADAR since the auto-correlation of noise is a Dirac function at 0-delay.

The first demonstration of a properly operating noise RADAR is thus to generate a pattern on the emitter, delay by a known duration this emission before it reaches the reception, and correlating emission and reception to demonstrate the coherent measurement.

The range resolution is however limited by the rather narrowband SDR platforms and the limited sampling and communication bandwidth. The hardware sampling rate limitation is overcome by sequentially probing the environment response to illumination by electromagnetic waves emitted at varying carrier frequencies: frequency stacking.

The challenge of frequency stacking is that the local oscillator of both emitter and receiver must be adjusted simultaneously and only when stabilized can relevant data be collected. GNU Radio does not allow for discontinuous datastreams to be recorded so that the proposed approach is a TCP server running from the GNU Radio flowchart to control the local oscillator frequency under external control, and streaming collected I/Q data to the external control software through a non-blocking UDP-like socket implemented as a ZeroMQ publish socket. Our implementation of the control software is with GNU/Octave and its `zeromq` and `socket` toolboxes since digital signal processing (correlation, threshold, range compressed signal display) is most easily implemented with this processing framework, although all functions have been ported to

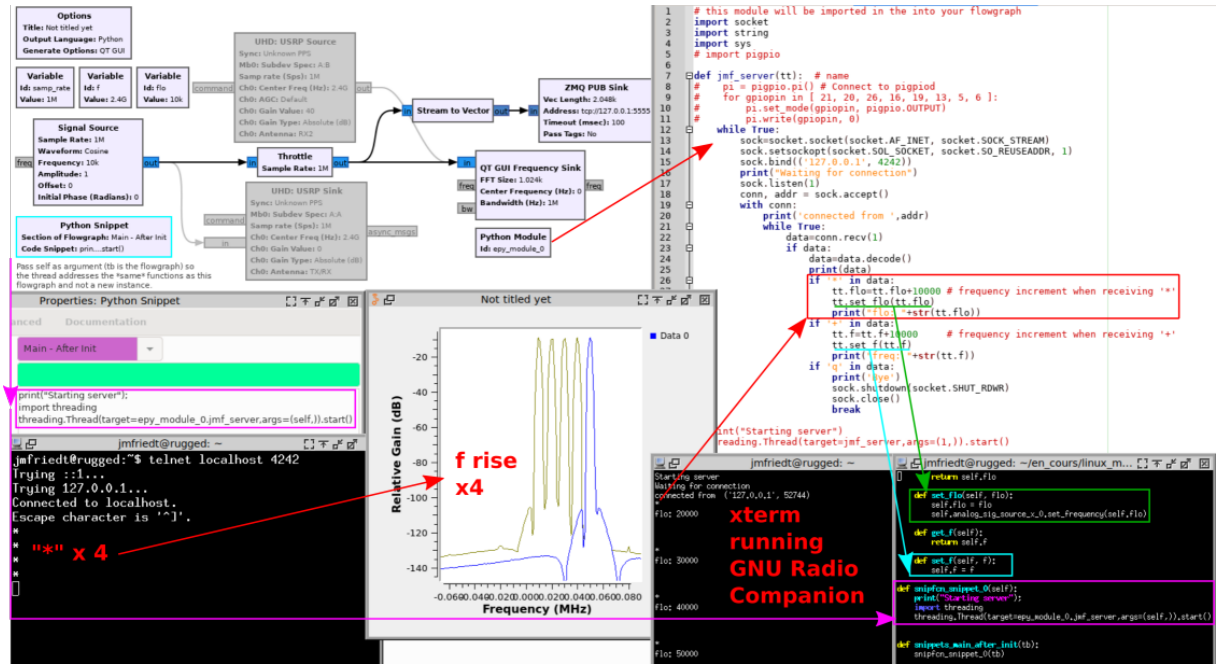

Figure 2. Demonstration of the inclusion of a TCP server in the Python Module as part of a GNU Radio Companion application with the additional thread launched by the Python snippet. Accessing the original GNU Radio Companion objects is demonstrated by tuning the Signal Source frequency by sending control commands from a `telnet` client.

Python3 in order to achieve a fully autonomous RADAR implementation running on the single board computer Raspberry Pi 4. The consistent embedded GNU/Linux generation framework Buildroot was used in order to provide the cross-compilation toolchain, the kernel, bootloader and userspace libraries and applications including GNU Radio running on the embedded board, `libuhd` supporting the Ettus Research B210 dual-channel receiver connected to the USB-3 communication bus, `gr-iio` and `libiio` for sending emission streams to the PlutoSDR connected to the USB-2 communication bus, and `numpy` for embedded signal processing (Fig. 3).

Rather than modify the Python code generated by GNU Radio Companion, which prevents returning to the graphical user interface once the Python code has been appended with new functionalities, we use the Python Module (Fig. 2) to add custom code – namely a separate thread running a TCP server able to access the methods provided by the calling program – and execute this thread from the main program by appending the initialization function with the thread call. Calling the thread with the `self` argument allows to modify asynchronously the flowgraph properties and all variable defining the SDR properties (e.g. gain or local oscillator frequency: Fig. 1).

Having found the ability to tune the SDR hardware during a measurement, we must save data when acquisition conditions are stable. Because the continuous I/Q datas-

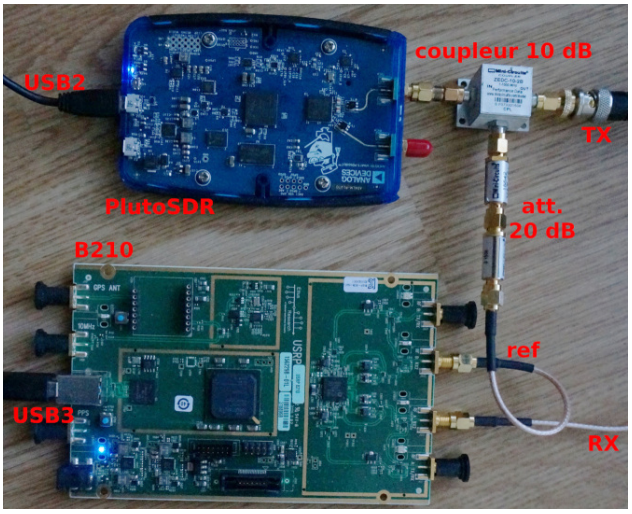

Figure 3. Experimental setup, here using a PlutoSDR for emission and a B210 for reception. The passive additional radiofrequency components are limited to a minimum with a coupler and attenuators keeping maximum flexibility of the software approach to RADAR signal acquisition.

tream includes the state transition as the hardware properties are modified, the external software controlling the hardware can also synchronize acquisition by fetching the broadcast data only when the hardware is known to be in

a stable condition. Rather than using the UDP Sink from GNU Radio Companion, we have selected to use the higher level ZeroMQ framework and its equivalent Publish server. A Subscribe client connects to this server whenever the datastream is expected to be stable following hardware reconfiguration, and data are otherwise lost if no listening socket has been connected to the server. Since ZeroMQ has been ported to multiple languages, all are suitable for running the client: we have selected GNU/Octave (running on a x86-based laptop) or Python (running on Raspberry Pi 4 for a fully embedded noise RADAR implementation) for fetching and processing data. Indeed while the PlutoSDR emitter is only fitted with a USB-2 interface, the B210 can benefit from the bandwidth of the USB-3 bus. Since the B210 must stream twice as many data than the PlutoSDR since both reference and measurement channels are fetched, sharing USB busses can improve speed and avoid data loss. Since the Panasonic CF-19 laptop used to run these experiments lacks a USB-2 bus, the Raspberry Pi 4 platforms appears ideally suited to implement an embedded version of this experiment.

3. Azimuth compression

The objective of this presentation is not to detail processing techniques which, although described in multiple textbooks, only become relevant when implemented on practical hardware. We here demonstrate how the collected data allow to investigate beyond range compression by addressing azimuth compression for target location identification.

An antenna beamwidth is determined by its aperture with respect to wavelength: the larger the antenna, the better the focusing capability and the narrower the illuminated target range. However, large antennas are impractical under some environments, either due to excessive dimensions with respect to available space, restrictive antenna ordinances or sensitivity to wind. Synthetic aperture RADAR (SAR) aims at simulating a large antenna of aperture D by moving a single broadbeam antenna along a path $D = Nd$ with N steps of equal length d. Because a broadmean antenna is compact and hence lightweight, it can be setup on a motorized rail for automated displacement.

SAR processing can be intuitively understood (see appendix A for the full demonstration) by considering that moving an antenna along a linear path whose normal is at an angle ϑ with respect to the incoming plane wave with wavelength λ will introduce a phase shift $\frac{2\pi}{\lambda} nd\sin(\vartheta)$ when moving by n. Since n is incremented in discrete steps, this expression of the argument of a complex exponential can be considered as a Fourier series with variables nd and $2\pi\sin(\vartheta)/\lambda$ as dual quantities between space as the antenna position and the wavevector. Since the inverse Fourier transform computed during the range compression correlation implemented as the inverse Fourier transform of the product of the Fourier transform of the reference channel with the complex conjugate of the Fourier transform of the measurement channel, range-azimuth compression ends up as the 2D-inverse Fourier transform of the matrix whose columns are the frequency domain spectra resulting from the Fourier transform of the time-domain sequences measured on the reference and measurement channels (whos product yields the correlation), and the lines are the successive antenna positions.

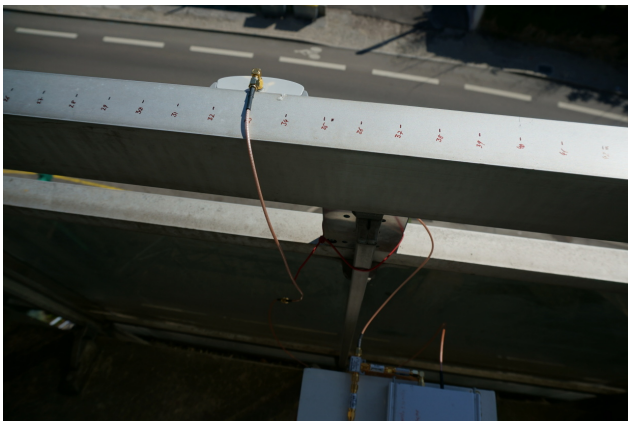

Figure 4. Experimental setup for demonstrating azimuth compression. The compact, broadbeam Wi-Fi patch antenna, is moved with $\lambda/4 = 3.1$ cm steps along the balcony railing.

Such abstract concepts are best understood when experimenting with data collected with the proposed setup. At first, a periodic graduation was drawn on the balcony railing and the 2.45 GHz patch antenna used in this experiment moved along this linear track while keeping the emitting antenna fixed. Under these conditions, the sampling theorem states that the maximum step must be half wavelength to avoid aliasing (appendix B). Furthermore, the emitting antenna should be located as far as possible (in our case on the floor of the balcony or about 1 m below the receiving antenna) to avoid excessive cross-talk when the two antennas are closest to each other along the track (Fig. 4).

Meeting these conditions allows for azimuth compression (Figs. 5, 6). Notice that once the experimental conditions are set with the frequency range and antenna displacement step defined by the experimental setup, then all quantities are defined when generating the radargram and no degree of freedom remains when overlapping with aerial images. Indeed the range is given by $c \times T/2$ when observing a target with two-way time delay T, with a range resolution of $c/(2B)$ since incrementing time with steps δT requires a bandwidth $B = 1/\delta T$, and the azimuth angle is determined by the displacement step.

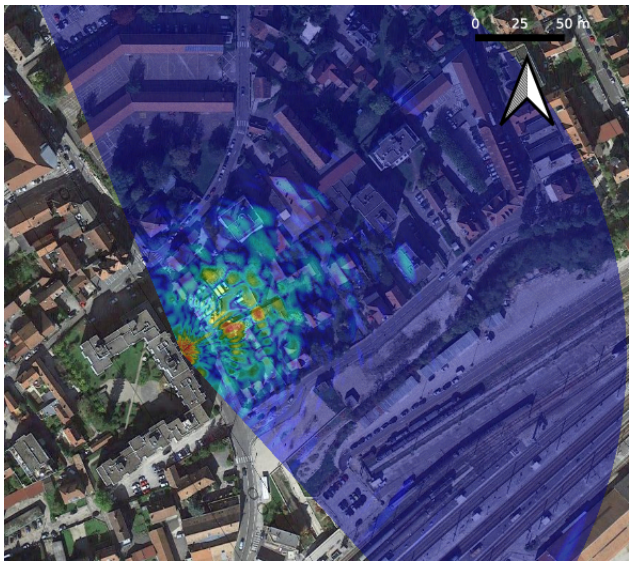

Figure 5. Range-azimuth compressed map overlaid over an aerial photography of the area surrounding the experimental setup: pseudo-random phase modulated noise RADAR using the PlutoSDR. Wide range display exhibiting reflections from targets further than 100-m from the emitter limited to −30 dBm (30 dB attenuation of the PlutoSDR output observed to emit a 0-dBm continuous wave when no attenuation is applied).

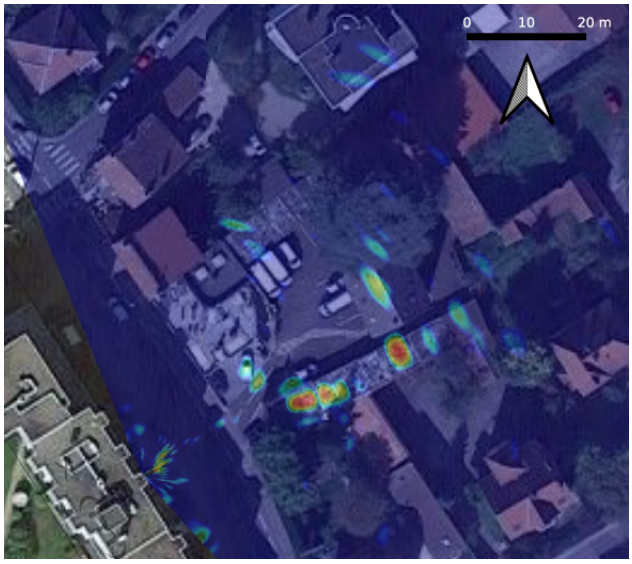

Figure 6. Range-azimuth compressed map overlaid over an aerial photography of the area surrounding the experimental setup: pseudo-random phase modulated noise RADAR using the PlutoSDR. Zoom on the nearby structures emphasizing the consistency of geographic features – house roof, parking box roofs acting as dihedral corner reflectors – and the targets detected by the noise RADAR.

The same experimental procedure as noise RADAR can be used with a Wi-Fi USB dongle as radiofrequency source (Fig. 7). In this case, the two subtleties are that the OFDM modulated signal does not exhibit a flat spectrum and cancelling the magnitude fluctuation as a function of frequency due to the adjacent subcarrier requires replacing the product of the Fourier transform of the reference signal with the complex conjugate of the Fourier transform of the measurement signal with the ratio of these two Fourier transform. While the effect of multiply conjugate or division is the same on the argument of the complex exponential, considering the ratio cancels the magnitude fluctuation common to both spectra and avoids grating sidelobes. Furthermore, despite the near continuous Wi-Fi emission thanks to B. Bloessl's `packetspammer` software (`github.com/bastibl/gr-ieee802-11/tree/maint-3.8/utils/packetspammer`) requiring a Wi-Fi chipset compatible with monitoring mode, the successive Wi-Fi packets remain discontinuous with durations without emission dependent on the operating system load. Under such conditions, the reference channel power is monitored and the data collected from the ZeroMQ stream only saved if this power is above a threshold. Thus, collecting a complete measurement sequence might require more measurements with Wi-Fi than with the noise RADAR which emits continuously. Nevertheless, Wi-Fi channels are separated by 5 MHz and allow for bigger frequency steps than the PlutoSDR noise emission limited by communication bandwidth to 2 to 2.5 MHz without loss of samples.

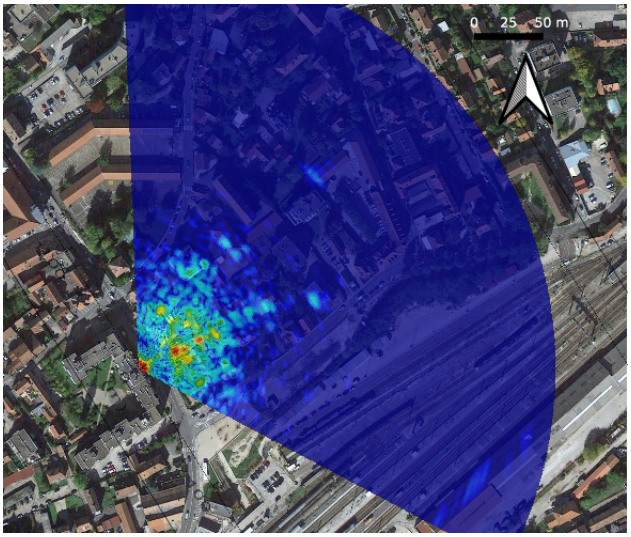

Figure 7. Range-azimuth compressed map overlaid over an aerial photography of the area surrounding the experimental setup: Wi-Fi emitter. Notice the consistent pattern with Fig. 5 despite the completely different waveform used to probe the targets.

4. Interferometric Synthetic Aperture RADAR

Finally, interferometric SAR (InSAR) involved complementing the returned power magnitude analysis (target location) with phase analysis. Since the phase rotates by 2π every time a target moves by half the wavelength (because of the two-way trip), phase analysis allows for sub-wavelength displacement detection independently of range resolution determined by the bandwidth. Such measurements require multiple SAR range-azimuth compressed image acquired under the exact same conditions, i.e. with antenna positions reproducible with much greater resolution than the wavelength. With a 12.2 cm wavelength at 2.45 GHz, reproducible manual positioning of the antenna is excessively challenging and an automated setup was assembled with the receiving antenna attached to a sliding rail driven by a lead-screw controlled by a stepper motor. This setup allows for sub-mm resolution reproducibility whatever the antenna motion direction (Fig. 8). Again the client-server software architecture is used to send commands to the microcontroller dedicated to stepper motor control to address antenna position while GNU Radio keeps on continuously streaming I/Q coefficients and the control software only collecting data as the position is stable.

Figure 8. Motorized rail for reproducibly positioning with sub-mm accuracy the receiving antenna while keeping the emitting antenna static. Inset bottom-right: closeup on the lead-screw setup using a stepper motor controlled by a dedicated microcontroller receiving control commands through its USB (virtual serial) port.

Detecting sub-wavelength position variation requires a target with a large RADAR cross-section with respect to surrounding static targets – the strongest reflectors being cars modeled as a first approximation as 2.5 m diameter spheres – and yet easily manipulated with centimeter resolution. A corner trihedral square corner reflector with 30 cm long sides exhibits the same RADAR cross section than a 2.5 m sphere: such a cooperative reflector is located at a 40 m range from the RADAR setup and moved with 1-cm steps. The phase measurement φ at each position is converted to a displacement by considering that $\varphi = 2\vec{k}\vec{r} = \frac{4\pi r}{\lambda_c}\cos\vartheta_0 = \frac{4\pi r \cdot f_c}{c}\cos\vartheta_0$ with $\vec{k}$ the wavevector and $\vec{r}$ the displacement vector, these two vector exhibiting an angle ϑ_0, and f_c the center frequency (λ_c its corresponding wavelength) of the RADAR signal.

Figure 9. Corner reflector located at a range of 40 m of the RADAR setup, in front of car defining the needed RADAR cross section and hence cube side length of 30 cm.

The result of such a measurement is given in Figs. 11, 12 and 13 in which the corner reflector is moved by 1 cm with respect to the reference acquisition, by 2 cm and brought back to its initial position respectively. The arrow indicates the corner reflection location as identified by subtracting a reference image with corner reflector from an acquisition with no corner reflector (Fig. 10).

5. Embedded implementation

We have mentioned multiple times the use of the Raspberry Pi 4 single board computer as suitable for implementing this experiment. As part of this project, all necessary GNU Radio blocks were ported to Buildroot to allow for generating the firmware running on the embedded board. Thanks to this effort, GNU Radio but also `libuhd` (for B210 control), `libiio` and `gr-iio` (for controlling the PlutoSDR) and even Qt5 for graphical display output on the micro-HDMI port were ported and now provide a stable and efficient development framework. In addition to providing an energy and cost efficient, as well as compact embedded solution, the Raspberry Pi 4 is fitted with multiple General Purpose Input Output pins allowing for controlling such features as antenna position, orientation or switch control. Such features are unfortunately no longer available on consumer grade computers, making the Raspberry Pi 4 attractive.

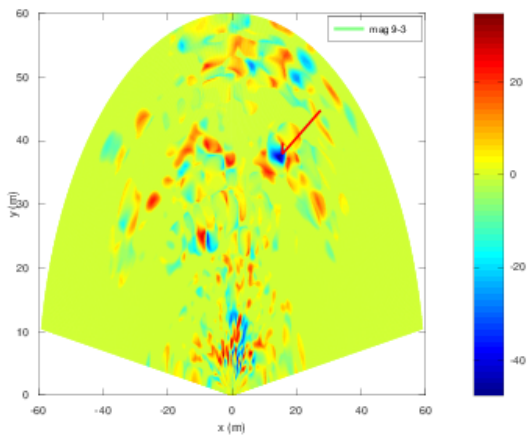

Figure 10. Magnitude difference between an acquisition without corner reflector and a map with corner reflector: the location is identified as a drop in reflectivity as shown by the red arrow.

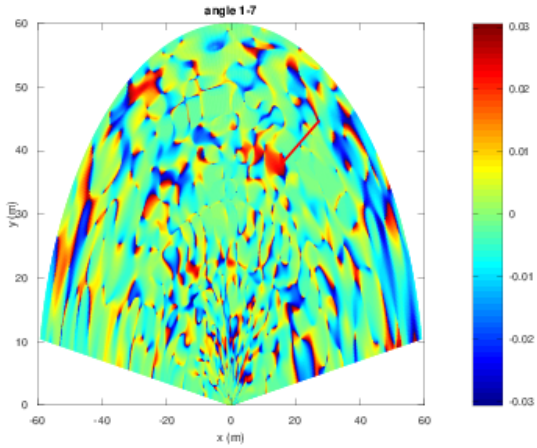

Figure 12. Range-azimuth map of the phase converted to displacement when the corner reflector was moved by 2 cm. The location of the corner reflector is given by the red arrow.

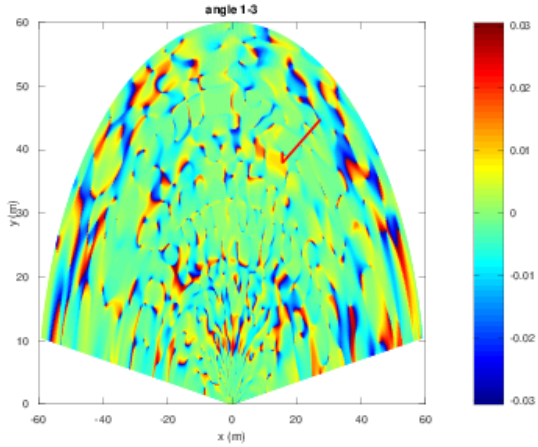

Figure 11. Range-azimuth map of the phase converted to displacement when the corner reflector was moved by 1 cm. The location of the corner reflector is given by the red arrow.

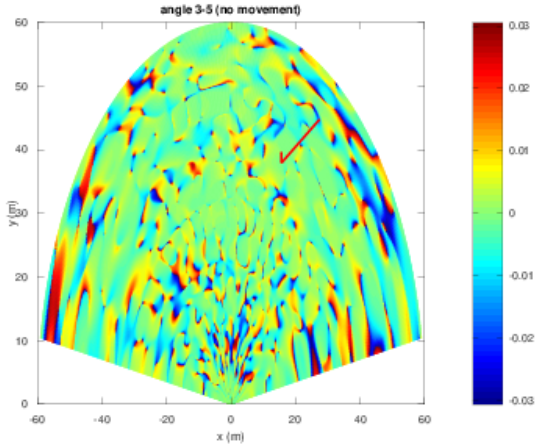

Figure 13. Range-azimuth map of the phase converted to displacement when the corner reflector was moved by 0 cm, i.e. brought back to its initial position at the end of the experiment. The location of the corner reflector is given by the red arrow.

6. Conclusion

We have demonstrated how the flexibility of opensource SDR allows for implementing various RADAR schemes more efficiently than the basic pulsed RADAR commonly associated with such measurements. We have separated the various parts of a RADAR system – emitter, receiver, control and processing – so that each part can be handled independently by the most efficient hardware and software framework. We have shown how azimuth compression and InSAR measurements could be achieved with such basic hardware, providing the educational opportunity of hands-on experimenting with otherwise abstract concepts. The meter range-resolution, sub-cm InSAR resolution and a few-degree azimuth resolution are ideally suited for urban SAR monitoring, especially when using the Wi-Fi emission for covert emission of RADAR signals complemented with digital communication.

Acknowledgement

The technical support of P. Abbé (FEMTO-ST Time & Frequency, Besançon, France) who assembled the corner reflector and the motorized rail is acknowledged. G. Goavec-

Merou (FEMTO-ST Time & Frequency, Besançon, France) ported all the software needed to run this experiment on the Rapsberry Pi 4 to Buildroot. These functionalities are now part of the official Buildroot releases.

References

Charvat, Gregory L. *Small and short-range radar systems*. CRC Press, 2014.

Friedt, J.-M and Feng, W. Noise radar implementation using software defined radio hardware. In *Software Defined Radio Academy (SDRA)*, 2020. https://www.youtube.com/watch?v=SPORRWjQqbA%22.

Prager, Samuel, Thrivikraman, Tushar, Haynes, Mark S, Stang, John, Hawkins, David, and Moghaddam, Mahta. Ultrawideband synthesis for high-range-resolution software-defined radar. *IEEE Transactions on Instrumentation and Measurement*, 69(6):3789–3803, 2019.

A. SAR principle demonstration

The underlying principle of SAR azimuth compression is based on the knowledge [1] that the Fourier transform of $\exp(j2\pi f_y \cdot y_0) \times \exp(j2\pi f_x \cdot x_0) \xrightarrow{\text{2D FT}} \delta(\underbrace{x_0, y_0}_{target})$ so that if the quantities f_x and f_y dual of the spatial target position $(x_0, y_0) = (r_0\cos(\vartheta_0), r_0\sin(\vartheta_0))$ can be separated, then the inverse Fourier transform of measuring f_x and f_y will allow for recovering (x_0, y_0).

1. f_q (frequency $\rightarrow$ range r_0) and x_p (antenna position $\rightarrow$ azimuth ϑ_0) must be separated
2. the signal at position p when frequency band q is emitted is $s(p,q) \underbrace{\propto}_{RCS} \exp\left(j\frac{4\pi}{c} f_q \cdot R_p(\underbrace{r_0, \vartheta_0}_{target})\right)$ with $R_p = \sqrt{(x_p - r_0\sin\vartheta_0)^2 + (r_0\cos\vartheta_0)^2}$ where $x_0 = r_0\sin\vartheta_0$ and $y_0 = r_0\cos\vartheta_0$ (cartesian $\rightarrow$ polar coordinates)
3. $\frac{\partial}{\partial x_p}\left(\sqrt{(x_p-a)^2+b^2}\right) = \frac{x_p-a}{\sqrt{(x_p-a)^2+b^2}} = \frac{-a}{\sqrt{a^2+b^2}}$ at $x_p \simeq 0 \Rightarrow$ Taylor expansion $R_p \simeq r_0 - x_p\sin\vartheta_0$ since $a^2 + b^2 = r_0^2$
4. $s(p,q) \propto \exp\left(j\frac{4\pi}{c} f_q \cdot (r_0 - x_p\sin\vartheta_0)\right) \simeq \exp\left(j2\pi(\underbrace{2f_q \cdot r_0/c}_{range\ \alpha} - \underbrace{2x_p\sin\vartheta_0/\lambda_c}_{azimuth\ \beta})\right)$ assuming that $f_q/c = 1/\lambda \simeq 1/\lambda_c$ the wavelength at center frequency since $\frac{1}{x} = \sum_n (-1)^n \cdot (x-1)^n \simeq 1$ around $x \simeq 1$ (keep only $n = 0$)
5. $r_0 = \alpha \times c/(2f_q)$ and $\sin(\vartheta_0) = \beta/(2\lambda_c)$ in polar coordinates or
6. $x_0 = r_0\sin\vartheta_0 = \alpha\beta \cdot c \cdot \lambda_c/4$ and $y_0 = r_0\cos\vartheta_0 = c\alpha/2\cos(\text{asin}(\lambda_c\beta/2))$ in cartesian coordinates: conversion from (f_q, x_p) to (x_0, y_0) using 2D-FT thanks to variable separation.

B. Antenna step size demonstration

The spatial equivalent to sampling theorem determines the maximum step size allowing for synthetic antenna analysis when moving a single broadbeam antenna at positions x_p, $p \in \mathbb{N}$:

1. the received signal phase is $\varphi = 2 \times \frac{2\pi f_c R}{c}$ where $R = \sqrt{(x_p - x_0)^2 + y_0^2}$
2. far field measurements so that $R \simeq r_0 + x_p \cdot \sin\vartheta_0$
3. $\Rightarrow \varphi \simeq \underbrace{2 \times 2\pi f_c \cdot r_0/c}_{static} + 2 \times 2\pi \underbrace{f_c/c}_{\lambda_c^{-1}} \cdot x_p \cdot \sin\vartheta_0$
4. avoid φ ambiguity with $2x_p/\lambda_c < 1 \Leftrightarrow x_p < \lambda_c/2$

Notice that if both TX and RX move (as opposed to keeping TX fixed and moving RX), then the phase unambiguity condition becomes $x_p < \lambda_c/4$

[1] https://hforsten.com/synthetic-aperture-radar-imaging.html