Software–Defined Ground Penetrating Radar Using COTS SDRs and GNU Radio Companion

Gavin Messerly, Christian W. Hearn, A. Tye Gardner

***Abstract*—This paper investigates the use of Software-Defined Radio (SDR) for Ground Penetrating Radar (GPR) to detect shallowly buried objects, and evaluates its performance against a commercial off-the-shelf (COTS) GPR system. Traditional GPR implementations typically employ separate transmitting and receiving antennas on handheld devices or vehicle-mounted platforms. Small rotor-based unmanned aerial vehicles (UAVs) paired with inexpensive sensors can reduce cost, effort, and time in GPR surveys. However, achieving high resolution often requires bulky, expensive, and power-intensive systems that are unsuitable for UAV payloads. SDR presents a promising alternative by offering compact, low-cost, and reconfigurable radar capabilities. This project demonstrates the viability of an SDR-based GPR (SD-GPR) as a capable and cost-effective alternative to conventional systems for detecting shallowly buried targets. The SD-GPR is implemented in GNU Radio Companion (GRC) as a stepped frequency continuous wave (SFCW) radar, with measurements collected in a controlled test environment containing known buried targets. Simulations and post-processing in Python are further used to generate and analyze GPR B-scans, providing cross-sectional images of the subsurface response.**

I. Introduction

GROUND Penetrating Radar (GPR) has been applied across diverse domains such as archaeology, civil engineering, and construction since the early to mid-1900s [1]. Traditional GPR systems consist of one transmitting and one receiving antenna mounted on handheld devices or ground vehicles such as carts or cars [2]. More recently, the use of small rotor-based unmanned aerial vehicles (UAVs) with inexpensive sensors has been shown to reduce cost, effort, and time in GPR measurements [3]. However, traditional radar systems are often expensive, bulky, and power-intensive, making them unsuitable for small UAV payloads. These limitations have led to interest in compact commercial SDRs for such applications.

The objective of this project is to determine whether a Software-Defined Ground Penetrating Radar (SD-GPR) can serve as a suitable replacement for traditional systems in detecting shallowly buried targets. To this end, a prototype SD-GPR was developed, encompassing both hardware components and software-defined radar processing. Its performance is evaluated by characterizing its detection capability against known targets and comparing the results to those of a COTS GPR and the replication of previously reported SD-GPR results [2]–[5].

A key design choice in this prototype is the use of a stepped-frequency continuous-wave (SFCW) waveform. SFCW is widely employed in ground-penetrating radar because it combines a high signal-to-noise ratio (SNR) with low power requirements. By transmitting discrete frequencies across a predefined band and measuring both magnitude and phase, it enables the reconstruction of a synthetic impulse response with high range resolution [6]. These advantages make SFCW particularly well-suited for lightweight platforms such as UAVs, where efficient power use and sensitivity are critical for detecting shallow targets. Figure 1 shows a time and frequency waveform for an SFCW signal.

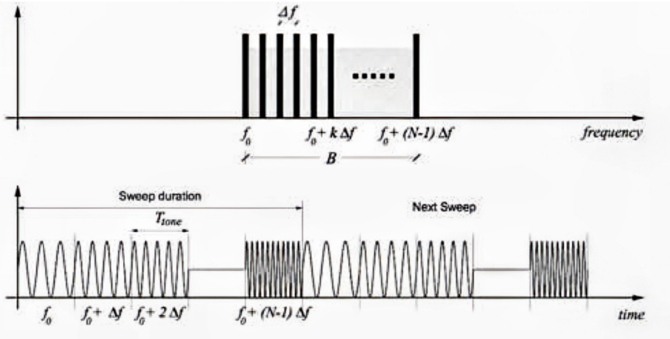


Fig. 1: SFCW Waveform in Time and Frequency [7]

II. Stepped Frequency Continuous Wave Radar

A. Theory of Operation

SFCW radar has been established for many decades [1]. The transmitter generates a synthetic pulse using N discrete frequencies spaced uniformly by a frequency step Δf. The frequencies are transmitted sequentially over a period t_{pri}, the pulse repetition interval (PRI). The frequency is given by

$$f_n = f_0 + n\Delta f \qquad n = 0, 1, 2, \ldots, N \tag{1}$$

where f_0 is the starting frequency. The total bandwidth of the system is defined as

$$B = N\Delta f \tag{2}$$

The system bandwidth is directly related to the range resolution, given by

$$\Delta R = \frac{v_m}{2B} \tag{3}$$

where v_m is the propagation speed in the half-space medium[1]. Because this ΔR is constant, the maximum unambiguous range of an SFCW system is given by

$$R_{max} = \Delta R \cdot N \tag{4}$$

[1] It is assumed that v_m is equal to the speed of light.

A reflection from a target at a given distance causes a two-way time-delayed and attenuated version of the transmitted signal to appear at the RF front-end. This signal is given by

$$s_n(\omega_n, t) = A_n \exp(-j(\omega_n(t+\tau))) \tag{5}$$

where A_n represents the attenuated amplitude of the received signal, $\omega_n = 2\pi f_n$, and τ is the two-way time delay. This signal is demodulated into an in-phase and quadrature (IQ) baseband signal, which preserves the magnitude and phase information for each received frequency [8]. The IQ baseband signal for all frequencies is given by

$$\begin{aligned} C_n(\omega_n, \tau) &= A_n[I_n(\omega_n, \tau) + jQ_n(\omega_n, \tau)] \\ &= A_n \exp(-j\omega_n\tau) = A_n \exp(-j\phi_n) \end{aligned} \tag{6}$$

where ϕ_n represents the phase shift induced by a reflection from a target at delay τ. A complex vector V would correspond to one range bin of size N:

$$V = [C_0, C_1, \ldots, C_{N-1}] \tag{7}$$

It can be observed that V contains magnitude and phase information for N frequencies separated by Δf which corresponds to the discrete Fourier transform (DFT) of some signal y_n. y_n can be described as the synthetic pulse response from the reflection and is then obtained by applying an inverse DFT (IDFT) on vector V:

$$y_n = \frac{1}{M} \sum_{i=0}^{M-1} C_i \exp\left[\frac{j2\pi ni}{M}\right] \tag{8}$$

where $0 \leq n < M$. Zero-padding V to a length L (a power of 2) before applying the IDFT increases the sampling density in the reconstructed range profile, providing finer interpolation of target responses. Taking the magnitude of the resulting transform gives:

$$|y_n| = \left| \frac{1}{L} \sum_{i=0}^{L-1} C_i \exp\left[\frac{j2\pi ni}{L}\right] \right| \tag{9}$$

If a target is present, the coherent combination of the stepped-frequency returns produces a synthetic pulse in the reconstructed range profile, located at the position corresponding to the target's round-trip time delay [2].

B. Advantages

A synthesized system such as this allows the received signal to be integrated for as long as is practical relative to the radar survey speed, providing a significant boost in SNR. The receiver bandwidth and corresponding thermal noise floor are also much lower than that of a time-domain radar [1], enabling effective operation in lossy media. Another advantage arises from transmitting each frequency individually rather than as a combined time-domain pulse. Doing so permits a higher average radiated power per spectral line, allowing equivalent performance with lower overall power requirements compared to pulsed systems. Finally, the operational frequency range of an SFCW radar can be flexibly tailored to match the material properties and target characteristics under investigation.

While SFCW radar offers high SNR, low power requirements, and considerable flexibility, it suffers from notable disadvantages. The large number of frequency steps N and the switching time required between them lead to long acquisition times and, consequently, long PRIs. Additionally, the unambiguous range is limited, making SFCW radar inherently better suited for short-range applications [2].

Aliasing is also a problem for SFCW systems where reflections from objects beyond R_{max} are detected, and it becomes impossible to distinguish between an object within the unambiguous range and an object outside this range. The only way to mitigate this phenomenon is to choose R_{max} large enough so that the signal scattered by a target deeper than R_{max} reaches the surface attenuated and is not detected by the receiving antenna. R_{max} should be based on the expected perceivable maximum depth [9].

III. Simulation

The goal of the simulation was to generate both an A-scan and a B-scan of a perfectly electrically conducting (PEC) cylinder buried in a half-space to verify that the chosen parameters and signal processing techniques could successfully detect an object in the intended environment, and will be referenced frequently in subsequent sections.

An A-scan represents the reflected signal as a function of time (or range) at a single radar position. Each peak corresponds to a reflection from a subsurface interface or target. Sequentially acquiring A-scans along a survey line produces a B-scan, a two-dimensional cross-sectional image of the subsurface. In B-scans, the horizontal axis corresponds to the radar's position, and the vertical axis represents time delay or depth. Figure 2 shows a B-scan where its A-scans are distinctly visible.

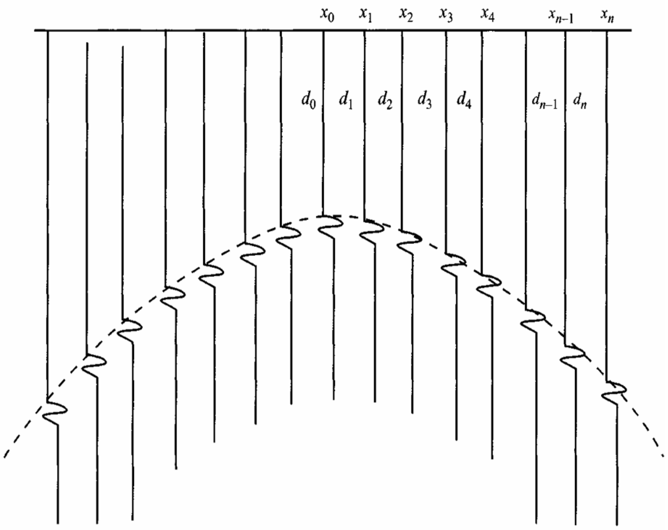


Fig. 2: B-Scan Showing Distinct A-scans [1]

The characteristic hyperbolic response in a B-scan arises from the acquisition of A-scans as the GPR passes over a

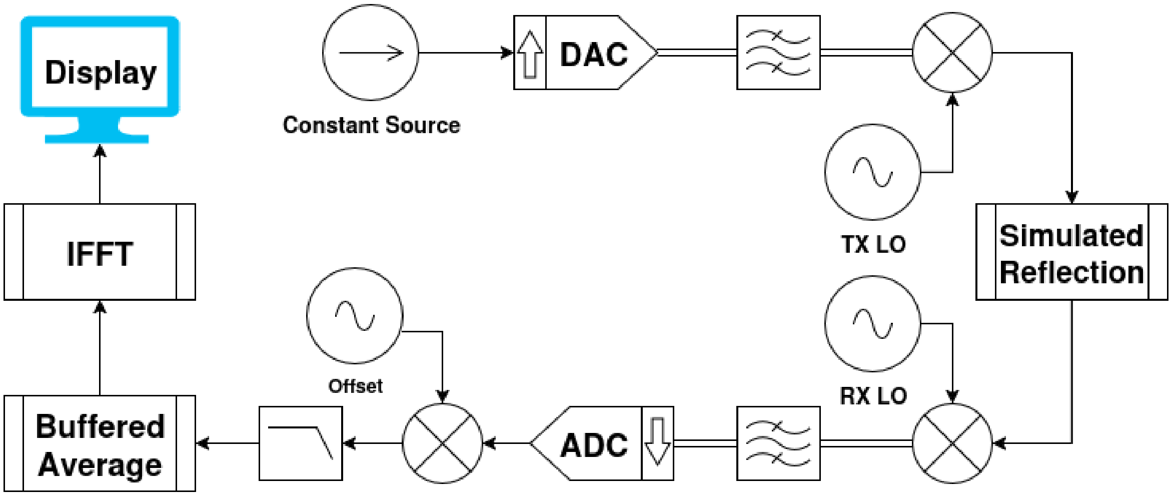


Fig. 3: Simulated Low-IF Architecture

subsurface target. The apex of the hyperbola (or the center of a cluster of peaks) indicates the lateral position of the target, while the depth is inferred from the corresponding time delay.

A. Radar Parameters

Table I lists the parameters that define the radar system's operation. These values are based on the project requirements for which this simulation was developed.

It is important to note that although the actual system sample rate is f_s, simulating analog modulation at higher carrier frequencies requires significantly more samples during all stages leading up to the simulated ADC. Section III-D provides further discussion of this sampling strategy.

Parameter	Value	Symbol
Sample Rate	6 MHz	f_s
Number of steps	100	N
Frequency Step	25 MHz	Δf
Intermediate Frequency	2 MHz	f_{IF}
Start Frequency	600 MHz	f_0
Stop Frequency	3.1 GHz	f_{N-1}
Total Bandwidth	2.5 GHz	B
Length of IFFT	2^{16} pts	L
Range Resolution	6 cm	ΔR
Max Unambiguous Range	6 m	R_{max}

TABLE I: Radar Parameters

B. Simulated Architecture

The final implementation of the SD-GPR is on a BladeRF SDR (see section IV-A), which has a direct conversion receiver (DCR) architecture. Direct conversion receivers can introduce a large DC offset from local oscillator (LO) leakage [2], [5]. This issue can be mitigated by shifting the DCR to a low intermediate frequency (low-IF) receiver and centering the baseband around an offset rather than DC before sampling. While this introduces additional challenges, such as the need to filter out image frequencies with an anti-imaging filter, these challenges are easier to overcome than the DC offset issue [5]. A low-IF receiver architecture was used and simulated in this work. Figure 3 illustrates the overall architecture used for the simulation.

In a traditional low-IF receiver, the signal processing chain proceeds as follows:

1) **Transmit path:** I and Q values are generated separately with a two-channel DAC.
2) **Upconversion:** This signal is mixed with the transmit LO before being radiated via the antenna.
3) **Reception:** The reflected signal (see Section III-C for generation details) is received and downconverted to a frequency offset from the desired carrier using a quadrature demodulator.
4) **Filtering:** I and Q channels are filtered for alias suppression.
5) **Sampling:** The signal is sampled. The ADC is simulated via decimation at f_s.
6) **Digital downconversion:** The sampled signal is shifted to a 0 Hz baseband and the DC offset is filtered out.
7) **Averaging:** 100 coherent samples are averaged to improve SNR [10].
8) **Spectral analysis:** Fourier analysis is performed, as described in Section III-E, to synthesize the final time-domain pulse.

C. The LTI Assumption and gprMax

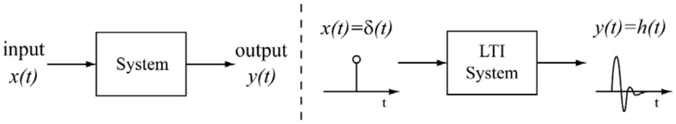
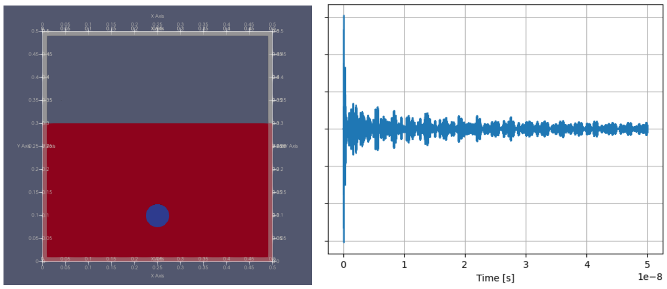


Fig. 4: LTI System

A GPR survey is modeled as a Linear Time-Invariant (LTI) system because the propagation medium and target geometry remain stationary over the duration of a single stepped-frequency sweep, and platform motion is sufficiently slow that Doppler effects are negligible. Under these conditions, the system response can be approximated as a scaling and shifting operation of the transmitted signal [11]. Meaning that if an impulse were used as a waveform for accurately simulating a GPR environment, the received signal (or the output of the GPR system) would be considered the system's impulse response, $h[n]$ (see Figure 4). Any valid radar waveform, x_n, can then be convolved with $h[n]$ to simulate the system. For an SFCW waveform, each output from the frequency synthesizer can be convolved with $h[n]$ to simulate a reflection.

The simulation of propagation, reflection from a target, and the return to the antenna was conducted using electromagnetic simulation software called "gprMax" [11] [12]. This software allows the setup of the environment by defining system boundaries, the target to detect, the half-space in which the target is buried, the placement of the TX and RX antennas, and the waveform. Figure 5a shows a visualization of the simulated environment. Sample gprMax outputs from a radar using a Gaussian Dot Norm waveform that are used as comparisons can be seen in Figures 6a and 6b.

In Figure 5a, the red region represents a half-space with a dielectric constant of 3, while the blue object represents a PEC cylinder[2]. Figure 5b shows the generated $h[n]$ when the waveform was switched from a Gaussian Dot Norm to an impulse, and is the function used for simulating the reflection. Figure 6b shows two pulses in the time-domain waveform (an A-scan): the first and largest pulse represents the direct coupling from the TX to the RX antenna, and the second, smaller pulse represents the reflection off the buried target. Figure 6a illustrates this same behavior, but in a B-scan mode after the antennas have swept the environment.



(a)

(b)

Fig. 5: Simulated GPR environment and corresponding impulse response generated by gprMax.

D. Sampling

To digitally simulate the analog signal processing of this system, a much higher sample rate had to be used than the f_s defined for the system. This "analog sample rate" will be referred to as $\tilde{f_s}$. If f_{N-1} is 3.1 GHz, then $\tilde{f_s}$ must be at least 6.1 GHz to satisfy the Nyquist theorem. Given that t_{step} is 50 μs, this means that at 6.1 GHz, each frequency would require at least 305e3 samples (assuming equal sampling across all frequencies). It would take approximately 244 MB to store the entire transmitted signal in RAM, which is feasible for most hardware setups.

For the convolution with $h[n]$ to be valid, both signals must have the same sample rate. To analyze the frequency components in the simulation, gprMax over samples at approximately 424 GHz[3]. A sample rate of 424 GHz for t_{step} yields about 21 million samples per frequency. Storing the entire transmitted signal in RAM would take approximately 17 GB, which is not feasible for most hardware setups.

To simulate this system while maintaining a manageable simulation size, both the synthesized frequency steps and the impulse response were resampled to 32.614 GHz. This sample rate was chosen as the maximum RAM usage the hardware running the simulation could handle while allowing an integer decimation rate of the impulse response, and minimizing information loss. The synthesized frequency steps were generated at this sample rate, and the impulse response was decimated by a factor of 13.

E. Simulation Operation

This system was simulated using Python 3.12.4 on a PC running Ubuntu 22.04. Before the Python program could be executed, the gprMax software was called to generate the impulse response. The Python program then loaded the impulse response, synthesized each frequency step, and stored it in a 2D NumPy array. To avoid spectral leakage and ringing in the final time-domain impulse, each generated frequency amplitude was ramped up using an envelope lasting for 4 cycles of the frequency.

The synthesized frequency steps are then convolved with the downsampled impulse response to simulate the reflection. These reflections follow the receiver signal path, as described in Section III-B. Fourier analysis can begin after coherent integration. The output from the coherent integration is the complex vector $\mathbf{V}$, as described in Section II-A, which is then prepared for the inverse FFT. It is also necessary to create the conjugate symmetry in the FFT to properly represent the negative frequency components. Once the vector $\mathbf{V}$ is padded and the conjugate reflection is added for the negative frequencies, the IFFT can be taken to generate the synthetic time-domain impulse.

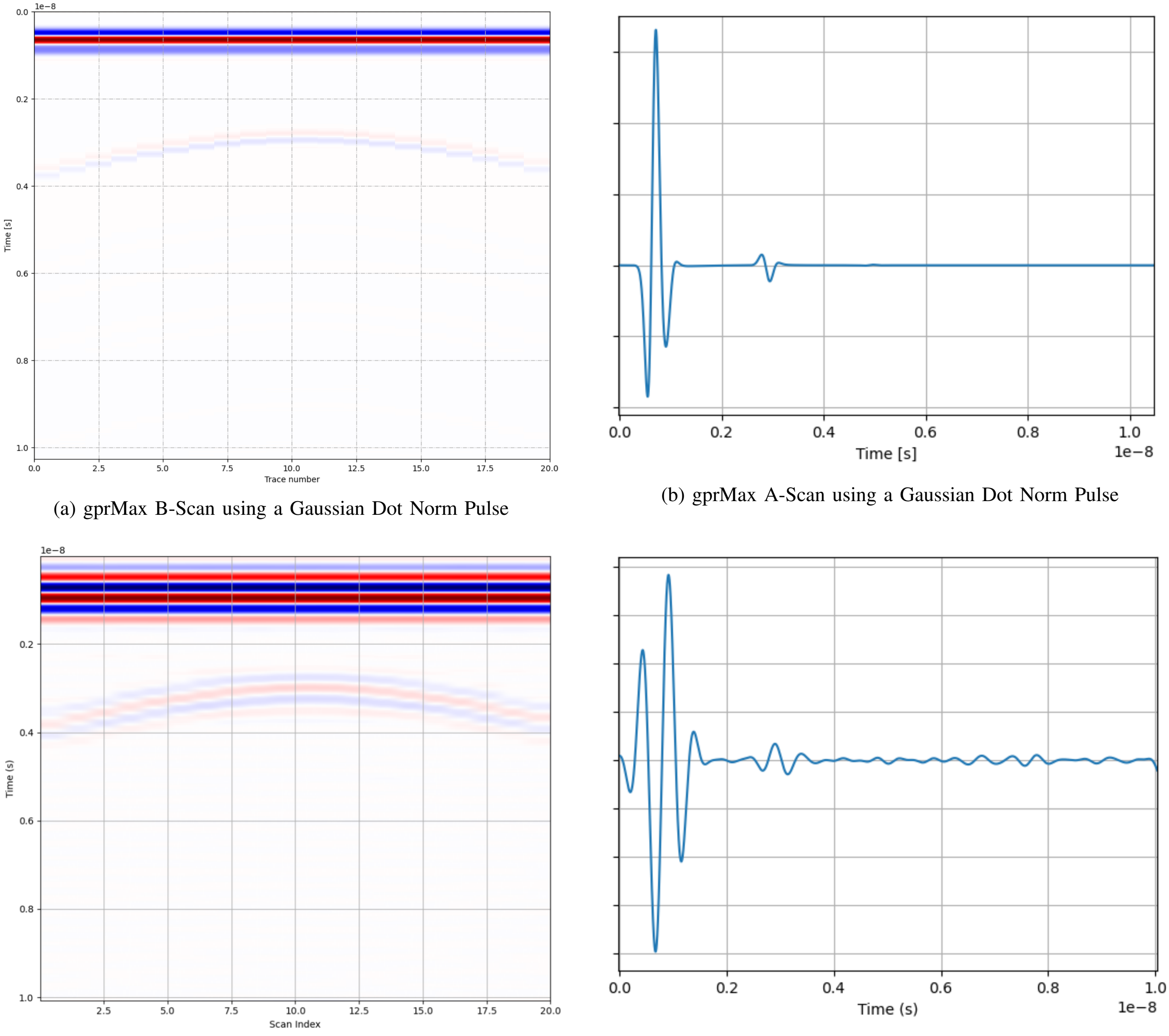
F. Simulation Results

Figures 6c and 6d show the results from running the simulation using the impulse function shown in Figure 5b that was generated using the geometry shown in Figure 5a. Comparing these results to Figures 6a and 6b demonstrates that the output of the Python simulation using the simulated SFCW radar setup matches the output generated by gprMax using a standard Gaussian Dot Norm waveform to validate the results.

Analyzing Figure 6 indicates that as the target moves closer to the antennas, it would be lost in the direct coupling between them. This poses a potential issue in practical applications when trying to detect targets that are within the same distance

[2] There are two antennas present in the simulation for the RX and TX, but gprMax does not display them in the visualization.

[3] This can vary across different simulations



(a) gprMax B-Scan using a Gaussian Dot Norm Pulse

(b) gprMax A-Scan using a Gaussian Dot Norm Pulse

(c) Python Simulation B-Scan Output

(d) Python Simulation A-Scan Output

Fig. 6: Comparison of simulation output from gprMax and the python simulation

as the antenna separation. Proper shielding placed between the antennas has been shown to reduce this coupling, allowing the target reflection to be more easily observed [2].

IV. Proposed SD-GPR Design

The proposed SD-GPR architecture for detecting shallowly buried targets, shown in Figure 7, is divided into two main subsystems: software and hardware. The software subsystem consists of the embedded software running on the BladeRF SDR, custom code utilizing the GNU Radio toolkit via GNU Radio Companion for real-time signal processing, and Python for post-processing and data display. The hardware subsystem consists of a bi-static antenna configuration utilizing Vivaldi antennas from RFSpace and the BladeRF SDR.

Signal generation, phase offset removal, range correlation, data display, and storage are performed by the software subsystem. The signal generation, phase offset removal, and data storage are done in real-time using the GNU Radio toolkit, while the range correlation and display are in post-processing after the full scans are taken. The correlation operation is performed via the Inverse Fast Fourier Transform (IFFT). Each transmitted frequency is synthesized by modulating a constant DC signal onto a carrier frequency that corresponds to the current f_n. By periodically stepping up the carrier frequency generated by the transmitter's local oscillator (TX LO), the stepped-frequency waveform is created. The receiver is a low-IF receiver, meaning the receiver's local oscillator (RX LO) is offset from the desired frequency step. Post sampling from the ADC, the signal is then downconverted further to center

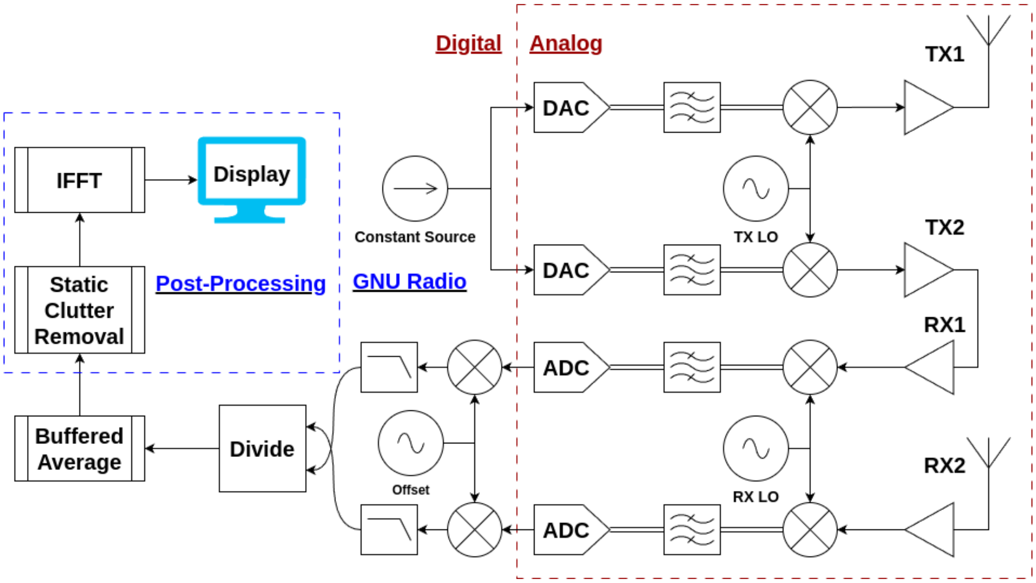


Fig. 7: Proposed SD-GPR Architecture

the baseband around 0 Hz and remove the DC offset.

A. Hardware Overview

The hardware subsystem is comprised of two main components: the BladeRF SDR and the antennas. The system is configured in a bi-static radar approach with transmitting and receiving antennas separated by 30 cm and isolated by conductive shielding placed between them.

It has been shown in previous implementations of SD-GPR units that Vivaldi antennas are well-suited for wideband radar due to a broad range of operating frequencies and the ability to produce high-quality GPR measurements [2]–[4], [13], [14]. TSA600 Vivaldi Antennas were selected to be used for this project.

The BladeRF 2.0 Micro is a 2x2 MIMO SDR that consists of the AD9361 IC transceiver from Analog Devices and the Intel Cyclone V FPGA. The transceiver is integrated with 12-bit DACs and ADCs and has a tuning range of 47 MHz to 6 GHz. It was designed for modern communication systems that require large bandwidths and full duplex communication on separate frequency channels [5], meaning there are two different oscillators. TX LO is used for both transmitters, and RX LO is used for both receivers, with both oscillators sharing a common clock. With a max sample rate of 56 MHz, a tunable filter can adjust the baseband bandwidth between 1 and 28 MHz. The transmitter block contains a variable RF power amplifier, and the receiver block contains a variable gain low-noise amplifier. Each has a maximum gain of 66 dB.

It is common for radar front-ends to make use of a single oscillator for the transmitter and receiver for better phase estimation of the target. It has been shown that the BladeRF TX LO and RX LO produce the same carrier frequency to a high degree of precision, but have different phases [5]. Phase estimation becomes difficult as the phase difference between TX LO and RX LO becomes non-deterministic [15].

To fix this relative phase error, the 2nd transceiver on the BladeRF is used as a reference line to measure a known line length. This reference measurement is divided out from the received measurement of the antenna later down the RX path to isolate the actual magnitude ratio and phase difference between the transmitted and received signal caused by some target.

B. Software Overview

The software subsystem performs the main controller, processing, and data handling. It consists of custom Python code for a main controller and a hardware interface created using the GNU Radio toolkit. The controller is used as a user interface (UI) to stop and start acquisition, as well as post-processing and data display after a scan is taken. The GNU Radio module is used to initialize the BladeRF based on the parameters passed to it from the main controller and to perform scans by stepping through each frequency transmitted and received. The GNU Radio module also performs some DSP, including the digital modulation to a 0 Hz baseband, filtering, dividing the reference phase offset out, and performing the buffered averaging.

Figure 8 shows a block diagram of the software subsystem. The software is started by executing the main controller Python script, which prints a simple text-based UI where the user can select to run either an A-scan or B-scan and input the parameters of the radar. Table I shows the default values. These input parameters were then passed to the GNU Radio module to initialize the BladeRF. The module will then generate a constant signal to be output from the DACs on both transmitter lines (TX1 and TX2). This constant signal would be up-converted with TX LO using quadrature modulators to create each frequency step f_n.

The receiving hardware then down-converts each received f_n to an intermediate frequency f_{IF} by RX LO using quadrature demodulators. Once filtered and sampled, the signal is then digitally modulated further by f_{IF} to a 0 Hz baseband and filtered again. After the phase offset is removed by division of the reference line, all the samples for each frequency are gathered in a buffer and coherently averaged together to save a single complex IQ sample per f_n.

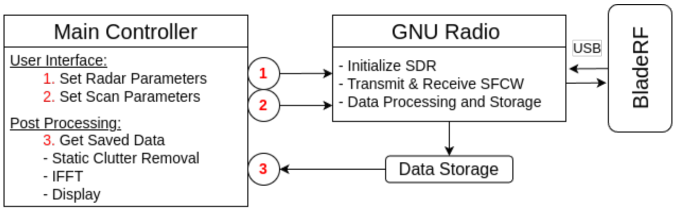


Fig. 8: Block Diagram for Software Subsystem

The module terminates once the specified number of scans has been completed for either an A-scan or a B-scan acquisition. An A-scan corresponds to a single PRI transmission with the SD-GPR held stationary, producing a one-dimensional depth profile of reflected amplitudes versus time or range. In contrast, a B-scan consists of multiple sequential A-scans collected as the SD-GPR is moved along a survey line, generating a two-dimensional cross-sectional image of the subsurface.

After acquisition, the main controller initiates post-processing by loading the saved samples and applying a static clutter removal algorithm to suppress direct coupling between the antennas. Each A-scan is then zero-padded to length L and mirrored using Hermitian symmetry to represent negative frequencies in the DFT. The IFFT is applied to synthesize the time-domain response for each A-scan. For B-scans, these time-domain A-scans are spatially stacked to form a complete image. The final result is displayed for user inspection, with the option to save the processed data.

V. Calibration

It has been shown that the magnitude and phase response vary with frequency in both the transmitter and receiver [5]. The proposed SD-GPR's transmitter and receiver chains need to be calibrated to provide accurate measurements. Calibration of the SD-GPR's transmitter and receiver chains is described in the following two sections.

A. Receiver Calibration

Two types of phase offsets, ϕ_{LO} and ϕ_{HW}, must be compensated to properly calibrate the receiver. The first, ϕ_{LO}, is the phase difference between the TX LO and RX LO. The second, ϕ_{HW}, is the deterministic phase offset introduced by signal propagation through the hardware and antennas before transmission. The oscillator phase difference ϕ_{LO} prevents any synthetic time-domain signal from conveying meaningful information, while the hardware-induced offset ϕ_{HW} shifts the time-domain pulse such that the return time of a reflected signal appears incorrect. Correcting ϕ_{LO} enables the creation of a meaningful time-domain pulse, and correcting ϕ_{HW} ensures that this pulse displays accurate measurements.

ϕ_{LO} is non-deterministic and cannot be compensated by applying a single fixed phase shift to the received signal. It is handled dynamically by using the second transceiver pair on the BladeRF as a reference line. Two measurements were taken for each sample: one from the reference line and one from the receiving antenna. The absolute difference in magnitude and phase caused by a target reflection can then be isolated from ϕ_{LO} by dividing the received signal by the reference signal. The propagation delay between the reference and receive paths is short compared to the timescale of LO phase variation, so any residual phase error due to the propagation time is negligible [5]. The received signal is normalized by the transmitted signal to isolate only the changes introduced by the propagation environment.

Unlike ϕ_{LO}, the offset ϕ_{HW} introduced by the hardware signal path is deterministic, since the signal path does not vary. It can be measured once and corrected by applying a constant phase shift to the received signal. This calibration was performed by placing a metal plate 1 m from the radar in free space. The time-domain reflection from the plate was measured, and the two-way travel time of the signal was calculated using equation 10.

$$\tau = \frac{2R}{c} \tag{10}$$

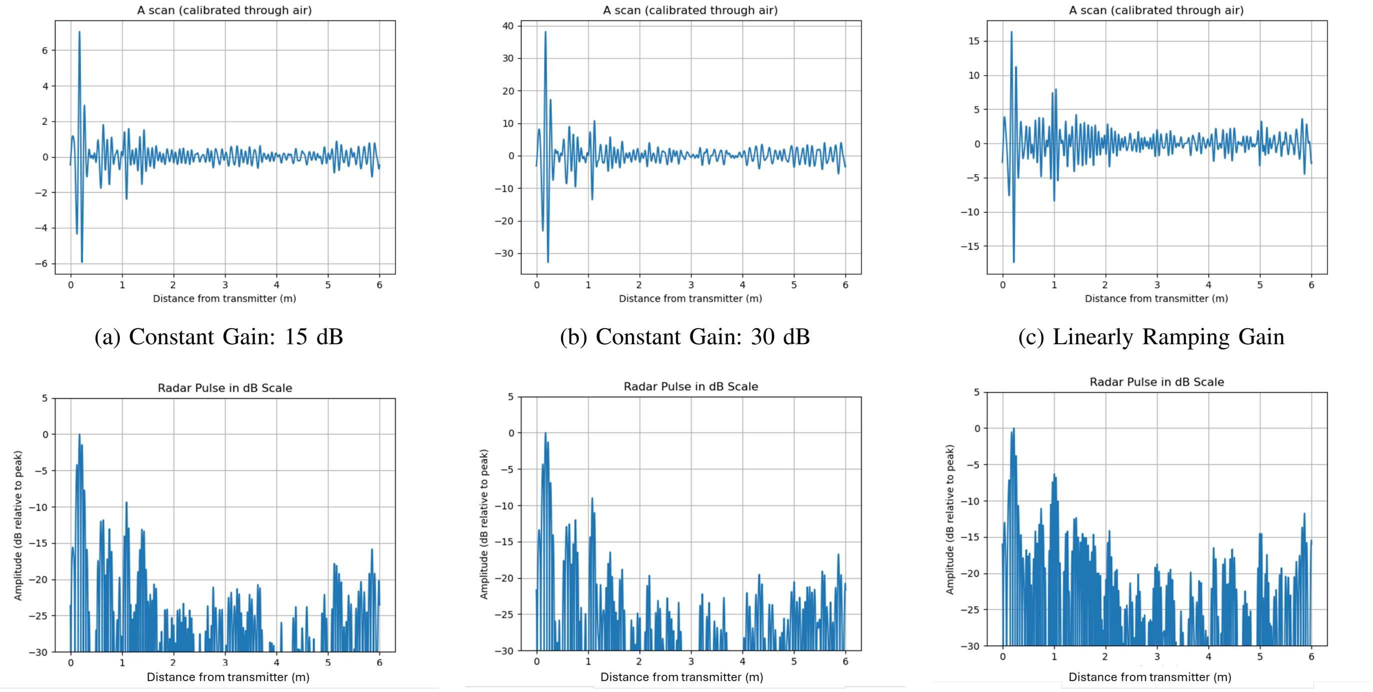
where R is the distance to the target and c is the speed of light. By substituting $R = 1$ m, the expected return time was calculated. The difference between the expected time and the measured time corresponds to the constant phase offset ϕ_{HW}. By applying this correction to received signals, the reflection from any target can be reconstructed in the time domain.

B. Transmitter Calibration

It can be observed that the transmit power frequency response varies as the frequency increases [5]. To increase the dynamic range of the receiver and maximize the power of the return signal, the transmitter gain should also be varied with frequency to accommodate this variation [2]. Another important consideration when selecting transmitter gain values is the effectiveness of shielding to reduce direct coupling. Direct coupling is problematic when observing targets located at the same distance as, or closer than, the antenna separation. Previous work has shown that most metallic shielding is more effective at attenuating high-frequency signals than low-frequency ones [2]. Consequently, shielding effectiveness can be improved by applying less gain to lower frequencies and more gain to higher frequencies.

The proposed calibration technique for this SD-GPR design was to linearly increase the gain with each frequency increment Δf. Figure 10 shows the output power of the BladeRF measured with a spectrum analyzer. The blue trace represents the absolute measured power using the default transmitter gain of 15 dB, while the orange trace represents the output power after the linearly ramped gain was applied.

An experiment was conducted to compare the effectiveness of linearly ramped gain against constant gain across the entire band. The test target was a soda can placed 1 meter away in free space. Figure 9 shows the results of these measurements.



(a) Constant Gain: 15 dB

(b) Constant Gain: 30 dB

(c) Linearly Ramping Gain

(d) Signal Power, 15 dB

(e) Signal Power, 30 dB

(f) Signal Power, Ramping Gain

Fig. 9: Results from testing the effectiveness of ramped versus constant transmitter gain

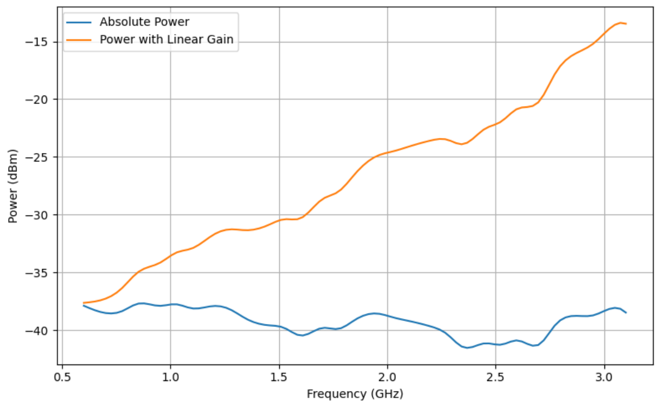


Fig. 10: Output power delivered to the antennas

The top row of plots in Figure 9 shows the reconstructed time-domain impulse response of a single A-scan of the soda can. In each case, the large initial impulse corresponds to direct antenna coupling, while the smaller spike at the 1-meter mark corresponds to the target reflection. The bottom row shows the relative signal power over time, where the spikes at approximately 0 m and 1 m represent direct coupling and the soda can reflection, respectively. The first two columns correspond to constant-gain tests (15 dB and 30 dB), while the third column corresponds to the linearly ramped gain shown in Figure 10.

One might initially assume that increasing the transmitter gain would improve the detectability of weak targets. However, comparison of the first and second columns shows that simply raising the gain amplifies both the desired signal and the unwanted noise and coupling by the same amount, yielding no practical improvement in detection. In practice, higher gain can also introduce nonlinear effects such as harmonic distortion in the transmitter chain, further reducing signal fidelity and potentially degrading overall performance. The linearly ramped gain increases the representation of higher-frequency components, which both improves shielding effectiveness (reducing direct coupling) and increases the relative amplitude of the target return. As seen in the bottom row, the ramped-gain configuration achieves a 3–4 dB improvement in the reflected signal compared to constant-gain cases.

VI. Experimentation

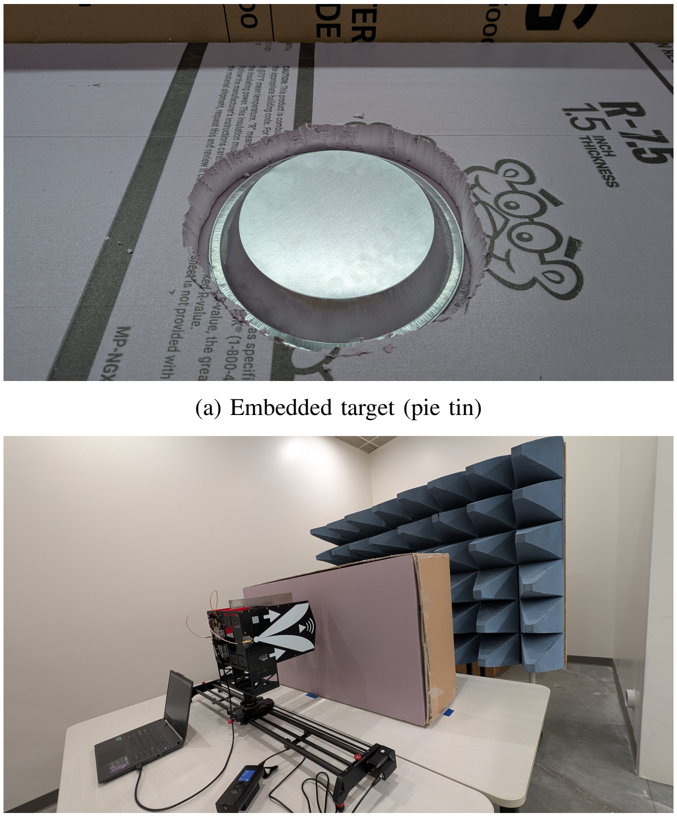
A. Setup

While normally aimed at the ground, the test rig for the proposed SD-GPR was configured to transmit horizontally so that Radar Absorbing Material (RAM) could be placed behind the target to suppress unwanted reflections during testing. The bi-static system was mounted on a 3D-printed bracket that attached to a motorized camera slider. The slider enabled controlled and repeatable sweeps of the GPR across the test environment.

The test environment consisted of the target and a lossy medium into which the target was embedded. The medium selected was extruded polystyrene (XPS) insulation foam board, chosen for its uniform structure and suitability as a controlled testing material. With a relative permittivity of $\epsilon_r \approx 1.1$ and a relative permeability of $\mu_r \approx 1$, XPS provides a low-loss environment ideal for low-power GPR experiments.

The targets embedded in the foam included a soda can and a pie tin. A laptop running Ubuntu 22.04 was connected to the BladeRF via USB to provide power, control, and data transfer. Figure 11 shows the physical test setup and one of the test targets, the pie tin, embedded in the foam.

For an A-scan, the slider remained stationary to capture a single recording. For a B-scan, the slider was activated to move the rig across the foam board. Once the B-scan was initiated, the SD-GPR transmitted and received each frequency while the slider carried the antennas across the test bed, continuing until the rig reached the opposite end of the track. After the sweep was completed, the post-processing stage of the Python controller was executed, and the results were displayed to the user, with an option to save the scan.



(a) Embedded target (pie tin)

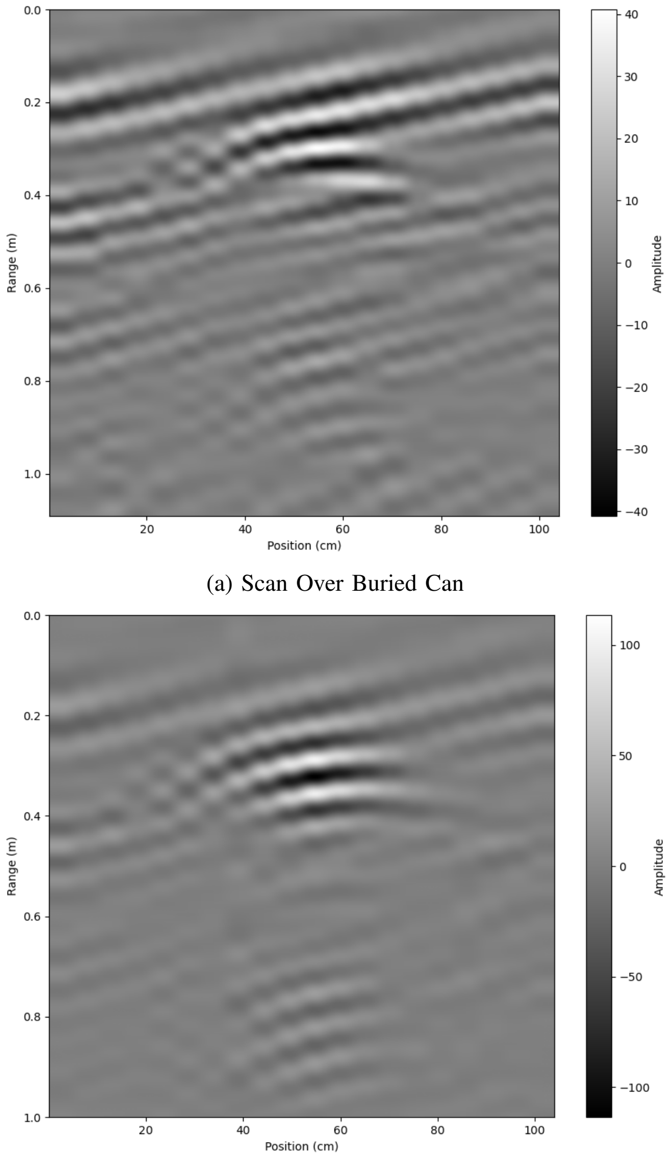
(b) Physical test environment

Fig. 11: Physical test environment and embedded target

A COTS GPR was used in the same test environment to validate the results obtained from the designed SD-GPR.

B. Results

For both the proposed SD-GPR and the COTS GPR, a B-scan was generated for each of the two targets embedded in the foam. In both cases, the target was buried approximately 50 cm from the edge of the test environment and 15 cm below the surface of the foam. Figure 12 presents the initial results obtained using the SD-GPR, while Figures 13a and 13b show the results from the COTS GPR.



(a) Scan Over Buried Can

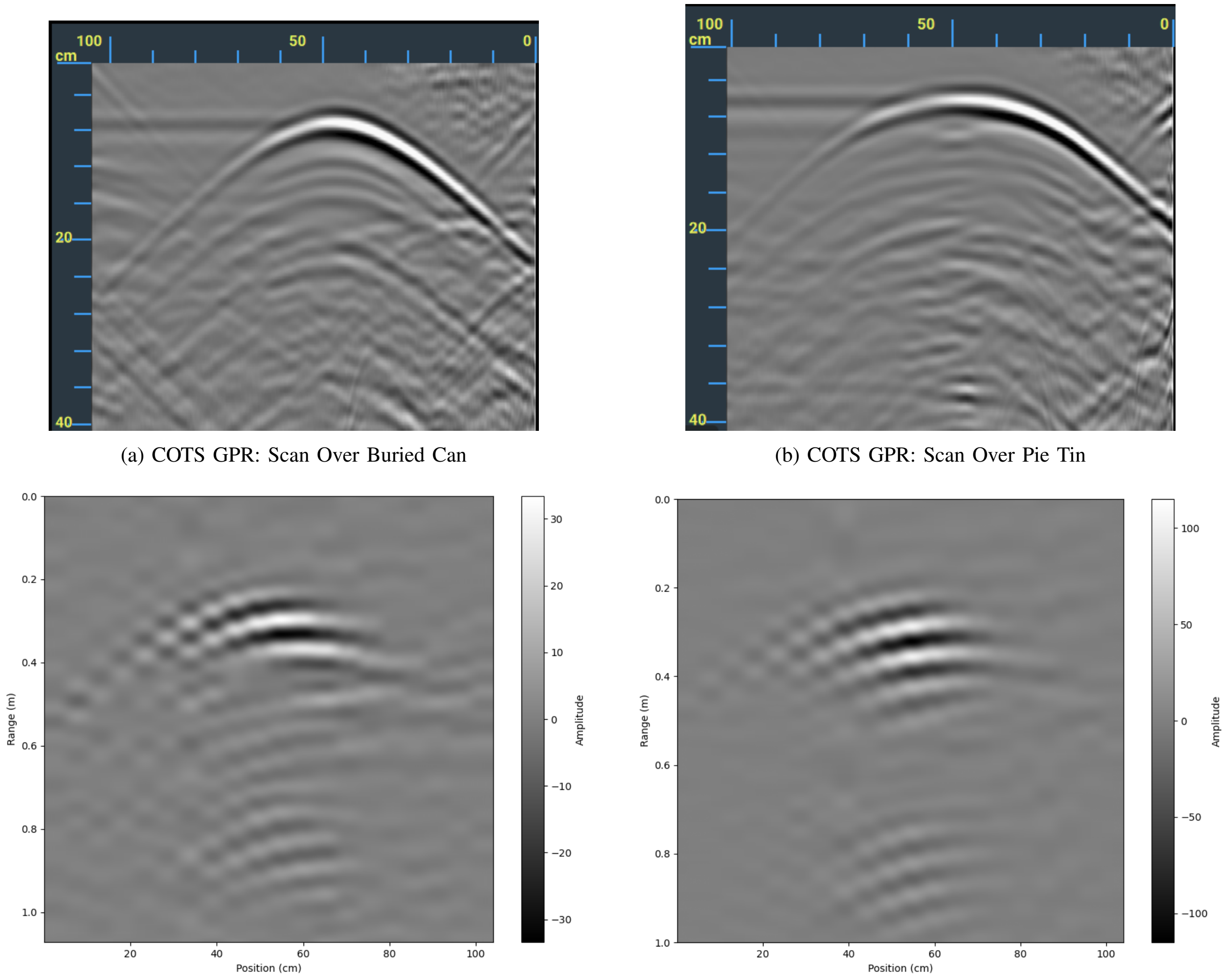
(b) Scan Over Pie Tin

Fig. 12: Initial Sub-Surface Results Using the SD-GPR

From Figure 12, it can be observed that when a target is too close to the antennas, its reflection can be obscured by direct coupling. Upon closer inspection, both the pie tin and soda can are faintly visible, but the strong coupling makes them difficult to clearly distinguish. A post-processing method for removing direct coupling was implemented, allowing the targets to be separated from the background medium more effectively.

A subtraction-based method of removing direct coupling was implemented due to its simplicity and effectiveness. A reference scan with no target present was acquired and subsequently subtracted from all future scans. Since direct coupling is relatively constant across measurements, this approach provides reliable suppression. Figures 13c and 13d show the results from re-running the scans with the additional reference scan and subtraction algorithm included.

The subtraction method proved effective in suppressing direct coupling, allowing the targets to be more easily distinguished.



(a) COTS GPR: Scan Over Buried Can

(b) COTS GPR: Scan Over Pie Tin

(c) SD-GPR: Scan Over Buried Can

(d) SD-GPR: Scan Over Pie Tin

Fig. 13: Comparison of sub-surface results obtained using the COTS GPR (top row) and the SD-GPR (bottom row).

VII. Discussion

Despite its simplicity, the subtraction method of removing direct coupling has several limitations, including how its effectiveness is sensitive to environmental and system changes. Relocating the radar, adjusting gain, or bandwidth settings typically requires acquisition of a new reference scan. Future improvements may include a dynamic or adaptive algorithm to remove static clutter in radar scans, which would more effectively remove the direct coupling without the need for the reference scan [10].

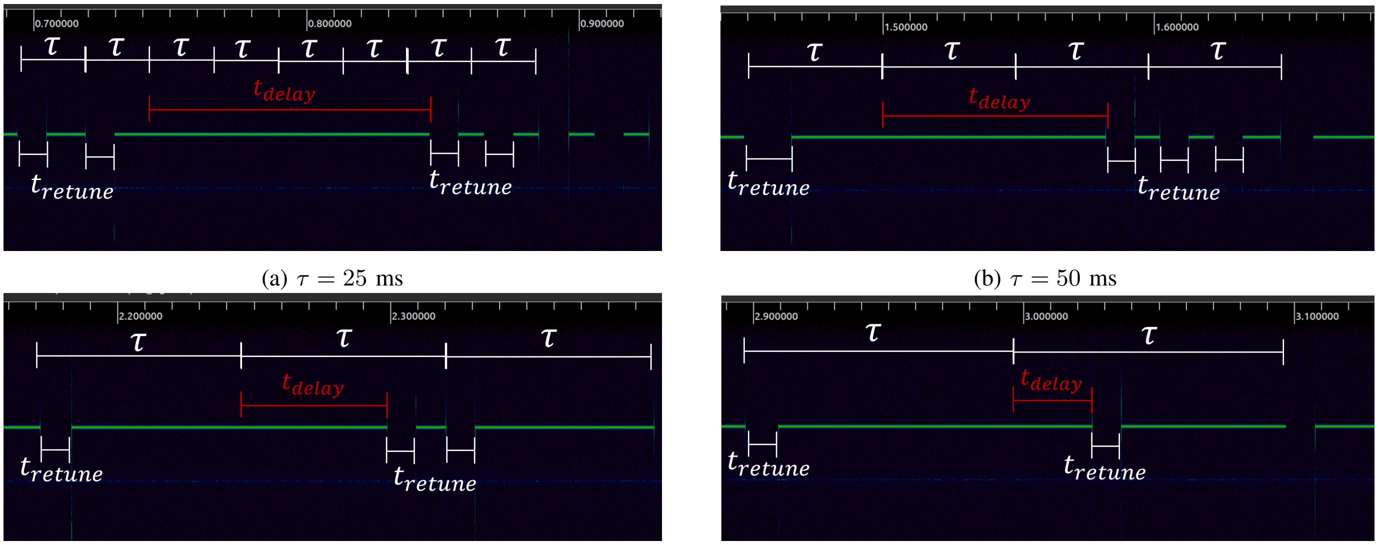
The performance metrics for each GPR can be seen in Table II. The absolute distance error, e_d, was calculated by finding the distance between the reported coordinates of each GPR for each target and the actual coordinates of each target, being a lateral position of 50 cm and a depth of 15 cm. The average distance error, $\bar{e_d}$, was calculated by taking the average between the pie tin and soda can errors for each GPR.

GPR	Target	Lateral	Depth	e_d	$\bar{e_d}$
COTS	Pie Tin	42 cm	5 cm	13 cm	11.5 cm
	Soda Can	46 cm	6 cm	10 cm	
SD-GPR	Pie Tin	55 cm	25 cm	11 cm	9 cm
	Soda Can	55 cm	20 cm	7 cm	

TABLE II: Comparison of reported target positions and absolute errors for COTS and SD-GPR.

Comparing the COTS GPR scans to the SD-GPR scans in Figure 13 and analyzing the average error between the two GPRs, it is evident that the proposed SD-GPR design demonstrates performance comparable to that of the COTS GPR in identifying shallowly buried targets.

Detailed analysis of Figure 13 shows that while the proposed SD-GPR can locate buried targets with the same accuracy as the COTS GPR, the scan resolution of the COTS GPR is significantly higher. The limitation found in the SD-GPR implementation that prevented higher horizontal resolution stems from the time required for the oscillators to switch between frequencies, referred to as the retune time t_{retune}.



(a) $\tau = 25$ ms

(b) $\tau = 50$ ms

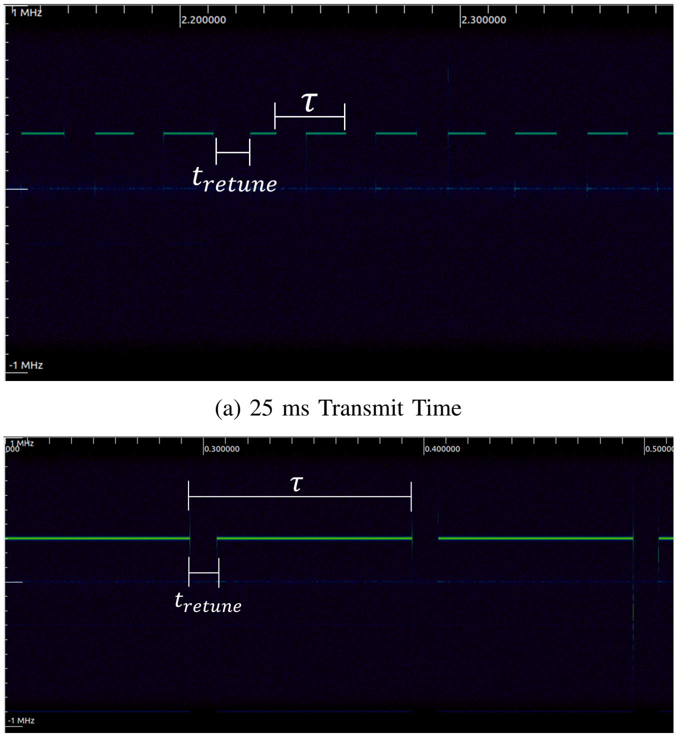
(c) $\tau = 75$ ms

(d) $\tau = 100$ ms

Fig. 14: Crossing 1.35 GHz.

A. Retune Time Analysis

To characterize t_{retune}, a simple wired-echo test was conducted. TX1 was directly connected to RX1, and the stepped-frequency waveform was saved to a file. The resulting spectrogram was displayed using the open-source software Inspectrum [16]. Figure 15 shows spectrograms for different transmit times. The green lines represent a signal being received at the offset frequency and the gaps between them are the SDR switching from one frequency to another.



(a) 25 ms Transmit Time

(b) 100 ms Transmit Time

Fig. 15: Spectrograms for 25 ms and 100 ms Transmit Times

The results show that, when using GNU Radio Companion (GRC) and Python, t_{retune} is approximately 10 ms. This delay appears as blank regions in the spectrogram, representing periods where the receiver is idle during oscillator retuning. This retune time is considerably slower than expected. The total time per frequency step, τ, can be expressed as

$$\tau = t_{retune} + t_{RX} \tag{11}$$

where t_{RX} is the duration of sample collection at each frequency.

The simulation described in Section III assumed $\tau = 50, \mu s$. However, the measured t_{retune} of $\sim$ 10 ms makes this assumption infeasible. Furthermore, t_{retune} was observed to be non-deterministic, varying by 1–2 ms between frequency steps.

An additional anomaly was identified when the oscillator frequency crossed 1.35 GHz: t_{RX} would be extended by an additional delay before retuning to the next frequency. Figure 14 illustrates this extended retune time for varying τ with the delay time being referred to as t_{delay}.

Regardless of the chosen τ, t_{RX} consistently increases to $\sim$110 ms when crossing the 1.35 GHz boundary. Although this occurs only once per sweep, it has a significant impact on operation with smaller τ values. In these cases, the software attempts to tune to the next frequency before the hardware has completed retuning, causing several frequency steps to be skipped. This occurs for any frequency step size Δf as long as the overall bandwidth B includes the 1.35 GHz boundary. To ensure reliable operation without rewriting the software, τ was set to 100 ms, approximately 2000 times slower than the value assumed in simulation.

At this setting, the SD-GPR has a total PRI of 10 seconds. Even at the slowest possible setting (approximately 3.5 minutes to move across the span), this limits the system to 20 A-scans across a single slider pass, which explains the lower horizontal resolution compared to the COTS GPR.

Further research into the BladeRF documentation shows the SDR should achieve retune times on the order of 400 μs [17], [18]. While no mention is made of the 1.35 GHz anomaly, these faster times are likely obtained when directly using the C/C++ API rather than GRC and Python.

It is hypothesized that the BladeRF oscillators operate in discrete bands and that crossing a band boundary (e.g., at 1.35 GHz) requires additional switching time. Future work using a lower-level implementation in C/C++ would enable further experimentation and validation of this hypothesis. Conducting the same experiment across multiple SDR platforms would also provide valuable insight.

VIII. Conclusion

This work presented the design and implementation of a software-defined ground penetrating radar using commercially available software-defined radios and GNU Radio Companion. Through experimental validation, it was shown that the system can effectively mitigate direct coupling using subtraction techniques, allowing buried targets to be more clearly distinguished. The comparison between the SD-GPR and a commercial off-the-shelf GPR demonstrated that the proposed design achieves target detection performance comparable to that of the COTS unit for shallowly buried targets despite the limited scanning speed and image resolution.

These results highlight the potential of SDR-based approaches as a cost-effective and flexible alternative to proprietary GPR systems. By leveraging open-source software and readily available hardware, researchers and practitioners can adapt and extend the platform for a wide range of subsurface sensing applications.

Future work will focus on preparing the SD-GPR for aerial deployment. In its current state, the system is not feasible for drone implementation due to timing constraints inherent to frequency-stepped SDR operation, which limit both measurement speed and image resolution. The present 10-second PRI is unsuitable for airborne surveys. Furthermore, data collection and processing were performed offline for simplicity. Future iterations must enable real-time, wireless data transmission and processing to support rapid drone-based surveys.

Additional efforts will aim to refine the static clutter removal algorithm. As the operational environment grows more dynamic, effectively suppressing stationary artifacts like ground reflections will become increasingly important for reliable subsurface imaging. Addressing these challenges is essential to achieving a practical, drone-mounted SD-GPR capable of real-time operation and high-resolution target detection.

IX. Acknowledgment

The authors would like to thank the 309th Software Engineering Group (309 SWEG) of the Air Force Sustainment Center (AFSC) at Hill Air Force Base, Utah, and the Department of Electrical and Computer Engineering at Weber State University for their support through the Educational Partnership Agreement (EPA) between the two organizations. The authors are grateful to the 309 SWEG for providing the funding necessary to acquire the required equipment and hardware. Special thanks are extended to Shreeya Maharjan for assistance with COTS GPR measurements and to Neil Rogers, Technical Chair of the GNU Radio Conference, for his valuable feedback during the preparation of this work.

X. References

References

[1] D. J. Daniels, *Ground Penetrating Radar - 2nd Edition*. The Institution of Electrical Engineers, London, United Kingdom, 2004.

[2] D. Šipoš and D. Gleich, “A lightweight and low-power uav-borne ground penetrating radar design for landmine detection,” *Sensors*, vol. 20, no. 8, 2020.

[3] S. Prager and M. Moghaddam, “Application of ultra-wideband synthesis in software defined radar for uav-based landmine detection,” in *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pp. 10115–10118, 2019.

[4] S. Prager, G. Sexstone, D. McGrath, J. Fulton, and M. Moghaddam, “Snow depth retrieval with an autonomous uav-mounted software-defined radar,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–16, 2022.

[5] J. Marimuthu, K. S. Bialkowski, and A. M. Abbosh, “Software-defined radar for medical imaging,” *IEEE Transactions on Microwave Theory and Techniques*, vol. 64, no. 2, pp. 643–652, 2016.

[6] J. M. Weiss, “Continuous-wave stepped-frequency radar for target ranging and motion detection,” in *The Midwest Instruction and Computing Symposium*, 2009.

[7] C. Gentile, *Application of Radar Technology to Deflection Measurement and Dynamic Testing of Bridges*. 01 2010.

[8] R. G. Lyons, “A quadrature signals tutorial: Complex, but not complicated.” https://www.dsprelated.com/showarticle/583.php, 2008. Accessed: 2025-08-20.

[9] R. Persico, *Introduction to Ground Penetrating Radar: Inverse Scattering and Data Processing*. Hoboken, NJ: Wiley-IEEE Press, 2014.

[10] M. A. Richards, *Fundamentals of Radar Signal Processing*. New York: McGraw-Hill Education, 2nd ed., 2014.

[11] C. Warren, A. Giannopoulos, and I. Giannakis, “gprmax: Open source software to simulate electromagnetic wave propagation for ground penetrating radar,” *Computer Physics Communications*, vol. 209, pp. 163–170, 2016.

[12] C. Warren, A. Giannopoulos, and I. Giannakis, “gprmax: Electromagnetic simulation software.” https://www.gprmax.com/, 2025. Accessed: 2025-04-24.

[13] S. Prager, T. Thrivikraman, M. Haynes, J. Stang, D. Hawkins, and M. Moghaddam, “Ultra-wideband synthesis for high-range resolution software defined radar,” in *2018 IEEE Radar Conference (RadarConf18)*, pp. 1089–1094, 2018.

[14] D. J. Daniels, W. van Verre, F. Podd, and A. J. Peyton, “Antenna design considerations for ground penetrating radar landmine detection,” *IEEE Transactions on Antennas and Propagation*, vol. 70, no. 6, pp. 4273–4286, 2022.

[15] C. H. J. Jenks and S. Pennock, “The use of a software defined radio as an ofdm gpr,” in *2017 9th International Workshop on Advanced Ground Penetrating Radar (IWAGPR)*, pp. 1–4, 2017.

[16] miek, “Inspectrum: A tool for analysing captured radio signals.” https://github.com/miek/inspectrum, 2025. Accessed: 2025-08-30.

[17] Nuand, “bladerf: Software, firmware, and hdl for the bladerf sdr platform.” https://github.com/Nuand/bladeRF, 2025. Accessed: 2025-08-20.

[18] Nuand, *libbladeRF Documentation, Version 2.5.0*. Nuand LLC, 2025. Accessed: 2025-08-20.