Progress In Electromagnetics Research B, Vol. 90, 91–108, 2021

Hardware Enabled Acceleration of Near-Field Coded Aperture Radar Physical Model for Millimetre-Wave Computational Imaging

Rahul Sharma[1, *], Okan Yurduseven[1], Bhabesh Deka[2], and Vincent Fusco[1]

Abstract—There is an increasing demand in real-time imagery applications such as rapid response to disaster rescue and security screening to name a few. The throughput of a radar imaging system is mainly controlled by two parameters; data acquisition time and signal processing time. To minimize the data acquisition time, various methods are being tried and tested by researchers worldwide. Among them is the computational imaging (CI) technique, which relies on using coded apertures to encode the radar back-scattered measurements onto a set of spatio-temporally incoherent radiation patterns. Such a CI-based imaging approach eliminates the requirement for a raster scan and can substantially simplify the physical hardware architecture. Equally important is the processing time needed to retrieve the scene information from the coded back-scattered measurements. In CI, the simplification in the hardware layer comes at the cost of increased complexity in the signal processing layer due to the indirect mapping and compression of the scene information through the spatio-temporally incoherent transfer function of the coded apertures. To address this particular challenge, this paper presents a hardware-based solution for CI signal processing using a Field Programmable Gate Array (an Xilinx Virtex-7 (XC7VX485T) FPGA chip) architecture. In particular, the proposed method consists of calculating the CI sensing matrix using the FPGA chip and storing it on the FPGA platform for image reconstruction. For the adjoint operation, the calculated sensing matrix is applied on the measured back-scattered waves from the target object. We demonstrate that the FPGA based calculation can reach 21.9 times faster speed than conventional brute-force solutions.

1. INTRODUCTION

Radars have been used for detecting and ranging objects ever since they came into existence. They are also used for imaging purposes. Imaging at millimetre-waves (mmW) has many applications such as security screening [1–3], automotive radars [4, 5], coincidence imaging [6], autonomous robotics [7–9], remote sensing [10–12], and concealed weapon detection [13–15]. In comparison to optical modalities, mmW signals can penetrate most optically opaque materials and operate in all-weather conditions. Moreover, unlike X-rays, radiation at mmW frequencies does not exhibit ionizing effects, and hence does not pose health hazards. The mmW spectrum contains several frequency bands suitable for both short range and long range communication and radar networks [16]. In comparison to optical and X-ray frequencies, imaging at such longer wavelengths often requires a larger aperture and associated hardware to achieve the desired resolution [17]. A large aperture is generally synthesized by mechanical raster scanning (Synthetic Aperture Radar, SAR) [14, 18, 19] or by electronic beam forming (phased arrays) [20, 21]. A significant limitation associated with these techniques is that they require beam synthesis to sample the scene information on a pixel-by-pixel basis. As a result, both these methods produce excellent imaging performance, and they often come at the cost of large bulky systems, high

Received 23 November 2020, Accepted 28 December 2020, Scheduled 8 January 2021
* Corresponding author: Rahul Sharma (rsharma05@qub.ac.uk).
[1] Institute of Electronics, Communications and Information Technology, Queen's University Belfast, UK. [2] Department of Electronics and Communication Engineering, Tezpur University, India.

power consumption, and slow data acquisition time [22]. To improve such conventional methods, a new perspective, computational imaging (CI), has recently gained significant traction [23–27]. CI approaches can relax hardware constraints to a great degree and enable alternative aperture architectures to be explored. Different CI techniques or algorithms can be employed to counter the conventional raster scanning requirement of SAR and phased-array based modalities. One such technique is frequency diversity, which uses frequency-diverse apertures that can generate distinct radiation patterns as a function of frequency [28]. Such waveforms can multiplex a scene information into a set of backscattered measurements, which can be decoded using computational algorithms. Another such example is a coded aperture, in which the transmitted and received wavefronts probing the imaged scene and collecting the back-scattered information are passed through a set of masks with spatio-temporally varying transparency values [29–32]. Such a modulation can be realized using semiconductor based elements, such as PIN diodes and varactors, to load the radiating elements across the coded aperture and tune their radiation characteristics to create spatio-temporally varying masks [31–33]. This spatio-temporal mapping aperture concept can be thought of as a large aperture antenna that can radiate complex, quasi-random radiation patterns or measurement modes. A single frequency operation and measurement mode synthesis through dynamically reconfiguring the antenna aperture brings several advantages from a hardware perspective such as faster data acquisition, simplified synthesizer architecture for single frequency operation, and increased system phase calibration robustness [33].

The total operation time of a radar imaging system is dictated primarily by the data acquisition speed and the computational complexity of the signal processing layer. Adopting the CI concept substantially simplifies the hardware architecture and improves the data acquisition speed. However, this simplification comes at the cost of increased computational complexity on the reconstruction layer because CI modalities collect the scene information in an indirect manner by illuminating the scene with spatio-temporally incoherent field patterns and encoding the back-scattered measurements through the transfer function of the coded aperture. Hence, retrieving the scene information back from these compressed measurements conventionally requires the processing of large sensing matrices, increasing the computational burden for the retrieval problem [28].

Recently, various parallelization techniques have been shown to be a promising candidate to handle large datasets in conventional imaging modalities, such as general-purpose computing on graphics processing units (GPGPUs) [34–37] and field programmable gate arrays (FPGAs) [38–42]. Among them, FPGAs are particularly promising as they offer customer reprogrammable hardware-based solutions to maximize the image reconstruction throughput on the signal processing layer while giving us the liberty to implement problem specific hardware and allow more dedicated special-purpose architectures. This becomes particularly important when the algorithm is implemented as a long pipeline, which is the case in this paper. Moreover, because an FPGA can be configured to directly access hardware I/O without the latency of internal bus structures common to GPUs, we propose using an FPGA-based hardware solution to parallel-process the sensing matrix calculation in CI imaging. Particularly, this paper demonstrates the application of an FPGA integrated forward-model to parallelize the reconstruction step for near-field coded aperture radar imaging at mmW frequencies. FPGA-based radar research has recently received considerable attention, such as in [40–42], most of the research deals with the traditional, multi-pixel SAR-based solutions. This paper presents the physical model for a single-pixel mmW CI radar system, eliminating the need for raster scanning, and instead, leveraging spatio-temporally incoherent bases to probe the scene information in a compressive manner. Therefore, the physical model implemented using the FPGA-accelerated technique in this paper fundamentally differs from the previous FPGA-based radar research. To the best of our knowledge, this is the first time that the physical model of a coded-aperture based CI system is parallelized on a hardware layer enabled by an FPGA platform. Specifically, by performing a discreet calculation of the sensing matrix on the FPGA, we achieve a substantial simplification in the computational burden of the developed forward-model for the CI problem. It is worth mentioning that parallelization of the sensing matrix using an FPGA-assisted hardware layer can have several challenges, such as the limited memory capacity and the need to convert the data to a hardware compatible form (binary or hex) before being used for computation — a process that can affect the quality of the reconstructed images. However, we demonstrate that the fast computation capabilities enabled by the presented FPGA integrated CI forward-model outweigh these challenges, paving the way for real-time mmW CI radar technology.

The remainder of this paper is organized as follows. In Section 2, we provide a brief insight into the CI concept leveraging coded apertures as an enabling technology. Section 3 demonstrates the proposed FPGA-based framework to parallel-process the sensing matrix for the forward-model and the adjoint operation. Section 4 presents the imaging results and provides a quantitative analysis on the reconstruction metrics for the investigated computational technique with and without the FPGA. Finally, Section 5 presents the concluding remarks.

2. COMPUTATIONAL IMAGING USING CODED APERTURES

A coded aperture can be considered as a computational aperture that can radiate spatio-temporally varying radiation patterns to probe the scene information [30, 33]. A distinctive feature of this technique is that the radiation pattern of the aperture changes substantially as new imaging modes are generated [43]. The spatio-temporal modulation of the radiated fields probing the scene information is achieved by creating a set of dynamic masks to vary the field patterns radiated by the aperture [44, 31, 32]. In this technique, the encoded back-scattered radar measurements are correlated to the imaged scene through the *forward-model*, which can be given as follows:

$$\mathbf{g}_{Mx1} = \mathbf{H}_{MxN}\mathbf{f}_{Nx1} + \mathbf{n}_{Mx1} \tag{1}$$

In Equation (1), $\mathbf{g}$ denotes the back-scattered measurements; $\mathbf{f}$ is the reflectivity distribution of the descritized voxels within the imaged scene (or region of interest, RoI); $\mathbf{H}$ is the sensing matrix; and $\mathbf{n}$ is the measurement noise, which is modeled as a Gaussian distribution with zero mean and governed by system signal-to-noise (SNR) level [45]. It should be noted that the physical model developed in this paper does not directly consider hardware specific losses, such as the free-space propagation loss and efficiencies of the coded apertures. However, the back-scattered measurements, $\mathbf{g}$, exhibit a finite SNR level of 20 dB to ensure a realistic channel response in the presence of additional loss factors. Our previous works in near-field imaging radars [22, 45] justify this SNR selection.

For the forward-model defined in Equation (1), we use the scalar wave definition (single polarization) and adopt the bold font only to denote the vector-matrix notation. It should be noted that a polarimetric approach to probe different polarization bases in reconstructed images is also possible, and in this case, the forward-model in Equation (1) can be solved individually for the desired polarization bases [46, 47]. From Equation (1), it is evident that the mapping of the scene onto measurements is achieved through the sensing matrix of the radar aperture. The sensing matrix is of size $M \times N$, where M is the number of measurements ($number\ of\ transmitters = 1\times\ number\ of\ receivers = 1\times\ number\ of\ masks = M$), and N is the number of pixels used to define the scene.

In this work, as depicted in Figure 1, we make use of the active modulation technique to create the spatio-temprorally varying field patterns from the aperture at a single frequency. This suggests that for each measurement mode, $1 \rightarrow M$, the aperture distributions of the transmitting and receiving antennas are modulated, which collectively result in a quasi-random basis to probe the scene information. As a result, each row in the sensing matrix corresponds to an individual modulation mask for a given transmitter-receiver pair at a particular frequency. As evident from Figure 1(a), the system uses a single channel for the transmitting and receiving coded apertures.

The depiction in Figure 1 is made for a single measurement of an arbitrary scene encoded by a single transmitting-receiving mask pair. On the other hand, using the coordinate system defined in Figure 1(a), the columns of the sensing matrix are formed by the dot product of the transmitter and receiver field patterns for each voxel in the scene as follows:

$$\mathbf{H} = \mathbf{E}_{Tx}\mathbf{E}_{Rx} \tag{2}$$

The calculation in Equation (2) is done for all measurement modes (M in total) for all voxels in the scene with coordinates $\mathbf{r}(x, y, z)$. For the calculation of the sensing matrix in Equation (2), we make use of the first Born approximation, suggesting that the total field inside the imaged scene is proportional to the radiated fields by the transmitting and receiving apertures, $\mathbf{E}_{Tx}$ and $\mathbf{E}_{Rx}$, respectively, and is not perturbed by the scattered field [48, 49]. At single frequency, the terms $\mathbf{E}_{Tx}$ and $\mathbf{E}_{Rx}$ are the radiated electric field patterns from the coded apertures propagated to the scene using the Green's function [30]

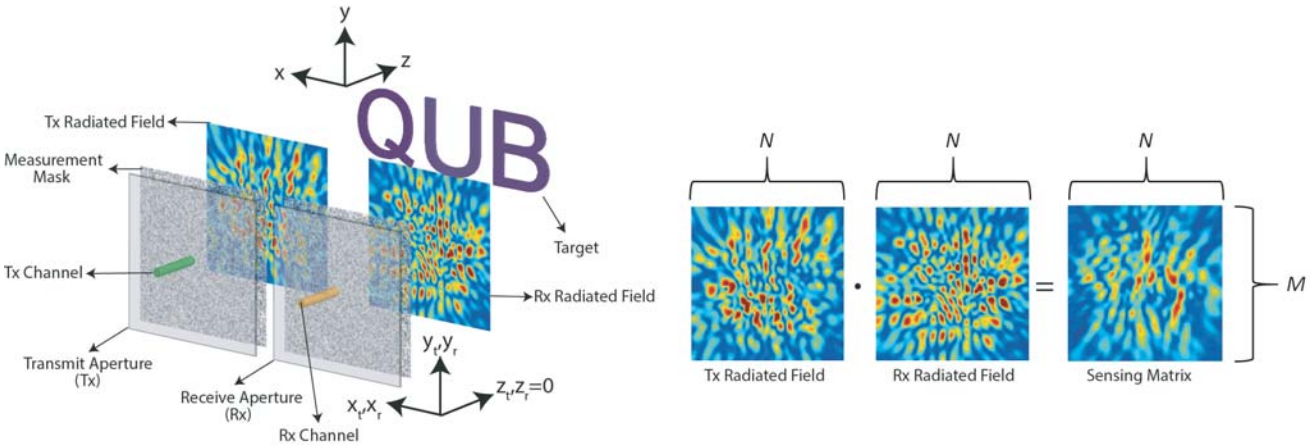

Figure 1. Imaging layout. (a) CI setup depicting two coded apertures, transmit (Tx) and receive (Rx), and an object (the word "QUB") located within the scene. The co-ordinates of the transmit aperture, receive aperture and scene are $\mathbf{r}_t(x_t, y_t, z_t = 0)$, $\mathbf{r}_r(x_r, y_r, z_r = 0)$, and $\mathbf{r}(x, y, z)$, respectively. The aperture radiated fields are also depicted for the selected measurement mask configuration. (b) Depiction of the calculation of the sensing matrix for the selected measurement mask configuration.

as follows:

$$\mathbf{E}_{Tx}(x, y, z) = \sum_{x_t} \sum_{y_t} \mathbf{J}_{Tx}(x_t, y_t) \times e^{-jk\sqrt{(x_t-x)^2+(y_t-y)^2+(0-z)^2}} \tag{3}$$

$$\mathbf{E}_{Rx}(x, y, z) = \sum_{x_r} \sum_{y_r} \mathbf{J}_{Rx}(x_r, y_r) \times e^{-jk\sqrt{(x_r-x)^2+(y_r-y)^2+(0-z)^2}} \tag{4}$$

In Equations (3) and (4), $\mathbf{J}_{Tx}(x_t, y_t)$ and $\mathbf{J}_{Rx}(x_r, y_r)$ are the surface current distributions on the transmitting and receiving antenna apertures, which are modeled as a random set of complex numbers forming the measurement masks for M different measurements, and k represents the wave number. The values of $\mathbf{J}_{Tx}(x_t, y_t)$ and $\mathbf{J}_{Rx}(x_r, y_r)$ range from $[-1.0 - j1.0]$ to $[+1.0 + j1.0]$ to account for both the positive and negative phases. From Equations (1) and (2), it is evident that the forward-model is of a linear form, which is the outcome of the first Born approximation [50]. Solving Equation (1) to recover an estimate of the scene information, $\mathbf{f}_{est}$, is an inverse problem. For the CI technique employed in this work, we can leverage several algorithms to this end, such as matched-filtering [45] and least-squares techniques [51, 52]. The matched-filter technique is advantageous in that it is a single-shot reconstruction algorithm, and hence the computational load of this algorithm can be minimal. Therefore, in this work, it is the choice for the adjoint operation to recover $\mathbf{f}_{est}$ in Equation (1). Using the matched-filter technique, $\mathbf{f}_{est}$ can be retrieved by means of a simple phase compensation of the sensing matrix, $\mathbf{H}^{\dagger}$, applied to the computational back-scattered measurements of the scene, $\mathbf{g}$, as:

$$\mathbf{f}_{est} = \mathbf{H}^{\dagger}\mathbf{g} \tag{5}$$

In Equation (5), symbol † denotes the complex conjugate operation applied to the sensing matrix, $\mathbf{H}$. It should also be noted that for a given RoI and radar aperture, the sensing matrix needs to be calculated just once, and it is not dependent on the target object in the scene that is being imaged. However, if RoI parameters, such as RoI centre, size and discretization of the scene, are varied, then the sensing matrix, $\mathbf{H}$, needs to be recalculated. Therefore, whereas it is convenient that we keep the RoI parameters fixed in this work, the developed solution is generic and also applicable to such scenarios. Moreover, from Equations (1) and (5), it is evident that the calculation of the sensing matrix, $\mathbf{H}$, is a prerequisite step to recover an estimate of the scene information $\mathbf{f}_{est}$ from the compressed back-scattered measurements, $\mathbf{g}$. Therefore, in this work, we consider the calculation of the sensing matrix as part of the reconstruction step.

For the mmW imaging problem at hand, reducing the acquisition time is of the essence, especially in applications where the evaluation process needs to be done in real-time, such as security-screening and nondestructive testing. This is especially important for cases where the scene dynamics change rapidly to prevent unwanted effects, such as motion blur [53, 54]. As the total speed of an imaging process is governed by not only the data acquisition time but also by the reconstruction time, it is vital that the reconstruction time is also minimized. However, achieving this goal for CI can be challenging. This is because the reconstruction process consists of calculating the large sensing matrix, **H** (Equation (2)), and the final scene estimation (Equation (5)). In this process, a major chunk of the total reconstruction time is taken up by the system to calculate the sensing matrix. Hence, speeding up the computation process of the sensing matrix is vital for facilitating the reconstruction step. Therefore, in this work, a major emphasis is given to the fast computation of the sensing matrix by means of a hardware based parallelization principle. In CI, calculating the adjoint operation for the sensing matrix can require the inversion of large sensing matrices. As an example, in [28], Yurduseven et al. showed that matrices as large as 90 GB needed to be inverted to recover mmW images in CI. Therefore, calculation of the sensing matrix can be quite intensive computationally, especially for cases where the CI platform uses multiple frequencies and multiple apertures (as in a MIMO platform in [28]). As a result, one can conclude that the CI framework, while substantially simplifying the hardware layer, increases the computational burden of the image reconstruction process. In order to speed up this complex image reconstruction step, computations can be carried out using parallel-programming architectures. In this work, an FPGA based hardware layer is added to facilitate the calculation of the sensing matrix **H** for the CI forward-model in Equation (1) and adjoint operation in Equation (5). The FPGA implementation of the CI imaging concept is detailed in the next section.

3. FPGA IMPLEMENTATION FOR THE CI MMW RADAR

This section describes, in detail, how the FPGA performs the calculation of the sensing matrix, **H**, for the CI forward-model in Equation (1) and the adjoint operation in Equation (5). The role of the FPGA in the whole computation process is depicted in Figure 2. The scene and aperture parameters

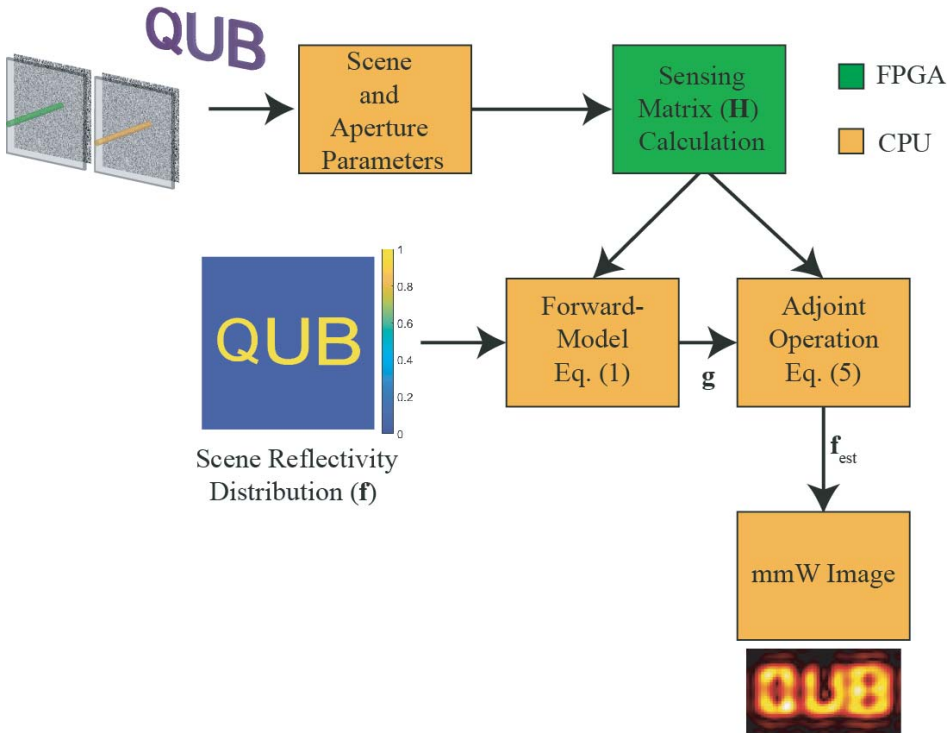

Figure 2. Figure outlines the roles of both CPU and FPGA in the computation process.

(aperture size, number of transmitting and receiving coded antennas, number of pixels in the imaged scene, etc.) are chosen in the design stage of the forward-model. This step is performed on the CPU and is communicated to the FPGA. The FPGA uses its memory and fast computation capabilities enabled by its parallel-architecture to store and compute the sensing matrix **H** values in real or quasi-real time. After completion, the FPGA communicates back the results to the CPU to perform the rest of the computation to recover an estimate of the scene information from a set of M spatio-temporally incoherent coded measurements.

The target FPGA device used in this work is XC7VX485T (Virtex 7) [55]. Since the forward-model has been developed in a complete software-based environment (MATLAB), the Xilinx System Generator^{TM} has been used to act as a bridge between the software and the hardware environment as it aids in checking for feasibility in introducing hardware elements in the model. Therefore, before programming the FPGA, the validation for hardware implementation is carried out using the Simulink block models via the System Generator^{TM}. After this check has been carried out, the generated Hardware Description Language (HDL) design can be synthesized for implementation on Xilinx FPGAs and System-on-Chips (SoCs) (integrated circuits which integrate most of the components of a computer system such as memory, central processor unit, or input/output ports). System Generator simulations are faster than traditional HDL simulators, and results are easier to analyse.

Figure 3 shows the generic design flow of the CI problem studied in this paper.

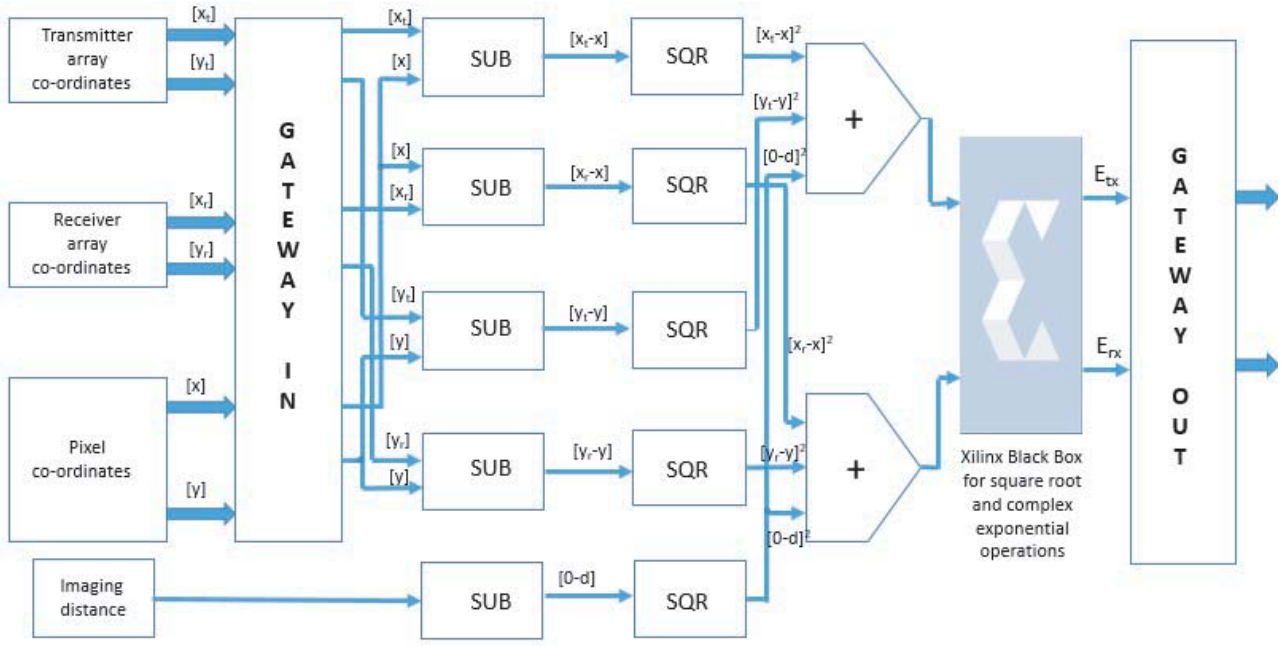

Figure 3. Generic design flow (Single loop period).

It is worth mentioning that the FPGA has been used to carry out the loop iterations involved in the sensing matrix calculations, which takes the bulk of the computation time. The main calculation that the FPGA performs is Equation (2). The block diagram shown in Figure 3 shows the calculation of a single loop of operation for a single voxel coordinate (x, y, z) and a single set of transmitter (x_t, y_t) and receiver (x_r, y_r) coordinates. The FPGA boundary is depicted by Gateway In and Gateway Out blocks. The SUB and SQR are the subtractor and square blocks, respectively, performing a part of the operation of calculating the Euclidean distance between a particular transmitter/receiver location and a voxel location in the RoI, $((x_t - x)^2 + (y_t - y)^2 + (0 - z)^2; (x_r - x)^2 + (y_r - y)^2 + (0 - z)^2)$. The Xilinx Black Box has the subroutines to perform the final square root and complex exponential operations, Equations (3) and (4). We use the CORDIC algorithms [56] to perform the exponential functions.

3.1. Data Representation

The data involved in the calculation mostly are the coordinate values of the transmitter and receiver as well as the pixel location of the scene. All these values are decimal point values (up to four decimal points, in this case), and we use a fixed-point representation for these values. Because we have to deal with a maximum length up to four decimal points, we choose a scale value of 16 (2^4). For example, a coordinate value of 0.5125 will be represented as 8.2 in fixed-point representation (0.5125×16) because it is easier to perform calculations using 8.2 as compared to 0.5125. In this case, we consider a maximum of three bits to represent the fraction part. Hence, the scaled number '8.2' is represented as '1000010' with imaginary decimal point 'fixed' after the 3rd bit (1000.010). The FPGA will perform all the calculations using the binary number '1000010'. To get back the original decimal number, all we have to do is to divide it by the scale value. The reason for not choosing a scale value other than powers of two (for example 10, 100, 1000 which are much easier to analyse) is that at the end when it is required to divide the result by the scale value, additional hardware is not required for the division operation. Division by two or powers of two can be achieved by just performing shift operations to the right.

As the requirement goes, this problem requires the coordinate values of the transmitter, receiver, and the pixels in the scene to calculate the Euclidean distance and eventually the values of the $\mathbf{E}_{Tx}$ and $\mathbf{E}_{Rx}$. The coordinate values consist of decimal point numbers with an equal mix of positive and negative values. In our case, considering the range of coordinate values used, a size of 7 bits is sufficient to represent each coordinate value. Each coordinate pair (of transmitter, receiver, and pixels), (x, y), is represented using a total of 16 bits of memory, as shown in Figure 4. In Figure 4, the 7th and 15th bit positions are used as a sign bit, and '1' in these bit locations represents a negative coordinate value. The same rule is followed to process the complex numbers taken up by the surface currents, $\mathbf{J}_{Tx}(x_t, y_t)$ and $\mathbf{J}_{Rx}(x_r, y_r)$, on both the transmitting and receiving apertures. The whole 16 bits of memory is used to represent a single complex number: 8 bits each for real and imaginary parts. Similar to the coordinate system representation, 7 bits are used to represent the decimal values, and the 7th and 15th bit positions are reserved as sign bits. As the values of the transmitting and receiving coded-aperture surface currents, $\mathbf{J}_{Tx}(x_t, y_t)$ and $\mathbf{J}_{Rx}(x_r, y_r)$, are random in nature, the computation uses different values for each measurement mode. This demands a constant communication between the CPU and the FPGA after every round of computation for each measurement mode. To save time and resources, these values are generated in the CPU itself prior to the computation process, stored in the FPGA memory using the above representation scheme and used as required in the computation process.

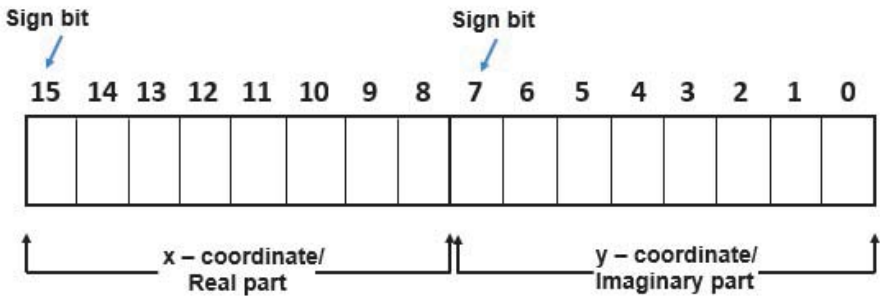

Figure 4. Co-ordinate/Complex number representation in memory.

3.2. Pipelining

As evident from the Equations (3) and (4), the most time consuming and the one that uses multiple loops, step in the calculation of the coded aperture radiated fields, is the Euclidean distance calculation between every pixel coordinate ((x, y, z), z is the imaging distance which is fixed) and every aperture coordinate ((x_t, y_t) and (x_r, y_r), with z_t, $z_r = 0$). The present imaging scenario has about 9409 pixel and 3362 aperture coordinate values, so computing the distance for each of these values using just CPU-brute force is bound to take huge amount of time. This is where the FPGA-parallelization method comes into

the picture. Rather than running the whole process in a sequential manner, the entire CI sensing matrix calculation process in the physical layer is shifted to the FPGA. The concept of pipelining is employed to run multiple operations in a single clock cycle, exploiting the available memory space, DSP blocks, and other internal FPGA logic blocks. The detailed operation is explained in the following paragraphs.

The XC7VX485t FPGA has a block RAM space of 37080 kbits, which is used for storing the operands. Each Block RAM has a space of 36 Kbits. The main operations involved in this problem are subtractions and multiplications. The DSP485E1 [57], which is embedded in most 7-series Xilinx FPGAs is used to perform such calculations. In this FPGA, there are about 2800 DSP slices, ordered in 20 columns with each column containing 140 DSP slices. The two registers inside the DSP slices, A and B, are used for storing the required operands for a particular operation, from the memory. Each DSP slice will perform two operations: a subtraction and a multiplication (square operation).

The full sensing matrix will have the dimension $M = number\ of\ transmitters \times number\ of\ receivers \times number\ of\ measurement\ masks \times N = total\ voxels\ in\ the\ scene$, which can result in a rather large number of elements to process. Therefore, to carry out all the processes simultaneously (in parallel), the concept of pipelining can be employed. It is an extension of parallel code execution that works within a single process. By partitioning the code into smaller segments, parallel code execution can be achieved. The smaller code segments run in parallel in the same loop, thus reducing the critical path in each loop iteration, which in turn will reduce the execution time. The layout of the pipelining process is shown in Figure 5.

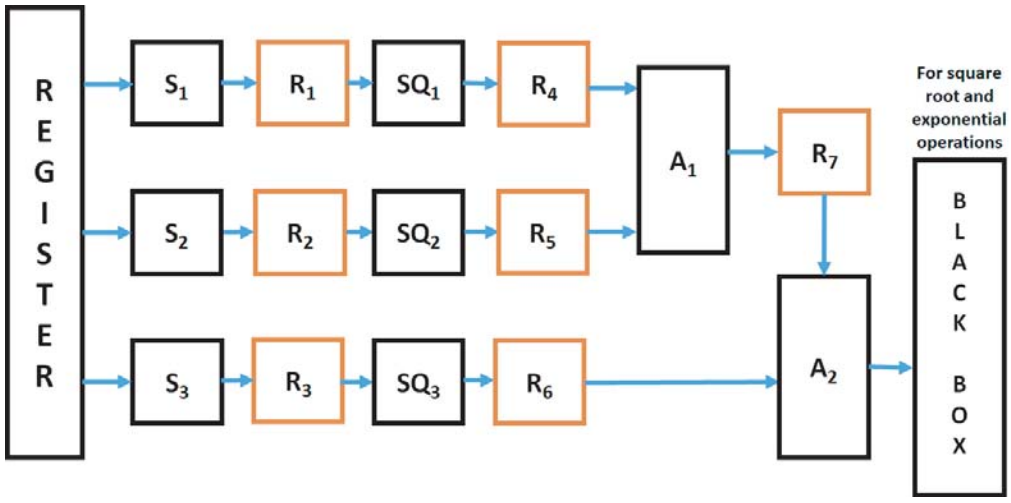

Figure 5. Pipelining process. Register blocks are depicted in orange colour.

Generally in a calculation process leveraging FPGA-based parallelization, the two most time consuming functions are analog input and output operations. The data processing is a relatively shorter operation. By using registers in appropriate places in the design, we can reduce the latency period. To show a picture of how pipelining is aiding in reducing the total computation time (Figure 5), let us look into the basic operation $\sqrt{(x_t - x)^2 + (y_t - y)^2 + (0 - z)^2)}$. The subtractors ($S_i$) have holding register blocks R_1 through R_3, and the square blocks (SQ_i) have holding register blocks R_4 through R_6, and so on. The main purpose of these register blocks is to free-off the arithmetic blocks after it completes an operation. Suppose that after the first clock tick, the subtractors produce the respective results and store the values in their respective registers. These values are required by the square blocks to perform the rest set of operation. If the register blocks were not present, the data would have to be made available at the output of the subtractors till the square block had finished reading the data, making the subtractor blocks remain idle during this whole process. Once the output of the subtractor is transferred to the respective registers, the subtractors are free to perform the operations of the next set of data, rather than doing nothing. So, on every clock count, we will be getting new values on the output buses, hence utilizing our clock cycles efficiently. As evident from Figure 5, this is a classic

case of linear arithmetic pipelining. The task of calculating the square root and complex exponential takes the maximum time. These two tasks take about 16 clock cycles, each. It is important to note that the calculations for the real and imaginary parts are done separately. The maximum time period in the pipelining is about 80 ns. Each holding register (or latch) has a delay about 5 ns. Hence, the maximum time period of the pipelining process is 85 ns. One thing to consider is the total memory available in the FPGA used. In this case, the FPGA has a block RAM space of 37080 kbits. So, the configuration of the imaging system and representation of the values in the FPGA may have to be altered in accordance with the size available with the FPGA. The CORDIC algorithm [58] is used to calculate the negative complex exponential term, which can be expressed as cosine and sine terms. The square of the subtractor products can be performed by DSP slice's multiplier, which uses a 25×18 bit multiplier.

3.3. Hardware Details

The following table lists the hardware details for the project:

Table 1. Summary of hardware specifications.

CPU	Intel Core i7-8700 @ 3.20 GHz
Memory	15.8 GB
FPGA	XC7VX485t (Virtex-7) @ 200 MHz
Memory	37080 kbits
OS	Microsoft Windows 10 Enterprise
HDL Software	Vivado 2018.3

The FPGA development board (ADM-XRC-7V1) is plugged into the PC backplane which supports PCI-express generation 3.0 with 16 lanes. The FPGA unit supports 4 lane generation 2.0 PCI-express interface. Xilinx provides PCIe DMAs and PCIe Bridge IP blocks to easily implement PCIe-based designs. In any transfers between the CPU and FPGA, the FPGA acts as a PCIe bus master, sending or requesting data as required. In this particular design, the FPGA requests data from the CPU only once throughout the whole process. A memory mapping is carried out between the FPGA memory and the CPU virtual memory, whose addresses are registered in the operating system through memory-mapped files.

The concept of pipelining depicted in Figure 5 is used in this application. Algorithm 1 details the processes involved in the calculation of the Euclidean distance between the transmitter or receiver coordinates and the pixel coordinates of the scene. This algorithm is only for a single pixel coordinate. This process is required to run for all the pixel values. The output at the end of the algorithm gives the sum of squares of differences (the exponential term in Equations (3) and (4) inside the square root). The obtained sum is used to calculate the square root and subsequently the exponential operation of the Green's function in Equations (3) and (4) using the CORDIC algorithm. This algorithm will continue for every aperture coordinates stored in the memory, which is total of 1681 in this particular case. As mentioned in Figure 4, each coordinate takes a space of 16 bits. So, the total space required to store the entire co-ordinates is 26896 bits (almost 27 kbits). The algorithm makes use of only three DSP slices and 5 holding registers for temporarily storing the results (two of 8 bits each and three of 16 bits each).

4. RESULTS AND DISCUSSION

The CI radar layout consists of two coded apertures, transmitter and receiver respectively, each with a size of $0.5\,\text{m} \times 0.5\,\text{m}$ operating in a bistatic mode. The imaged scene is of size $30\,\text{cm} \times 30\,\text{cm}$ and placed within the near-field of the radar, at a distance $d = 0.5\,\text{m}$ from the coded apertures. A depiction of the radar layout is shown in Figure 1(a). It should be noted that synthesizing the same aperture using the conventional SAR technique would require 144 channels for each aperture (at Nyquist rate for 12 GHz frequency). This provides a substantial simplification in the hardware layer.

Algorithm 1 Parallel algorithm for calculating the Euclidean distances between aperture co-ordinates (Tx/Rx) and a single pixel co-ordinate.

Input: Co-ordinate value as represented in Figure 4 in the memory.
Output: Sum S1

1: *Initialisation* :Imaging distance, d.
2: **For 1st clock cycle**
3: 1st aperture co-ordinate read from memory to register A.
4: Pixel co-ordinate read from memory to register C.
5: Write d = register A of DSP 1 (A_{DSP1}).
6: **For next clock cycle**
7: A[14:8] - C[14:8] := store in holding register reg1 (8 bits).
8: A[6:0] - C[6:0] := store in holding register reg2 (8 bits).
9: Write d = register B of DSP 1 (B_{DSP1}).
10: **For next clock cycle**
11: $A_{DSP1} \times B_{DSP1}$:= store in reg5 (16 bits).
12: Next aperture co-ordinate read from memory to register A.
13: Write reg1 = register A of DSP slice 2 (A_{DSP2}).
14: Write reg2 = register A of DSP slice 3 (A_{DSP3}).
15: **For next clock cycle**
16: Write reg1 = register B of DSP slice 1 (B_{DSP2}).
17: Write reg2 = register B of DSP slice 2 (B_{DSP3}).
18: **For next clock cycle**
19: A_{DSP2} x B_{DSP3} := store in reg3 (16 bits).
20: A_{DSP2} x B_{DSP3} := store in reg4 (16 bits).
21: A[14:8] - C[14:8] := reg1.
22: A[6:0] - C[6:0] := reg2.
23: **For next clock cycle**
24: Write reg3 = register A of DSP slice 1 (A_{DSP1}).
25: Write reg4 = register C of DSP slice 1 (C_{DSP1}).
26: Write reg1 = register A of DSP slice 2 (A_{DSP2}).
27: Write reg2 = register A of DSP slice 3 (A_{DSP3}).
28: Next aperture co-ordinate read from memory to register A.
29: **For next clock cycle**
30: $A_{DSP1} + C_{DSP1}$:= reg3.
31: Write reg1 = register B of DSP slice 2 (B_{DSP2}).
32: Write reg2 = register B of DSP slice 3 (B_{DSP3}).
33: A[14:8] - C[14:8] := store in reg1.
34: A[6:0] - C[6:0] := store in reg2.
35: **For next clock cycle**
36: Next aperture co-ordinate read from memory to register A.
37: $A_{DSP2} \times B_{DSP3}$:= store in reg3.
38: $A_{DSP2} \times B_{DSP3}$:= store in reg4.
39: Write reg3 = register A of DSP slice 1 (A_{DSP1}).
40: Write reg5 = register C of DSP slice 1 (C_{DSP1}).
41: **For next clock cycle**
42: A[14:8] - C[14:8] := store in reg1.
43: A[6:0] - C[6:0] := store in reg2.
44: $A_{DSP1} + C_{DSP1}$:= reg3. **(First output S1)**

It should also be noted that in a practical CI scenario for near-field imaging, the system phase calibration plays a crucial role, governing the fidelity of the reconstructed images [59, 60]. To this end, in order to achieve accurate positioning information of the coded apertures and the scene, additional techniques, such as integrating the apertures with optically-sensitive fiducials and optical scanning these fiducials, can be used [61, 22]. In this work, we focus on the numerical modeling of the CI technique with coded apertures and its parallelization using an FPGA platform. Therefore, we assume that the position information of the transmitting and receiving coded apertures, $\mathbf{r}_t(x_t, y_t)$ and $\mathbf{r}_r(x_r, y_r)$, and the imaged scene, $\mathbf{r}(x, y, z)$, are known a-priori in our forward-model in Equation (1).

To form the spatio-temporally incoherent radiation patterns, we employ an active modulation scheme by defining a set of random masks for the transmitting and receiving apertures, each of whose combination forms an individual row of the sensing matrix. Therefore, each mask configuration is used to illuminate the RoI, and the back-scattered measurements are collected at the receiving aperture and compressed into a single channel using the transfer function of the aperture. From radar theory, for a finite size aperture D and frequency-bandwidth, B, the diffraction limited resolution limits can be calculated as follows [45]:

$$\delta_{cr} = \lambda d/D \tag{6}$$

$$\delta_r = c/2B \tag{7}$$

δ_{cr} and δ_r denote the cross-range resolution and the range resolution of the radar, respectively. Equations (6) and (7) strictly hold in the far field, operating in the near field of the aperture brings the advantage that the range resolution is finite despite the single frequency operation [33, 44]. In this work, the synthesized CI radar aperture operates at 12 GHz frequency. In view of this, we first test the resolution limit of the synthesized CI system depicted in Figure 1(a) by imaging a point scatter [45]. For this analysis, the diameter of the point scatter is selected to be below the theoretical cross-range resolution limit of the radar, $\delta_{cr} = 1.2$ cm. The reconstructed image for this target reveals the point spread function (PSF) of the radar and is shown in Figure 6.

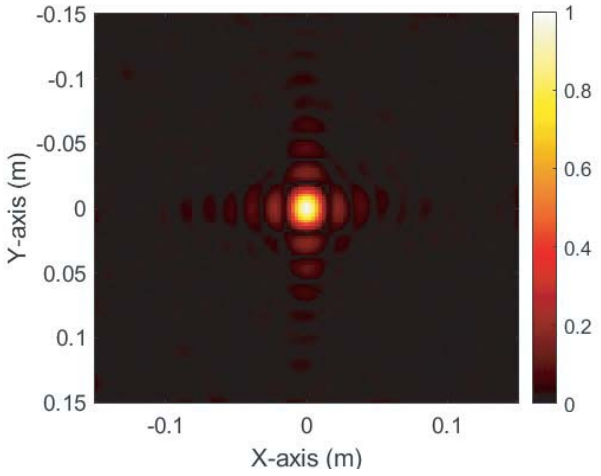

Figure 6. Reconstructed PSF pattern of the CI radar.

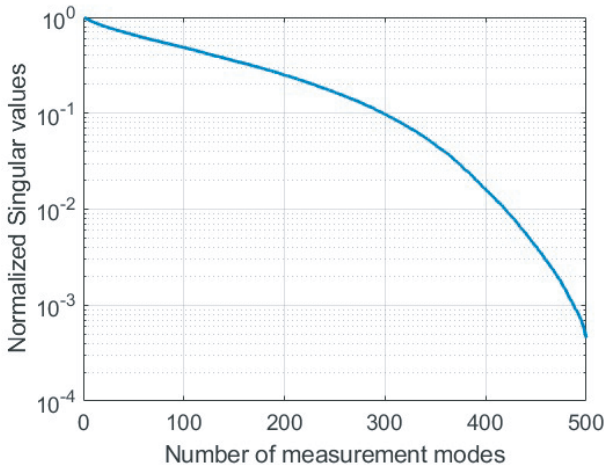

Figure 7. Singular Values of sensing matrix, **H**.

Performing a PSF analysis in Figure 6, the cross-range resolution limit is calculated to be 1.7 cm, which is in good agreement with the theoretical limit, 1.2 cm. For the next imaging scenarios, we fix the discretization pixel size of the imaged scene at 0.3125 cm, selected below the resolution limit of the synthesized aperture to smooth the reconstructed images. Choosing a discretization size smaller than this value would have a similar effect to interpolating the reconstructed images on a finer grid and further smoothing the reconstructions for plotting purposes, but would not bring new information about the imaged object. Therefore, choosing the discretization size substantially smaller than the resolution limit only increases the size of the sensing matrix, and hence the computational expense of the retrieval problem.

Following the analysis of the PSF patterns and confirmation of the diffraction limited resolution, we next image a set of more complicated targets, a gun phantom and the letters "Q", "U", and "B", forming the word "QUB", as shown in Figures 8 and 9. It is worth mentioning that all objects imaged in this paper, from the point scatter to the gun phantom and the letters 'QUB', are assumed to be metal with a normalized reflectivity value of 1. Varying the object reflectivity value can change the reflectivity contrast in the reconstructed images, but it would not change the computation complexity of the reconstruction step. An important aspect for the imaging problem at hand is the determination of the number of measurements (and hence aperture masks) to be used to retrieve the images of these targets to a good fidelity. The orthogonality of measurement modes produced by a CI system can be assessed by doing a singular value decomposition (SVD) analysis on the sensing matrix [25, 17, 43]. To this end, we first analyze the singular values of the sensing matrix as shown in Figure 7.

From Figure 7, it can be concluded that beyond 400 modes, the singular values drop substantially, suggesting that the amount of new information acquired from the scene is becoming negligible (redundant information). As a result, we set the upper bound limit for the number of measurement modes slightly above this limit, to 500 modes, for imaging. The reconstructed images, using both CPU and FPGA, are shown in Figures 8 and 9, respectively.

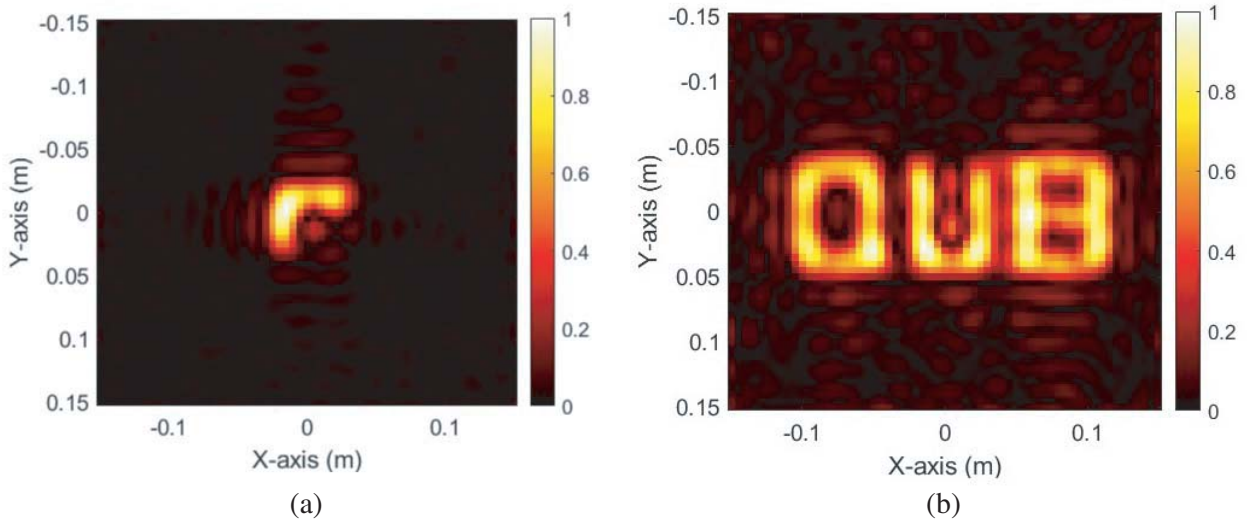

Figure 8. Images using CPU-brute force. (a) Reconstructed image of the gun phantom. (b) Reconstructed image of the QUB letters. The overall size of the gun phantom is 5 cm (x-axis) $\times$5 cm (y-axis). The overall size of the "QUB" word is 22 cm (x-axis) $\times$8 cm (y-axis).

The images of the gun phantom and "QUB" target are reconstructed to a good fidelity using the CPU-based brute-force solution and the proposed FPGA-based technique as can be seen in Figures 8 and 9, respectively. These reconstructions are from simulated data using the developed physical model given in Equations (1)–(5) only. Using the imaging setup as shown in Figure 1(a), the total electric field of the transmitting and receiving apertures is calculated according to Equations (3) and (4). These values are used to calculate the sensing matrix in Equation (2), and the scene is reconstructed from the calculated sensing matrix in Equation (5). In comparison to the ground truth, the Mean-Square-Error (MSE) is calculated for both the targets. The values are listed in the table below. It shows that there is a slight increase in MSE for both targets reconstructed using the FPGA integrated CI physical model as compared to the CPU-based brute-force calculations.

Table 3 lists the computation time as well as memory consumed by both the CPU-based and FPGA-based methods to complete the whole computation process. From the table, it is evident that the FPGA-based method provides a massive speed-up of the sensing matrix calculation (about 21 times). It also shows that the FPGA-based method uses about 72% of the available memory but still requires less resources than the CPU-based technique. This is also supported by a qualitative assessment; comparing Figures 8 and 9, we find that the quality of reconstruction is slightly degraded in the FPGA-based

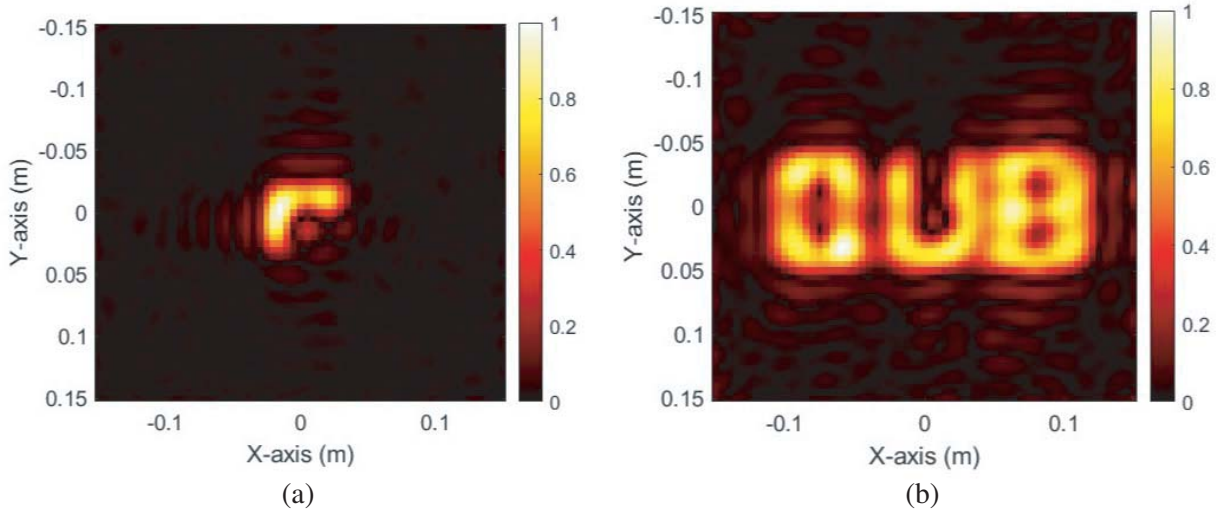

Figure 9. Images using the developed FPGA-based parallelization technique in physical model. (a) Reconstructed image of the gun phantom. (b) Reconstructed image of the QUB letters. The overall size of the gun phantom is 5 cm (x-axis) $\times$5 cm (y-axis). The overall size of the "QUB" word is 22 cm (x-axis) $\times$8 cm (y-axis).

Table 2. MSE metrics for both CPU- and FPGA-based computation methods.

	MSE-CPU	**MSE-FPGA**
Gun Phantom	0.0051	0.0068
Acronym 'QUB'	0.0367	0.0467

reconstructions. This is because the coordinate values used for calculating the sensing matrix cannot be represented in decimal form in the FPGA and have to be converted into a hardware compatible form. In this case, we have used the fixed-point representation, where the coordinates are scaled by a factor of 16. This is an approximate representation of the decimal coordinate values. However, the accuracy of fixed-point representation depends mainly on the number of bits used to represent the fractional part. Here, we use a maximum of three binary bits. A fraction requiring more than three binary bits cannot be represented accurately and have to be approximated, hence affecting the accuracy of the final result to a certain degree. This can be overcome by using a floating point representation. However, this representation is both time and resource consuming, and can substantially increase the complexity of the imaging problem at hand. A more feasible option is to increase the number of bits to represent the fractional part. Here, each coordinate system uses a total of 16 bits of memory. If the memory space is increased to 32 bits or even higher, more bits can be assigned to represent the fractional part of the coordinate value, making it a more accurate representation. However, this requires a large memory space. Given the scene size of 0.3 m $\times$ 0.3 m, the discretization pixel size of 3.125 mm, the aperture

Table 3. Performance metrics for both CPU- and FPGA-based methods.

	Computation time (s)	**Memory (Mbits)**
CPU	504.6	602.176
FPGA	23.04	26.896

size of 0.5 m × 0.5 m each and that each coordinate is represented with a 16 bit binary number, the memory space required to represent these values has a total of 204,336 bits (150,544 bits for the scene and 53,792 bits for the transmit and receive aperture). The calculation results need a total space around 26,896,000 bits. The memory requirements are within the capability of the current FPGA. However, if a better approximation to decimal values is required, i.e., space to represent each coordinate value is increased to 32 bits or higher, the memory requirements will increase tremendously. Then, there comes the ability of the FPGA to perform as many calculations in parallel as possible. This requires the services of as many DSP slices as possible for the FPGA to handle at once. Here, for this particular algorithm and for just a single pixel coordinate, three such slices are used. To run the calculations in parallel for a 97×97 pixel size scene, the number of DSP slices required would be large, more than 2800 DSP slices present in the current FPGA. An FPGA with a higher memory capability and higher number of DSP slices can be used, or multiple FPGAs can be used in tandem to store the values and carry out the calculations. Using such an arrangement of multiple FPGAs could also be beneficial in imaging large-scale targets, a scenario which also has huge memory and computation requirements. Partitioning the imaging scene into smaller sections and involving each individual FPGA in handling computations of a particular section of the scene could be a way forward in improving the ability of this FPGA-based solution to handle large-scale imaging scenarios.

A significant advantage of the proposed FPGA-based parallelization scheme can be appreciated by comparing the calculation times of the sensing matrix with and without the FPGA. Using the FPGA to calculate the aperture radiated fields in the scene and the sensing matrix in Equation (2), the calculation time is recorded to be 23.04 seconds, while without the FPGA (i.e., CPU-based brute-force solution) the calculation time is 504.6 seconds (8.4 minutes) (Table 3). This suggests that the proposed FPGA-based parallelization scheme reduced the reconstruction time by a factor of 21.9.

5. CONCLUSIONS

In this paper, we have demonstrated a parallelization scheme for CI radars at mmW frequencies. The computational radar, presented in this paper, consists of spatio-temporally modulating coded apertures which illuminate the scene using quasi-random field patterns and compress the back-scattered measurements into a single channel. It has been shown that using an FPGA-based parallelization platform to calculate the spatio-temporally incoherent sensing matrix substantially reduces the reconstruction time in comparison to using a brute-force processing technique for the forward-model and adjoint operation. These results demonstrate that implementing the CI technique on an FPGA platform and calculating the adjoint operation in parallel, the reconstruction step can be accelerated. Combined with the hardware simplification facilitated by the CI technique, FPGA-parallelization of the CI physical model exhibits a significant potential for reducing the computational latency of CI radars. It is worth mentioning that in spite of being shown for radar imaging, the validity of the presented approach is independent of the type of specific application considered, and this technique can be readily adopted across a wide application spectrum in CI, from security-screening [22, 33, 43] to automotive radars [4] and computational remote sensing techniques [6]. This is because the compression of the back-scattered data on the physical layer is the same for such applications. Despite being presented for the mmW frequency spectrum, the developed method can be readily extended to different frequency bands. This can be particularly advantageous for even higher frequencies where the number of unknowns for the imaging problem increases quadratically due to the increased resolution of the imaging system in the cross-range. Based on the promising results that we have obtained so far, future work includes the development of the presented algorithm to reconstruct images of more complex nature, such as human-scale targets for security screening, and moving objects. As the complexity of the imaging target increases, the demand for more computation power and resources will increase exponentially.

ACKNOWLEDGMENT

This work was funded by the Leverhulme Trust under Research Leadership Award RL-2019-019.

REFERENCES

1. Ahmed, S. S., A. Genghammer, A. Schiessl, and L.-P. Schmidt, "Fully electronic e-band personnel imager of 2 m^2 aperture based on a multistatic architecture," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 61, No. 1, 651–657, 2012.
2. Yurduseven, O., "Indirect microwave holographic imaging of concealed ordnance for airport security imaging systems," *Progress In Electromagnetics Research*, Vol. 146, 7–13, 2014.
3. Wang, Z., T. Chang, and H. Cui, "Review of active millimeter wave imaging techniques for personnel security screening," *IEEE Access*, Vol. 7, 148336–148350, 2019.
4. Yurduseven, O., T. Fromenteze, C. Decroze, and V. Fusco, "Frequency-diverse computational automotive radar technique for debris detection," *IEEE Sensors Journal*, Vol. 20, No. 22, 13167–13177, 2020.
5. Castro, J., S. Singh, A. Arora, S. Louie, and D. Senic, "Enabling safe autonomous vehicles by advanced mm-wave radar simulations," *IEEE MTT-S International Microwave Symposium Digest*, Vol. 2019-June, 1476–1479, 2019.
6. Diebold, A., M. Imani, and D. Smith, "Phaseless radar coincidence imaging with a MIMO SAR platform," *Remote Sensing*, Vol. 11, No. 5, 2019.
7. Sarabandi, K., M. Vahidpour, M. Moallem, and J. East, "Compact beam scanning 240 GHz radar for navigation and collision avoidance," *Proceedings of SPIE — The International Society for Optical Engineering*, Vol. 8031, 2011.
8. Detlefsen, J., "Industrial applications of microwave imaging," *1991 21st European Microwave Conference*, Vol. 1, 108–119, 1991.
9. Bilik, I., O. Longman, S. Villeval, and J. Tabrikian, "The rise of radar for autonomous vehicles: Signal processing solutions and future research directions," *IEEE Signal Processing Magazine*, Vol. 36, No. 5, 20–31, 2019.
10. Shehab, S., J. Feng, and N. Karmakar, "Trends on remote sensing technology: Receiver architectures and antenna systems," *1st International Conference on Robotics, Electrical and Signal Processing Techniques, ICREST 2019*, 227–232, 2019.
11. Li, Q., K. Chen, W. Guo, L. Lang, F. He, L. Chen, and Z. Xiong, "An aperture synthesis radiometer at millimeter wave band," *2008 International Conference on Microwave and Millimeter Wave Technology Proceedings, ICMMT*, Vol. 4, 1699–1701, 2008.
12. Piddyachiy, V., V. Shulga, V. Myshenko, A. Korolev, A. Myshenko, and A. Antyufeyev, "Ground-based 3 mm-wave radiometer for spectroscopic observations of atmospheric ozone and carbon monoxide," *2010 International Kharkov Symposium on Physics and Engineering of Microwaves, Millimeter and Submillimeter Waves, MSMW'2010*, 2010.
13. Sheen, D. M., D. L. McMakin, and T. E. Hall, "Three-dimensional millimeter-wave imaging for concealed weapon detection," *IEEE Transactions on Microwave Theory and Techniques*, Vol. 49, No. 9, 1581–1592, 2001.
14. Martínez-Lorenzo, J., F. Quivira, and C. Rappaport, "SAR imaging of suicide bombers wearing concealed explosive threats," *Progress In Electromagnetics Research*, Vol. 125, 255–272, 2012.
15. Demirci, S., H. Cetinkaya, E. Yigit, C. Ozdemir, and A. Vertiy, "A study on millimeter-wave imaging of concealed objects: Application using backprojection algorithm," *Progress In Electromagnetics Research*, Vol. 128, 457–477, 2012.
16. Hansen, H., A. Kulessa, and G. Brooker, "Millimetre-wave radars in targeting and data linking operations," *2003 Proceedings of the International Conference on Radar, RADAR 2003*, 230–234, 2003.
17. Fromenteze, T., O. Yurduseven, M. F. Imani, J. Gollub, C. Decroze, D. Carsenat, and D. R. Smith, "Computational imaging using a mode-mixing cavity at microwave frequencies," *Applied Physics Letters*, Vol. 106, No. 19, 2015.
18. Qi, F., I. Ocket, D. Schreurs, and B. Nauwelaers, "A system-level simulator for indoor mmW SAR imaging and its applications," *Optics Express*, Vol. 20, No. 21, 23811–23820, 2012.

19. Laviada, J., A. Arboleya-Arboleya, Y. Alvarez-Lopez, C. Garcia-Gonzalez, and F. Las-Heras, "Phaseless synthetic aperture radar with efficient sampling for broadband near-field imaging: Theory and validation," *IEEE Transactions on Antennas and Propagation*, Vol. 63, No. 2, 573–584, 2015.
20. Charvat, G., L. Kempel, E. Rothwell, C. Coleman, and E. Mokole, "An Ultrawideband (UWB) switched-antenna-array radar imaging system," *IEEE International Symposium on Phased Array Systems and Technology*, 543–550, 2010.
21. Withington, S., G. Saklatvala, and M. Hobson, "Partially coherent analysis of imaging and interferometric phased arrays: Noise, correlations, and uctuations," *Journal of the Optical Society of America A: Optics and Image Science, and Vision*, Vol. 23, No. 6, 1340–1348, 2006.
22. Gollub, J., O. Yurduseven, K. Trofatter, D. Arnitz, F. Imani, T. Sleasman, M. Boyarsky, A. Rose, A. Pedross-Engel, H. Odabasi, M. Reynolds, and D. Smith, "Large metasurface aperture for millimeter wave computational imaging at the human-scale," *Scientific Reports*, Vol. 7, 2017.
23. Molaei, A., J. Heredia-Juesas, G. Ghazi, J. Vlahakis, and J. A. Martinez-Lorenzo, "Digitized metamaterial absorber-based compressive reflector antenna for high sensing capacity imaging," *IEEE Access*, Vol. 7, 1160–1173, 2019.
24. Barbastathis, G., A. Ozcan, and G. Situ, "On the use of deep learning for computational imaging," *Optica*, Vol. 6, No. 8, 921–943, 2019.
25. Fromenteze, T., E. L. Kpré, D. Carsenat, C. Decroze, and T. Sakamoto, "Single-shot compressive multiple-inputs multiple-outputs radar imaging using a two-port passive device," *IEEE Access*, Vol. 4, 1050–1060, 2016.
26. Hunt, J., T. Driscoll, A. Mrozack, G. Lipworth, M. Reynolds, D. Brady, and D. R. Smith, "Metamaterial apertures for computational imaging," *Science*, Vol. 339, No. 6117, 310–313, 2013.
27. Yurduseven, O., V. R. Gowda, J. N. Gollub, and D. R. Smith, "Printed aperiodic cavity for computational and microwave imaging," *IEEE Microwave and Wireless Components Letters*, Vol. 26, No. 5, 367–369, 2016.
28. Yurduseven, O., J. Gollub, A. Rose, D. Marks, and D. Smith, "Design and simulation of a frequency-diverse aperture for imaging of human-scale targets," *IEEE Access*, Vol. 4, 5436–5451, 2016.
29. Chi, W. and N. George, "Phase-coded aperture for optical imaging," *Optics Communications*, Vol. 282, 2110–2117, June 2009.
30. Don, M. L., C. Fu, and G. R. Arce, "Compressive imaging via a rotating coded aperture," *Applied Optics*, Vol. 56, No. 3, B142, 2017.
31. Watts, C. M., D. Shrekenhamer, J. Montoya, G. Lipworth, J. Hunt, T. Sleasman, S. Krishna, D. R. Smith, and W. J. Padilla, "Terahertz compressive imaging with metamaterial spatial light modulators," *Nature Photonics*, Vol. 8, No. 8, 605, 2014.
32. Sleasman, T., M. F. Imani, J. N. Gollub, and D. R. Smith, "Dynamic metamaterial aperture for microwave imaging," *Applied Physics Letters*, Vol. 107, No. 20, 204104, 2015.
33. Imani, M., J. Gollub, O. Yurduseven, A. Diebold, M. Boyarsky, T. Fromenteze, L. Pulido-Mancera, T. Sleasman, and D. Smith, "Review of metasurface antennas for computational microwave imaging," *IEEE Transactions on Antennas and Propagation*, Vol. 68, No. 3, 1860–1875, 2020.
34. Andrecut, M., "Fast GPU implementation of sparse signal recovery from random projections," *Engineering Letters*, Vol. 17, No. 3, 2009.
35. Zhou, B., Y. Peng, C. Yeh, and J. Tang, "GPGPU accelerated fast convolution back-projection for radar image reconstruction," *Tsinghua Science and Technology*, Vol. 16, No. 3, 256–263, 2011.
36. Park, S. and D. Shires, "CUDA optimization techniques for SAR imaging algorithm," *Proceedings of the 2010 International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV 2010*, Vol. 1, 36–40, 2010.
37. Clemente, C., M. Di Bisceglie, M. Di Santo, N. Ranaldo, and M. Spinelli, "Processing of synthetic aperture radar data with GPGPU," *IEEE Workshop on Signal Processing Systems, SiPS: Design and Implementation*, 309–314, 2009.

38. Rybalkin, V. and N. Wehn, "When massive GPU parallelism Ain't enough: A novel hardware architecture of 2D-LSTM neural network," *FPGA 2020 — 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 111–121, 2020.
39. Farhadi, M., M. Ghasemi, and Y. Yang, "A novel design of adaptive and hierarchical convolutional neural networks using partial reconfiguration on FPGA," *2019 IEEE High Performance Extreme Computing Conference, HPEC 2019*, 2019.
40. Zhou, X., Z. Yu, Y. Cao, and S. Jiang, "SAR imaging realization with FPGA based on VIVADO HLS," *ICSIDP 2019 — IEEE International Conference on Signal, Information and Data Processing 2019*, 2019.
41. Liu, R., D. Zhu, D. Wang, and W. Du, "High resolution SAR signal processing system using FPGA," *2019 International Applied Computational Electromagnetics Society Symposium-China, ACES 2019*, 2019.
42. Di, W., C. Chen, and Y. Liu, "FPGA-based parallel system for synthetic aperture radar imaging," *2018 International Conference on Electronics Technology, ICET 2018*, 430–433, 2018.
43. Yurduseven, O., M. A. B. Abbasi, T. Fromenteze, and V. Fusco, "Lens-loaded coded aperture with increased information capacity for computational microwave imaging," *Remote Sensing*, Vol. 12, No. 9, 1531, 2020.
44. Sleasman, T., M. Boyarsky, M. F. Imani, T. Fromenteze, J. N. Gollub, and D. R. Smith, "Single-frequency microwave imaging with dynamic metasurface apertures," *JOSA B*, Vol. 34, No. 8, 1713–1726, 2017.
45. Yurduseven, O., M. Imani, H. Odabasi, J. Gollub, G. Lipworth, A. Rose, and D. Smith, "Resolution of the frequency diverse metamaterial aperture imager," *Progress In Electromagnetics Research*, Vol. 150, 97–107, 2015.
46. Peng, R., O. Yurduseven, T. Fromenteze, and D. R. Smith, "Advanced processing of 3D computational microwave polarimetry using a near-field frequency-diverse antenna," *IEEE Access*, Vol. 8, 166261–166272, 2020.
47. Fromenteze, T., O. Yurduseven, M. Boyarsky, J. Gollub, D. L. Marks, and D. R. Smith, "Computational polarimetric microwave imaging," *Optics Express*, Vol. 25, No. 22, 27488–27505, 2017.
48. Lipworth, G., A. Rose, O. Yurduseven, V. R. Gowda, M. F. Imani, H. Odabasi, P.'Trofatter, J. Gollub, and D. R. Smith, "Comprehensive simulation platform for a metamaterial imaging system," *Applied Optics*, Vol. 54, No. 31, 9343–9353, 2015.
49. Mandel, L. and E. Wolf, *Optical Coherence and Quantum Optics*, Cambridge University Press, 1995.
50. Hecht, K. T., *The Born Approximation*, 462–476, Springer New York, New York, NY, 2000.
51. Rashidi-Ranjbar, E. and M. Dehmollaian, "Microwave imaging using frequency-diverse scattering of a random rough surface," *ICEE 2019 — 27th Iranian Conference on Electrical Engineering*, 1679–1681, 2019.
52. Venkatesh, S., N. Viswanathan, and D. Schurig, "W-band sparse synthetic aperture for computational imaging," *Optics Express*, Vol. 24, No. 8, 8317–8331, 2016.
53. Kowdle, A., C. Rhemann, S. Fanello, A. Tagliasacchi, J. Taylor, P. Davidson, M. Dou, K. Guo, C. Keskin, S. Khamis, V. Tankovich, and J. Valentin, "The need 4 speed in real-time dense visual tracking," *ACM Transactions on Graphics*, Vol. 37, No. 6, 2018.
54. Malczewski, K., "Rapid diffusion weighted imaging with enhanced resolution," *Applied Magnetic Resonance*, Vol. 51, No. 3, 221–239, 2020.
55. X. Inc., *Virtex-7 FPGA Design Summary*, p. 5, Xilinx, February 27, 2018.
56. X. Inc., *CORDIC v6.0 LogiCORE IP Product Guide*, Xilinx, December 20, 2017.
57. X. Inc., *7 Series DSP48E1 Slice User Guide*, Xilinx, March 27, 2018.
58. Andraka, R., "A survey of CORDIC algorithms for FPGA based computers," *Tech. Rep.*, 1998.

59. Yurduseven, O., T. Fromenteze, and D. R. Smith, "Relaxation of alignment errors and phase calibration in computational frequency-diverse imaging using phase retrieval," *IEEE Access*, Vol. 6, 14884–14894, 2018.
60. Yurduseven, O., J. N. Gollub, K. P. Trofatter, D. L. Marks, A. Rose, and D. R. Smith, "Software calibration of a frequency-diverse, multistatic, computational imaging system," *IEEE Access*, Vol. 4, 2488–2497, 2016.
61. Sleasman, T., M. F. Imani, O. Yurduseven, K. P. Trofatter, V. R. Gowda, D. L. Marks, J. N. Gollub, and D. R. Smith, "Near field scan alignment procedure for electrically large apertures," *IEEE Transactions on Antennas and Propagation*, Vol. 65, No. 6, 3257–3262, 2017.