remote sensing

Communication

GPU-Accelerated Signal Processing for Passive Bistatic Radar

Xinyu Zhao [1], Peng Liu [1,*], Bingnan Wang [2] and Yaqiu Jin [1]

[1] The Key Laboratory for Information Science of Electromagnetic Waves (MoE), Fudan University, Shanghai 200433, China; 23210720053@m.fudan.edu.cn (X.Z.); yqjin@fudan.edu.cn (Y.J.)
[2] The National Key Laboratory of Microwave Imaging Technology, Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China; wbn@mail.ie.ac.cn
* Correspondence: pliu@fudan.edu.cn

Abstract: Passive bistatic radar is a novel radar technology that passively detects targets without actively emitting signals. Since passive bistatic radar entails larger data volumes and computations compared to traditional active radiation radar, the development of hardware and software platforms capable of efficiently processing signals from passive bistatic radar has emerged as a research focus in this field. This research investigates the signal processing flow of passive bistatic radar based on its characteristics and devises a parallel signal processing scheme under graphic processing unit (GPU) architecture for computation-intensive tasks. The proposed scheme utilizes high-computing-power GPU as the hardware platform and compute unified device architecture (CUDA) as the software platform and optimizes the extensive cancellation algorithm batches (ECA-B), range Doppler and constant false alarm detection algorithms. The detection and tracking of a single target are realized on the passive bistatic radar dataset of natural scenarios, and experiments show that the design of this algorithm can achieve a maximum acceleration ratio of 113.13. Comparative experiments conducted with varying data volumes revealed that this method significantly enhances the signal processing rate for passive bistatic radar.

Keywords: passive bistatic radar; signal processing; GPU parallel computing; CUDA

check for updates

Citation: Zhao, X.; Liu, P.; Wang, B.; Jin, Y. GPU-Accelerated Signal Processing for Passive Bistatic Radar. *Remote Sens.* **2023**, *15*, 5421. https://doi.org/10.3390/rs15225421

Academic Editors: Eugin Hyun and Inoh Choi

Received: 19 October 2023
Revised: 16 November 2023
Accepted: 17 November 2023
Published: 19 November 2023

1. Introduction

With the advancement of information technology, radar, as a commonly used detection and measurement technology, has become an indispensable part of military, civil, scientific research and other fields [1–3]. A traditional radar system mainly adopts the active detection method, through its own transmission signal for target detection and measurement. Nevertheless, with the development of radar countermeasure technologies, active radar systems are confronted with various threats, including electronic interference, stealth technology, low-altitude penetration, and anti-radiation weapons [4]. To address these challenges, the concept of passive bistatic radar systems has emerged [5,6]. These radar systems make use of third-party non-cooperative civilian radiation sources to illuminate targets, eliminating the need for active signal transmission and enabling passive target detection. Unlike traditional radar systems, the passive bistatic radar system does not require a separate transmission device and does not emit electromagnetic radiation either. Moreover, it offers a significant appeal and competitive advantage in scenarios where the current spectrum is limited due to its independence from dedicated frequency bands [7].

The signal processing stage [8,9] is the most time-consuming component of the passive bistatic radar system, frequently encountering challenges like large data volumes and high computational complexity. The demand for computational resources is increasing to meet the real-time signal processing requirements of radar systems [10]. The graphic processing unit (GPU) offers higher cost-effectiveness [11], higher energy efficiency [12], and more convenient development methods compared to the traditional digital signal processor (DSP) and field-programmable gate array (FPGA). The powerful computational and parallel processing capabilities of GPU open up new possibilities for signal processing in passive bistatic radar systems [13–15].

In recent years, the development of high-performance computing technology has attracted the interest of researchers who are exploring the potential of GPU parallel acceleration for various radar signal processing algorithms. Bu et al. [13] designed a GPU-accelerated clutter suppression algorithm for digital television terrestrial multimedia broadcasting (DTMB)-based passive radar. Zhao et al. [14] proposed a parallel algorithm for passive bistatic radar signal processing based on the orthogonal frequency division multiplexing (OFDM) waveform, which effectively reduces the processing time in the signal processing stage. Wan et al. [15] present a parallel acceleration of target detection in a passive radar system. However, none of these efforts parallelize the entire signal processing system, so the acceleration performance improvement is limited. Furthermore, the GPU graphics cards in common use today have a computing power of 8.x or less [16], and there has been no further exploration of GPU with a computing power of 8.x or more or of the new features available in newer versions of the GPU.

This paper presents an effective solution for the GPU acceleration of passive bistatic radar signal processing. Parallel signal processing is implemented on the CUDA platform. A clutter suppression module utilizes the more robust extensive cancellation algorithm batches (ECA-B) to mitigate clutter [17], while the range Doppler module reduces computational complexity through filtering and decimation [18]. The constant false alarm ratio (CFAR) module adopts the cell-averaging CFAR (CA-CFAR) algorithm. Using a dataset of passive bistatic radar data collected from natural scenes [19], the paper demonstrates the successful detection and tracking of an individual target. The achieved acceleration ratio of the algorithm reaches up to 113.13 compared to that under the traditional processing approach.

2. Radar Signal Processing

Passive bistatic radar signal processing [20] involves performing correlation processing between the received echo signals and reference signals [5,6]. This processing is essential for subsequent target parameter estimation and tracking. In this section, the main modules of signal processing are designed, including clutter suppression, range Doppler processing, and CFAR detection, to counteract the impact of strong clutter and noise in complex environments on target detection.

2.1. Radar System Analysis

In passive radar systems, the maximum range and maximum Doppler frequency are crucial parameters that directly affect the system's performance [6]. The range and Doppler characteristics are determined using the characteristics of the transmitted signals and the geometry of the radar setup. The radar equation for passive bistatic radar is as follows:

$$\frac{P_r}{P_n} = \frac{P_t G_t G_r \lambda^2}{(4\pi)^3 R_t^2 R_r^2} \cdot \sigma \cdot \frac{1}{K T_0 B F} \cdot L \tag{1}$$

where P_r, P_n, and P_t are the received signal power, received noise power, and radiation source transmit power. B, F, and L are the receiver's effective bandwidth, the receiver's noise figure and system loss, $T_0 = 290K$. λ is the wavelength, K is Boltzmann's constant, G_r is the receiver antenna's gain, R_t is the distance between the target and the radiation source, R_r is the distance between the receiver and the target, and σ is the radar cross-section area (RCS) of the target.

The passive bistatic radar geometry is shown in Figure 1. The angle between the target transmitter and target receiver segments, denoted by β, is called the bistatic angle. The angle between the target velocity vector and bistatic bisector is denoted by δ. A canonical definition of bistatic Doppler shift, f_D, ignoring relativistic effects, is the rate of change in the total path length of the transmitted signal. The bistatic Doppler shift caused by target motion is written as follows [21]:

$$\begin{aligned} f_D &= \tfrac{v}{\lambda}\left[\cos\left(\delta - \tfrac{\beta}{2}\right) + \cos\left(\delta + \tfrac{\beta}{2}\right)\right] \\ &= \tfrac{2v}{\lambda} \cdot \cos(\delta)\cos\left(\tfrac{\beta}{2}\right) \end{aligned} \tag{2}$$

where v is the relative speed between the radar and the target, and λ is the wavelength.

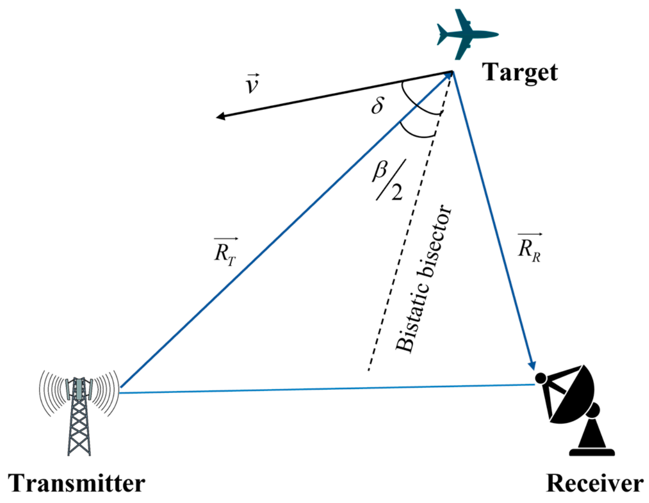


Figure 1. Geometry of passive bistatic radar.

The maximum range of a passive radar system depends on several factors, including the wavelength of the signal, the power of the signal, the sensitivity of the receiver and the processing capabilities of the system. The range resolution is inversely proportional to the bandwidth of the received signal. A larger bandwidth allows for a better range resolution. The maximum Doppler frequency is influenced by the carrier frequency of the received signal and the maximum relative velocity between the radar and the target. Higher carrier frequencies and faster relative velocities result in higher Doppler frequencies [20].

Overlapping signal blocks may allow for improved range and Doppler by using multiple signals with different characteristics simultaneously. However, this approach requires sophisticated signal processing techniques to handle the simultaneous reception of multiple signals. Achieving both a high maximum range and high maximum Doppler in passive radar design through overlapping signal blocks is challenging, and there are inherent trade-offs that need to be considered.

2.2. Clutter Suppression

In the clutter suppression module, the more robust ECA-B algorithm is utilized, and can effectively suppress direct wave signals while also exhibiting some suppression capability against multipath clutter originating from other static objects in the echo signals. In contrast to the traditional ECA algorithm, which necessitates the use of data from all sampling points to compute filter coefficients, the ECA-B algorithm utilized in this study achieves a reduction in the clutter subspace's dimensionality and computational storage requirements by segmenting both the surveillance and reference signals, leading to improved efficiency.

The clutter subspace is constructed by segmenting and overlapping the reference signal with its delay components. The least squares method is employed to process each segment of the surveillance signal, resulting in the generation of a clutter-suppressed signal [22]:

$$S^i_{ECA-B} = S^i_{surv} - X_i\left(X_i^H X_i\right)^{-1} X_i^H S^i_{surv} \tag{3}$$

where i is the segment number ($i = 0, 1, 2, \ldots, b-1$), b is the total number of segments, S^{i}_{surv} is the reference signal for the i-th segment, and X_i represents the i-th clutter subspace.

The entire surveillance signal after ECA-B clutter suppression is written as follows [13]:

$$S_{ECA-B} = \begin{bmatrix} S^{0}_{ECA-B} \\ S^{1}_{ECA-B} \\ \vdots \\ S^{b-1}_{ECA-B} \end{bmatrix} = \begin{bmatrix} S^{0}_{surv} - X_0\left(X_0^H X_0\right)^{-1} X_0^H S^{0}_{surv} \\ S^{1}_{surv} - X_1\left(X_1^H X_1\right)^{-1} X_1^H S^{1}_{surv} \\ \vdots \\ S^{b-1}_{surv} - X_{b-1}\left(X_{b-1}^H X_{b-1}\right)^{-1} X_{b-1}^H S^{b-1}_{surv} \end{bmatrix} \tag{4}$$

2.3. Range Doppler Processing

After clutter suppression, the energy of the target signals within the echo signal remains relatively feeble, and it is lower than the residual noise. Consequently, the target lacks effective differentiation from the background noise. To enhance the target's signal-to-noise ratio, the signal processing system must accumulate the energy of target echoes. This process is known as range Doppler processing, which facilitates the detection of weak target positions [23].

After applying the ECA-B algorithm, the reference signal and the echo signal can be represented as follows:

$$S_{echo}(t) = A_{echo} d(t-\tau_m) e^{j2\pi f_{am}(t-\tau_m)} \tag{5}$$

$$S_{ref}(t) = A_{ref} d(t-\tau_d) e^{j2\pi f_d(t-\tau_d)} \tag{6}$$

where A_{echo} is the amplitude of the echo signal, τ_m is the time delay of the echo signal, and f_{am} is the Doppler shift of the echo signal. Similarly, A_{ref}, τ_d, and f_d represent the respective characteristics (amplitude, time delay, and Doppler shift) of the reference signal, and $d(t)$ is the complex envelope of the direct signal. The radar receiver position is assumed to be constant, at $f_d = 0$.

The formula for utilizing the range Doppler two-dimensional cross-correlation is as follows [16]:

$$\begin{aligned} Y(t) &= S_{echo}(t) S^{*}_{ref}[t-\Delta\tau] \\ &= A_{echo} A^{*}_{ref} d(t-\tau_m) d^{*}(t-\tau_d-\Delta\tau) e^{j2\pi f_{am}(t-\tau_m)} e^{-j2\pi f_d(t-\tau_d-\Delta\tau)} \end{aligned} \tag{7}$$

The above equation is then fast Fourier-transformed to obtain the following:

$$\psi(\Delta\tau, f) = \int_0^{\Delta t} A_{echo} A^{*}_{ref} d(t-\tau_m) d^{*}(t-\tau_d-\Delta\tau) * e^{-j2\pi f_{am}\tau_m} e^{j2\pi f_{am} t} e^{-j2\pi f t} dt \tag{8}$$

where $(\tau_m - \tau_d, f_{am})$ represents the peak of the aforementioned function, which corresponds to the target position.

Given the substantial amount of data in range Doppler processing, filtering and decimation operations are necessary to reduce the number of points for fast Fourier transform (FFT). The filtering and decimation operation is shown in Figure 2 [24]. When the sampling rate is M:1, it reads continuous L-point data from the signal, starting at positions 1, M + 1, 2M + 1 and so on. Subsequently, each segment of data undergoes low-pass filtering to achieve signal anti-aliasing.

In range Doppler processing, the existence of a trade-off between the range resolution and Doppler resolution needs to be considered. The range resolution and Doppler resolution of passive bistatic radar are expressed as follows:

$$\Delta R = c \cdot \Delta\tau = \frac{c}{B} \tag{9}$$

$$\Delta V = \lambda \cdot \Delta f_d = \frac{\lambda}{T} \tag{10}$$

The bistatic range resolution depends on the ability to measure the relative delay between the echo and reference signals. The delay measurement resolution, $\Delta\tau$, is inversely proportional to the signal bandwidth, B; thus, $\Delta\tau = 1/B$. The Doppler frequency resolution, Δf_d, is inversely proportional to the observation or integration time, T; thus, $\Delta f_d = 1/T$.

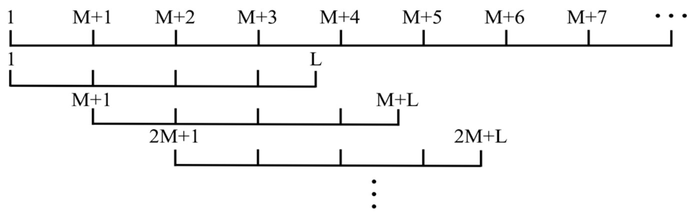


Figure 2. Filtering and downsampling operation.

From Equations (9) and (10), there is often a trade-off between the range and Doppler resolution in radar systems. Increasing the bandwidth of the received signal improves the Doppler resolution but can reduce the range resolution [8]. While the dilemma remains, there are several strategies and technologies that can be employed to mitigate its impact in passive radar setups. For example, using multiple-input multiple-output (MIMO) techniques, employing multiple antennas and utilizing spatial diversity can help improve the range and Doppler resolution, implementing adaptive waveform strategies [25]. In addition, an adaptive waveform strategy is implemented, using waveforms such as frequency-modulated continuous-wave (FMCW) signals that can adapt to changing environments and requirements.

2.4. CFAR Processing

CFAR detection [26] is a threshold-based process in which each target cell is assigned a threshold value. All cells that exceed the threshold value are retained, while cells that do not exceed the threshold value are filtered out. In this paper, the CA-CFAR detection algorithm is employed. This algorithm is capable of providing accurate thresholds and is well-suited to scenarios with non-homogeneous targets.

In the CA-CFAR detector, the rectangular reference window is commonly used, as shown in Figure 3. The length and width of the reference window are N_{ref} and M_{ref}, and the length and width of the protection window are N_{prot} and M_{prot}, respectively [27].

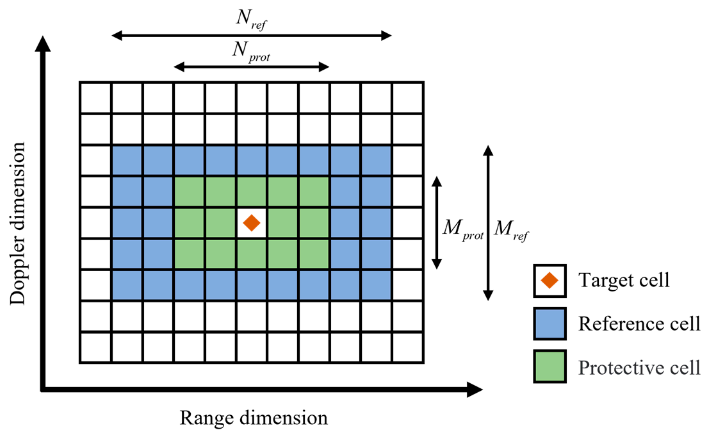


Figure 3. The structure model of the 2D-CA-CFAR detector.

The noise estimate of the target unit window can be expressed as Equation (11), where $Y_{i,j}$ denotes the value in the range Doppler two-dimensional matrix, and i and j denote the indexes of the range dimension and Doppler dimension.

$$Z_{i,j} = \frac{\sum_{i=-\frac{M_{ref}}{2},j=-\frac{N_{ref}}{2}}^{i=\frac{M_{ref}}{2},j=\frac{N_{ref}}{2}} Y(i,j) - \sum_{i=-\frac{M_{prot}}{2},j=-\frac{N_{prot}}{2}}^{i=\frac{M_{prot}}{2},j=\frac{N_{prot}}{2}} Y(i,j)}{M_{ref} \times N_{ref} - M_{prot} \times N_{prot}} \tag{11}$$

When the total number of selected reference cells is n, Equation (12) represents the functional relationship between the false alarm probability, P_{fa}, and the threshold factor, T, and the expression for the detection threshold, U_T, at (i, j) is as follows:

$$T = n\left(P_{fa}^{(-\frac{1}{n})} - 1\right) \tag{12}$$

$$U_T = T \times Z_{i,j} \tag{13}$$

3. GPU Acceleration Realization

Radar signal processing is a computationally intensive task. Parallelization is an effective means to enhance its efficiency and speed. By using the CUDA programming model [28], three parallelization algorithms were designed to harness the high parallelism and computational power of GPU in the clutter cancellation, range Doppler processing, and CFAR detection modules.

3.1. GPU Parallel Algorithm for Clutter Suppression

The most important steps in the clutter suppression ECA-B algorithm are the segmentation and overlap of reference signals, least squares computation and the solution of the remaining signals [13]. Algorithm acceleration is achieved by segmenting the computation tasks and utilizing GPU multi-threaded parallel processing.

In the clutter suppression ECA-B algorithm, the first step is to segment the echo signal and the reference signal, as shown in Figure 4. In this paper, signal segmentation is achieved by constructing the kernel function SigDivision. The pseudocode for this kernel function is presented in Algorithm 1. The segmented signal matrix, X_i, has dimensions of $N_B * b$, where N_B represents the length of data processed in each batch and b represents the total number of segments. Additionally, K represents the maximum-distance elimination unit.

Algorithm 1: Kernel Function SigDivision

```
input:  Reference signal $S_{ref}$ and segment number $i$.
output: Segmented signal matrix $X_i$
1 idx = threadIdx.x + blockIdx.x * blockDim.x + i * N_B;
2 idy = threadIdx.y + blockIdx.y * blockDim.y;
3 R = K − 1;
4 if $idy < K \wedge idx < (i+1) * N_B \wedge i * N_B \leq idx$ then
5     $b \leftarrow idy + (idx \bmod N_B) * K$;
6     $a \leftarrow R - idy + idx$;
7     $X_i[b].x \leftarrow ref[a].x$;
8     $X_i[b].y \leftarrow ref[a].y$;
9 end
```

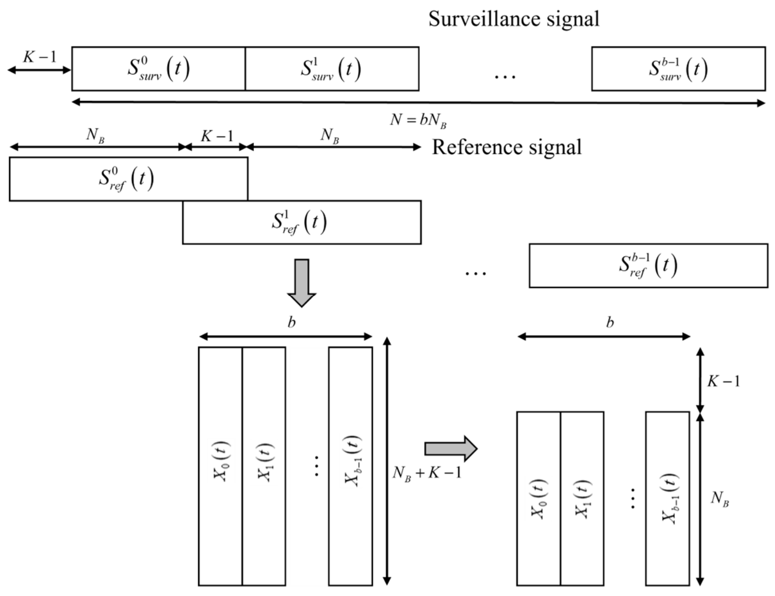


Figure 4. Signal segmentation in the ECA-B algorithm.

The key to accelerating the ECA-B algorithm lies in the computation of the least squares, $X_i\left(X_i^H X_i\right)^{-1} X_i^H S^i_{surv}$, which involves a significant amount of matrix operations. The CUDA platform provides libraries like CUBLAS and CUSOLVER, which provide a range of functions that can be utilized for matrix operations on the GPU [29].

3.2. GPU Parallel Algorithm for Range Doppler Processing

Range Doppler processing aims to generate a two-dimensional matrix with range and Doppler dimensions. The conjugate multiplication process of the reference and echo signals can be achieved by using the FFT- Multiply -IFFT calculation [30], and the data volume can be reduced by using filtering and downsampling. Since there is no specific order governing the formation of each data segment, the acceleration of the algorithm can be attained by leveraging data-level parallelism, as shown in Figure 5. The parallel acceleration steps for this process are as follows, and Figure 6 illustrates the flowchart of the parallelized range Doppler processing algorithm.

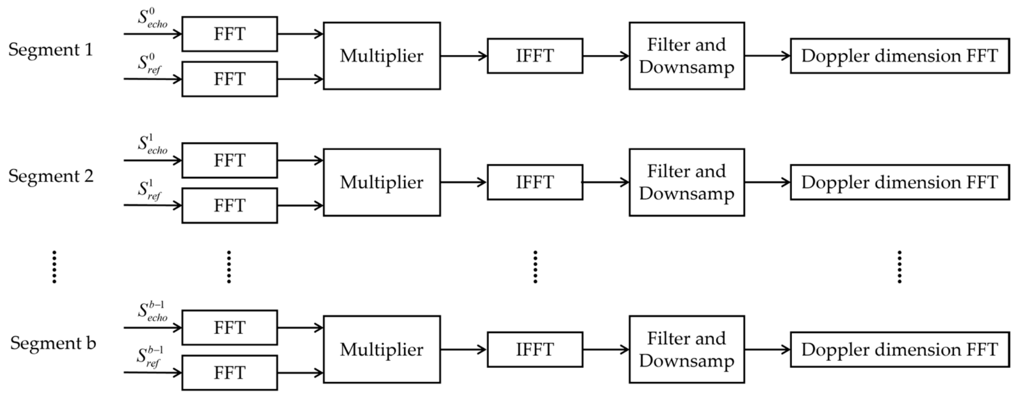


Figure 5. Parallelizability analysis of range Doppler processing algorithm.

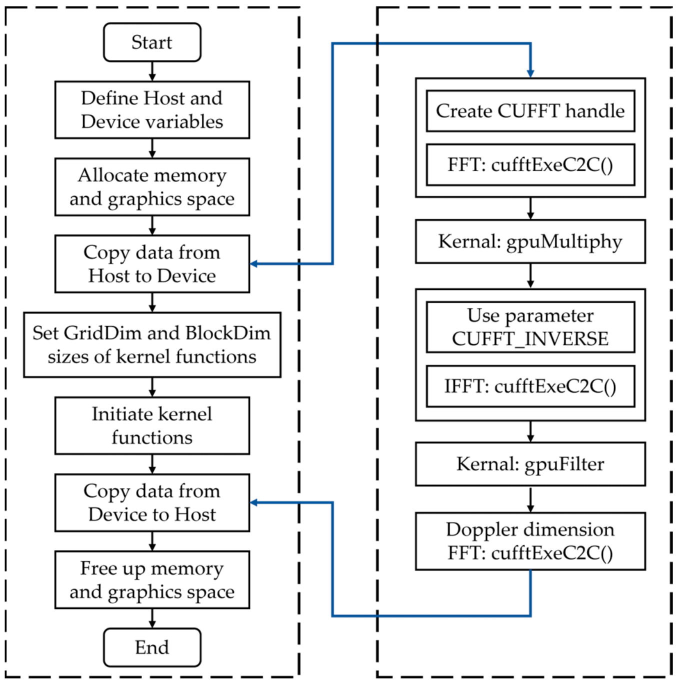

Start
Define Host and Device variables
Allocate memory and graphics space
Copy data from Host to Device
Set GridDim and BlockDim sizes of kernel functions
Initiate kernel functions
Copy data from Device to Host
Free up memory and graphics space
End
Create CUFFT handle
FFT: cufftExeC2C()
Kernal: gpuMultiphy
Use parameter CUFFT_INVERSE
IFFT: cufftExeC2C()
Kernal: gpuFilter
Doppler dimension FFT: cufftExeC2C()

Figure 6. Parallel implementation of range Doppler processing algorithm.

(1) Using the cufftPlan2d function, zero-padding and FFT computations are performed on the clutter-suppressed echo signal, S_{echo}, and the reference signal;
(2) Complex multiplication on the transformed signals using a kernel function is performed. After that, the range–domain correlation operation can be completed by performing IFFT computations;
(3) gpuFilter() is used to achieve downsampling. To avoid aliasing issues stemming from downsampling, it is imperative to apply anti-aliasing filtering concurrently during the downsampling procedure;
(4) FFT is performed on the correlated data along the Doppler dimension.

The FFT and IFFT operations in range Doppler processing can be directly accelerated by using the CUFFT function library in CUDA.

The clutter suppression and distance Doppler schemes proposed in this paper are also applicable to waveforms with cyclic prefixes, but need to be corrected according to the characteristics of the waveforms in use. Taking the digital terrestrial multimedia broadcast (DTMB) signal as an example, which uses the cyclic extension structure of the PN sequence as a prefix, if this feature is not handled, it will produce a number of regular secondary peaks in the result of the cross-correlation computation, which will have an effect on the detection of the target. In engineering, header zeroing is often used to process the reference signal. For this type of signal, our design idea is first, to use the standard signal frame data to highlight the header with the captured reference channel in the inter-correlation operation; then, to search for the header region in the inter-correlation result; and lastly, to zero the header region in the searched signal.

3.3. GPU Parallel Algorithm for CFAR Processing

Figure 7 illustrates the parallel implementation of the two-dimensional CA-CFAR (2D-CA-CFAR) algorithm. The input to this system is the signal that has undergone range Doppler processing [27].

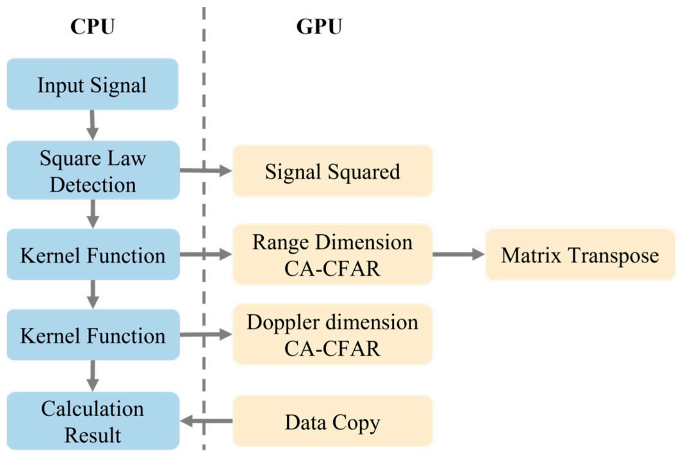


Figure 7. Parallel implementation of the 2D-CA-CFAR algorithm.

(1) Malloc() and cudaMalloc() are used to allocate space for CPU variables and GPU variables;
(2) The cudaMemcpy() function is used with the cudaMemcpyHostToDevice parameter to copy CPU variables to GPU;
(3) The thread grid and block sizes are allocated and the square law detection kernel function on the GPU, which can be called from official CUDA libraries, is executed;
(4) Threshold calculation and the decision making kernel function on the detection results are called;
(5) The cudaMemcpy() function is used with the cudaMemcpyDeviceToHost parameter to copy the results from the GPU back to the CPU;
(6) Free() and cudaFree() are called to free the memory resources consumed on the CPU and GPU.

4. Experimental Results

In this paper, the experimental simulation and testing section primarily aims to validate the feasibility and practical effectiveness of GPU-based parallel acceleration in processing radar signals originating from passive bistatic radar. In the experimental phase, real GPU and CPU devices are utilized, combined with CUDA programming to implement parallel signal processing; this approach serves to establish the reliability and precision of the algorithms presented in this paper. In the testing phase, authentic radar data are utilized alongside selected acceleration ratio metrics to evaluate the acceleration ratios of crucial algorithms and of the whole algorithm of radar signal processing under different data processing workloads.

4.1. Experimental Settings

In this experiment, the CPU model utilized is the Intel i9-10980XE processor with a base frequency of 3.0 GHz and a boost frequency of 4.6 GHz. The GPU employed is the NVIDIA RTX 3090 series, equipped with the computing capability of 8.6 and 10496 CUDA cores. The hardware specifications are shown in Table 1. In addition, the software platform primarily encompasses Microsoft Visual Studio 2019 for C++ development and VS Code for C language development, while the CUDA development environment employed is CUDA 11.6.

Table 1. Hardware specifications.

Device Type	Model	Key Parameters
CPU	Intel i9-10980XE	Base clock: 3.0 GHz Boost clock: 4.6 GHz Cores: 18 Threads: 36 L3 cache: 24.75 MB
GPU	NVIDIA RTX3090	CUDA cores: 10496 GPU frequency: 19.5 GHz GPU memory: 24 GB(GDDR6) GPU computing power: 8.6

The experimental signal source is derived from a FM radio broadcast [31], detecting and targeting a flying aircraft in the Halifax area, while the utilized radar data format is HDF5 [32]. Figure 8 shows the position of the PBR system and the FM station; the target is a Boeing-787 airliner, flying at roughly Mach 0.85, and the flying altitude is about 11.3 km. The relevant parameters of the radar dataset are shown in Table 2; the dataset is of a a total size of 6 GB and operates at a sampling frequency of 2.4 MHz. The maximum bistatic range is 200.11 km, with the maximum Doppler frequency shift reaching 256.04 Hz.

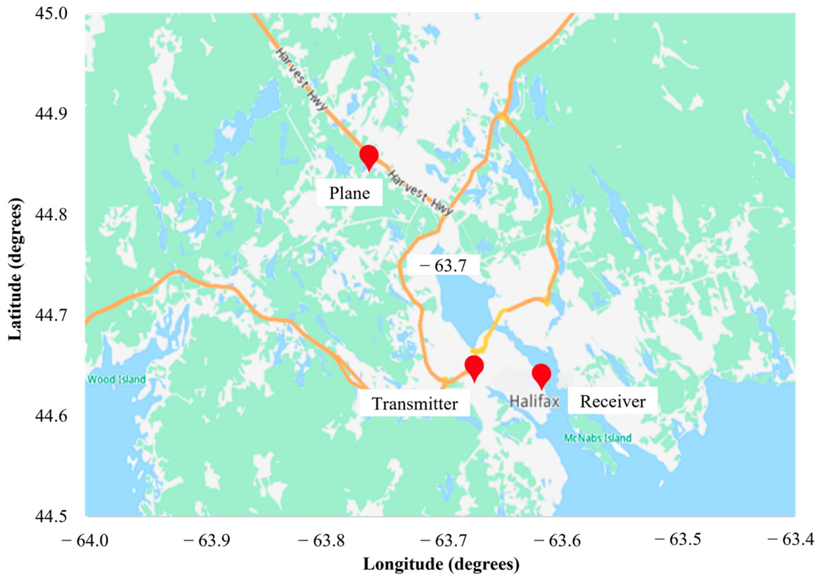


Figure 8. Position of the PBR system.

Table 2. Radar dataset and related parameters.

Parameters	Value
Signal type	FM radio broadcast
Center frequency	101.9 MHz
Band width	200 kHz
Sampling rate	2.4 MHz
Maximum bistatic range	200.11 km
Range resolution	1.14 Km
Maximum Doppler frequency shift	256.04 Hz
Doppler resolution	0.5 Hz

The experiment employs the speedup ratio of the CUDA program as a metric to assess the efficacy of parallel acceleration on the GPU. The specific derivation of this ratio is outlined below:

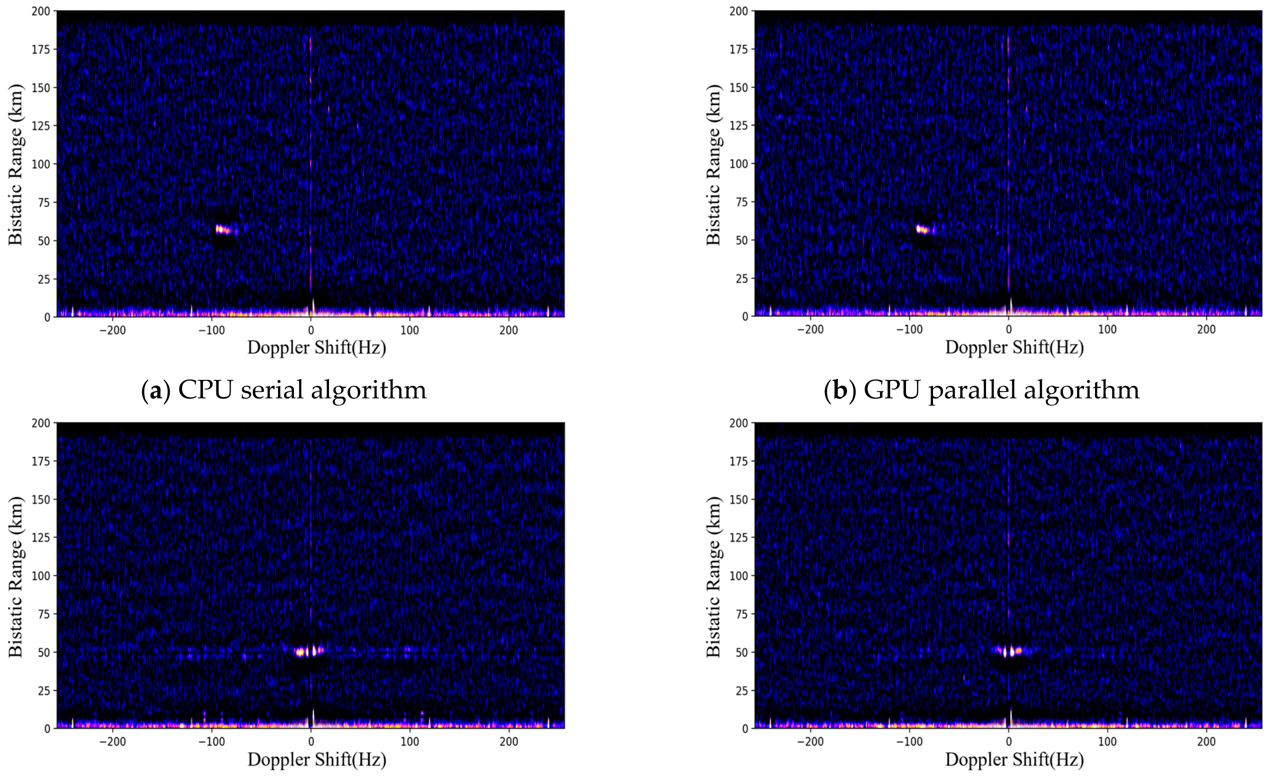
$$Speedup = \frac{W_s + W_p}{W_s + W_p/n} \tag{14}$$

where W_s represents the execution time of the sequential algorithm, W_p is the execution time of the parallel implementation of the sequential algorithm, and n is the number of processors or CUDA threads in the case of GPU programs.

4.2. GPU Parallel Algorithm Correctness Verification

To ascertain the correctness of the parallel signal processing algorithm proposed in this paper, an evaluation is conducted by comparing the processing outcomes of the GPU-based parallel signal processing algorithm with those of the CPU-based serial algorithm using the authentic radar data.

Figure 9 displays the target detection results acquired by executing the serial CPU and parallel GPU signal processing algorithms. It is evident that the parallel algorithm employed in this design effectively detects the target aircraft. Moreover, within an acceptable margin of error, the bistatic range and Doppler frequency shift of the targets detected by the parallel algorithm remain consistent with those detected by the serial algorithm.



(**a**) CPU serial algorithm (**b**) GPU parallel algorithm

(**c**) CPU serial algorithm (**d**) GPU parallel algorithm

Figure 9. (**a**–**d**) Target detection results after signal processing by serial CPU and parallel GPU algorithm.

4.3. GPU Parallel Algorithm Acceleration Performance Verification

During the phase of performance acceleration testing, this design utilizes 50 frames of radar echo data, amounting to a total size of 400 MB, as the initial dataset. Both the CPU serial algorithm and GPU parallel algorithm are executed on an identical hardware platform, with their execution times being measured and recorded. In the serial program, the total program runtime is determined by commencing the timer after the completion of data file reading and concluding it once the CFAR detection phase is finalized. For each signal processing module, timing begins when the module starts running and stops when the module completes its calculations. In the parallel program, the overall program runtime is gauged by initiating the timer after the data file reading process is complete and halting it once the CFAR detection phase concludes. As for the signal processing modules, timing is initiated when a thread is launched and halted when all computational modules within that thread have completed their tasks.

The CPU serial algorithm and GPU parallel algorithm for processing the passive bistatic radar signal underwent ten separate tests. In each run, the execution time for each of the three pivotal algorithms and the total processing time for the entire workflow were measured and recorded in seconds. The average time for each algorithm was computed by taking the mean of the ten experiments and rounding it to three decimal places. The acceleration ratio of the proposed algorithm was determined using the CUDA program acceleration ratio metric. The pertinent algorithm execution time and acceleration ratio are presented in Table 3.

Table 3. Average time and acceleration ratios of serial and parallel algorithms.

Signal Processing	CPU (s)	GPU (s)	Speedup
ECA-B	0.122	0.009	14.37
RD-Processing	12.490	0.344	36.31
2D-CA-CFAR	6.440	0.026	247.69
Whole Algorithm	19.052	0.379	50.34

As illustrated in Figure 10, the parallel algorithm yields significant speedup in each module when compared to the CPU platform. The overall algorithm achieves a speedup of 50.34.

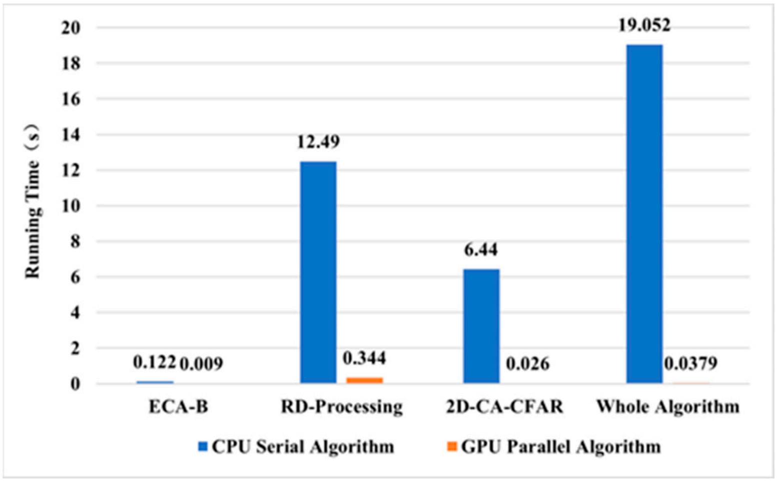


Figure 10. Average execution time of serial and parallel algorithms for passive bistatic radar signal processing in various stages.

Due to the dynamic nature of radar echo data volume in real-world environments, further testing is necessary to evaluate the impact of varying data sizes on the acceleration performance of the proposed parallelized algorithm. The experiment consisted of six different configurations of radar echo data volumes. For each group of data, both the passive bistatic radar CPU serial and GPU parallel algorithms were run. Similarly to the previous experiments, each group of data was repeated ten times, and the runtime for each module algorithm and the total processing time for the entire algorithm were recorded for each run. The average time in seconds was calculated by averaging the results of the ten experiments for each group and rounding it to three decimal places, representing the experimental result for that group. The average processing time of each module in the parallel and serial algorithms was recorded for each data volume configuration, and the speedup was calculated and shown in Table 4.

Table 4. Speedup of each key algorithm under different data volume.

Data Volume	ECA-B	RD-Processing	2D-CA-CFAR	Whole Algorithm
10 frames (80 MB)	8.21	24.77	61.24	29.29
20 frames (160 MB)	10.52	23.28	214.25	31.99
50 frames (400 MB)	14.37	36.31	247.69	50.34
100 frames (800 MB)	17.87	55.91	243.49	74.72
150 frames (1200 MB)	23.61	92.11	227.39	113.13
200 frames (1600 MB)	25.95	90.54	271.02	112.95

As illustrated in Figure 11, there is a noticeable increase in the CUDA acceleration ratios of the three critical algorithms for passive bistatic radar signal processing, as well as in those if the overall algorithm, with there was an increase in radar echo data size. However, as the input data size reaches a certain threshold, the rate of increase gradually diminishes, and the acceleration ratios tend to stabilize. Furthermore, it is evident that the acceleration ratio of the 2D-CA-CFAR algorithm significantly outpaces that of the other algorithms, displaying rapid growth in the initial stages. In contrast, the clutter suppression algorithm exhibits a lower acceleration ratio in comparison to that of the other algorithms, ultimately stabilizing at approximately 25 times. The acceleration ratio of the range Doppler algorithm follows a relatively consistent trend as the echo data size increases, ultimately stabilizing at around 90 times.

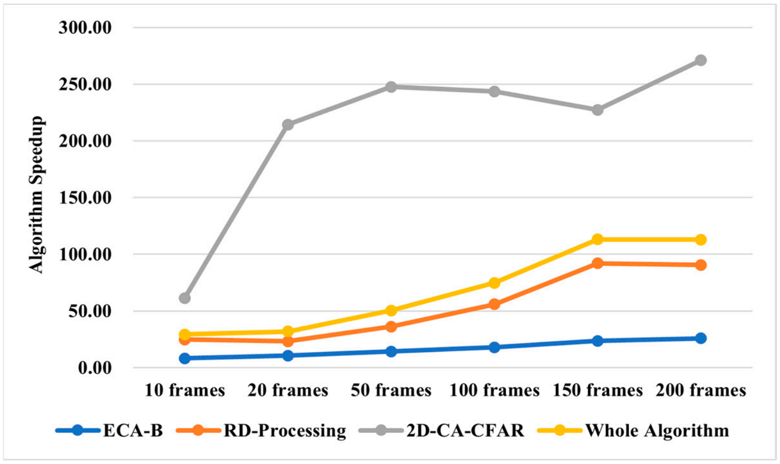


Figure 11. Changes in speedup of key algorithms and overall algorithm with different data sizes.

The observed trends can be attributed to several factors in the parallelization approach presented in this paper. The filtering and downsampling operations in the range Doppler processing reduce the data size significantly. Additionally, range Doppler processing involves a substantial number of FFT operations and their inverse transformations, which can be effectively parallelized using the cuFFT library in CUDA; this, combined with the parallelization strategy employed in this paper, achieves a notable acceleration ratio. The 2D-CA-CFAR detection algorithm involves numerous convolution operations, which traditionally can be time-consuming in CPU serial processing. However, in this design, GPU parallel processing takes advantage of the parallelization capabilities of GPUs, leading to significantly improved performance.

The overall algorithm acceleration ratio in this design also demonstrates an initial upward trend and subsequently stabilizes as the data volume continues to increase, eventually reaching an acceleration ratio of approximately 110 times. Furthermore, the traditional CPU serial radar signal processing algorithms exhibit increasing execution times as the data volume grows. In contrast, the GPU parallel algorithms complete equivalent tasks in less than 1 s. This stark contrast underscores the effectiveness of the GPU/CPU heterogeneous coordination mode and the CUDA programming architecture in achieving parallel acceleration for passive bistatic radar signal processing algorithms.

5. Conclusions

This paper presents a study on the GPU parallel acceleration of signal processing for the passive bistatic radar. By utilizing the CUDA programming technique and the immense parallelism and computational capabilities of GPU, parallel acceleration algorithms are developed for the clutter suppression ECA-B algorithm, range Doppler processing, and two-dimensional CA-CFAR detection. In real passive bistatic radar datasets, the acceleration ratio of the proposed algorithm reaches 113.13 compared to that under the traditional CPU-based methods. This substantial speedup greatly enhances the efficiency of passive bistatic radar systems, enabling the real-time processing of passive bistatic radar data.

Future work will optimize the various aspects of passive bistatic radar signal processing. In view of the incomplete clutter suppression, more innovative clutter suppression algorithms such as the iterative cancellation method with singular value decomposition (SVD) combined with entropy change will be considered. In addition, considering the problems of obvious branching delay and slow data intercommunication between the host side and the device side, when the amount of data is too large or the complexity of the program increases, the further design of the algorithms can be accomplished by using parallel methods such as CUDA streams and tensor cores, etc. It is envisioned that such systems could play an important role in the current generation of spectrum limitation and can be used with the critical applications of the system information that requires very low latency (e.g., self-driving cars).

Author Contributions: X.Z. and P.L. conceived and developed the processing algorithm and analyzed the processing results; B.W. and Y.J. discussed the processing procedure and examined the final results. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (NSFC) Project under Grant 62231021 and 61771142.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank the editors and reviewers for their time and effort spent in going through this manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, G.S.; Gu, H.; Su, W.M.; Sun, H.B.; Zhang, J.H. Random signal radar-a winner in both the military and civilian operating environments. *IEEE Trans. Aerosp. Electron. Syst.* **2003**, *39*, 489–498.
2. Zhang, J.A.; Liu, F.; Masouros, C.; Heath, R.W.; Feng, Z.; Zheng, L.; Petropulu, A. An Overview of Signal Processing Techniques for Joint Communication and Radar Sensing. *IEEE J. Sel. Top. Signal Process.* **2021**, *15*, 1295–1315. [CrossRef]
3. Sun, M.; Pan, J.; Le Bastard, C.; Wang, Y.; Li, J. Advanced signal processing methods for ground-penetrating radar: Applications to civil engineering. *IEEE Signal Process. Mag.* **2019**, *36*, 74–84. [CrossRef]
4. Yan, Q.; Xia, D.; Zhao, Y. Radar Waveform Design Based on Linear Frequency Modulation Signal. In Proceedings of the 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Chengdu, China, 12–15 April 2019; pp. 574–578.
5. Kuschel, H.; Cristallini, D.; Olsen, K.E. Tutorial: Passive radar tutorial. *IEEE Aerosp. Electron. Syst. Mag.* **2019**, *34*, 2–19. [CrossRef]
6. Griffiths, H.; Baker, C. *An Introduction to Passive Radar*; Artech House Radar Library, Artech House: Norwood, MA, USA, 2017.
7. Tan, D.K.P.; Sun, H.; Lu, Y.; Lesturgie, M.; Chan, H.L. Passive radar using global system for mobile communication signal: Theory, implementation and measurements. *IEEE Proc.-Radar Sonar Navig.* **2005**, *3*, 116–123. [CrossRef]
8. Palmer, J.E.; Harms, H.A.; Searle, S.J.; Davis, L. DVB-T passive radar signal processing. *IEEE Trans. Signal Process.* **2012**, *61*, 2116–2126. [CrossRef]
9. Berger, C.R.; Demissie, B.; Heckenbach, J.; Willett, P.; Zhou, S. Signal processing for passive radar using OFDM waveforms. *IEEE J. Sel. Top. Signal Process.* **2010**, *4*, 226–238. [CrossRef]
10. Wan, X.R. An Overview on Development of Passive Radar Based on the LowFrequency Band Digital Broadcasting and TV Signals. *J. Radars* **2012**, *1*, 109–123.
11. Nickolls, J.; Dally, W.J. The GPU computing era. *IEEE micro.* **2010**, *30*, 56–69. [CrossRef]

12. Mittal, S.; Vetter, J. A Survey of CPU-GPU Heterogeneous Computing Techniques. *ACM Comput. Surv.* **2015**, *47*, 1–35. [CrossRef]
13. Bu, Z.; Zhu, H.; Qian, J. GPU-Accelerated Multipath Clutter Cancellation Algorithm for DTMB-Based Passive Radar. In Proceedings of the 2020 International Conference on Information Science, Parallel and Distributed Systems (ISPDS), Xi'an, China, 14–16 August 2020; pp. 216–219.
14. Zhao, Z.; Wan, X.; Yi, J.; Xie, R.; Wang, Y. Radio frequency interference mitigation in OFDM based passive bistatic radar. *AEU-Int. J. Electron. Commun.* **2016**, *70*, 70–76. [CrossRef]
15. Liu, Y.; Wan, X.; Sun, X. GPU parallel acceleration of target detection in passive radar system. In Proceedings of the 2016 CIE International Conference on Radar (RADAR), Guangzhou, China, 10–13 October 2016; pp. 1–4.
16. Barkhatov, A.; Kozlov, A. Fast Calculation of Cross-Correlation Function with Video Cards in Coherent Radar. In Proceedings of the 2020 9th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 8–11 June 2020; pp. 1–5.
17. Colone, F.; Palmarini, C.; Martelli, T.; Tilli, E. Sliding extensive cancellation algorithm for disturbance removal in passive radar. *IEEE Trans. Aerosp. Electron. Syst.* **2016**, *52*, 1309–1326. [CrossRef]
18. Moscardini, C.; Petri, D.; Capria, A.; Conti, M.; Martorella, M.; Berizzi, F. Batches algorithm for passive radar: A theoretical analysis. *IEEE Trans. Aerosp. Electron. Syst.* **2015**, *51*, 1475–1487. [CrossRef]
19. Zhang, P.; Wu, Y.; Wang, J. Real-time signal processing for FM-based passive bistatic radar using GPUs. In Proceedings of the 2014 19th International Conference on Digital Signal Processing (DSP), Hong Kong, China, 20–23 August 2014; pp. 536–540.
20. Malanowski, M. *Signal Processing for Passive Bistatic Radar*; Artech House: Norwood, MA, USA, 2019.
21. Mathews, Z.; Quiriconi, L.; Schüpbach, C. Learning Resource Allocation in Active-Passive Radar Sensor Networks. *Front. Signal Process.* **2022**, *2*, 822894. [CrossRef]
22. Wu, Y.; Chen, Z.; Peng, D. Target Detection of Passive Bistatic Radar under the Condition of Impure Reference Signal. *Remote Sens.* **2023**, *15*, 3876. [CrossRef]
23. Ning, X.; Yeh, C.; Zhou, B.; Gao, W.; Yang, J. Multiple-GPU accelerated range-doppler algorithm for synthetic aperture radar imaging. In Proceedings of the Radar Conference (RADAR), Kansas City, MO, USA, 23–27 May 2011; pp. 698–701.
24. Feng, W.; Friedt, J.M.; Cherniak, G.; Sato, M. Batch compressive sensing for passive radar range-Doppler map generation. *IEEE Trans. Aerosp. Electron. Syst.* **2019**, *55*, 3090–3102. [CrossRef]
25. Liu, G.; Yang, W.; Li, P.; Qin, G.; Cai, J.; Wang, Y.; Wang, S.; Yue, N.; Huang, D. MIMO Radar Parallel Simulation System Based on CPU/GPU Architecture. *Sensors* **2022**, *22*, 396. [CrossRef] [PubMed]
26. Liu, C.; Liu, Y.; Li, Q.; Wei, Z. Radar target MTD 2D-CFAR algorithm based on compressive detection. In Proceedings of the 2021 IEEE International Conference on Mechatronics and Automation (ICMA), Takamatsu, Japan, 8–11 August 2021; pp. 83–88.
27. Venter, C.; Grobler, H.; AlMalki, K. Implementation of the CA-CFAR algorithm for pulsed-Doppler radar on a GPU architecture. In Proceedings of the 2011 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), Amman, Jordan, 6–8 December 2011; pp. 1–6.
28. Nishino, R.; Loomis, S.H.C. CuPy: A NumPy-compatible library for NVIDIA GPU calculations. In Proceedings of the Workshop on Machine Learning Systems (LearningSys) in the Thirty-first Annual Conference on Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, 4–9 December 2017.
29. Yang, H.; Zhang, T.; He, Y.; Dan, Y.; Yin, J.; Ma, B.; Yang, J. GPU-Oriented Designs of Constant False Alarm Rate Detectors for Fast Target Detection in Radar Images. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5231214. [CrossRef]
30. Wu, S.; Xu, Z.; Wang, F.; Yang, D.; Guo, G. An improved back-projection algorithm for GNSS-R BSAR imaging based on CPU and GPU platform. *Remote Sens.* **2021**, *13*, 2107. [CrossRef]
31. Malanowski, M.; Kulpa, K.; Kulpa, J.; Samczynski, P.; Misiurewicz, J. Analysis of detection range of FM-based passive radar. *IET Radar Sonar Navig.* **2014**, *8*, 153–159. [CrossRef]
32. Passive Radar_20191102_1011.hdf5. Available online: https://drive.google.com/file/d/18dG__HnbuHJtG6WCHtPq3c_PRLqJA2O/view (accessed on 18 October 2023).