remote sensing

MDPI

Article

An Efficient BP Algorithm Based on TSU-ICSI Combined with GPU Parallel Computing

Ziya Li [1,2,3,4], Xiaolan Qiu [1,2,4,5,*], Jun Yang [4,5], Dadi Meng [1,2], Lijia Huang [1,2] and Shujie Song [1,2,3,4]

[1] Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China; liziya21@mails.ucas.ac.cn (Z.L.)
[2] Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Chinese Academy of Sciences, Beijing 100190, China
[3] School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China
[4] Suzhou Key Laboratory of Microwave Imaging, Processing and Application Technology, Suzhou 215124, China
[5] Suzhou Aerospace Information Research Institute, Suzhou 215124, China
* Correspondence: xlqiu@mail.ie.ac.cn

Abstract: High resolution remains a primary goal in the advancement of synthetic aperture radar (SAR) technology. The backprojection (BP) algorithm, which does not introduce any approximation throughout the imaging process, is broadly applicable and effectively meets the demands for high-resolution imaging. Nonetheless, the BP algorithm necessitates substantial interpolation during point-by-point processing, and the precision and effectiveness of current interpolation methods limit the imaging performance of the BP algorithm. This paper proposes a TSU-ICSI (Time-shift Upsampling-Improved Cubic Spline Interpolation) interpolation method that integrates time-shift upsampling with improved cubic spline interpolation. This method is applied to the BP algorithm and presents an efficient implementation method in conjunction with the GPU architecture. TSU-ICSI not only maintains the accuracy of BP imaging processing but also significantly boosts performance. The effectiveness of the BP algorithm based on TSU-ICSI is confirmed through simulation experiments and by processing measured data collected from both airborne SAR and spaceborne SAR.

Keywords: synthetic aperture radar (SAR); backprojection algorithm (BPA); GPU parallel computing; improved cubic spline interpolation; upsampling

check for updates

Citation: Li, Z.; Qiu, X.; Yang, J.; Meng, D.; Huang, L.; Song, S. An Efficient BP Algorithm Based on TSU-ICSI Combined with GPU Parallel Computing. *Remote Sens.* **2023**, *15*, 5529. https://doi.org/10.3390/rs15235529

Academic Editor: Timo Balz

Received: 26 August 2023
Revised: 12 November 2023
Accepted: 25 November 2023
Published: 27 November 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Synthetic aperture radar (SAR) operates as an active remote sensing imaging device within the microwave frequency band [1], capable of creating two-dimensional high-resolution images of observed scenes [2]. Increased resolution enhances the detail of scene and target information, which underscores the consistent drive towards high resolution in SAR technology development. For instance, microwave photonic (MWP) SAR can transmit signals with over 5 GHz bandwidth, yielding images with centimeter-level resolution [3–5]; spaceborne SAR, such as Capella, achieves high azimuth resolution through long-time staring spotlight processing [6,7]. However, these high-resolution SAR imaging techniques are confronted with problems such as severe echo signals coupling and significant parametric spatial variability, making batch-processing frequency domain algorithms (such as Chirp Scaling (CS), Omega-K, etc.) limited in their applications and unable to achieve the desired focusing precision. In contrast, the classical backprojection (BP) algorithm conducts point-by-point phase compensation and coherent accumulation and does not necessitate approximation during the processing. This results in high imaging accuracy and broad applicability, thus meeting the demands for high-resolution SAR imaging [8,9].

However, the main issue with the BP algorithm lies in its high computational complexity and low processing efficiency. Currently, to improve the processing efficiency of the BP

algorithm, there are mainly two types of methods. One type of method modifies the model and structure of the classical BP algorithm to form a fast BP algorithm, such as fast back-projection (FBP) [10], fast factorized backprojection (FFBP) [11], and Cartesian factorized backprojection (CFBP) [12]. However, these fast BP algorithms increase model complexity, and the imaging process involves approximations, which somewhat compromises the image quality [13,14]. Another type of method maintains the structure of the classical BP algorithm model, with primary enhancements focused on the core steps that impact the efficiency of the BP algorithm, notably the performance of interpolation, and deeply analyzes the characteristics of the BP algorithm to fully combine the computational benefits of advanced processing devices such as graphics processing units (GPUs) to achieve parallel acceleration [15–19]. At present, the interpolation methods commonly used in engineering that are suitable for the BP algorithm include linear or neighbor interpolation after FFT upsampling, sinc interpolation, and weighted sinc interpolation, etc. Literature [20] pointed out that the projection of ground range uniform grids to slant range non-uniform grids in the BP algorithm constitutes a type of non-uniform sampling, which can employ the non-uniform fast Fourier transform (NUFFT) method [21] to implement interpolation from the frequency domain data after range compression to image grids. This method was optimized on the GPU, yielding positive results. The study [22] utilizes 2-D NUFFT to carry out the two-dimensional interpolation required in FBP and FFBP, mitigating the truncated error effect in conventional interpolation methods. Nonetheless, NUFFT remains complex and leaves room for efficiency improvement. In 2020, Lin et al. [23] proposed an improved cubic spline interpolation (ICSI) method, which optimized the solution conditions of the traditional cubic spline interpolation, and improved the frequency response performance of the interpolation kernel by additional coefficient solving degrees of freedom, which can maintain the low computational complexity of traditional cubic spline interpolation while enhancing interpolation accuracy. Lin et al. have successfully applied this method to wireless communication signal processing [24].

This paper, inspired by previous research, is the first to incorporate the ICSI method into the BP algorithm. It combines this method with time-shift upsampling to propose an efficient BP algorithm based on time-shift upsampling-improved cubic spline interpolation (TSU-ICSI). By harnessing the parallel features of the proposed BP algorithm and the architectural characteristics of GPUs, we have realized an efficient BP algorithm based on TSU-ICSI combined with GPU parallel computing, which has achieved better performance than that of previously established interpolation methods.

The rest of this paper is organized as follows. In Section 2, we introduce the signal model, algorithm flow, parameter selection, and computational complexity analysis of the BP algorithm based on TSU-ICSI. In Section 3, an efficient GPU-based TSU-ICSI BP implementation method is presented. In Section 4, we conduct both simulation and measured data experiments on the proposed method, confirming its effectiveness and analyzing its contribution to enhancing processing efficiency. A discussion and conclusions are given in Sections 5 and 6, respectively.

2. BP Algorithm Based on TSU-ICSI

2.1. Signal Model and Algorithm Process

The transmitted signal of SAR is typically described by:

$$s(t) = \text{rect}\left(\frac{t}{T}\right)\exp\left(j2\pi f_0 t + j\pi k_r t^2\right), \tag{1}$$

where f_0 represents the carrier frequency, k_r represents the frequency-modulated rate of the transmitted signal, T represents the pulse duration, and rect( ) represents the rectangular window. Figure 1 illustrates the schematic diagram of the BP algorithm principle. Let (x_i, y_j, z_{ij}) denote the coordinates of the grid points in the imaging area. At azimuth

moment η, with the SAR positioned at $(x(\eta), y(\eta), z(\eta))$, the distance between the radar and the grid points can be expressed as:

$$R_{ij}(\eta) = \sqrt{(x_i - x(\eta))^2 + (y_j - y(\eta))^2 + (z_{ij} - z(\eta))^2}. \tag{2}$$

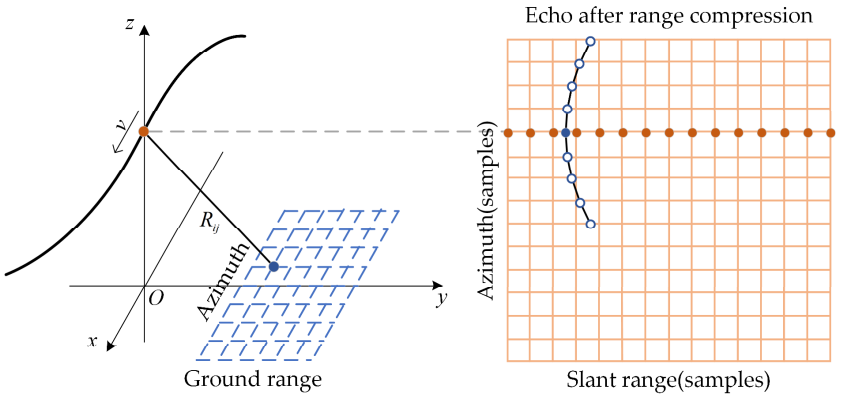


Figure 1. Schematic diagram of BP algorithm principle.

The received signal captured by the receiver can be expressed as follows:

$$s_r(t, \eta) = \sum_{i,j} \sigma_{ij} G(t, \eta; x_i, y_j) s\left\{ t - \frac{2R_{ij}(\eta)}{c} \right\}, \tag{3}$$

and the demodulated baseband signal can be represented as follows:

$$s_r(t, \eta) = \sum_{i,j} \sigma_{ij} G(t, \eta; x_i, y_j) s\left\{ t - \frac{2\sqrt{(x_i - x(\eta))^2 + (y_j - y(\eta))^2 + (z_{ij} - z(\eta))^2}}{c} \right\}, \tag{4}$$

where σ_{ij} represents the backscatter and $G(t, \eta; x_i, y_j)$ represents the antenna pattern.

Given that the range matched filter is fixed, a frequency domain method is commonly used for efficient range matched filtering. The result after range compression can be expressed as:

$$s_M(t, \eta) = \sum_{i,j} \sigma_{ij} G(t, \eta; x_i, y_j) \mathrm{sinc}(t - 2R_{ij}(\eta)/c) \exp(-j4\pi f_0 R_{ij}(\eta)/c), \tag{5}$$

accordingly, the core formula for BP imaging can be illustrated as:

$$f(x_i, y_j, z_{ij}) = \int_{\eta} s_M[t_{ij}(\eta), \eta] \exp(j4\pi f_0 R_{ij}(\eta)/c) d\eta. \tag{6}$$

Hence, the process of reconstructing the image from the echo using the BP algorithm is illustrated in Figure 2. The SAR echo initially undergoes range matched filtering to obtain the range compressed echo data. Subsequently, based on parameters like the slant range, azimuth observation range, sampling rate, and azimuth resolution, the ground grid range and grid spacing are determined, and the imaging grid is partitioned accordingly. Following this, for each azimuth time η, the imaging grid points within the beam illumination range are identified. The slant range between each of these grid points and the radar position at that specific moment η is computed to derive the time delay $t_{ij}(\eta)$. The echo pixel location corresponding to the time delay $t_{ij}(\eta)$ is then obtained from the echo $s_M(t, \eta)$ at the azimuth time η. As seen in Figure 1, the echo position often does not match the radar sampling points, necessitating interpolation to obtain the corresponding echo value. In this paper, the proposed TSU-ICSI method is employed for interpolation. Following this, the

obtained echo values undergo phase compensation, or in other words, are multiplied by $\exp\left(j4\pi f_0 R_{ij}(\eta)/c\right)$ and the compensated echo values are then accumulated. This process is carried out until all echo pulses are processed, yielding the final focused SAR image.

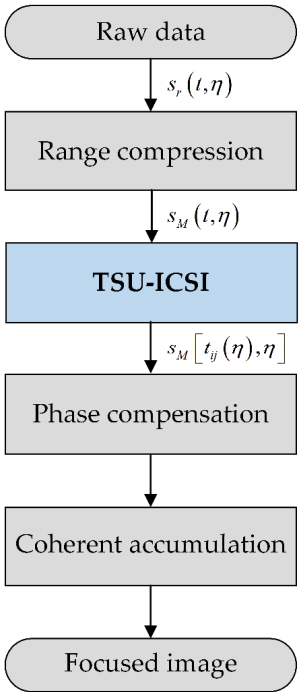


Figure 2. Flowchart of BP algorithm based on the TSU-ICSI method.

2.2. TSU-ICSI Method

2.2.1. Interpolation Method

The time-shift property of Fourier transform states that a rightward shifting by n_0 of a signal in the time domain results in

$$\begin{aligned} x(n-n_0) &= \frac{1}{N}\sum_{k=0}^{N-1} X(k)\exp\left\{j\frac{2\pi k(n-n_0)}{N}\right\} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\left(\exp\left\{-j\frac{2\pi kn_0}{N}\right\}X(k)\right)\exp\left\{j\frac{2\pi kn}{N}\right\} \end{aligned}, \tag{7}$$

in other words, a shifting to the right by n_0 in the time domain is equivalent to multiplying the signal by a negative exponential linear phase in the frequency domain, followed by performing the inverse Fourier transform (IFT). Based on this principle, upsampling by time-shift property can enable integer multiple upsampling. If upsampling by a factor of l is desired, the spectrum function of the original signal can be multiplied by the linear phase $e^{-j\frac{2\pi ki}{lN}}$, $i = 1, \ldots, l-1$, then the inverse Fourier transform can be executed to derive a signal that is shifted to the right by $\frac{i}{l}$, $i = 1, \ldots, l-1$. The process of interleaving the shifted signals can subsequently realize a signal upsampled by a factor of l.

When upsampling by a factor of l is applied to each row of a matrix containing $M \times N$ points, the computational complexity of the zero-padding upsampling method is $4lMN\log_2(lN)$. In contrast, the computational complexity of the time-shift upsampling method proposed in this paper is $4lMN\log_2 N$. Evidently, time-shift upsampling provides a significant advantage in decreasing computational complexity and time.

Building upon the foundation of upsampling, cubic spline interpolation can be improved in the time domain to achieve favorable interpolation results. The classic cubic spline interpolation function is assembled from piecewise cubic polynomials, requiring the first and second derivatives of the piecewise polynomials to be continuous at the junction points. However, these conditions alone are insufficient to exclusively determine the

coefficients of the piecewise polynomial. Additional boundary conditions are typically necessary. If the second derivative is zero at both endpoints, it is referred to as the natural boundary condition.

For uniform data points, a schematic diagram of cubic spline interpolation using eight-point data is depicted in Figure 3. If the interval between two adjacent data points is T_S, the x-coordinate of the initial data point is $(n+i)T_s$, $i = -3, \ldots, 4$, the y-coordinate is $y[n+i]$, $i = -3, \ldots, 4$, and the piecewise cubic polynomial function is $S_{n+i}(x)$, $i = -3, \ldots, 3$, $x \in [0, T_S]$. The interpolation points are represented as brown points in Figure 3, and the distance from nT_S is αT_S, $\alpha \in [0, 1)$. The value at the interpolation point can be denoted as

$$y[n+\alpha] = S_n(\alpha) = a_n + b_n\alpha + c_n\alpha^2 + d_n\alpha^3. \tag{8}$$

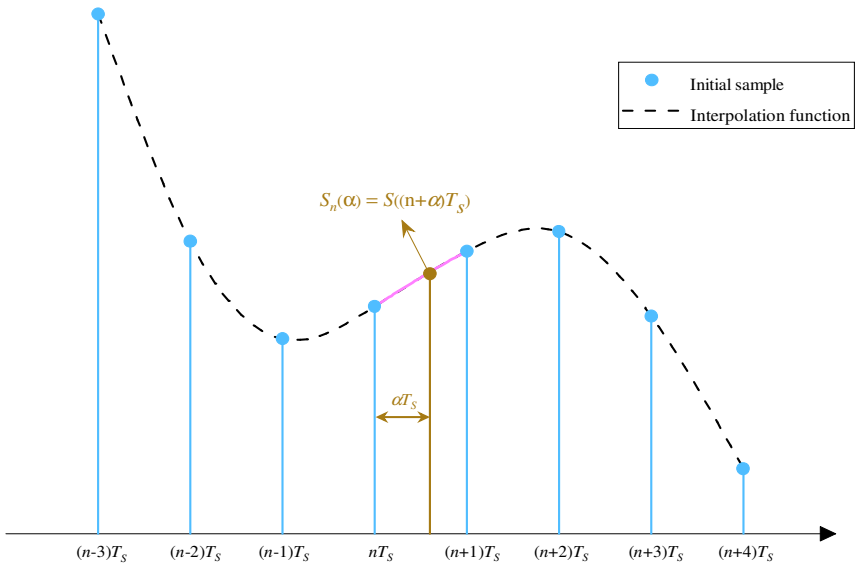


Figure 3. Schematic diagram of cubic spline interpolation.

The improved cubic spline interpolation proposed in [23] maintains the condition of continuous second derivatives at the junction points and introduces a condition where the second derivative can be represented as a linear combination of adjacent samples with adjustable coefficients a_k, as described in Equation (10). The coefficients a_k are computed by approximating the frequency response of the improved cubic spline interpolator to the ideal interpolator within an allowable error range, thereby implementing the coefficient solution for the improved cubic spline interpolation method. The specific derivation process is as follows.

Based on the retained conditions, the following equation can be derived

$$\begin{cases} d_n = \frac{S_{n+1}''(0) - S_n''(0)}{6} \\ c_n = \frac{S_n''(0)}{2} \\ b_n = y[n+1] - y[n] - d_n - c_n \\ a_n = y[n] \end{cases}. \tag{9}$$

For an M-sample interpolation with an even integer M, assuming that the second derivative can be expressed as a weighted sum of the data points, it follows that

$$S_n''(0) = \sum_{k=-\frac{M}{2}+1}^{\frac{M}{2}-1} a_k y[n+k], \tag{10}$$

where, a_k is a real number and $a_k = a_{-k}$. By simplifying Equations (9) and (10), $S_n(\alpha)$ can be expressed as a function of a_k. Equation (11) provides the expression for $S_n(\alpha)$ with eight-sample.

$$\begin{aligned} S_n(\alpha) &= \begin{bmatrix} y[n-3] & \cdots & y[n+4] \end{bmatrix} \begin{bmatrix} -\frac{a_3}{6} & \frac{a_3}{2} & -\frac{a_3}{3} & 0 \\ \frac{a_3-a_2}{6} & \frac{a_2}{2} & -\frac{a_3+2a_2}{6} & 0 \\ \frac{a_2-a_1}{6} & \frac{a_1}{2} & -\frac{a_2+2a_1}{6} & 0 \\ \frac{a_1-a_0}{6} & \frac{a_0}{2} & -\frac{a_1+2a_0+6}{6} & 1 \\ \frac{a_0-a_1}{6} & \frac{a_1}{2} & -\frac{a_0+2a_1-6}{6} & 0 \\ \frac{a_1-a_2}{6} & \frac{a_2}{2} & -\frac{a_1+2a_2}{6} & 0 \\ \frac{a_2-a_3}{6} & \frac{a_3}{2} & -\frac{a_2+2a_3}{6} & 0 \\ \frac{a_3}{6} & 0 & -\frac{a_3}{6} & 0 \end{bmatrix} \begin{bmatrix} \alpha^3 \\ \alpha^2 \\ \alpha \\ 1 \end{bmatrix}. \\ &= \sum_{k=-3}^{4} y[n+k]F_k(\tau) \end{aligned} \tag{11}$$

Thus, the impulse response of the improved cubic spline interpolation is illustrated in Equation (12), and the corresponding frequency response is depicted in Equation (13).

$$h[n] = F_{\lfloor \frac{n}{L} \rfloor + 1}\left(1 - \frac{n(\mathrm{mod}L)}{L}\right), \tag{12}$$

$$H\left(e^{j\omega}\right) = h[0] + 2\sum_{m=1}^{\frac{8L}{2}-1} h[m]\cos m\omega, \tag{13}$$

where L represents the upsampling factor, and the frequency response can be optimized by adjusting the coefficients $a_0 \sim a_3$ to approach the ideal rectangular low-pass filter $G\left(e^{j\omega}\right)$, thereby enhancing the spectral performance of the improved cubic spline interpolation method. The closer the spectrum of the interpolation kernel approaches the spectrum of the ideal low-pass filter, the higher the interpolation accuracy. The calculation steps for improved cubic spline interpolation are fundamentally consistent with traditional cubic spline interpolation. Therefore, compared to the traditional cubic spline interpolation method, the improved cubic spline interpolation method enhances interpolation accuracy while maintaining low computational complexity.

2.2.2. Analysis of Computational Complexity for Interpolation Methods

Assume the size of the echo matrix to be $N_a \times N_r$, the number of grid points in the imaging area to be $N_x \times N_y$, the upsampling factor to be l, and the number of interpolation points to be m. Let $\alpha(\eta)$, $\alpha(\eta) \in [0, 1]$ denote the proportion of grid points illuminated at the azimuth time η, calculated based on the azimuth antenna pattern and other information. In spotlight mode, $\alpha(\eta)$ remains constant at 1, while in stripmap or sliding spotlight modes, $\alpha(\eta)$ varies with the azimuth time η.

In Equation (11), we present the computation formula for the improved cubic spline interpolation method, where $x[n+i]$ are complex numbers; a_k and α are real numbers. The multiplication of the first two matrices involves $4 \times 2m$ real number multiplications and $4 \times 2(m-1)$ real number additions, yielding a 1×4 dimensional complex vector. Multiplying this vector by the third 4×1 dimensional real vector requires 8 real number multiplications and 6 real number additions. Consequently, for a single azimuth time, an improved cubic spline interpolation at a grid point necessitates $(16m + 6)$ instances of real number multiplications and additions. Considering the above-mentioned time-shifted upsampling computational complexity, we can calculate the theoretical computational complexity of the TSU-ICSI method as follows

$$O(\mathrm{TSU-ICSI}) = 4lN_aN_r\log_2 N_r + \alpha(\eta)N_aN_xN_y(16m+6). \tag{14}$$

The sinc interpolation kernel is centered at the coordinate of the interpolation point. After calculating the weights at the sampling points, we obtain the weighted sum of the sample points as follows

$$g(x) = \sum_i g_d(i)\mathrm{sinc}(x - i), \tag{15}$$

where, $g_d(i)$ are complex numbers and $\mathrm{sinc}(x-i) = \frac{\sin[\pi(x-i)]}{\pi(x-i)}$ are real numbers. Thus, the sinc interpolation of m points requires $(4m-2)$ real number additions, $5m$ real number multiplications, and m sin operations (the computational complexity of sin operations is represented as $O(\sin)$ below). We then compute the theoretical computational complexity of the sinc interpolation method as follows

$$O(\mathrm{sinc}) = \alpha(\eta)N_aN_xN_y(9m - 2 + mO(\sin)). \tag{16}$$

When the number of interpolation points m is constant, we calculate the difference in the theoretical computational complexity between these two interpolation methods can be expressed as

$$O(\mathrm{sinc}) - O(\mathrm{TSU-ICSI}) = \alpha(\eta)N_aN_xN_y\left[mO(\sin) - 7m - 8 - \frac{4lN_r\log_2 N_r}{\alpha(\eta)N_xN_y}\right]. \tag{17}$$

During the measured data processing, we generally find that $\frac{4lN_r\log_2 N_r}{\alpha(\eta)N_xN_y} < 1$. Across different programming languages and platforms, the computational complexity of the sin function fluctuates, but it is common to find that $mO(\sin) - 7m - 8 > 1$. As a result, the computational complexity of sinc interpolation is commonly higher than that of the TSU-ICSI method.

The ratio of the theoretical computational complexities for the two interpolation methods can be denoted as

$$\frac{O(\mathrm{sinc})}{O(\mathrm{TSU-ICSI})} = \frac{\alpha(\eta)N_aN_xN_y(9m - 2 + mO(\sin))}{4lN_aN_r\log_2 N_r + \alpha(\eta)N_aN_xN_y(16m+6)}. \tag{18}$$

When $\alpha(\eta) \neq 0$, we obtain

$$\frac{O(\mathrm{sinc})}{O(\mathrm{TSU-ICSI})} = \frac{9m - 2 + mO(\sin)}{\frac{4lN_r\log_2 N_r}{\alpha(\eta)N_xN_y} + (16m+6)}. \tag{19}$$

From the above equation, it is evident that the value of $\frac{O(\mathrm{sinc})}{O(\mathrm{TSU-ICSI})}$ increases as $\alpha(\eta)$ grows. This is one of the reasons why the processing efficiency of the TSU-ICSI method shows further improvement compared to the sinc interpolation method, which is consistent with the measured data results in Section 4.2.2.

In Equation (14), the upsampling factor l is a value that is relatively important for both interpolation efficiency and precision. A simulation in Section 4.1.2 will present an optimal choice for the parameter l.

3. Efficient Implementation of the GPU-Based TSU-ICSI BP Algorithm

The SAR backprojection algorithm consists of a substantial amount of independent computing units, where the backprojection computation of each grid point does not correlate with other grid points. This abundance of parallel computing and data processing demand aligns perfectly with the hardware characteristics of GPUs. Under the compute unified device architecture (CUDA), by intelligently organizing threads, managing memory, and optimizing function usage, we can effectively harness the multi-core parallel processing capabilities of GPUs to efficiently implement the TSU-ICSI BP algorithm.

Figure 4 illustrates the simplified diagram of GPU architecture. GPU is constructed around an array of streaming multiprocessors (SMs). Each GPU incorporates several SMs,

and each SM consists of multiple streaming processors (SPs), also referred to as CUDA cores, which are responsible for specific instructions and task computations. The simultaneous computation by multiple SPs constitutes the parallel processing of the GPU.

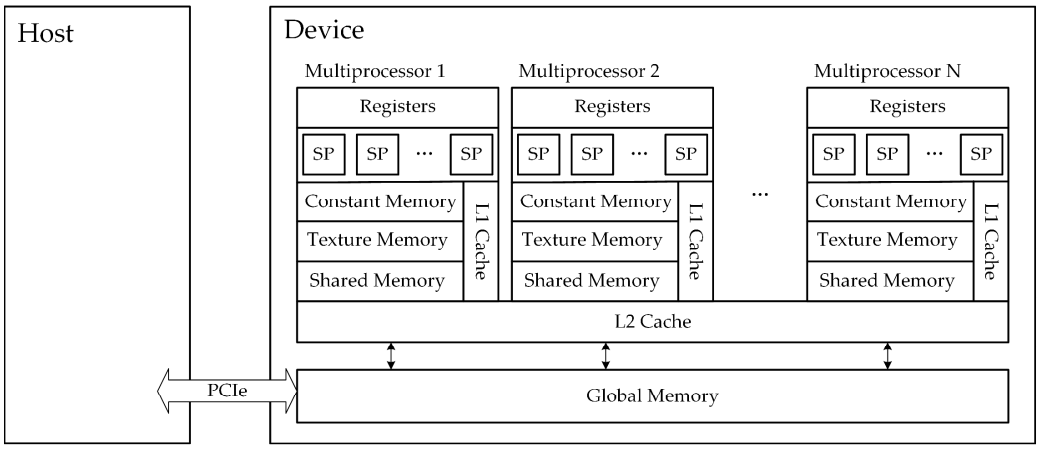


Figure 4. Simplified diagram of GPU architecture.

The GPU memory system is typically a multi-tiered structure comprising different types of memory. Registers offer the fastest access speed within the GPU, with each thread having its own registers, though in limited numbers. Shared memory can facilitate data sharing among threads within a block of threads, with low latency and access speed only second to registers. Texture memory, optimized for 2D spatial locality, facilitates efficient spatial locality queries. Constant memory is designed to broadcast constants to all threads and offers high read speeds when multiple threads access the same data. Lastly, Global memory, or device memory, accessible to all threads and the host, has the largest storage capacity despite its relatively higher latency and slower speed. In the proposed thread organization method, Figure 5, every thread forming an image grid needs to access the SAR platform orbit coordinates. Hence, the CUDA kernel function "TSU-ICSI BP kernel" stores the orbit coordinates in shared memory for fast access, while each section of the two-dimensional echo data is stored in global memory binding with texture memory to accelerate reading.

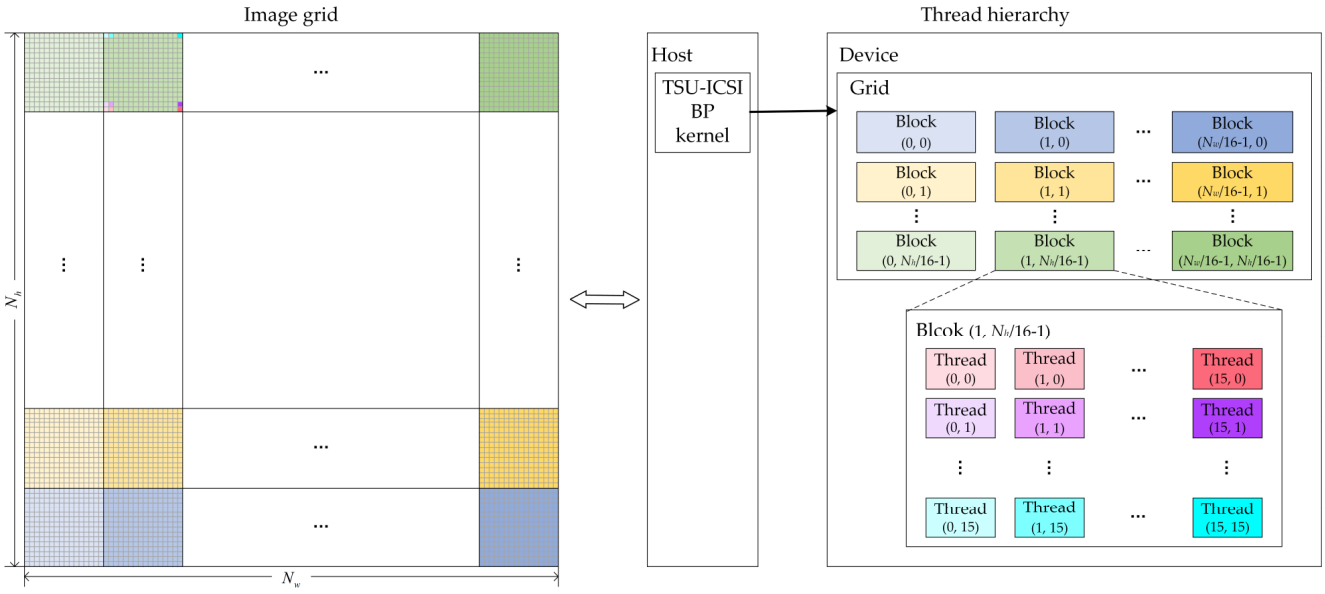


Figure 5. Schematic diagram of thread organization for TSU-ICSI BP kernel.

When the host triggers the "TSU-ICSI BP kernel" function, it needs to define the organization of threads as launch parameters to control the function's execution on the device. The proposed method for efficient GPU implementation binds two-dimensional image pixels and threads, configures threads and pixels consistently, and arranges them into two-dimensional grid and two-dimensional blocks, with each pixel assigned a thread to fully utilize the GPU SPs. In terms of the selection of the size of two-dimensional blocks, since a warp is the basic execution unit in SM, in most modern NVIDIA GPUs, a warp contains 32 threads. When the "TSU-ICSI BP kernel" is launched, the corresponding grids are also launched. After the blocks contained in the grid are allocated to the SMs, they are divided into multiple warps. In a warp, all threads execute in a single-instruction, multiple-thread (SIMT) manner. Thus, if the size of a block is an integral multiple of the warp size, it can maximize hardware utilization. When processing two-dimensional data with two-dimensional grids and two-dimensional blocks, this paper chooses a block size of 16×16. On the one hand, this is because 256 threads contain an integer number of warps and can meet the upper limit of the number of threads in a block on the hardware (usually limited to 1024 threads). On the other hand, this configuration can effectively utilize the hardware's memory-sharing function. Such a size can usually map well to physical hardware and demonstrates excellent performance in many applications. As a result, to generate an image of $N_w \times N_h$ pixels, we need to construct blocks of size 16×16 and a grid comprising $(N_w/16) \times (N_h/16)$ blocks, as shown in Figure 5.

This method of thread organization significantly boosts the utilization of SMs. In the GPU efficient implementation method of the BP algorithm based on TSU-ICSI, threads within the same block can fast access the orbit coordinates stored in shared memory. At the same time, threads are organized such that adjacent threads access adjacent echo data, thereby binding echo data with texture memory can speed up access. Lastly, optimized floating-point trigonometric functions are used to improve processing efficiency while ensuring imaging precision. Subsequent experimental results demonstrate that the GPU efficient implementation method of the BP algorithm based on TSU-ICSI proposed in this paper achieves further efficiency improvements compared to conventional GPU parallel methods that use one-dimensional threads and global memory.

The pseudocode is shown in Figure 6.

```
Divide the echo into several batches of pulses
Allocate host and GPU memory for the orbit coordinates
Allocate host and GPU memory for the image coordinates
Allocate host and GPU memory for the image grid
Allocate host and GPU memory for a batch of pulses after upsampling
For each batch of pulses
    Time-shift upsampling of the echo after range compression
     Copy a batch of pulses to GPU memory
     Bind valid echo data to texture cache
     Mapping a thread for each image pixel
     Call TSU-ICSI BP kernel
        Copy orbit coordinates to shared memory
        For each thread (image pixel)
            For each pulse in batch
                Calculate distance and angle
                Calculate expected phase
                Use ICSI to calculate corresponding echo values
                Phase compensation
                Accumulate the results
    Unbind and free texture cache resources
Copy the image result from the GPU
Save the complex image
Free host and GPU memory resources
```

Figure 6. Pseudocode for the GPU-based implementation of the TSU-ICSI BP algorithm.

4. Experimental Validation and Analysis

4.1. Simulation Validation

4.1.1. Simulation Parameters

The simulation parameters are as shown in Table 1. In the range direction, a one-dimensional SAR simulated echo is set up with several point targets, and complex white Gaussian noise with a signal-to-noise ratio of 30 dB is added.

Table 1. Main Parameters for Simulation.

Parameter	Value
Carrier frequency	9.4 GHz
Signal bandwidth	100 MHz
Sampling frequency	120 MHz
Transmitted pulse duration	10 μs
Near range	29.12560 km
Altitude	10 km
Effective radar velocity	250 m/s
Doppler bandwidth	433 Hz
Pulse repetition frequency	600 Hz
Azimuth beamwidth	1.624°
Squint angle	0°

4.1.2. Simulation Results

First, we conduct simulation experiments in the one-dimensional range direction, comparing the precision and efficiency of various interpolation methods: linear, sinc, 8-point natural boundary condition cubic spline, improved cubic spline, and 7-point non-equispaced results inverse fast Fourier transform (NERIFFT) under different upsampling factors. After performing range compression on the echo, we carry out the time-shift upsampling. The upsampled signal is then interpolated using the aforementioned methods. Note, that sinc interpolation contains less information after upsampling, hence we do not perform sinc interpolation after upsampling. Also, NERIFFT cannot be used without upsampling as it requires upsampling during implementation.

In this paper, the results of the non-uniform inverse discrete Fourier transform (NUIDFT) are considered as the truth value, and the expression for NUIDFT is shown in Equation (20). The root mean square error (RMSE) is used to measure the interpolation error, as shown in Equation (21). The error curves under different upsampling factors are shown in Figure 7.

$$z_l = \sum_{k=-N/2}^{N/2-1} e^{2\pi i x_l k/N} \hat{z}_k,\ l = 1, \ldots, M, \tag{20}$$

where, $\hat{z}_k$ represents the uniformly sampled data at equal intervals with a total number of points N, and z_l denotes the time-domain data at non-uniform grid points $x_l \in \left[-\frac{N}{2}, \frac{N}{2}\right]$, obtained through the non-uniform discrete Fourier inverse transform of $\hat{z}_k$, with a total number of points M.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - y_i)^2}{N}}, \tag{21}$$

where x_i represents the truth value, y_i is the estimated value, and N represents the number of data points.

According to Figure 7, when there is no upsampling (Upsampling factor = 1) among the interpolation methods used in the experiment, the 32-point sinc interpolation method has the highest precision, while the improved cubic spline interpolation has a higher precision than the traditional cubic spline interpolation. However, it is worth noting that the improved cubic spline interpolation is the most sensitive to upsampling. When

upsampling by a factor of 3, the precision of the 8-point improved cubic spline interpolation method significantly improves, and it greatly surpasses other interpolation methods used in the experiment. Figure 8 shows that the improved cubic spline interpolation method maintains the low computational complexity of the traditional cubic spline interpolation method, resulting in shorter runtime compared to the sinc interpolation and NERIFFT. Therefore, compared with the existing commonly used interpolation methods, the TSU-ICSI method, which combines the threefold time-shift upsampling and 8-point improved cubic spline interpolation, can simultaneously improve the precision and efficiency of interpolation. It is now applied to BP imaging interpolation processing, thereby achieving highly efficient and precise BP imaging.

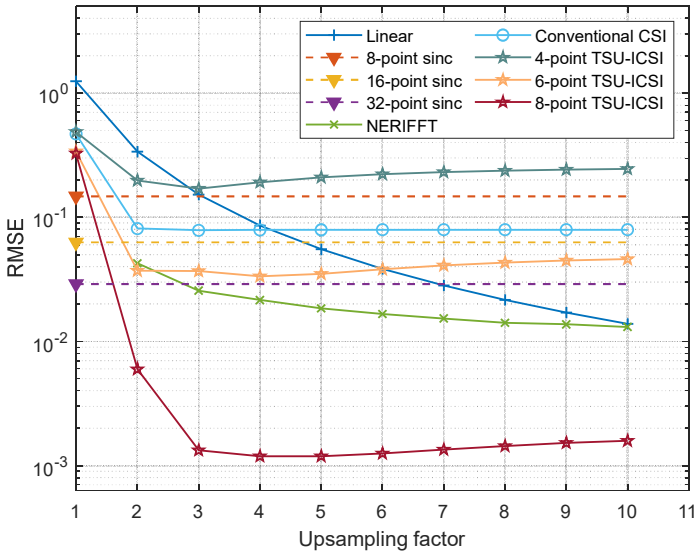


Figure 7. RMSE curves of interpolation methods under different upsampling factors.

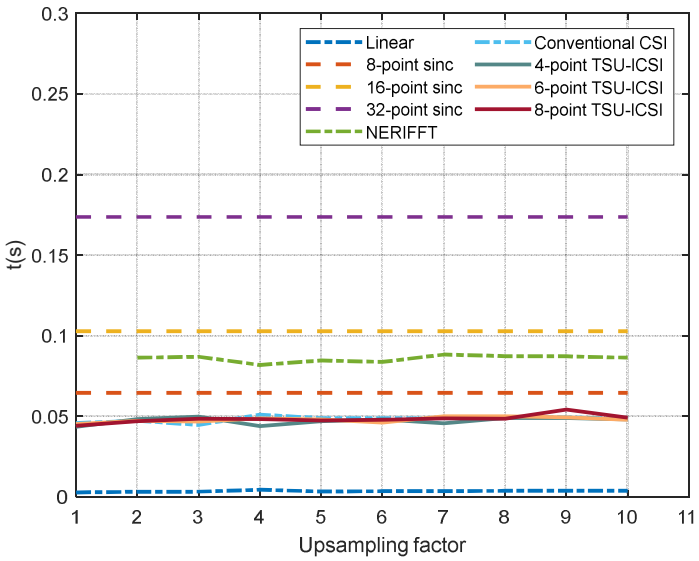


Figure 8. Time taken by interpolation methods under different upsampling factors.

A two-dimensional point target echo simulation was also conducted using the parameters listed in Table 1. Nine point targets were set, and the BP algorithm based on the TSU-ICSI method was used for imaging. The imaging results are shown in Figure 9. BP imaging based on the aforementioned interpolation methods was performed. The range profiles and azimuth profiles of point 9 (in the third row and third column) are shown in Figures 10 and 11, respectively.

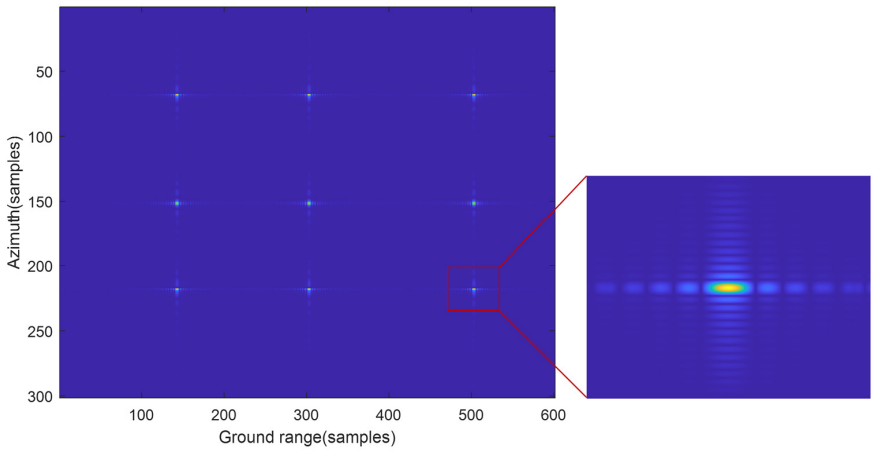


Figure 9. Imaging results of the BP algorithm based on TSU-ICSI.

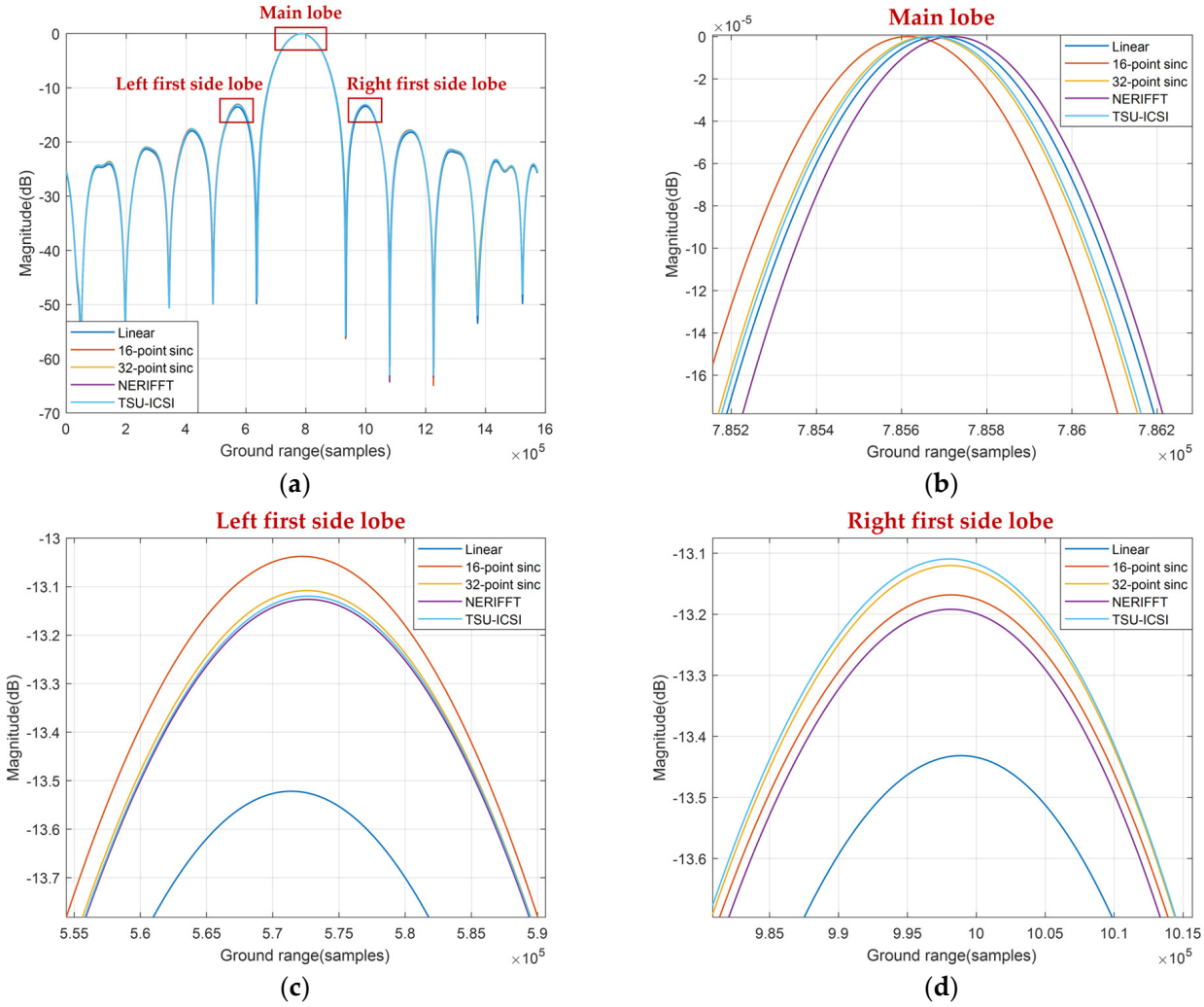


Figure 10. (**a**) Range profiles of point 9; (**b**) The main lobe of range profiles; (**c**) The left first side lobe of range profiles; (**d**) The right first side lobe of range profiles.

Figure 9 shows that the simulated points are well-focused. As indicated in Figures 10 and 11, among the illustrated interpolation methods, the 32-point sinc interpolation, NERIFFT, and TSU-ICSI present similar shapes for the main lobe and the first side lobes in the point, demonstrating good performance and yielding more accurate imaging results. Therefore,

the two-dimensional simulation experiment indicates that the BP algorithm based on the TSU-ICSI method provides correct imaging results and good imaging quality.

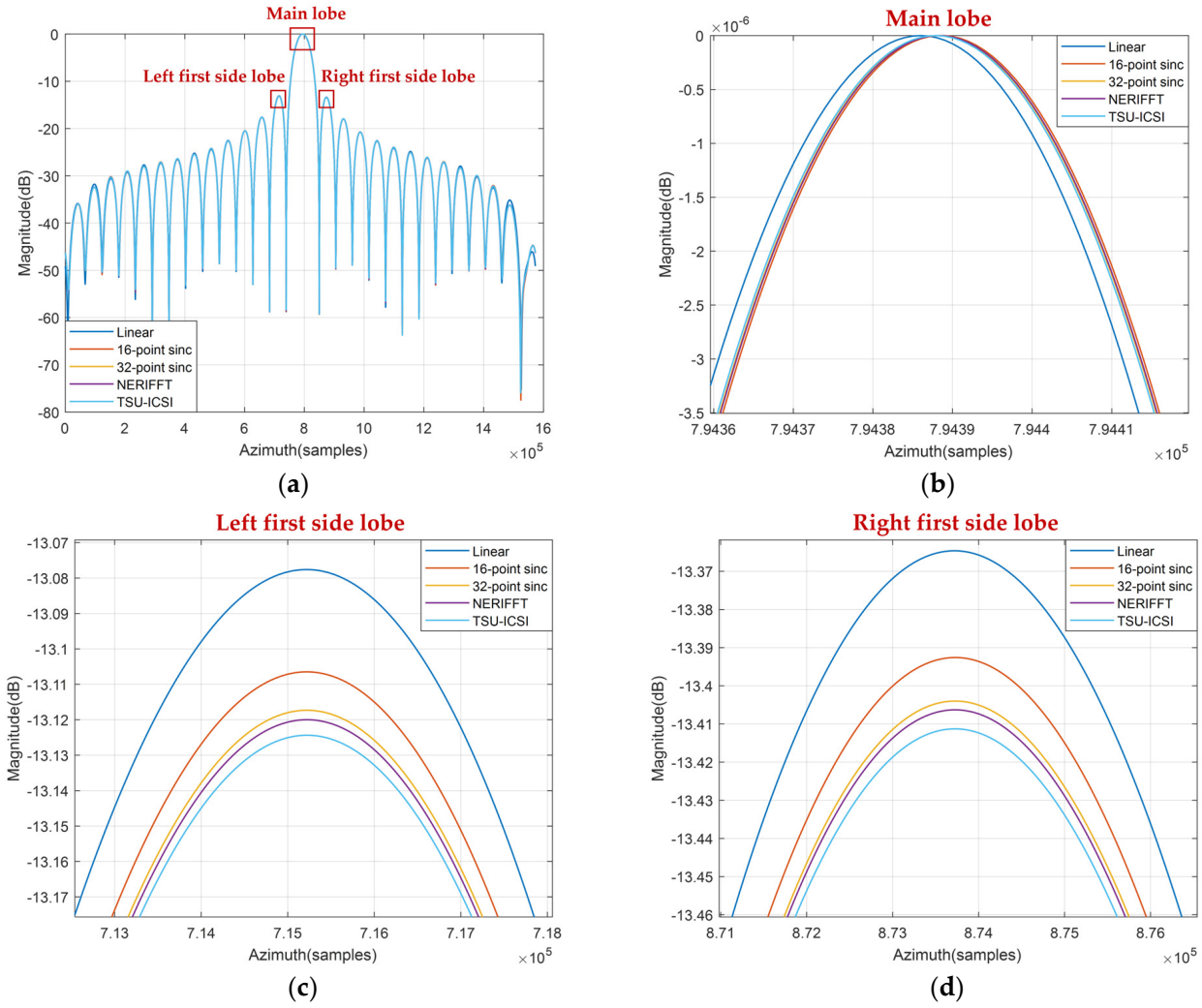


Figure 11. (**a**) Azimuth profiles of point 9; (**b**) The main lobe of azimuth profiles; (**c**) The left first side lobe of azimuth profiles; (**d**) The right first side lobe of azimuth profiles.

4.2. Validation and Analysis of Measured Data

4.2.1. Measured Data Parameters

In this section, imaging is conducted using measured data from airborne stripmap mode and spaceborne spotlight mode, with the parameters of the measured data shown in Tables 2 and 3, respectively.

Table 2. Parameters of the airborne measured data.

Parameter	Value
Carrier frequency	15.2 GHz
Signal bandwidth	600 MHz
Sampling frequency	625 MHz
Transmitted pulse duration	28 μs
Near range	4747.1245 km
Doppler bandwidth	245.0095 Hz
Pulse repetition frequency	625 Hz
Azimuth beamwidth	3°
Squint angle	0°

Table 3. Parameters of the spaceborne measured data.

Parameter	Value
Carrier frequency	16.7 GHz
Signal bandwidth	600 MHz
Sampling frequency	800 MHz
Transmitted pulse duration	53.92 μs
Near range	640.7464 km
Pulse repetition frequency	6843.6901 Hz
Squint angle	0°

4.2.2. Analysis of Measured Data

The GPU-based TSU-ICSI BP high-efficiency parallel method proposed in the previous section is implemented on a workstation equipped with an AMD Ryzen 9 3950X 16-Core Processor and NVIDIA Quadro RTX 8000. The imaging results of the airborne measured data are shown in Figure 12. The runtime comparison between the proposed GPU efficient implementation method and the conventional GPU parallel method utilizing one-dimensional threads and global memory is illustrated in Table 4. The table reveals that the proposed GPU efficient implementation method has further improved the imaging efficiency. The normalized mean square error (NMSE) between the GPU and CPU processing results is 3.5854×10^{-5}, where the NMSE is calculated as shown in Equation (22). Selecting the strong scattering point in the red box of Figure 12, the comparison of the profiles processed by CPU and GPU is presented in Figures 13 and 14. The results indicate that the influence of GPU processing on accuracy is minor.

$$\mathrm{NMSE} = \frac{\sum_{i=1}^{N} (x_i - y_i)^2}{\sum_{i=1}^{N} x_i{}^2}, \tag{22}$$

where, x_i represents the truth value, y_i is the estimated value, and N represents the number of data points.

Table 4. Runtime comparison between the proposed GPU efficient implementation method and the conventional GPU parallel method.

GPU Parallel Algorithms	The Proposed GPU Efficient Implementation Method	Conventional GPU Parallel Method
BP algorithm's GPU runtime based on TSU-ICSI (s)	152	273
BP algorithm's GPU runtime based on 16-point sinc interpolation (s)	477	856
BP algorithm's GPU runtime based on 32-point sinc interpolation (s)	848	1515

Spaceborne measured data are used for GPU-based TSU-ICSI BP imaging. The imaging results are shown in Figure 15, with the range and azimuth profiles of the strong scattering points depicted in Figures 16 and 17. The results indicate that both TSU-ICSI and 32-point sinc interpolation methods have high accuracy, consistent with the results of simulation experiments, further proving the validity and accuracy of the TSU-ICSI method. The runtime of GPU-based BP imaging with different interpolation methods for airborne and spaceborne measured data is shown in Table 5. As can be observed, the BP algorithm's runtime based on the TSU-ICSI method is significantly reduced compared to the BP algorithm's runtime based on sinc interpolation, whether in the airborne or spaceborne case, achieving an enhancement in imaging efficiency. As shown in Table 6, compared with the stripmap mode, there are more azimuth points in the synthetic aperture time under the spotlight mode. According to Equation (19), as $\alpha(\eta)$ increases, the ratio of $\frac{O(\mathrm{sinc})}{O(\mathrm{TSU-ICSI})}$ increases. Moreover, since sinc interpolation contains more flow control

instructions like loops and judgments than improved cubic spline interpolation, it results in thread divergence, significantly affecting the effective instruction throughput and subsequently increasing GPU computation time. Therefore, under the spaceborne spotlight mode, as the number of azimuth points within the synthetic aperture time increases, the ratio of GPU runtime of the BP algorithm based on sinc interpolation to that based on TSU-ICSI increases compared with the airborne stripmap mode, and the imaging efficiency of the BP algorithm based on TSU-ICSI is more notably improved, further demonstrating the superiority of this method.

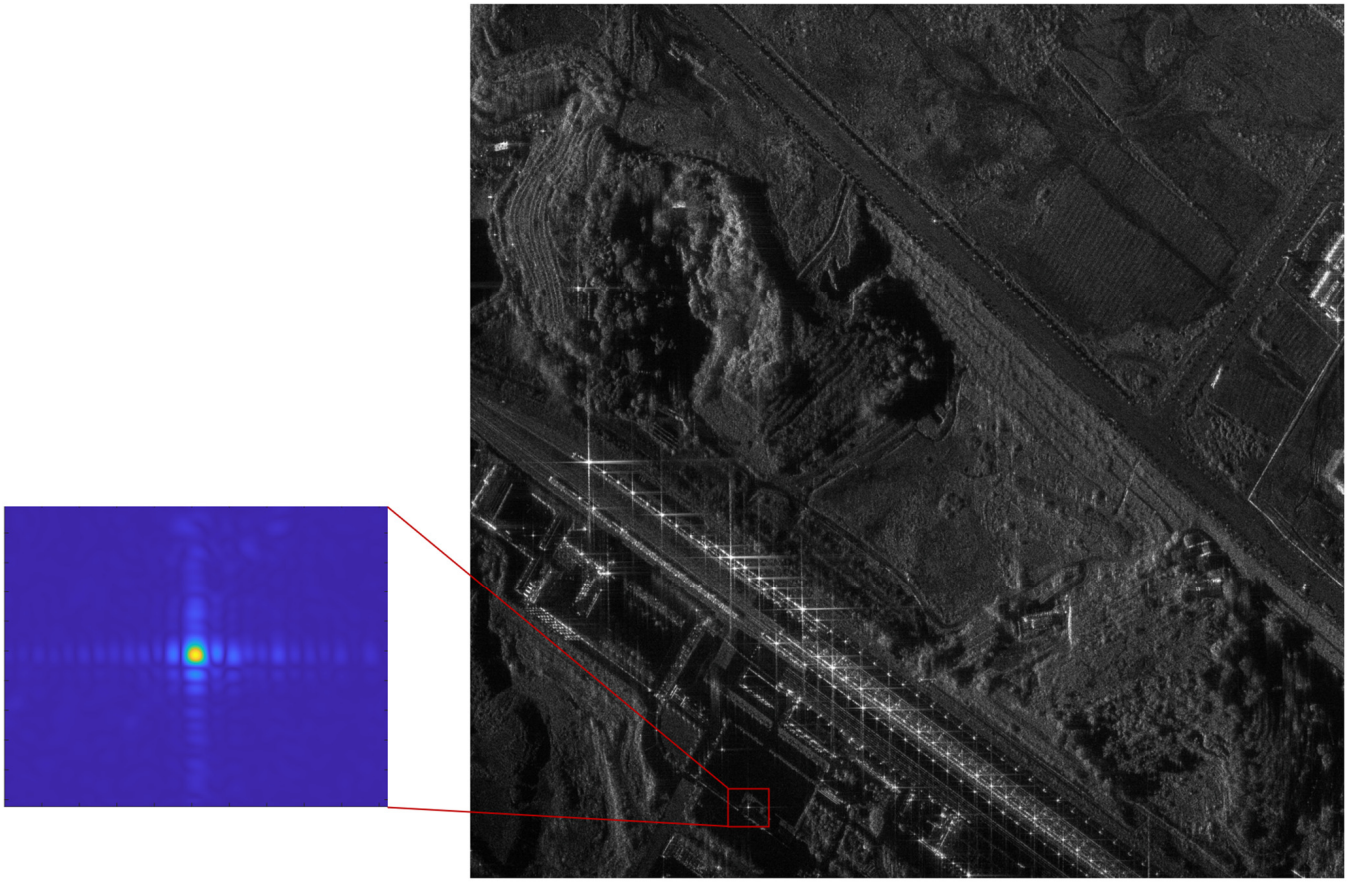


Figure 12. An example of airborne data processed by the BP algorithm based on TSU-ICSI implemented using GPU.

Table 5. Parameters and imaging time of airborne and spaceborne data.

SAR Platform	Airborne	Spaceborne
Imaging mode	Stripmap mode	Spotlight mode
Size of echo $(N_a \times N_r)$	10,000 × 8000	10,000 × 57,600
Size of image $(N_w \times N_h)$	6000 × 6000	6000 × 6000
Size of image grid (m)	0.1	0.3
Number of azimuth points within synthetic aperture time	4043	10,000
BP algorithm's GPU runtime based on TSU-ICSI (s)	152	223
BP algorithm's GPU runtime based on 16-point sinc interpolation (s)	477	1126
BP algorithm's GPU runtime based on 32-point sinc interpolation (s)	848	2150

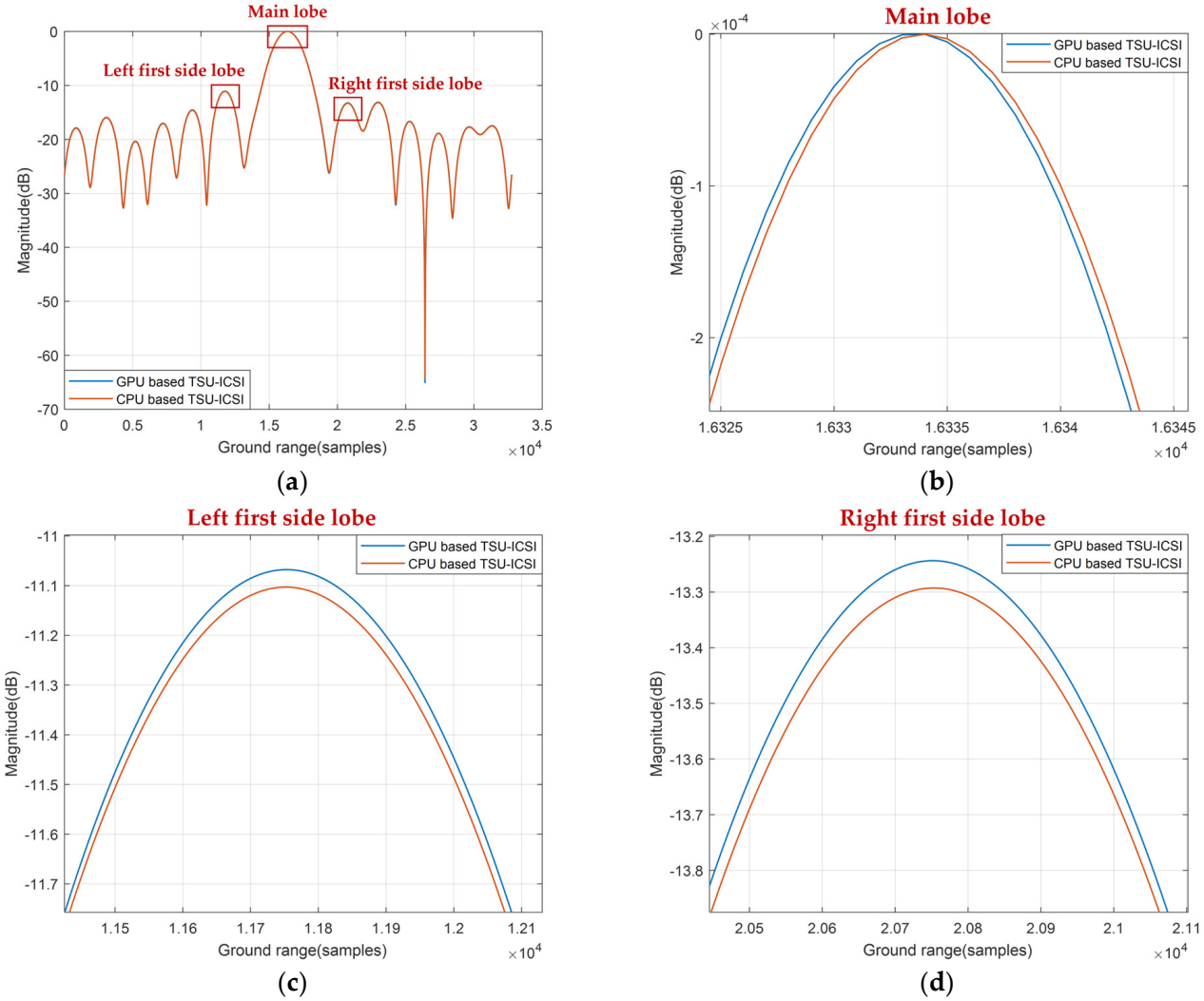


Figure 13. (**a**) Range profiles of strong scattering points in CPU and GPU processing results; (**b**) The main lobe of range profiles; (**c**) The left first side lobe of range profiles; (**d**) The right first side lobe of range profiles.

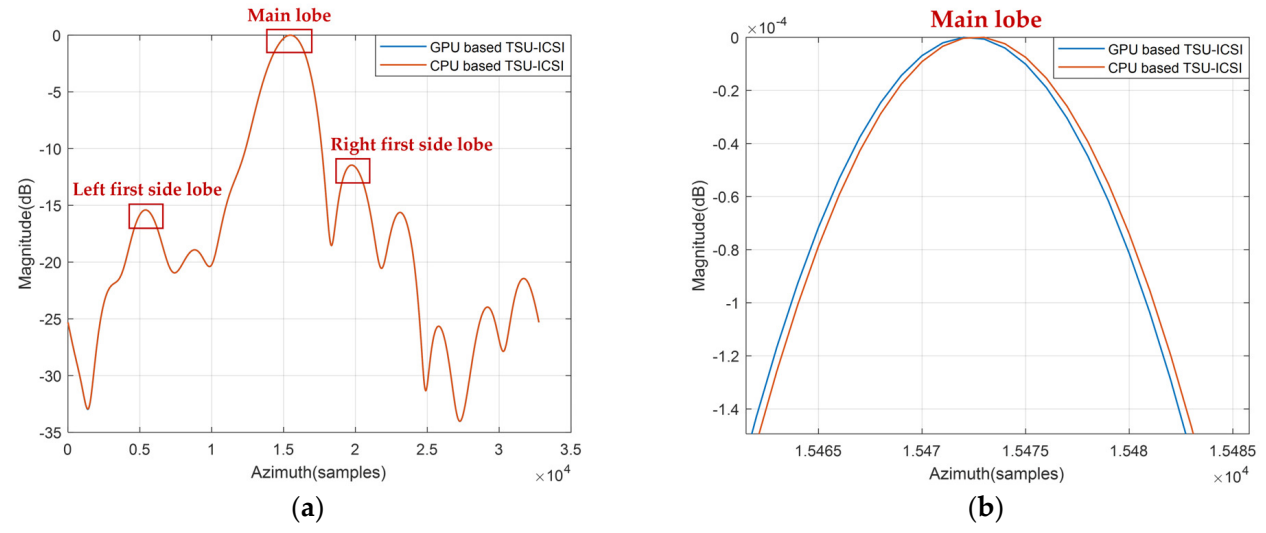


Figure 14. *Cont.*

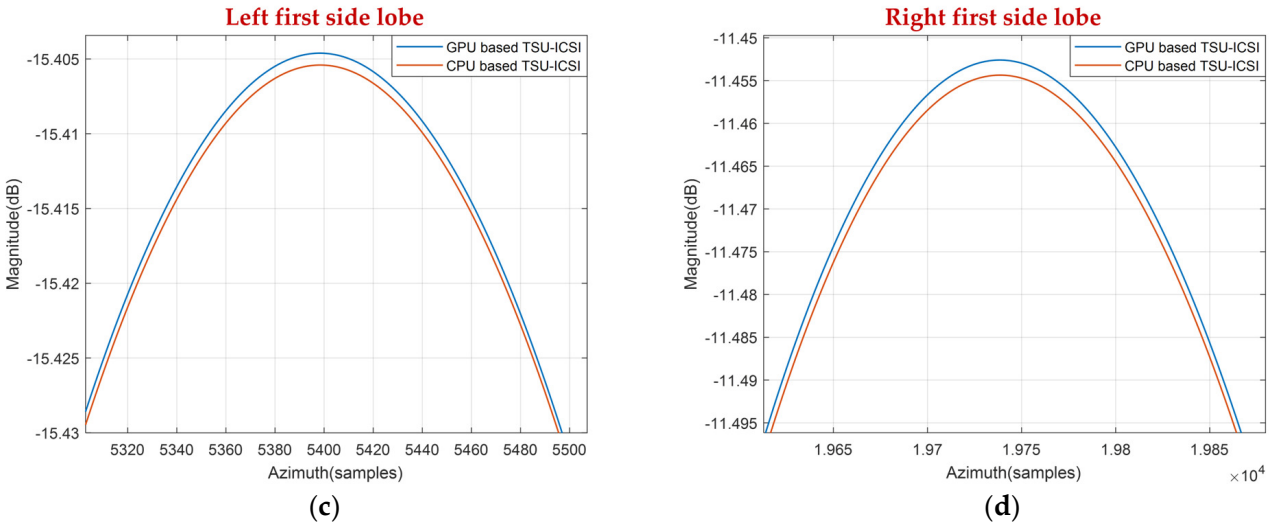


Figure 14. (**a**) Azimuth profiles of strong scattering points in CPU and GPU processing results; (**b**) The main lobe of azimuth profiles; (**c**) The left first side lobe of azimuth profiles; (**d**) The right first side lobe of azimuth profiles.

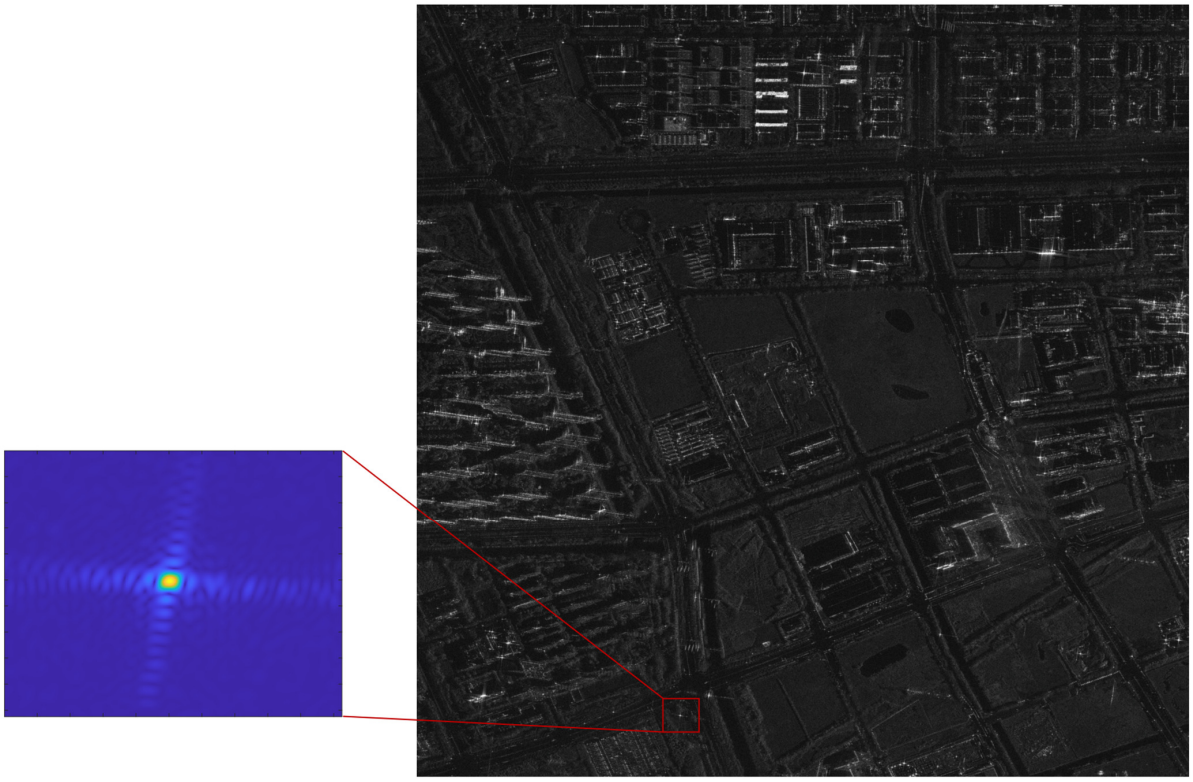


Figure 15. An example of spaceborne data processed by the BP algorithm based on TSU-ICSI implemented using GPU.

Table 6. Ratio of GPU runtime of BP algorithms based on different interpolation methods.

SAR Platform	Airborne	Spaceborne
Number of azimuth points within synthetic aperture time	4043	10,000
BP algorithm's GPU runtime based on 16-point sinc interpolation/BP algorithm's GPU runtime based on TSU-ICSI	3.1382	5.0493
BP algorithm's GPU runtime based on 32-point sinc interpolation/BP algorithm's GPU runtime based on TSU-ICSI	5.5789	9.6413

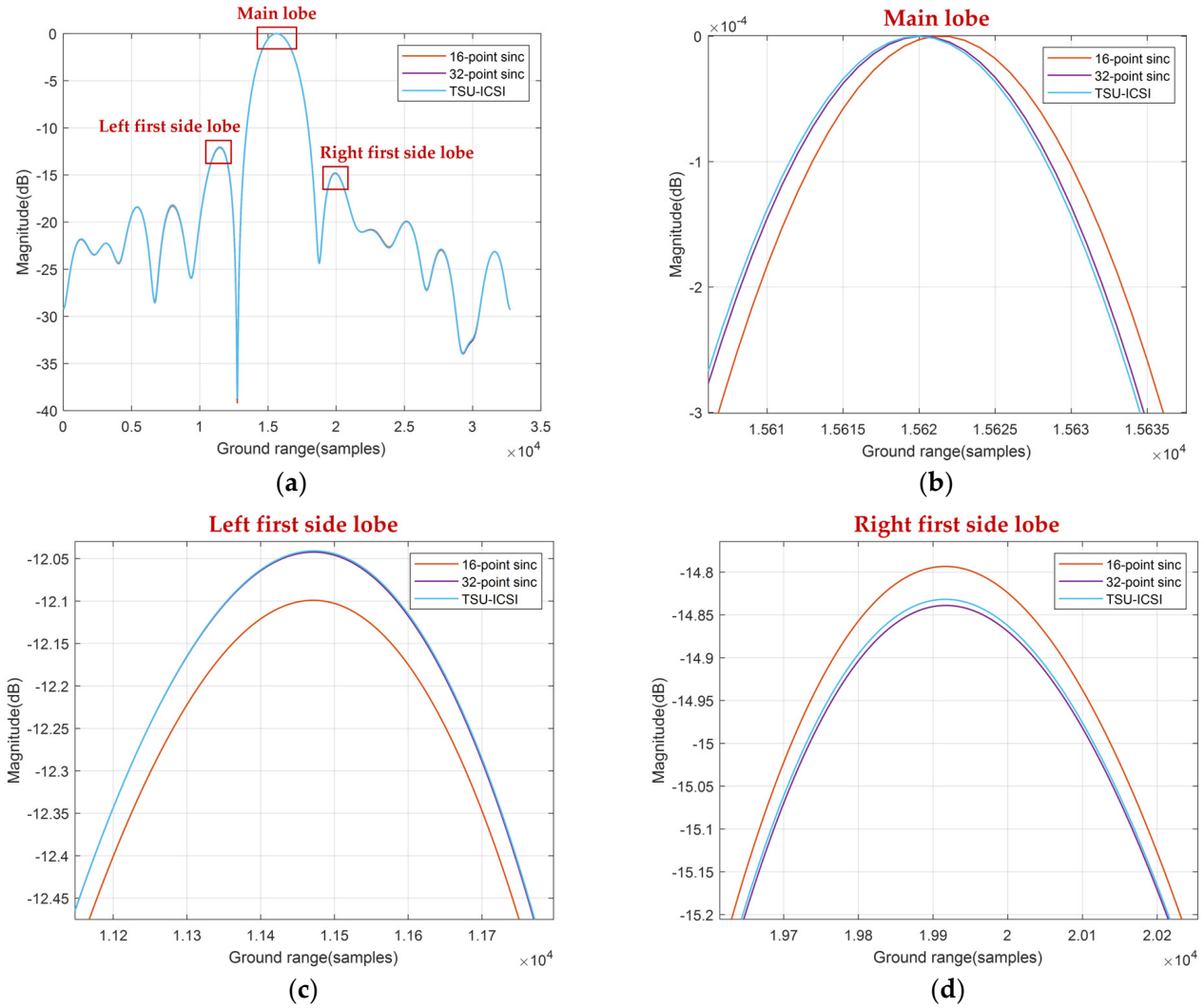


Figure 16. (**a**) Range profiles of strong scattering points in the imaging results of BP algorithms based on different interpolation methods; (**b**) The main lobe of range profiles; (**c**) The left first side lobe of range profiles; (**d**) The right first side lobe of range profiles.

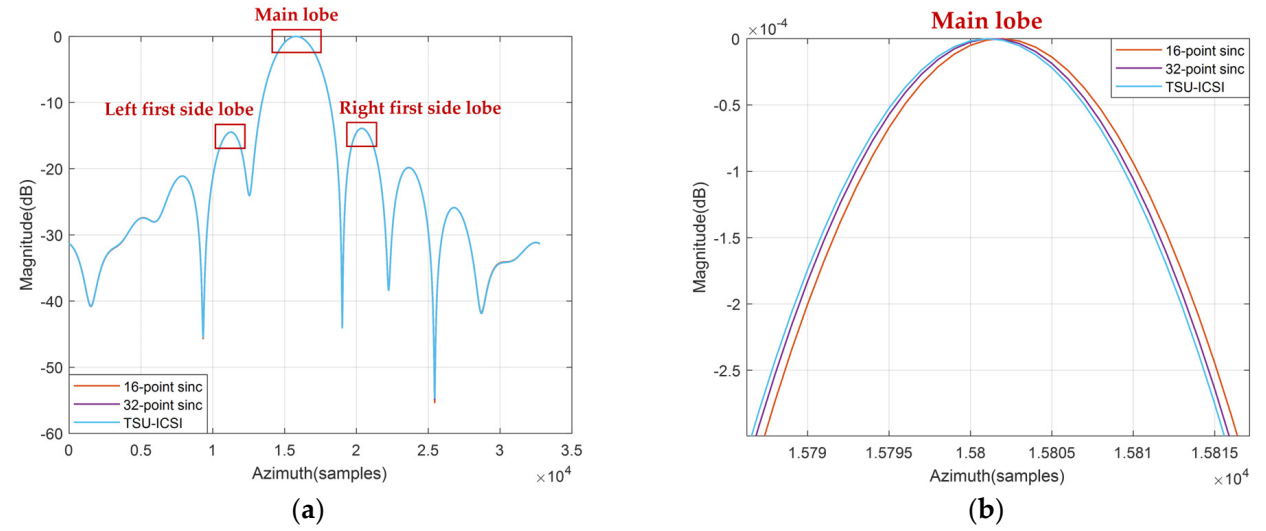


Figure 17. *Cont.*

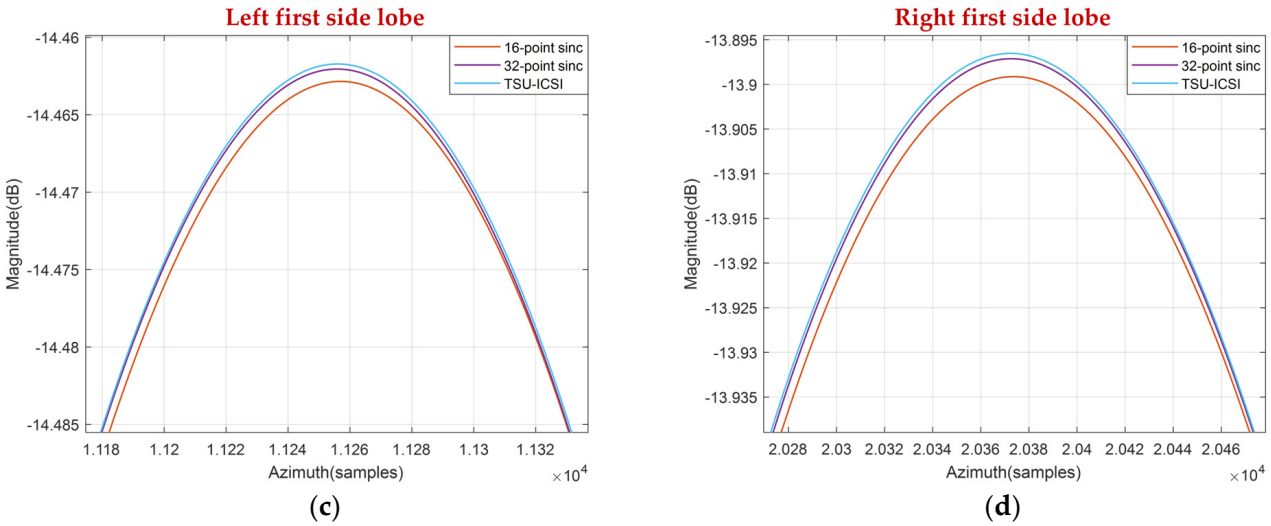


Figure 17. (**a**) Azimuth waveforms of strong scattering points in the imaging results of BP algorithms based on different interpolation methods; (**b**) The main lobe of azimuth profiles; (**c**) The left first side lobe of azimuth profiles; (**d**) The right first side lobe of azimuth profiles.

5. Discussion

The experimental results from both simulations and measured data demonstrate that the 8-point TSU-ICSI method surpasses sinc interpolation in terms of efficiency and precision under upsampling by a factor of 3. When the GPU is utilized for interpolation implementation, the TSU-ICSI method, due to the virtual absence of branch structures like loops and judgments during the interpolation process, allows the GPU to fully exploit the parallel advantage of the CUDA features. This results in improved interpolation efficiency compared to sinc interpolation. Additionally, the increase in the number of azimuth points within the synthetic aperture time leads to a rise in the number of accumulations in the azimuth direction of the BP algorithm, making the advantages of the TSU-ICSI method more evident. These experimental results are consistent with our theoretical analyses regarding the GPU's efficient implementation and the computational complexity of interpolation methods. They provide strong evidence for the superior performance of TSU-ICSI under GPU parallel computing conditions. Looking forward, we aim to further refine and optimize the TSU-ICSI method to ensure higher efficiency and robustness. This includes exploring broader applications of the algorithm in SAR processing, particularly in enhancing SAR algorithms through interpolation. Additionally, we plan to investigate strategies for further enhancing the implementation of GPU parallel computing, adapting to the growing demands for data processing and algorithmic optimization.

6. Conclusions

In this paper, our aim is to refine the interpolation method of the BP algorithm used in synthetic aperture radar, enhancing its precision and efficiency. We confirm the superior performance of our proposed TSU-ICSI method over traditional interpolation methods through one-dimensional and two-dimensional simulation experiments. These illustrate its distinct advantages in both accuracy and efficiency. To further boost processing efficiency, we developed a highly efficient implementation of the TSU-ICSI BP method based on GPU. Validation using both airborne and satellite data revealed that the GPU-based TSU-ICSI BP method significantly enhances processing efficiency. Looking ahead, we believe this innovative technology can unleash substantial benefits in future synthetic aperture radar applications, whether for earth observation or resource exploration.

Author Contributions: Conceptualization, Z.L. and X.Q.; methodology, Z.L., X.Q. and D.M.; software, Z.L. and J.Y.; validation, Z.L., J.Y. and L.H.; formal analysis, Z.L., X.Q. and L.H.; investigation, Z.L.;

resources, X.Q., J.Y., D.M. and L.H.; data curation, X.Q. and J.Y.; writing—original draft preparation, Z.L. and X.Q.; writing—review and editing, Z.L. and X.Q.; visualization, Z.L. and X.Q.; supervision, X.Q., J.Y., D.M., L.H. and S.S.; project administration, X.Q. and J.Y.; funding acquisition, X.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Key R&D Program of China, No. 2018YFA0701903 and NSFC, No. 62022082.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Acknowledgments: The authors would like to thank the staff of Suzhou Aerospace Information Research Institute for their support and providing measured data.

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

References

1. Wiley, C.A. Synthetic Aperture Radars. *IEEE Trans. Aerosp. Electron. Syst.* **1985**, *AES-21*, 440–443. [CrossRef]
2. Raney, R.K.; Runge, H.; Bamler, R.; Cumming, I.G.; Wong, F.H. Precision SAR processing using chirp scaling. *IEEE Trans. Geosci. Remote Sens.* **1994**, *32*, 786–799. [CrossRef]
3. Chen, M.; Qiu, X.; Li, R.; Li, W.; Fu, K. Analysis and Compensation for Systematical Errors in Airborne Microwave Photonic SAR Imaging by 2-D Autofocus. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2023**, *16*, 2221–2236. [CrossRef]
4. Li, R.; Li, W.; Dong, Y.; Wen, Z.; Zhang, H.; Sun, W.; Mo, Z. PFDIR—A Wideband Photonic-Assisted SAR System. *IEEE Trans. Aerosp. Electron. Syst.* **2023**, *59*, 4333–4346. [CrossRef]
5. Deng, Y.; Xing, M.; Sun, G.C.; Liu, W.; Li, R.; Wang, Y. A Processing Framework for Airborne Microwave Photonic SAR with Resolution up to 0.03 m: Motion Estimation and Compensation. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5121017. [CrossRef]
6. Farquharson, G.; Woods, W.; Stringham, C.; Sankarambadi, N.; Riggi, L. The Capella Synthetic Aperture Radar Constellation. In Proceedings of the 2018 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2018), Valencia, Spain, 22–27 July 2018; pp. 1873–1876. [CrossRef]
7. Castelletti, D.; Farquharson, G.; Brown, J.; De, S.; Yague-Martinez, N.; Stringham, C.; Yalla, G.; Villarreal, A. Capella Space VHR SAR Constellation: Advanced Tasking Patterns and Future Capabilities. In Proceedings of the 2022 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2022), Kuala Lumpur, Malaysia, 17–22 July 2022; pp. 4137–4140. [CrossRef]
8. Ferro-Famil, L.; Tebaldini, S.; Davy, M.; Leconte, C.; Boutet, F. Very high-resolution three-dimensional imaging of natural environments using a tomographic ground-based SAR system. In Proceedings of the 8th European Conference on Antennas and Propagation (EuCAP 2014), Hague, The Netherlands, 6–11 April 2014; pp. 3221–3224. [CrossRef]
9. Chen, Z.; Zeng, Z.; Fu, D.; Huang, Y.; Li, Q.; Zhang, X.; Wan, J. Back-Projection Imaging for Synthetic Aperture Radar with Topography Occlusion. *Remote Sens.* **2023**, *15*, 726. [CrossRef]
10. Yegulalp, A.F. Fast backprojection algorithm for synthetic aperture radar. In Proceedings of the 1999 IEEE Radar Conference. Radar into the Next Millennium (Cat. No. 99CH36249), Waltham, MA, USA, 22 April 1999; pp. 60–65. [CrossRef]
11. Ulander, L.M.H.; Hellsten, H.; Stenstrom, G. Synthetic-aperture radar processing using fast factorized back-projection. *IEEE Trans. Aerosp. Electron. Syst.* **2003**, *39*, 760–776. [CrossRef]
12. Dong, Q.; Sun, G.-C.; Yang, Z.; Guo, L.; Xing, M. Cartesian Factorized Backprojection Algorithm for High-Resolution Spotlight SAR Imaging. *IEEE Sens. J.* **2018**, *18*, 1160–1168. [CrossRef]
13. Xu, G.; Zhou, S.; Yang, L.; Deng, S.; Wang, Y.; Xing, M. Efficient Fast Time-Domain Processing Framework for Airborne Bistatic SAR Continuous Imaging Integrated with Data-Driven Motion Compensation. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5208915. [CrossRef]
14. Hu, X.; Xie, H.; Zhang, L.; Hu, J.; He, J.; Yi, S.; Jiang, H.; Xie, K. Fast Factorized Backprojection Algorithm in Orthogonal Elliptical Coordinate System for Ocean Scenes Imaging Using Geosynchronous Spaceborne–Airborne VHF UWB Bistatic SAR. *Remote Sens.* **2023**, *15*, 2215. [CrossRef]
15. Stringham, C.; Long, D.G. GPU Processing for UAS-Based LFM-CW Stripmap SAR. *Photogramm. Eng. Remote Sens.* **2014**, *80*, 1107–1115. [CrossRef]
16. Wijayasiri, A.; Banerjee, T.; Ranka, S.; Sahni, S.; Schmalz, M. Multiobjective Optimization of SAR Reconstruction on Hybrid Multicore Systems. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 4674–4688. [CrossRef]
17. Lee, S.; Ban, I.; Lee, M.; Jung, Y.; Lee, W. Architecture Exploration of a Backprojection Algorithm for Real-Time Video SAR. *Sensors* **2021**, *21*, 8258. [CrossRef] [PubMed]
18. Cao, Y.; Guo, S.; Jiang, S.; Zhou, X.; Wang, X.; Luo, Y.; Yu, Z.; Zhang, Z.; Deng, Y. Parallel Optimisation and Implementation of a Real-Time Back Projection (BP) Algorithm for SAR Based on FPGA. *Sensors* **2022**, *22*, 2292. [CrossRef] [PubMed]
19. Gong, H.; Liu, Y.; Chen, X.; Cheng, W. Scene optimization of GPU-based back-projection algorithm. *J. Supercomput.* **2023**, *79*, 4192–4214. [CrossRef]
20. Capozzoli, A.; Curcio, C.; Liseno, A. Fast GPU-based interpolation for SAR backprojection. *Prog. Electromagn. Res.* **2013**, *133*, 259–283. [CrossRef]

21. Fourmont, K. Non-Equispaced Fast Fourier Transforms with Applications to Tomography. *J. Fourier Anal. Appl.* **2003**, *9*, 431–450. [CrossRef]
22. Breglia, A.; Capozzoli, A.; Curcio, C.; Liseno, A. NUFFT-Based Interpolation in Backprojection Algorithms. *IEEE Geosci. Remote Sens. Lett.* **2021**, *18*, 2117–2121. [CrossRef]
23. Lin, S.C.; Chuang, K.; Chen, J.H. Efficient implementation of cubic spline interpolator. In Proceedings of the 2020 IEEE Radio and Wireless Symposium, San Antonio, TX, USA, 30 March 2020; pp. 287–290. [CrossRef]
24. Lin, S.C.; Chuang, K.; Chang, C.W.; Chen, J.H. Efficient interpolation method for wireless communications and signal processing applications. *IEEE Trans. Microw. Theory Tech.* **2021**, *69*, 2753–2761. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.