Postprint of FPGA-Based Frequency Domain Lucky Imaging Algorithm

Authors: Xueming Huang, Binhua Li, Wang Jinliang

Date: 2021-07-30T00:00:00+00:00

Abstract

Lucky imaging is a post-processing image restoration technique that can effectively mitigate the effects of atmospheric turbulence on image quality. Traditional frequency-domain lucky imaging algorithms outperform traditional spatial-domain algorithms and primarily consist of three stages: image preprocessing, data selection, and data stacking. This paper proposes a frequency-domain lucky imaging algorithm based on Field-Programmable Gate Array (FPGA) and implements an experimental frequency-domain lucky imaging system on an FPGA platform. The system performs two-dimensional Fast Fourier Transform (2D-FFT) on astronomical image data by decomposing the two-dimensional operation into two one-dimensional implementations, employs the sum of squares of the real and imaginary parts of frequency-domain data rather than amplitude for data selection, and produces high-resolution images essentially identical to those obtained from Central Processing Unit (CPU)-based algorithmic processing. This experimental system can process 2,000 frames of 128 128 pixel images using frequency-domain lucky imaging, achieving a speed more than 13 times faster than traditional CPU-based frequency-domain lucky imaging algorithms, thereby accelerating the conventional frequency-domain lucky imaging algorithm and providing a feasible technical solution for future real-time implementation of frequency-domain lucky imaging.

Full Text

Preamble

FPGA-based Lucky Imaging Algorithm in Frequency Domain

Huang Xueming[1], Li Binhua[1],[2], Wang Jinliang[1]
(1. Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China;

2. Key Laboratory of Applications of Computer Technologies of the Yunnan Province, Kunming University of Science and Technology, Kunming 650500, China)

Abstract: Lucky imaging is a post-processing image restoration technique that effectively mitigates the impact of atmospheric turbulence on image quality. Traditional frequency-domain lucky imaging algorithms outperform their spatial-domain counterparts and consist primarily of three components: image preprocessing, data selection, and data superposition. This paper proposes a frequency-domain lucky imaging algorithm based on Field-Programmable Gate Array (FPGA) and constructs an experimental system for frequency-domain lucky imaging on an FPGA platform. The system performs two-dimensional Fast Fourier Transform (2D-FFT) on astronomical image data by decomposing it into two one-dimensional operations, and employs the sum of squares of the real and imaginary parts of frequency-domain data instead of amplitude for data selection. The resulting high-resolution images are essentially identical to those processed by CPU-based algorithms. The experimental system can process 2000 frames of 128$×$128 pixel images using frequency-domain lucky imaging, achieving more than 13 times the speed of traditional CPU-based frequency-domain lucky imaging algorithms. This implementation accelerates the conventional frequency-domain lucky imaging algorithm and provides a feasible technical solution for future real-time frequency-domain lucky imaging applications.

Keywords: Lucky imaging; Atmospheric turbulence; 2D-FFT; Frequency domain; FPGA

Atmospheric turbulence is an irregular, random motion in the atmosphere that causes the actual resolution of telescopes to fall far below their diffraction-limited resolution, representing a major factor limiting the spatial resolution of ground-based optical telescopes [1]. To obtain images approaching the diffraction limit of telescopes from ground observations, post-processing image restoration or reconstruction techniques can be employed, with lucky imaging being one such technique for mitigating atmospheric turbulence effects [2,3]. Lucky imaging can be performed in either the spatial or frequency domain. However, spatial-domain algorithms select entire image frames, which fails to consider high-frequency component information in certain directions, leading to wasted image data information during selection and processing. Consequently, in 2012, Garrel et al. proposed a Fourier transform-based lucky imaging algorithm comprising image preprocessing, data selection, and data superposition, and conducted simulation experiments to validate the method. Experimental results demonstrated optimal imaging performance when selecting 10% of the data [4]. In 2013, Mackay further confirmed the effectiveness of frequency-domain lucky imaging using real observational data, while also noting the algorithm' s computational complexity disadvantage compared to spatial-domain methods [3]. In 2019, Hu Xing et al. from our laboratory improved the traditional frequency-domain lucky imaging algorithm by implementing grouped data selection in the frequency domain to accelerate processing, though this approach remained in

the post-processing category [5].

All the aforementioned frequency-domain lucky imaging algorithms are CPU-based, and their serial processing nature makes real-time processing difficult, preventing observers from accurately understanding celestial object information during astronomical observations. Therefore, real-time processing represents an inevitable development trend for lucky imaging technology. In recent years, two major parallel hardware accelerators have been introduced to the image processing field [6-7]: Graphics Processing Units (GPUs) and FPGAs. While both offer powerful parallel processing capabilities, FPGAs are semi-custom circuits that can be programmed according to actual requirements, whereas GPUs are fixed after design, making them far less flexible than FPGAs [8]. Consequently, this paper selects FPGA to implement acceleration of the frequency-domain lucky imaging algorithm.

Leveraging the powerful parallelism and flexibility of FPGA, this paper proposes a frequency-domain lucky imaging algorithm based on FPGA. Following FPGA parallel and pipeline design methodologies, the algorithm was implemented using Verilog Hardware Description Language (HDL) and realized on a Xilinx Spartan-7 platform with relevant testing experiments. This paper is organized into four sections: Section 1 describes the algorithm, Section 2 introduces the system design process on the Spartan-7 FPGA, Section 3 presents experimental results and discussion, and Section 4 provides conclusions.

1 Frequency-Domain Lucky Imaging Algorithm Suitable for FPGA

Traditional frequency-domain lucky imaging algorithms consist of three main components: image preprocessing, frequency-domain data selection, and data superposition. Data selection in the frequency domain consumes substantial clock resources, resulting in very slow algorithm execution. Only Hu Xing has addressed this issue, but still within the post-processing framework. This section proposes a new frequency-domain lucky imaging algorithm accelerated on FPGA, with its effectiveness verified on a desktop computer.

1.1 Overall Algorithm Design

The FPGA-based frequency-domain lucky imaging algorithm proposed in this paper differs from both the traditional frequency-domain lucky imaging algorithms in references [3] and [4] and the improved frequency-domain lucky imaging algorithm in reference [5]. In reference [4], Carrel et al. proposed selecting image data based on amplitude in the frequency domain. In reference [3], Mackay confirmed the effectiveness of frequency-domain lucky imaging while highlighting its significant computational complexity. In reference [5], Hu Xing accelerated the frequency-domain lucky imaging algorithm using a grouping method, but the processing speed remained very slow. Therefore, implementing

frequency-domain lucky imaging on FPGA represents an effective approach to increasing processing speed.

Three technical challenges exist for frequency-domain lucky imaging on FPGA:

First, direct 2D FFT of image data cannot be performed on FPGA. While 2D Fourier transforms can be executed directly on image data in MATLAB, only 1D Fourier transforms are possible on FPGA. According to the principle of 2D Fourier transforms, for an N×M spatial-domain image f(x,y), its frequency-domain representation F(u,v) is a complex number defined by equation (1):

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})} \tag{1}$$

Decomposing equation (1) yields equation (2):

$$F(u,v) = \sum_{x=0}^{M-1} e^{-j2\pi \frac{ux}{M}} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi \frac{vy}{N}} \tag{2}$$

where x and y represent spatial image row and column indices ranging from x=0,1,⋯,M-1 and y=0,1,⋯,N-1; u and v represent frequency-domain row and column indices ranging from u=0,1,⋯,M-1 and v=0,1,⋯,N-1.

As shown in equations (1) and (2), 2D Fourier transforms can be implemented using a two-stage 1D approach, decomposing 2D-FFT into row-wise FFT and column-wise FFT performed sequentially. Generally, the order of 1D FFT operations does not affect the 2D-FFT result. Since this system processes 128$×$128 images, both N and M equal 128.

Second, traditional frequency-domain lucky imaging algorithms select data based on amplitude, calculated using equation (3):

$$M(u,v) = \sqrt{Re[F(u,v)]^2 + Im[F(u,v)]^2} \tag{3}$$

where Re[F(u,v)] and Im[F(u,v)] represent the real and imaginary parts of frequency-domain data, and M(u,v) is the magnitude.

While integer addition, subtraction, multiplication, and division can be conveniently performed on FPGA, computing complex amplitude is algorithmically complex and time-consuming unless using a CORDIC IP core, which consumes substantial hardware resources and is unsuitable for implementation on small-to-medium-scale FPGAs. Therefore, this paper proposes using the sum of squares of the real and imaginary parts instead of amplitude for data selection, as this sum follows the same variation pattern as amplitude. Equation (4) replaces equation (3) for data selection:

$$M(u,v) = Re[F(u,v)]^2 + Im[F(u,v)]^2 \tag{4}$$

Third, on-chip FPGA memory resources are limited. Since this system must store substantial frequency-domain image data, and on-chip RAM resources are insufficient, external high-capacity FPGA memory is used for storing frequency-domain image data.

The workflow of the proposed FPGA-suitable frequency-domain lucky imaging algorithm is as follows:

1. Image input with Gaussian filtering;
2. Row-wise Fast Fourier Transform of filtered images;
3. Transposition of row FFT results in RAM using ping-pong operations;
4. Reading data from RAM for column-wise Fast Fourier Transform, with results stored in DDR3;
5. Reading data from DDR3 and selecting 10% of data;
6. Data superposition;
7. Row-wise inverse Fast Fourier Transform of superposed data;
8. Transposition of row IFFT results in RAM;
9. Reading data from RAM for column-wise inverse Fast Fourier Transform;
10. Splitting column IFFT results into two paths: one for binarization and HDMI display, the other for transmission back to the host computer.

In this workflow, steps (1) through (4) are processed in parallel due to FPGA' s powerful parallel processing capabilities, as are steps (5) through (8). Image data is input via Reduced Gigabit Medium Independent Interface (RGMII) in step (1). Steps (2) through (4) constitute the forward Fourier transform process (spatial to frequency domain), while steps (7) through (9) constitute the inverse Fourier transform process (frequency to spatial domain).

1.2 Algorithm Verification

To verify the feasibility of the proposed frequency-domain lucky imaging algorithm and demonstrate its superior high-resolution image reconstruction compared to spatial-domain lucky imaging, two sets of experiments were conducted on a CPU+MATLAB platform. The first set reproduced the traditional frequency-domain algorithm based on amplitude selection, while the second set simulated the proposed algorithm' s FPGA implementation on MATLAB and compared results with reference [10]. The hardware platform was a Dell Precision T5500 graphics workstation with 16GB memory, Xeon E5620 CPU, and NVIDIA GTX1080 GPU, running Windows 7 and MATLAB R2017a.

The frequency-domain lucky imaging algorithm was applied to 2000 randomly selected 128$×$128 pixel images in MATLAB. Based on research by Garrel et al. and Hu Xing et al., a 10% data selection ratio yields optimal frequency-domain lucky imaging performance. The first experiment reproduced Garrel

et al.' s traditional frequency-domain algorithm, whose correctness has been verified multiple times and will not be detailed here. The second experiment validated the proposed algorithm' s correctness. Results from both experiments are shown in [Figure 1: see original paper] and [Figure 2: see original paper].

[Figure 1: see original paper] shows the traditional algorithm results: (a) 3D grayscale distribution map, (b) 2D grayscale image, and (c) peak data. [Figure 2: see original paper] shows the proposed algorithm results in the same format. Visually, no errors are apparent. Data analysis from [FIGURE:1(c)] and [FIGURE:2(c)] reveals that since the second experiment simulates FPGA implementation, all operations process integers. The results show that the proposed FPGA-suitable frequency-domain algorithm differs from the traditional algorithm only in fractional parts—that is, truncation errors caused by FPGA' s integer-only processing.

To further analyze the quality of high-resolution images reconstructed by the two frequency-domain algorithms, three objective image quality evaluation functions were employed for numerical comparison. FWHM is a common metric for astronomical images where smaller values indicate better quality, while the other two are classic digital image sharpness metrics where larger values indicate better quality. presents the image quality evaluation metrics for high-resolution results from three algorithms: the two frequency-domain lucky imaging algorithms and the spatial-domain lucky imaging algorithm from reference [10].

shows that compared to the traditional frequency-domain algorithm, the proposed algorithm yields identical FWHM values, with slightly lower Tenengrad gradient and Laplacian gradient values due to truncation errors during FPGA operations. This indicates marginally inferior image quality compared to the traditional algorithm. However, comparison with reference [10]' s spatial-domain data demonstrates that the proposed frequency-domain algorithm produces significantly better image quality (sharpness) than spatial-domain algorithms, confirming its feasibility and effectiveness.

Since the proposed algorithm is feasible and effective, proper FPGA design can reduce processing time, accelerate frequency-domain lucky imaging on FPGA, and provide a viable technical solution for real-time implementation. Therefore, following algorithm design, FPGA implementation of the frequency-domain lucky imaging system becomes the next critical issue.

2 FPGA Design and System Implementation of Frequency-Domain Lucky Imaging Algorithm

Algorithm design and implementation are typically hardware-platform dependent. The hardware implementation platform for this frequency-domain lucky imaging algorithm consists of a development board centered on Xilinx's Spartan-7 series XC7S50 chip and a workstation with Xeon E5620 CPU. The design environment is Vivado 2019.1, with verification and debugging performed using Vivado' s embedded logic analyzer and ModelSim SE 10.7. The proposed

algorithm was logically designed in Verilog HDL and implemented on FPGA hardware. This section focuses on the overall system design and the FPGA implementation of the FFT/IFFT module and data selection module, with details presented in three subsections.

2.1 Overall System Design of Frequency-Domain Lucky Imaging

The system comprises three main components: data transmission, data computation, and data storage. For future real-time applications requiring high data transmission speeds, Gigabit Ethernet is employed. For data computation, the system involves 2D-FFT, which cannot be performed directly on FPGA, so a two-stage 1D implementation is used for astronomical image data. For data storage, Double-Data-Rate Three Synchronous Dynamic Random Access Memory (DDR3) is utilized because processing 2000 frames of 128$×$128 pixel images exceeds FPGA on-chip RAM resources, necessitating external DDR3 storage. The system design was implemented in Verilog HDL, with the overall structure shown in [Figure 3: see original paper].

[Figure 3: see original paper] illustrates the overall system structure. Due to limitations in on-chip RAM resources on the FPGA hardware platform, the system currently processes 2000 frames of 128$×128$pixelshort - exposureimagesasrawdata, ultimatelydisplayingfrequency—domainluckyimagingresultsat128×$128 pixel resolution.

2.2 FFT/IFFT Algorithm and Module Design

Traditional spatial-domain lucky imaging algorithms perform image selection and superposition in the spatial domain, whereas frequency-domain lucky imaging algorithms perform data selection and superposition at each frequency point of the Fourier image. Therefore, implementing frequency-domain lucky imaging requires Fourier transformation of image data followed by selection and superposition in the frequency domain. Xilinx provides a 1D FFT IP core on the Vivado platform, but since image data is two-dimensional, the 1D FFT IP cannot be applied directly to spatial image data. Consequently, this module employs a two-stage 1D implementation for 2D Fourier transforms. The FFT/IFFT module design structure is shown in [Figure 4: see original paper].

[Figure 4: see original paper] shows the 2D-FFT module structure. The workflow is described as follows:

1. Preprocessed data undergoes FFT row transformation. The row transformation results are transposed in RAM1 and RAM2 using ping-pong operations. Two simple dual-port RAMs with depth 16384 and width 48 bits are used (original image data is 16-bit; after one forward Fourier transform, real and imaginary parts each become 24-bit, hence 48-bit width for transposition). When RAM1 fills with one frame of image data, it switches to read mode for column FFT while RAM2 switches to write mode. When RAM2 fills, it switches to read mode for column FFT while RAM1 switches

to write mode. The final row-column transformed frequency-domain data is stored in DDR3 for selection and superposition.

2. Selected and superposed data first undergoes row IFFT. The inverse transform results are stored in RAM using a simple dual-port RAM with depth 16384 and width 64 bits for transposition. After RAM stores the row IFFT results, column IFFT begins, yielding the superposed spatial-domain image data. This data is split into two paths: one for averaging and high-resolution image display, the other for transmission back to the host computer via the Ethernet module.

2.3 Data Selection Algorithm and Module Design

Traditional data selection methods sort amplitudes of all frequency points across all images to establish index relationships and select the top 10% frequency point data. This approach is computationally intensive and consumes substantial hardware resources, exceeding even high-end Xilinx Vertex-7 series FPGA capabilities. Therefore, this paper employs: (1) data selection without sorting, and (2) sum of squares of real and imaginary parts instead of amplitude for selection. The module design structure is shown in [Figure 5: see original paper].

[Figure 5: see original paper] illustrates the data selection module structure. The workflow is as follows:

First, frequency-domain image data is read from DDR3. After reading the first data point of frame 1 into RAM via FIFO, the corresponding data point of frame 2 is read and stored similarly, continuing until frame 200. From frame 201 onward, corresponding data points are first stored in FIFO. Before storing in RAM, the sum of squares of real and imaginary parts is compared with the minimum value among the previous 200 corresponding points to find the smallest value and its address. The new data' s sum of squares is compared with this minimum value: if larger, it replaces the data at the minimum value' s address; if smaller, it is discarded. This process repeats from frame 202 to frame 2000. After frame 2000' s corresponding data is processed, the 200 data points in RAM are sent to the superposition module. Second, this process repeats for each image' s second through final data points.

3 Experimental Results and Comparative Analysis

To verify the correctness of the proposed data selection method on FPGA, an embedded logic analyzer was used to capture data from the data selection module. To validate the frequency-domain lucky imaging algorithm' s implementation on the FPGA hardware platform, the same algorithm was simulated on a CPU+MATLAB platform, with results compared between the two platforms.

The experimental data consists of 10,000 frames of observational images of the astronomical binary star HDS 70 captured on October 20, 2016, at the Lijiang

Observatory Station of Yunnan Astronomical Observatories. Observation conditions and parameters are described in reference [9]. Due to FPGA hardware resource limitations, 2000 frames were randomly selected from the 10,000 frames and cropped to 128$×$128 pixel short-exposure images for processing.

3.1.1 Data Selection Algorithm Results

This system employs a data selection method that chooses 200 frequency point data from each corresponding point across 2000 frames (10% selection ratio). To make the embedded logic analyzer data more intuitive, partial data from the first 10 rows was selected at 10% ratio (1 row selected from 10 rows). The data selection results are shown in [Figure 6: see original paper].

[Figure 6: see original paper] shows the data selection results: (a) partial data from the first 10 rows, and (b) selected data. The figures demonstrate that the data with maximum sum of squares of real and imaginary parts (correspondingly maximum amplitude) from each row is selected, confirming the correctness of the embedded logic analyzer capture and validating the proposed data selection algorithm.

3.1.2 Frequency-Domain Lucky Imaging Results on FPGA and MATLAB

The hardware platform for this system is a development board centered on Xilinx' s Spartan-7 series XC7S50 chip. The FPGA-based frequency-domain lucky imaging algorithm verification uses a 10% data selection ratio. Since high-resolution images from MATLAB and FPGA are difficult to compare directly, both platforms applied identical sharpening to highlight targets or regions of interest, yielding the sharpened images shown in Figure 7: see original paper and 7(b).

Figure 7: see original paper shows the sharpened result of image data transmitted back to the host computer via Ethernet and displayed using MATLAB functions, while Figure 7: see original paper shows the result processed by frequency-domain lucky imaging algorithm on MATLAB with identical sharpening. The two images are completely identical, verifying the correctness of the FPGA implementation. However, since FPGA processes only integers, truncation errors occur during FFT. These errors are very small, typically within one count. For more intuitive comparison, before averaging the reconstructed high-resolution image data (i.e., after superposition of 200 frames), random segments of FPGA-processed data (captured by the embedded logic analyzer) were compared with MATLAB-processed data. Most data errors were within 200, meaning after averaging the errors were essentially within 1, indicating minimal impact of truncation errors on image quality. The random data segments are shown in Figure 8: see original paper and 8(b).

3.2 Clock Consumption Comparison

The proposed FPGA-based frequency-domain lucky imaging algorithm and Zhao et al.' s FPGA-based lucky imaging algorithm (spatial domain) from reference [10] were compared for processing 1000 frames of raw images, evaluating algorithm runtime, image data reading time, and total runtime. Results are presented in .

shows that despite frequency-domain lucky imaging processing substantially more data and having much higher algorithmic complexity than spatial-domain lucky imaging, the proposed frequency-domain algorithm on FPGA processes 0.51 seconds faster than Zhao' s spatial-domain system, demonstrating full utilization of FPGA' s parallel and flexible advantages. This system' s pixel reading speed is 8.4 times faster than Zhao' s spatial-domain system due to Gigabit Ethernet replacing SD card transmission. The overall average processing speed is 6.4 times faster, confirming the rationality and correctness of the FPGA implementation.

4 Conclusion

Following careful analysis of Garrel et al.' s Fourier transform-based lucky imaging algorithm and Hu Xing et al.' s improved frequency-domain lucky imaging algorithm, this paper proposes an FPGA-based frequency-domain lucky imaging algorithm and achieves acceleration on FPGA. First, the algorithm was overviewed and validated as feasible and effective on MATLAB, demonstrating superior imaging performance compared to spatial-domain algorithms. Second, the overall design and implementation approaches for the 2D-FFT module, data selection module, and data superposition module were introduced. Finally, the system was implemented on a small-scale, low-cost FPGA development board, with test results and verification processes presented for relevant modules. Since MATLAB code efficiency is relatively low and unsuitable for astronomical observations but appropriate for algorithm verification and experimental analysis, the FPGA platform results were compared with MATLAB platform results to prove the system' s correctness on FPGA. Although the design has limitations and still falls short of dynamic real-time implementation, this work demonstrates specific FPGA implementation and acceleration of frequency-domain lucky imaging algorithms, providing a feasible technical solution for real-time frequency-domain lucky imaging.

Acknowledgments

The authors thank Zhang Xiliang of Yunnan Astronomical Observatories, Chinese Academy of Sciences, for providing the short-exposure raw image data of binary star HDS 70 used in this experiment. This research was supported by the National Natural Science Foundation of China (Project No. 11673009).

References

[1] OSCOZ A, REBOLO R, LOPEZ R, et al. FastCam: a new lucky imaging instrument for medium-sized telescopes [D]// Proceeding of SPIE, 2008.
[2] BALDWIN J E, MACKAY C D, LAW N M. Lucky imaging: high angular resolution imaging in the visible from the ground [J]. Astronomy and astrophysics, 2006, 446(2): 739-745.
[3] MACKAY C D. High-efficiency lucky imaging [J]. Monthly Notices of the Royal Astronomical Society, 2013, 432(1): 702-710.
[4] GARREL V, GUYON O, BAUDOZ P. A Highly Efficient Lucky Imaging Algorithm: Image Synthesis Based on Fourier Amplitude Selection [J]. Publications of the Astronomical Society of the Pacific, 2012, 124(918): 861-867.
[5] HU X. Research on frequency-domain lucky imaging algorithm[D], Master' s degree thesis of Kunming University of Science and Technology, 2019, 3.
[6] GUO C X, CHEN H, SUN H, et al. Parallelism of in-memory sorting algorithm on modern hardware [J]. Chinese Journal of Computers, 2017, 40(9): 2070-2092.
[7] HEGARTY J, DALY R, DEVITO Z, et al. Rigel: flexible multi-rate image processing hardware [J]. ACM Transactions on Graphics, 2016, 35(04): 85.
[8] STEWART R, DUNCAN K, MICHAELSON G, et al. RIPL: a parallel image processing language for FPGAs [J]. ACM Transactions on Reconfigurable Technology and Systems, 2018, 11(1): 7.
[9] MAO L H, LI B H, ZHANG X L, et al. Experimental investigation of lucky imaging algorithm based on 2 m astronomical telescope [J]. Optical Technique, 2018, 44(5): 542-548.
[10] ZHAO P Z, LI B H, MAO L H, et al. Implementation of lucky imaging algorithm based on FPGA [J]. Astronomical Research & Technology, 2019, 16(2): 236-243.

Note: Figure translations are in progress. See original paper for figures.