GPU-Based Image Frame Selection for ONSET Real-Time Data Processing (Postprint)

Authors: Li Li, Deng Hui, Li Zhen, Mei Ying, Dai Wei, Yang Qiuping, Wang Feng

Date: 2018-05-29T00:00:00+00:00

Abstract

The Optical and Near-infrared Solar Eruption Monitoring Telescope acquires a large amount of solar image data daily. Real-time frame selection processing of these observational data can, on the one hand, reduce storage pressure, and on the other hand, improve the quality of subsequent image reconstruction. To meet the frame selection requirements during real-time observation, a real-time image frame selection processing module based on Graphics Processing Units (GPUs) was designed and implemented. The current module has already achieved high-speed parallel processing of two frame selection algorithms: the average gradient method and the spectral ratio method. The module implementation is discussed in detail, and the speedup ratios of the two frame selection methods are compared. Experiments show that the module operates stably and reliably. In terms of execution efficiency, for near full-disk image frame selection, the total execution time is as fast as 1.2s, representing a 7× improvement over the original serial implementation; for partial-disk images, it is as fast as 0.7s, representing an average 5× improvement. The overall module implementation and current performance can already meet the requirements for real-time observation and processing.

Full Text

GPU-Based Implementation of Image Frame Selection for ONSET Real-Time Data Processing

Li Li[1], Deng Hui[1]*, Li Zhen[3], Mei Ying[2], Dai Wei[12], Yang Qiuping[12], Wang Feng[12]

[1]Yunnan Key Laboratory of Computer Technology Application, Kunming University of Science and Technology, Kunming 650500, China
[2]Yunnan Observatories, Chinese Academy of Sciences, Kunming 650011, China

[3]School of Astronomy and Space Science, Nanjing University, Nanjing 210000, China

Abstract

The Optical and Near-infrared Solar Eruption Telescope (ONSET) acquires massive amounts of solar image data daily. Real-time frame selection of these observations can both alleviate storage pressure and improve the quality of subsequent image reconstruction. To meet the frame selection requirements during real-time observations, we have designed and implemented a GPU-based real-time image frame selection processing module that currently achieves high-speed parallel processing for both the average gradient method and spectral ratio method. This paper discusses the module implementation in detail and compares the acceleration ratios of the two frame selection approaches. Experimental results demonstrate that the module operates stably and reliably. In terms of execution efficiency, the fastest total execution time for near full-disk image frame selection is 1.2 seconds, representing a 7× improvement over the original serial implementation; for partial-disk images, the fastest time is 0.7 seconds with an average 5× speedup. The overall module implementation and current performance already satisfy the requirements for real-time observation and processing.

Keywords: Real-time frame selection; Optical and Near-infrared Solar Eruption Telescope (ONSET); Graphics Processing Unit (GPU)

1. GPU Parallel Technology

Graphics Processing Units (GPUs) were initially designed to provide basic operations for Central Processing Units (CPUs). However, with their evolution, GPUs have become high-performance processors with massive parallelism. Compute Unified Device Architecture (CUDA) is a programming interface introduced by NVIDIA to enhance the general-purpose computing capabilities of GPUs while maintaining ease of use. Programmers can write programs in C language under the CUDA architecture to run on NVIDIA GPUs. The CUDA Toolkit includes two important libraries: CUFFT, an optimized Fast Fourier Transform library, and CUBLAS, a Linear Algebra library containing Basic Linear Algebra Subprograms (BLAS).

In CPU-GPU data interactions, data is typically first copied from host memory to device memory, after which GPU threads perform predetermined operations on the data and return the final results to the host. By appropriately allocating grids, thread blocks, and threads per block in the GPU device, and employing numerous threads to maximize parallelism, we can leverage the GPU' s powerful computational throughput to achieve high-performance frame selection algorithm processing. Previous research has investigated GPU implementations of solar high-resolution image reconstruction algorithms, with one study implementing a GPU-parallelized Level-1 reconstruction program that accelerated

reconstruction speed by nearly 30×, achieving quasi-real-time performance. Another work developed a real-time cloud contamination detection and restoration system for H full-disk solar images under CUDA. These examples demonstrate that GPU parallel technology has found numerous applications in astronomical software development, providing valuable insights for developing real-time frame selection capabilities for ONSET observation data. However, given the specific data characteristics and frame selection requirements of ONSET, further research is needed to develop parallel implementation techniques suitable for its frame selection algorithms to meet real-time demands.

2. Design of Real-Time Frame Selection Processing Module

ONSET enables multi-channel, multi-wavelength observations with different data types and image characteristics, potentially requiring different frame selection algorithms for different wavelengths. To satisfy real-time frame selection requirements during observations, we designed and implemented a dedicated processing module.

During ONSET observations, the frame selection process typically involves: acquiring a set of candidate images, computing quality evaluation metrics for each frame using image quality assessment methods, selecting frames based on predetermined selection ratios according to these metrics, and finally storing selected and rejected frames in separate directories to complete the process.

[Figure 1: see original paper]

Considering the heterogeneous nature of CPU-GPU architectures, the parallel implementation flowchart for our frame selection module is shown in Figure 1. The implementation comprises the following functional steps: (1) reading a set of ONSET speckle images (e.g., 100 frames) into host memory; (2) allocating GPU device memory and copying the image data; (3) calculating the required number of threads for kernel execution based on each frame's dimensions; (4) executing the specific frame selection algorithm logic on the GPU (our experiments implement both average gradient and spectral ratio methods); (5) performing reduction summation of GPU computation results and copying them back to host memory; (6) performing quicksort on the host based on the computed evaluation values for each frame; and (7) separating good frames (e.g., top 70% by evaluation value) from bad frames into different directories based on the sorting results.

The illustrative code for the real-time frame selection processing module is shown in Table 1. For future ONSET development, we have designed the module with strong openness and extensibility to accommodate evolving scientific objectives, observation terminals, and varying image characteristics. In the current implementation, GPU thread allocation is primarily based on the dimensions of each frame to be processed, enabling each pixel to be assigned to a thread. Each thread handles the computation for one grayscale value, resulting in parallel processing of all pixels across each frame. The frame selection algorithm logic

executed on the GPU represents the core of the process, where algorithms can be flexibly selected and integrated into future ONSET data processing systems based on the characteristics of the frames and selection requirements.

Overall, the module implementation optimizes frame selection execution efficiency with high parallelism. During reduction summation of frame evaluation values, we employ parallel reduction using GPU shared memory. We first obtain the maximum threads per block available on the current GPU. If the number of pixels per frame is less than this maximum, a single thread block is allocated; otherwise, multiple blocks are calculated based on the pixel count. For summation within each thread block, we allocate the maximum available shared memory buffer (with offset equal to thread index), store each thread's computed values in this buffer, and then perform reduction iteration. All thread blocks operate simultaneously using this strategy to efficiently obtain the integrated evaluation results for the entire frame set.

3. GPU-Based Frame Selection Algorithm Implementation

Considering the characteristics of ONSET observation data and frame selection algorithms, we have currently implemented one spatial domain method (average gradient) and one frequency domain method (spectral ratio).

3.1 Average Gradient Method

The average gradient method reflects the intensity of edges and details in an image. The calculation formula is:

$$\text{Average Gradient} = \frac{1}{(M-1)(N-1)} \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} \sqrt{\frac{(f(i,j)-f(i+1,j))^2 + (f(i,j)-f(i,j+1))^2}{2}}$$

where M and N represent the total rows and columns of the image, and $f(i,j)$ is the grayscale value at row i, column j. The expression inside the square root represents the sum of squared differences between pixel F and its adjacent bottom and right pixels.

As evident from the formula, all pixels in each frame participate in the computation, creating substantial computational load. Without GPU acceleration, the average gradient method can only achieve speedup by computing partial image regions, which sacrifices some effective image quality evaluation information and inevitably affects selection accuracy. To address this, we simplified the approach by computing only a central region at half resolution: for near full-disk solar images of 2560×2160 pixels per frame, we select a central 1280×1080 region; for partial-disk images of 1700×1700 pixels, we select a central 850×850 region.

[Figure 2: see original paper]

When using GPU parallel acceleration with the real-time processing module, we compute the average gradient value for the entire image region. As shown in Figure 2, each thread participates in the following steps: (1) checking whether current row/column boundaries are exceeded and computing the squared difference between the current pixel and its bottom neighbor; (2) computing the squared difference between the current pixel and its right neighbor; and (3) summing these two squares, dividing by 2, and computing the mean. The key kernel function code is illustrated in Table 2.

3.2 Spectral Ratio Method

The spectral ratio method evaluates frames based on the strength of the target region's power spectrum. The approach normalizes each frame's power spectrum by the average power spectrum of all images in the sequence to eliminate target-related quantities, then integrates over the target frequency band. Frames with larger integral values are considered higher quality.

This method requires multiple Fourier transforms. Without GPU acceleration, only the FFTW library can be used for Fast Fourier Transform operations. Since Fourier transforms are computationally expensive, we adopt a central region cropping approach for both serial and parallel implementations: for near full-disk images, we define start row 540, start column 640, end row 1619, and end column 1919; for partial-disk images, start and end rows/columns are both 425 and 1274.

[Figure 3: see original paper]

With GPU parallel acceleration, as shown in Figure 3, the kernel function for computing spectral ratio evaluation values follows this process: (1) each thread converts corresponding point data from real to complex numbers within the cropped region, simultaneously performing the cropping operation; (2) multiplies the cropped region with a window function, performs complex-to-complex Fourier transform using CUFFT; (3) computes the power spectrum from the transformed values and converts to real numbers; (4) normalizes the power spectrum by the zero-frequency component (setting zero-frequency to 1); (5) obtains the mean of normalized power spectra for the sequence and computes relative power spectra; and (6) multiplies the relative power spectrum with a ring function and integrates to obtain the evaluation value in the target band. The parallel algorithm uses the CUFFT library instead of FFTW for FFT operations. To optimize thread allocation with one thread per pixel, we determine kernel thread counts based on the cropped region dimensions. For further optimization, we employ CUBLAS library functions: `cublasSscal` for computing average power spectra and `cublasSasum` for calculating integration over target frequency bands.

4. Experiments and Performance Comparison

We conducted comprehensive experiments to validate our implementation using five test datasets: one ONSET-observed H near full-disk solar image set (2560×2160 pixels/frame, 11 MB) and four partial-disk images with prominent solar activity (first two H , last two white-light; 1700×1700 pixels/frame, 6 MB). The selection target was to choose the best 70 frames from each 100-frame near full-disk set, and 50 frames from each 70-frame partial-disk set.

4.1 Average Gradient Method

Without GPU parallel acceleration, the average gradient method can only accelerate computation by processing partial image regions. As shown in Figure 4, this approach yields partial-region serial times significantly lower than full-region serial times, achieving an average 3.7× speedup. However, this improvement sacrifices effective image quality evaluation information, inevitably affecting selection accuracy.

With GPU parallel acceleration from our real-time processing module computing average gradient values for entire images, the parallel algorithm still demonstrates clear speed advantages over partial-region serial computation, achieving 6–8× speedup compared to full-region serial times. Experiments confirm that the average gradient frame selection module operates stably and reliably, with the parallel implementation delivering significantly superior execution efficiency compared to the original serial approach.

4.2 Spectral Ratio Method

As shown in Figure 5, with GPU parallel acceleration from our real-time processing module, the parallel spectral ratio algorithm demonstrates significantly superior execution efficiency compared to serial implementation for the same processing region. For near full-disk images, the fastest total frame selection time is 1.2 seconds, representing a 7× improvement over the original serial implementation; for partial-disk images, the fastest time is 0.7 seconds with an average 5× speedup. Experiments demonstrate that with spectral ratio method integration, our real-time frame selection module operates stably and reliably with remarkable efficiency improvements.

5. Conclusion

This paper presents the design and implementation of a GPU-based real-time frame selection processing module, discussing its implementation in detail and achieving high-speed parallel processing of both average gradient and spectral ratio methods. Experimental comparisons demonstrate excellent acceleration ratios, with performance already meeting ONSET' s real-time observation and processing requirements, establishing a solid foundation for real-time high-resolution image reconstruction.

Future work will focus on developing frame selection algorithms tailored to different data types and characteristics of ONSET observations, and integrating these into the future ONSET data processing system to meet evolving scientific research needs.

References

[1] Li Zhen. Observations and Research of Small-Scale Activities in the Solar Lower Atmosphere[D]. Nanjing: Nanjing University, 2016.

[2] Liu Changyu. Application of Frame Selection Technology in Solar High-Resolution Imaging[D]. Kunming: Yunnan Observatories, Chinese Academy of Sciences, 2013.

[3] Wang Z, Bovik A C, Sheikh H R, et al. Image quality assessment: from error visibility to structural similarity[J]. IEEE Transaction on Image Processing, 2004, 13(4): 600-612.

[4] Zheng Yanfang. Evaluation of Solar High-Resolution Data and Observational Study of Jets/Surges[D]. Kunming: Yunnan Observatories, Chinese Academy of Sciences, 2014.

[5] Zhao Qing, He Jianhua, Wen Peng. Image fusion method based on average grads and wavelet contrast[J]. Computer Engineering and Applications, 2012, 48(24): 165-168.

[6] Shi Zheng, Xiang Yongyuan, Deng Hui, et al. The frame selection for New Vacuum Solar Telescope Level 1 data processing based on GPU[J]. Chinese Science Bulletin, 2015, 60(15): 1408-1413.

[7] Liu Changyu, Xiang Yongyuan, Jin Zhengyu. Image processing with frame selection for H solar chromosphere images[J]. Astronomical Research and Technology—Publications of National Astronomical Observatories of China, 2014, 11(2): 140-144.

[8] Xiang Yongyuan. Research on High-Resolution High-Speed Reconstruction Algorithms for Solar Images[D]. Kunming: Yunnan Observatories, Chinese Academy of Sciences, 2016.

[9] Zhang Nengwei, Yang Yunfei, Li Ranyang, et al. A real-time image processing system for detecting and removing cloud shadows on H full-disk solar images[J]. Astronomical Research and Technology, 2016, 13(2): 242-249.

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