2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI) | 978-1-6654-6803-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCEAI55464.2022.00078

Research on SAR Image Processing Performance Based on OpenMP and CUDA Parallel Model

1st Yanbin Zhao
Innovation Academy for Microsatellites,
Chinese Academy of Sciences
Hefei, China
E-mail:1358813798@qq.com

2nd Yang LIU
Innovation Academy for Microsatellites,
Chinese Academy of Sciences
Shanghai, China
E-mail:lyhit1980@163.com

***Abstract*—In the field of remote sensing image processing, Synthetic Aperture Radar(SAR) image has a large amount of data and takes a long time to process. Especially in the military field and disaster detection, conventional serial processing methods cannot meet the requirements of real-time performance. To solve this problem, the high-performance computing advantages of parallel processing are utilized. The optimization effect of SAR image processing performance was studied based on OpenMP shared memory parallel processing method and CUDA parallel model. Combined with the use of OpenCV computer vision library, basic image processing tasks and parallel processing framework were designed. According to the comparison of the execution time between parallel processing and serial processing, the SAR image preprocessing speed is increased by 2 to 16 times. It is concluded that the parallel computing model is an effective fast SAR image processing optimization scheme.**

***Index Terms*—Remote sensing image processing, SAR image, OpenMP, CUDA, OpenCV**

I. Introduction

Synthetic aperture radar (SAR) is an active and high-resolution radar, which is widely used in resource exploration, disaster prevention and military fields because it can operate all day and all weather and is not limited by clouds and fog and other natural conditions [1]. Generally, SAR image processing mainly includes three parts: image preprocessing, image segmentation and image classification and recognition. This paper mainly studies the performance optimization effect of SAR image preprocessing operation in parallel mode. In recent years, Li Desheng et al. have studied the SAR image preprocessing method based on level set [2]. In order to improve the accuracy of subsequent target recognition, the region elimination method is mainly used to filter out clutter in the binarized image, but the processing speed of this method is insufficient. Xue Xiaorong et al. studied the application of parallel computing in SAR image processing [3] and used Message Passing Interface(MPI) as a parallel programming tool to achieve SAR image classification algorithm. However, the parallel mode was relatively single, and a more suitable parallel optimization scheme could not be selected for specific processing tasks. SAR imaging contrast ratio is higher, the object boundary between clearer, target more stereo, but the resolution of SAR imaging is much lower than that of visible imaging, and SAR images often contain strong speckle noise [4]. In order to improve image quality and reduce its impact on recognition performance, image preprocessing becomes an important link. Due to the particularity of SAR image application, not only accurate processing is required, but also the processing speed should be close to real-time. Parallel processing technology can meet the requirements of accelerating the calculation of different scales of data, so it is a great advantage to apply this technique to SAR image rapid processing.

The experimental data in this paper are SAR images of BTR60 armored vehicles from MSTAR public database made public by Defense Advanced Research Projects Agency(DARPA) and Air Force Research Laboratory(AFRL). Firstly, OpenMP programming technology is used to implement image enhancement algorithm, including image rotation, Hue Saturation Value(HSV) image conversion, luminance contrast adjustment, image free clipping and histogram equalization, as well as Wiener filtering processing used in image restoration task. In order to demonstrate the superiority of CUDA programming parallel computing performance, The CUDA programming model is used to implement gaussian filtering for SAR images. On the basis of serial programs, the running time difference between serial processing and parallel processing is compared to reflect the performance optimization effect of parallel computing. The overall experimental design framework is shown in Fig.1. The configuration of the experiment is as follows:

- The CPU is AMD dual-core processor with a basic frequency of 2.60GHz
- The graphics card is NVIDIA GeForce MX350, with a core frequency of 1354MHz and a memory size of 2048MB

The work was supported by the Key Deployment Project of National Defence Science and Technology Innovation of Chinese Academy of Sciences (Grant No. ZDRW-KT-2016-02-2)

978-1-6654-6803-9/22/$31.00 ©2022 IEEE
DOI 10.1109/ICCEAI55464.2022.00078

- programming environment: Visual Studio 2019, C++ programming language , CUDA 10.0, OpenCV library.

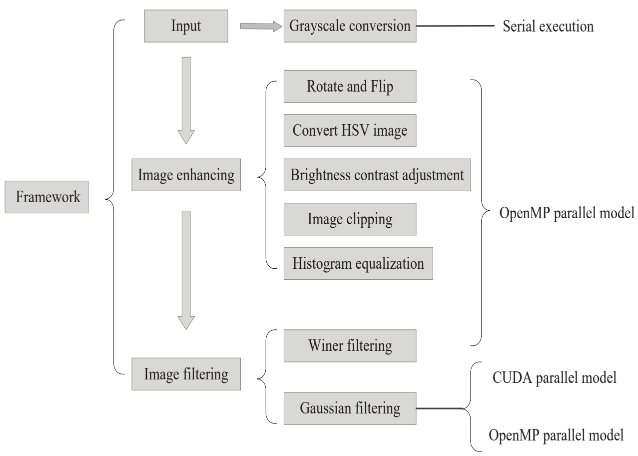


Fig. 1. Experimental design framework

II. Parallel programming model

A. OpenMP programming model

OpenMP is an extension of serial programming language. It is an application programming interface designed to write parallel programs on multi-processors of shared memory parallel system, especially suitable for parallel programming on multi-core Central Processing Unit(CPU) [5].

The basic idea of OpenMP parallel programming model is that there is only one main thread at the beginning of the program, and all the serial parts of the program are executed by the main thread. The parallel part is executed by spawning other threads. However, the serial part is not executed if the parallel part does not end. Fig.2 shows the structure of the parallel model.

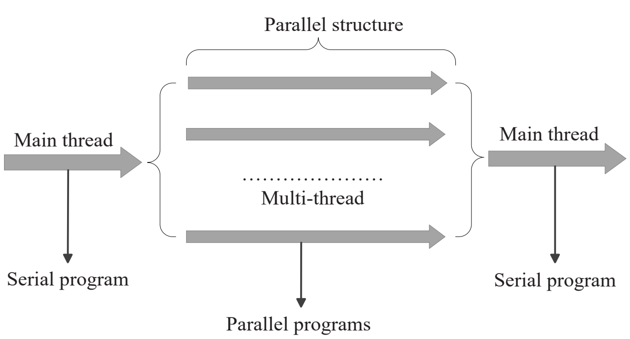


Fig. 2. Parallel model structure

In OpenMP parallel mode, the shared task structure divides the contained code among the members of a thread group for execution. The "fork-Join" model adopted in this experiment is the most common one, and its advantage is to develop serial computing programs in the way of increasing quantization [6]. Fig.3 is the process diagram of parallel computing. At the beginning, the program is executed sequentially, and only one main thread exists. When a parallel computing task is encountered, the main thread performs "Fork" operation to generate a group of threads, After the task is completed, each thread performs the operation of "Join" to return to the main thread.

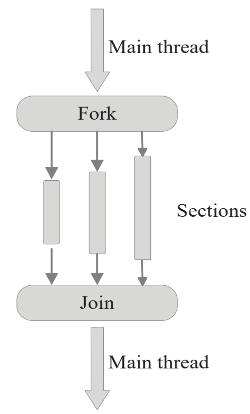


Fig. 3. Parallel computing process

B. CUDA programming model

CUDA is a general-purpose parallel computing architecture introduced by NVIDIA, which enables Graphical Processing Unit(GPU) to solve complex computing problems. It includes CUDA instruction set architecture(ISA) and a parallel computing engine inside the GPU. CUDA provides host-device programming mode, a large number of interface functions and scientific computing libraries, and achieves parallelism by executing a large number of threads at the same time [7].

The basic idea of CUDA is to support a large amount of thread-level parallelism, and to dynamically schedule and execute these threads in hardware. Cuda-based program code is actually executed in two types, one is host code running on CPU, the other is device code running on GPU. The CUDA parallel computing function running on the GPU is called kernel function, which is executed in multiple threads on the GPU [8]. A complete CUDA program is composed of a series of parallel kernel functions on the device side and serial processing part on the host side.Before the kernel function is executed, the number of threads in each thread block and the size of shared memory need to be configured.

III. Image processing parallel implementation

A. Image enhancement

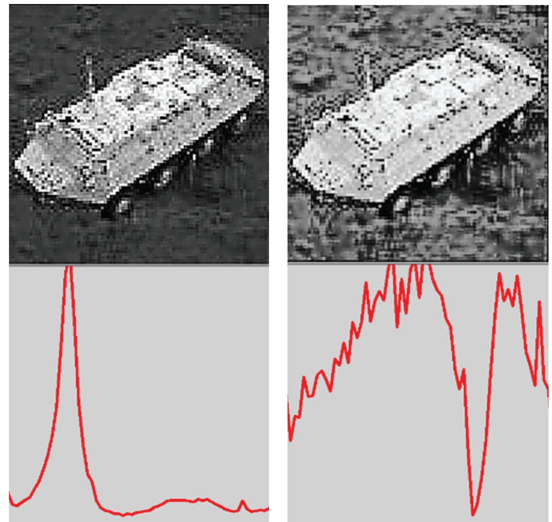
Image enhancement is a kind of processing to make the image more suitable for specific application than the original image, which can enhance the useful information in the image.

Under the deep learning category, image enhancement is to enrich the image sample characteristics, make model stronger generalization ability, through the existing image data can be flip, rotation, scaling, and luminance enhancement operation, to generate a new image to participate in the training and testing, this operation can increase the number of images several times, reducing the possibility of overfitting. Image enhancement methods used in this experiment include image rotation, HSV image conversion, luminance contrast adjustment, image free cropping, histogram drawing and histogram equalization.

The realization of image rotation firstly obtains the rotation matrix according to the set rotation angle and rotation center, and then realizes the rotation operation after obtaining the width and height of the transformed image. Image flipping and converting to gray scale are realized by setting corresponding parameters. Luminance contrast adjustment is achieved by linear fusion of two images of the same size and type, and the sliding control can dynamically modulate the Luminance and contrast of the image. Image free cropping is the operation of extracting regions of interest, processing monitored mouse action, drawing rectangle and clipping corresponding region, which can be used to make target detection annotation tool.

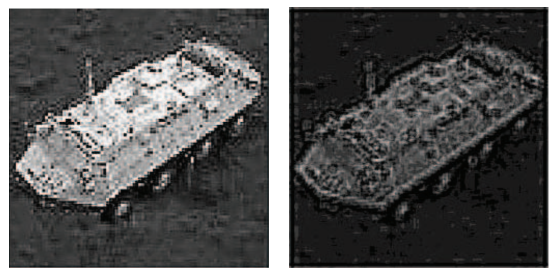
Histogram equalization is one of the simplest and most widely used contrast enhancement method [9], the histogram of grayscale image can be drawn by calling library functions. The principle of histogram equalization is the process of mapping one distribution to another distribution based on a functional relationship. The gray value is adjusted by using the cumulative probability density distribution function, specifically calculating the cumulative probability of each channel pixel level (0,L-1). Then, the original pixel is replaced by the cumulative probability multiplied by the upper limit of pixel level (L-1). Formula 1 for the histogram equalization transformation, where P_r(w) represents the probability density function of input r and s is the output. As shown in Fig. 4, to achieve histogram equalization, its input must be a single-channel image.

$$s = T(r) = (L-1)\int_0^r P_r(w)dw \tag{1}$$



(a) Before equalizing (b) After equalizing

Fig. 4. SAR image histogram equalization



(a) Before filtering (b) After filtering

Fig. 5. Wiener filtering

B. Wiener filtering

Because the image is usually inevitably disturbed by some noise in the process of acquisition and transmission, the image should be restored to its original true state as far as possible before image analysis. Image restoration is the key problem is to establish regression model, the wiener filter is a kind of adaptive minimum mean square error filter, Its ultimate goal is to minimize the mean square error(MSE) between the restored image and the original image [10]. In the field of signal processing, the original image is equivalent to useful signal s(t), restored image equal to the best estimate of the useful signal, Wiener filtering of the image is shown in Fig.5. Formula 2 establish frequency domain degradation model:

$$G(u,v) = F(u,v)H(U,V) + N(u,v) \tag{2}$$

In 2, G(u,v) is the degraded image as the filter input, F(u,v) is the original image, H(u,v) is the degraded function, and N(u,v) is the additive noise. The $\widehat{F}$(u,v) obtained by 3 is the best estimate of the original image, that is, the restored image.

$$\hat{F}(u,v) = [\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2 + S_n(u,v)/S_f(u,v)}]G(u,v) \tag{3}$$

In the above 3, $S_n(u,v)$ is the power spectrum of noise, and $S_f(u,v)$ is the power spectrum of desired signal. The derivation of the degradation function (also known as the diffusion function in the field of image processing) H(u,v) is the key point of using Wiener filter to restore the image. From time domain analysis, the degenerate function h(x,y) is the optimal solution of Wiener-Hopf equation when the minimum mean square error of the original image f(x,y) and the restored image $\widehat{f}$(u,v) reaches the minimum, and the solution is the

(a) SAR image (b) OpenMP (c) CUDA

Fig. 6. Gaussian filter

Wiener filter coefficient h_{opt}, which is solved by 4, the auto-correlation coefficient matrix R_{gg} of the filter input (degraded image g(x,y)) and the cross-variance matrix R_{gf} of the input and the desired original image f(x,y) are needed.

$$h_{opt} = R_{gg}^{-1} R_{gf} \tag{4}$$

C. Gaussian filtering

In the field of image processing, Gaussian filtering is a process of weighted average of the whole image. After filtering, the value of each pixel is obtained by weighted average of its own and other pixel values in the neighborhood. The filtering process is to use a template to scan every pixel in the image, and use the weighted average value of the neighborhood pixels determined by the template to replace the value of the center pixel of the template, and finally get the new filtered image [11]. The template coefficient of the Gaussian filter obeys the two-dimensional Gaussian distribution and decreases with the increase of distance from the template center, as shown in 5. Therefore, the Gaussian filter has a smaller degree of blurring to the image and can better maintain the overall details of the image.

$$f(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-ux)^2+(y-uy)^2}{2\sigma^2}} \tag{5}$$

Where (x,y) is the coordinates of any point in the template, (ux,uy) is the coordinates of the central point of the template (0,0), σ is the standard deviation, and I is the coordinates of the 3×3 template.

TABLE I
TABLE TYPE STYLES

(-1,1)	(0,1)	(1,1)
(-1,0)	(0,0)	(0,1)
(-1,-1)	(0,-1)	(-1,1)

The template coordinates are put into the Gaussian function successively to obtain the weighted coefficient of the template at each coordinate, and then the Gaussian template is formed after normalization. The Convolution calculation between the Gaussian template and the image pixel value at the corresponding position is carried out, and finally the gray value of the central pixel of the template is obtained. After processing SAR images with white noise, two parallel methods of OpenMP and CUDA are used to obtain the result output after Gaussian filtering, as shown in Fig.6.

Gaussian filter template coefficient generated in the test with the serial implementation way, template and image pixel traversal convolution computation time complexity is higher, which is implemented in parallel with CUDA programming model. Performance was measured by comparing serial and parallel computing time differences. Meanwhile, the program execution time of OpenMP and CUDA parallel modes is compared. The experiment timed the part of the convolution calculation program, the serial timing uses CPU host side timing function omp_get_wtime(), and the parallel process uses CUDA event timing Application Programming Interface(API).

IV. RESULTS AND ANALYSIS

In this paper, for each SAR image processing task, OpenMP model was used to implement the algorithm with a small amount of computation, and CUDA model was used to implement the program with a large amount of computation and complex computation process. The execution time of the program is used to compare the performance difference between the parallel mode and the serial mode, and the optimization effect of the parallel mode on the performance is measured.

By analyzing II, it can be concluded that the parallel mode of OpenMP shared memory can shorten the execution time of the program by at least 2 times, and the acceleration multiple is related to the number of processor cores and the amount of computation. When the amount of computation is larger, the parallel acceleration effect is more obvious. In general, CUDA parallel mode has better performance than OpenMP parallel mode. However, in the experiment, CUDA parallel execution takes more time, however, the phenomenon that CUDA parallel execution takes more time appears in the experiment, and the acceleration effect of CUDA is reflected by changing the size of convolution kernel to increase the amount of computation.

V. CONCLUSIONS

In this paper, based on remote sensing image processing of large amount of data and high processing precision, processing consumed for a long time, the SAR image preprocessing experiment framework design, including basic image enhancement operation, the wiener filter and gaussian filter.

TABLE II
COMPARISON OF EXPERIMENTAL RESULTS

Waiting task	Serial execution time (ms)	Parallel model	Parallel execution time (ms)	Accelerate rate
Rotate and Flip	246.734	OpenMP	100.258	2.461
Convert HSV image	226.063	OpenMP	71.642	3.155
Luminance contrast adjustment	384.000	OpenMP	169.004	1.959
Image clipping	208.880	OpenMP	96.052	2.174
Histogram equalization	619.304	OpenMP	73.301	8.449
Winer filtering	150.009	OpenMP	42.738	3.509
Gaussian filtering	154.534	OpenMP	79.989	1.932
		CUDA	9.234	16.735

By comparing the program execution time under serial and parallel programming models, according to the research results of acceleration performance of multi-core OpenMP parallel programming model and CUDA parallel computing architecture, the parallel computing method can greatly improve the efficiency of image processing under the condition of sufficient data processing, and under the same conditions, CUDA parallel computing architecture can make full use of the respective advantages of CPU and GPU, and its processing efficiency is higher than multi-core OpenMP parallel programming model. Finally, through the analysis of the experimental results of SAR image preprocessing, it is concluded that the image preprocessing speed can be effectively improved in the parallel environment, and the parallel model can provide an efficient and low-cost processing platform for remote sensing image processing.

REFERENCES

[1] E. Ponmani and P. Saravanan, "Image denoising and despeckling methods for sar images to improve image enhancement performance: a survey," *Multimedia Tools and Applications*, pp. 1–23, 2021.

[2] D. Li, X. Meng, G. Liu, H. Zhang, and L. Liu, "Pre-processing method for sar image based on level sets," *Fire Control Radar Technology*, vol. 000, no. 003, pp. 41–45, 2015.

[3] X. Xue and Q. Zeng, "Application of parallel computing in sar image processing," *COMPUTER APPLICATIONS*, vol. 26, no. B06, p. 2, 2006.

[4] W. Wang and C. Yang, "A review of object recognition methods based on image processing technology," *Computers and Information Technology*, vol. 28, no. 1, p. 5, 2020.

[5] D. Cheng and W. Wang, "Application of openmp in sar image processing," *COMPUTER APPLICATIONS*, no. S1, p. 4, 2017.

[6] J. Zhu and K. Wu, "Wheel set image processing method under the multiprocessor environment based on openmp parallel technology," *Journal of Mechanical Electrical Engineering*, vol. 32, no. 6, p. 4, 2015.

[7] X. Liu, C. Jiang, and C. Feng, "Research on cuda and opencv image parallel processing method," *Science of Surveying and Mapping*, vol. 37, no. 4, p. 3, 2012.

[8] H. Zeng, "Research and implementation of gpu acceleration technology for image processing," Ph.D. dissertation, Southwest University of Science and Technology, 2019.

[9] K. G. Dhal, A. Das, S. Ray, J. Gálvez, and S. Das, "Histogram equalization variants as optimization problems: A review," *Archives of Computational Methods in Engineering*, no. 5, 2020.

[10] R. Liu, Y. Li, H. Wang, and J. Liu, "A noisy multi-objective optimization algorithm based on mean and wiener filters," *Knowledge-Based Systems*, no. 9, p. 107215, 2021.

[11] Y. Wei, "Gauss filter openmp parallelization," *Communication World*, 2015.