GPU-Accelerated Circular SAR Echo Data Simulation of Large Scenes

Lingjuan Yu[1], Xiaochun Xie[2], and Lingling Xiao[3]

[1] Jiangxi University of Science and Technology, Ganzhou, China, 341000, yljsmile@163.com

[2] Gannan Normal University, Ganzhou, China, 341000, xiexiaochun@sina.com

[3] Jiangxi University of Science and Technology, Ganzhou, China, 341000, gzxll@aliyun.com

Abstract

In circular synthetic aperture radar (CSAR), echo data simulation can provide ideal echo data for the processing of imaging and performance analysis. However, when the target scene is large, the amount of echo data is also large according to Nyquist sampling theorem. If the traditional serial computing way based on CPU is used, then the execution time is long. In this paper, we adopt graphic processing unit (GPU) to improve the computing speed. This GPU-accelerated way can be simply described as the process that CSAR echo data with carrier information is generated on GPU, and the subsequent removal of carrier is implemented on CPU. Simulation results show the GPU- accelerated way is faster than the CPU-based way when the target scene is large.

1. Introduction

In circular synthetic aperture radar (CSAR), radar collects data over a circular trajectory [1]. However, it is hard to keep the trajectory to be a standard circle in the real data collection, which leads to the difficult in the processing of imaging and performance analysis [2-3]. One way to solve this problem is echo data simulation under ideal acquisition conditions. As we expected, synthetic aperture radar (SAR) echo data simulation was widely studied [4], and CSAR echo data simulation was also investigated in references [5]. In CSAR echo simulation, target scene can be seen as multiple discrete point targets. If the target scene is very large, then there will be large amounts of echo data, which means it will take up a great deal of time to generate echo data in the traditional calculation way based on CPU.

Graphic processing unit (GPU) proposed by NVIDIA Corporation has enormous float-point arithmetic capability and parallel processing capability [6]. So far, GPU has been widely used in the fields which refer to large amounts of data computing to increase calculation speed, for example, electromagnetic computing, electromagnetic imaging, SAR signal processing, and so on. In this paper, GPU is adopted to improve the speed of CSAR echo generation. The rest of this paper is organized as follows. In section 2, CSAR echo data acquisition is overviewed. In section 3, the principle of GPU- accelerated CSAR echo data simulation is illustrated. The CSAR echo simulation results are shown in section 4. Finally, section 5 concludes the paper.

2. CSAR System Model

In CSAR mode, radar platform travels around targets in a circular trajectory, which is different from the traditional linear SAR mode. The CSAR system model for data acquisition is shown in Fig.1 (a). Its top view and side view are shown in Fig.1 (b) and (c), respectively. Denote the radar height as zc and the flight radius as Rg. Then the slant range can be expressed as,

$$R_0 = \sqrt{R_g^2 + z_c^2} \tag{1}$$

and the slant depression angle can be calculated by,

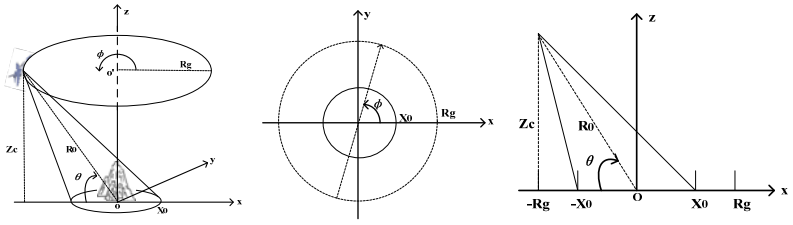
$$\theta = \arctan(z_c / R_g) \tag{2}$$

Denote the target location as (x, y, z), and the azimuthal angle of radar as ϕ. Then the range between radar and target can be written as [1],

$$R = \sqrt{(x - R_g \cos\phi)^2 + (y - R_g \sin\phi)^2 + (z - Z_c)^2} \tag{3}$$

Equation (3) is the case of three-dimensional target. For two-dimensional target, we only need to set $z = 0$.

978-1-4673-5225-3/14/$31.00 ©2014 IEEE



(a) (b) (c)

Fig. 1 (a) Geometry of CSAR system (b) Top view (c) Side view

3. The Principle of GPU- Accelerated CSAR Echo Data Simulation

3.1 The Baseband Echo Signal

In real CSAR echo data collection, let us assume that radar transmits a chirp signal modulated by carrier signal. The baseband chirp signal is in the form of,

$$s_0(t) = rect\left(t/T_p\right)\exp\left(j\pi Kt^2\right) \tag{4}$$

where T_p is the pulse duration, $rect\left(t/T_p\right)$ is standard rectangular window function, and K is linear frequency modulation rate. The carrier signal can be written as,

$$s_c(t) = \exp\left(j\omega_c t\right) \tag{5}$$

After the transmitted modulated signal interacting with targets, the received echo signal can be expressed as,

$$s\left(t\right) = s_0(t-\tau) = rect\left[\left(t-\tau\right)/T_p\right]\exp\left[j\omega_{cc}\left(t-\tau\right) + j\pi K\left(t-\tau\right)^2\right] \tag{6}$$

where the modified center frequency of echo signal is $\omega_{cc} = \omega_c - KT_p$, $\tau = 2R/c$ is time delays, and c is light speed.

Because the received signal carries carrier information, we need to remove it to obtain the baseband echo signal, that is,

$$s_b\left(t\right) = rect\left[\left(t-\tau\right)/T_p\right]\exp\left[j\omega_{cc}\left(t-\tau\right) + j\pi K\left(t-\tau\right)^2\right]\exp\left(-j\omega_c t\right) \tag{7}$$

3.2 GPU- Accelerated CSAR Echo Data Simulation

When the target scene is very large, the number of samples both in the range direction (corresponding to fast-time domain) and azimuthal direction is also very large. If we choose the number of azimuth pulses as the number of GPU threads, then it may cause that GPU memory is insufficient. If we choose the number of range units as the number of GPU threads, then there may exists memory conflicts. Considering these factors, the appropriate number of point targets in one range unit for each azimuth pulse is chosen.

From equation (6) and (7), echo generation mainly includes two steps. The first step is to generate echo data with carrier information, and the second step is the removal of carrier based on the results of the first step. In order to improve the calculating speed, the first step is implemented on GPU, while the second step is implemented on CPU. If the second step is also completed on GPU, which means that the processing is done for each point target, then the consumed time of this way will be longer than the case that the processing is done for each range unit for each azimuth pulse on CPU.

To achieve the good computing performance, the detailed steps of GPU- accelerated CSAR echo data generation are as follows.

(1) Set radar parameters such as carrier frequency, chirp bandwidth, chirp pulse duration, pulse repetition frequency, radar height, flight radius of radar platform, and so on.

(2) Set target parameters such as the size of target scene, the number of point targets in the observed scene, and so on.

(3) Calculate the azimuthal sampling number Na and range sampling number Nr, combing with radar and target parameters.

(4) Copy all the parameters of radar and targets from CPU to GPU for each azimuth pulse.

(5) Set the number of threads according to the number of point targets for each range unit.

(6) Calculate echo data with carrier information on GPU according to equation (6) for all threads, and accumulate echo data of all threads for each range unit.

(7) Copy results of step (6) from GPU to CPU, and remove carrier information according to equation (7), aiming to obtain the baseband echo signal.

(8) Repeat steps (5) – (7) until the baseband echo data of all the range units are generated for one azimuth pulse, and save results to a file.

(9) Repeat steps (4) – (8) until the baseband echo data of all the azimuth pulses are generated.

(10) Combine data in all files to form the final CSAR echo data.

The flow diagram of GPU- accelerated CSAR echo data generation is shown in Fig. 2, and the processing steps in dashed rectangular are the calculation on GPU.

4. Simulation Results

The CSAR system parameters are listed in Table 1 and parameters of simulation platform is described in Table 2. Simulation programs are written in CUDA. During the simulation, we choose SAR images as target scenes. Each pixel in a SAR image is seem as one point target, and each normalized pixel value is taken as the radar cross section (RCS) of one point target.

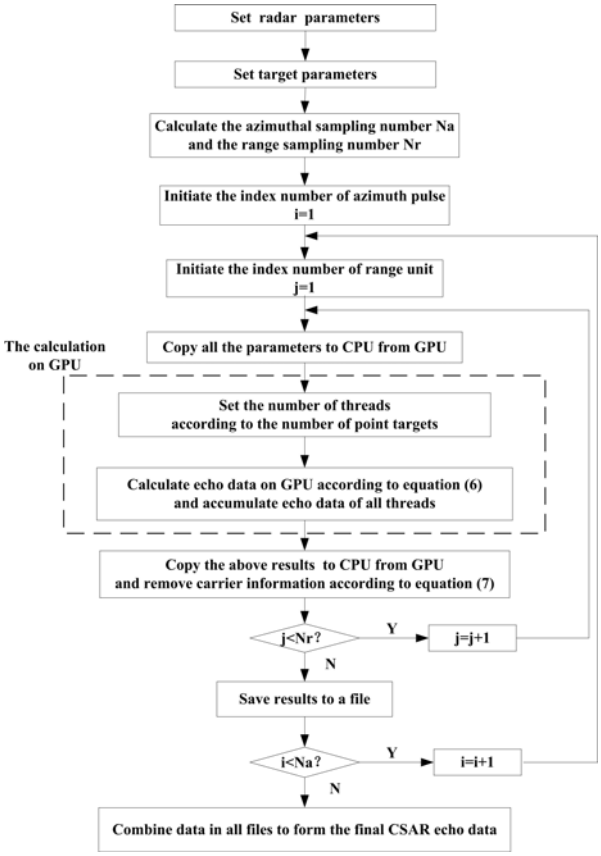


Fig. 2 Flow diagram of GPU- accelerated CSAR echo data generation

We change the size of the target scene, which means that the total number of point targets is also changed. If the total number of point targets is changed from 625 to 160000, the execution time curves of CPU-based and GPU-accelerated echo data generation are shown in Fig. 3.

When the total number of point targets is 625, the execution time of CPU-based way is about 0.121 seconds; while the GPU- accelerated way is 0.343 seconds. The latter is even longer than the former. When the total number of point targets is 2500, the execution time of CPU-based way is about 0.564 seconds; while the GPU- accelerated way is 0.539 seconds. The latter is almost the same as the former. When the total number of point targets is 160000, the execution time of CPU-based way is about 1408.1 seconds; while the GPU- accelerated way is 32.19 seconds. The ratio between the former and the latter is about 43.7.

From the above results, we can find that when the target scene corresponding to the total number of point targets is small, the CPU-based echo generation way is faster than the GPU- accelerated way, because the consumed time of data transmission between CPU and GPU can not be ignored. However, when the target scene is large, the GPU- accelerated way is faster than the CPU-based way.

Table 1 The CSAR system parameters

Carrier frequency	10GHz
Bandwidth	2GHz
Radius of radar platform	200m
Height of radar platform	200m

Table 2 The simulation platform parameters

CPU	Intel Core i5-3210M @ 2.5 GHz
Operation System	Windows 7 Professional 64-bit
GPU	NVIDIA GeForce 610M
CUDA	V 5.0

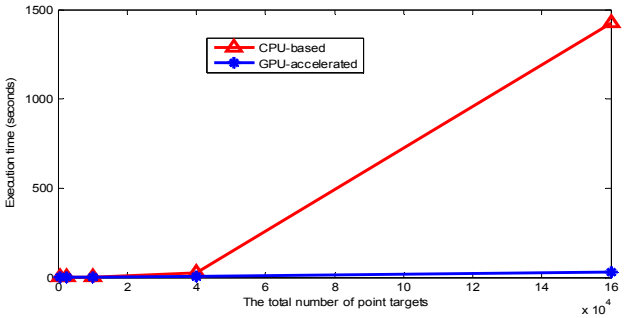


Fig. 3 The execution time of CPU-based and GPU- accelerated echo data generation

5. Conclusion

In this paper, GPU- accelerated CSAR echo generation is presented. This GPU- accelerated computing way mainly includes two important steps. The first step is that CSAR echo data with carrier information is generated on GPU, and the second step is that the subsequent removal of carrier is implemented on CPU. Simulation results show that, when the target scene is small, the performance of GPU- accelerated way may not as good as the CPU-based way; when the target scene is large, the GPU- accelerated way is faster than the CPU-based way.

Acknowledgements

This work was supported by Jiangxi Province Educational Committee Science Foundation for Youths (No. GJJ13387) and Jiangxi Natural Science Foundation (No. 20114BAB201020).

References

1. Soumekh M., “Synthetic aperture radar signal processing with MATLAB algorithms,” New York: J. Wiley, 1999.
2. Cantalloube H M J, Colin-Koeniguer E, Oriot H, “High resolution SAR imaging along circular trajectories,” IEEE International Geoscience and Remote Sensing Symposium, 2007: 850-853.
3. Oriot H, Cantalloube H, “Circular SAR imagery for urban remote sensing,” 2008 7th European Conference on Synthetic Aperture Radar, 2008: 1-4.
4. Franceschetti G, Iodice A, Perna S, and Riccio, D, “Efficient simulation of airborne SAR raw data of extended scenes,” IEEE Transactions on Geoscience and Remote Sensing, 2006, 44(10): 2851-2860.
5. Liu Q, Hong W, Tan W, Liu, and Y-R, T., “Efficient geosynchronous circular SAR raw data simulation of extended 3-D scenes,” Progress In Electromagnetics Research, 2012, 127: 335-350.
6. Sanders J, Kandrot E, “CUDA by example: an introduction to general-purpose GPU programming,” Addison-Wesley Professional, 2010.