基于 GPU 的多模式 SAR 实时成像算法研究

翟新刚[1,2,3] 韦立登[1,2] 汪丙南[1,2] 向茂生[1,2]
(1.中国科学院电子学研究所 北京 100190;2.微波成像技术重点实验室 北京 100190;
3.中国科学院大学 北京 100049)

摘 要：合成孔径雷达(SAR)成像处理需要较大的计算量,在基于中央处理器(CPU)平台开发的 SAR 成像系统上处理一般需要消耗很长时间,无法满足实时成像的要求。借助于计算统一设备架构(CUDA)编程模型,基于图形处理器(GPU)提出了一种适用于多模式 SAR 的实时成像方案。该方案通过数据分段处理技术解决了计算设备 GPU 显存容量不足的问题;通过分析成像处理任务的并行度,利用异步执行流处理技术减少数据处理对数据交互的等待时间;通过优化 GPU 内存访问机制并使用特殊函数单元(SFU)减少计算时钟周期。同时,该方案能够支持多 GPU 设备的并行处理,充分利用了 GPU 设备的计算资源。在 NVIDIA GT 740M 和 INTEL Q6600 上的实验结果表明,该方案与传统的基于 CPU 的单线程 SAR 成像技术相比,有了近 150 倍的速度提升,大大提高了 SAR 成像处理的计算效率,具有很好的工程应用前景。

关键词：多模式 SAR 实时成像 图形处理器(GPU) 流 异步执行
中图分类号：TN957.52 **文献标识码**：A **国家标准学科分类代码**：510.70

Research on real-time imaging algorithm for multi-mode SAR with GPU

Zhai Xingang[1,2,3] Wei Lideng[1,2] Wang Bingnan[1,2] Xiang Maosheng[1,2]
(1. Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China; 2. Science and Technology on Microwave Imaging Laboratory, Beijing 100190, China; 3. University of Chinese Academy Sciences, Beijing 100049, China)

Abstract: The SAR imaging system is time-consuming, which is developed based on central processing unit (CPU), because of the huge computation of SAR image processing, making real-time imaging impossible. Based on Computed Unified Device Architecture, This study proposes a new plan for real-time SAR imaging operated on graphic processing unit (GPU) which works for multiple working-mode SAR. This new proposal makes the plan work with small GPU memory by dividing SAR data into pieces, uses the asynchronous execution of multi-stream technology to cut down the time wasting in data transfer, optimizes memory management to get less access time of memory and uses light-weight function of special function unit (SFU) to reduce clock cycle. All computational resources are totally exploited because this plan is suitable for multi- GPU. It has been shown by an experiment on an NVIDIA GT740M and INTEL Q6600 that the proposed plan has an acceleration ratio of nearly 150. The proposed real-time SAR imaging system improves the imaging efficiency greatly and has a better application future.

Keywords: multi-mode SAR; real-time imaging; graphic processing unit (GPU); stream; asynchronous execution

1 引 言

合成孔径雷达(synthetic aperture radar,SAR)利用雷达平台的运动所形成的合成阵列来获得方位向的高分辨率,并通过雷达平台发射宽脉冲信号来获得距离向的高分辨率。高分辨率 SAR 成像需要处理的雷达回波数据量特别大,目前基于 CPU 平台实现的 SAR 成像一般都是利用较为复杂的编程手段进行后期处理,很少实时成像。

SAR 成像主要包括频域成像算法和时域成像算法,成像过程一般由快速傅里叶变换(FFT)、向量相乘、快速傅里叶逆变换(IFFT)以及插值重采样等运算模块组成。各运算模块均可以以较高效率并行化处理,因此借助于 CUDA 编程模型可以在 GPU 平台上快速实现 SAR 成像。基于

收稿日期:2015-12

GPU 的频域 SAR 成像算法需将一景 SAR 回波数据从内存转存到显存中进行处理，但支持 CUDA 编程模型的 GPU 产品显存容量一般不超过 6 GB，不足以容纳一景 SAR 回波数据[1]。后向投影(BP)算法是常见的时域 SAR 成像算法，逐方位时刻读取 SAR 回波数据的特点大大降低了该算法对 GPU 显存容量的要求；逐像素地对成像区域重建的特点使得该算法具有良好的成像质量，适用于多种 SAR 工作模式。

本文第 2 节对 SAR 成像技术中的频域算法和时域算法的并行结构特征进行分析；第 3 节介绍了所需的 CUDA 编程技巧，对基于 GPU 的实时成像算法进行优化加速；第 4 节给出了基于 GPU 的实时成像算法对条带 SAR 和圆迹 SAR 实测数据的成像结果，并与传统的基于 CPU 实现的成像结果进行对比分析；第 5 节总结了全文。

2　SAR 实时成像技术的并行结构分析

SAR 成像处理技术的根本是对 SAR 回波数据进行 2 维匹配滤波，包括频域算法和时域算法。频域 SAR 成像算法主要包括距离多普勒(RD)算法、Chirp Scaling(CS)算法和 wk 算法；时域 SAR 成像算法主要包括时域相关算法和 BP 算法。基于 GPU 实现的 3 种频域成像算法和 2 种时域成像算法的处理步骤如图 1 所示流程。

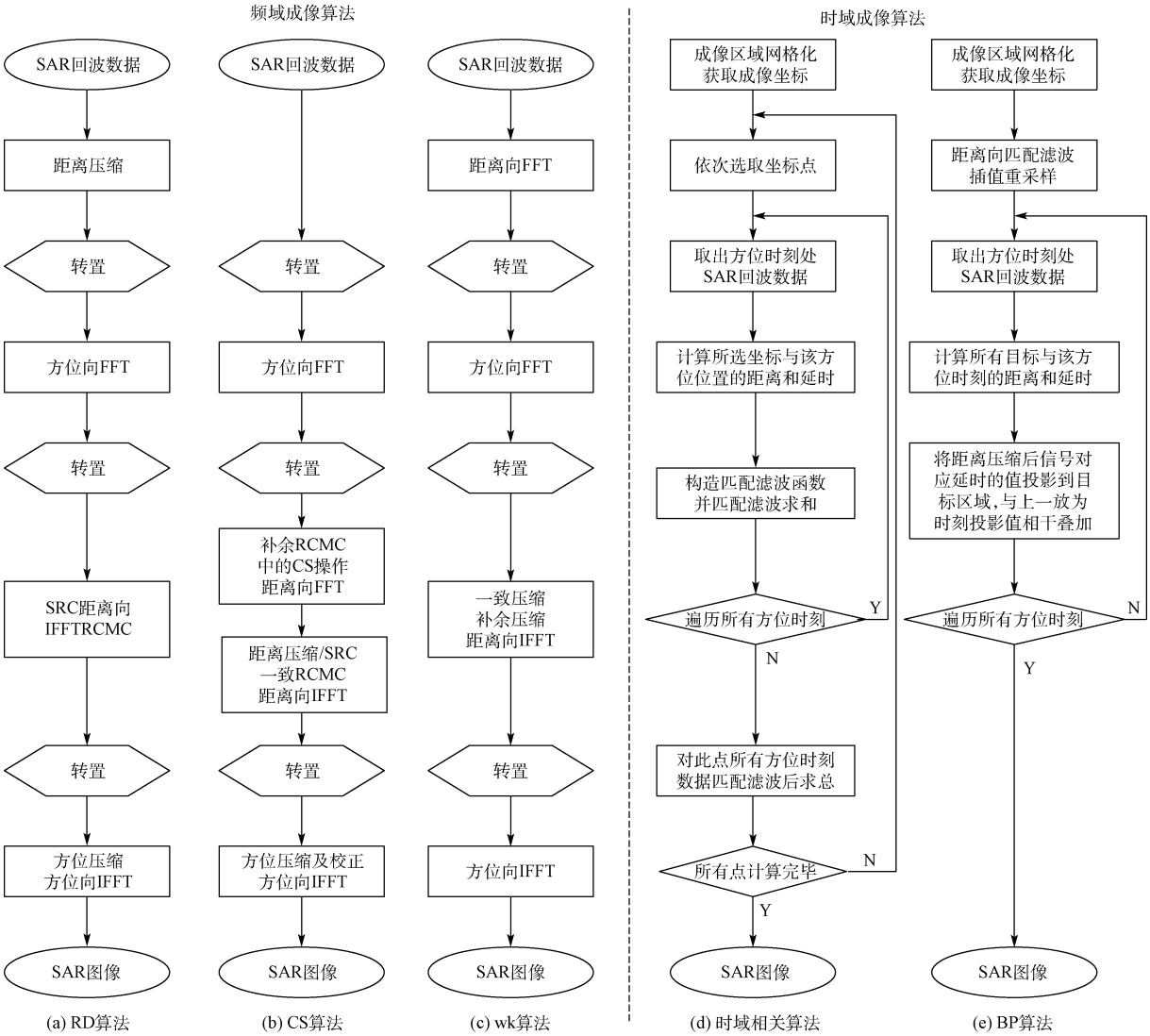


图 1　SAR 成像算法

由图 1 可知，基于 GPU 的 3 种频域 SAR 成像算法具有大致相同的算法结构，本文进行统一分析。该方案需要将一景 SAR 回波数据完全存储在内存和显存中，才可以进行成像处理。由于目前支持 CUDA 编程模型的 GPU 显存容量不超过 6 GB，一般需将 SAR 回波数据分块处理。由于计算机内存线性存储特点，为了提高处理设备对内存中数据的读写效率，该算法需要在 CPU 端进行 3 次转置操作，每次转置操作需要完成 1 次内存与显存的数据交互。

由于 GPU 端 PCI-E 接口带宽限制,频繁于内存和显存间交换数据会降低程序的运行效率和可操作性。时域 SAR 成像算法虽然计算复杂度高,但却可以在不依赖近似处理假设下进行精确成像,适用于多种 SAR 工作模式。时域相关算法和后向投影(BP)算法是常见的时域 SAR 成像算法,逐方位时刻读取 SAR 回波数据进行处理,逐像素地对成像区域重建,具有良好的成像质量。利用时域成像算法逐方位时刻的成像特点,可对 SAR 回波数据按方位向进行分块处理,降低对计算设备 GPU 显存容量的要求。由图 1 可知,基于 GPU 的时域相关算法需要对目标区域各目标点多次构造参考函数,工程实现复杂度极高;基于 GPU 的 BP 成像算法统一了参考函数进行匹配滤波,使用插值重采样得到目标区域目标点数据,这种处理在不改变成像适用范围的前提下大大提高了成像效率。本文采用有着更高效率且适用于多模式 SAR 成像的 BP 算法利用 CUDA 编程模型在 GPU 平台实现实时成像。

3 CUDA 编程的优化方案

CUDA 是由 NVIDIA 公司率先提出的统一计算设备体系结构,基于 CUDA 开发的程序代码在实际执行中分为两类,一类是运行在 CPU 端的串行部分,一类是运行在 GPU 端的并行部分,其中运行在 GPU 上的并行程序称为 Kernel 函数。基于 GPU 的 BP 成像算法主要包括两个过程,即距离向的脉冲压缩和方位向的后向投影。这两个步骤中,耗时短、不可并行实现的部分在 CPU 端串行实现;耗时长且可并行实现的部分在 GPU 端并行完成。利用 CUDA 编程模型将该方案在 GPU 上部署,并充分利用 GPU 设备的计算资源以及数据传输带宽,需要在 CUDA 编程中对程序进行优化处理。本文通过优化 GPU 内存访问机制,与 BP 算法中不同数据的不同用途相结合,减少了内存访问时间;利用 BP 算法逐方位时刻的成像方式,通过多 GPU 并行处理技术和异步执行流处理技术,提高数据并行度和任务并行度,充分利用 GPU 设备计算资源,提高计算效率;最后通过使用 SFU 提供的特殊函数,减少计算时钟周期,降低计算时间。

3.1 CUDA 内存访问机制的优化

GPU 平台的存储器类型包括全局内存、共享内存、本地内存、常量内存和寄存器等。其中全局内存采用 CPU 和 GPU 通用的访问机制,用于数据的传输和转存,空间大,所有线程都可对其进行读写操作;共享内存是片上内存,被同一个流多处理器(SM)上的线程所共享,访问其上的数据只需要 4 个时钟周期,避免了线程频繁访问全局内存造成的时间损失;常量内存采用了全局内存片上缓存来提高访问速度;寄存器只可被 1 个线程访问,同样拥有很快的读取速度[6]。

基于 GPU 的 BP 成像算法,首先从将 SAR 回波数据读取至内存,并由 CPU 端转存至 GPU 全局内存中。如果直接在全局内存进行成像处理,线程频繁访问全局内存将会造成大量的时间损失。在 Kernel 函数编写过程中,利用共享内存快速访问的特点,将全局内存中的 SAR 回波数据存储至共享内存中进行处理,避免了线程频繁访问全局内存造成的时间损失;在距离压缩过程中,参考函数只需读操作而不需写操作,与常量内存特点相匹配,选取常量内存存储参考函数可以提高距离压缩的速度;对于 Kernel 函数中可计算出数值的参数项存储至寄存器中,可减少每次执行不必要的计算。CUDA 内存访问机制的优化方案如图 2 所示。

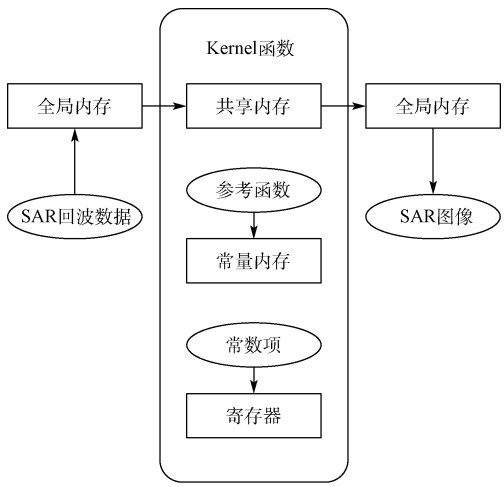


图 2 CUDA 内存访问机制优化方案

CUDA 编程模型中,线程为单指令多线程(SPMT)的管理办法,将待处理线程块中连续的 32 个线程统一进行调度管理,称为 1 个 Warp,但同时运行的实际是 16 个线程,即 Half-Warp。共享内存以 bank 为单位划分,每个 bank 占有 4 Byte。线程对 bank 的访问以 Half-Warp 为组进行。如果同时有多个线程访问同 1 个 bank,就会产生访问冲突。SAR 回波数据以复数形式存储,当在 CUDA 编程中以 8 Byte 的 float2 型定义共享内存数组时,同一 Half-Warp 中线程编号相差 8 的线程访问同一 bank,将会造成 2 路 bank 访问冲突,访问时间将根据线程数量成倍增加。因此,在编写 Kernel 函数时需定义 2 个 4 Byte 的 float 型共享内存数组分别用来存储 SAR 回波数据的实部和虚部,以避免 bank 访问冲突。

3.2 CUDA 并行处理技术

基于 GPU 的 BP 成像算法逐像素的成像方式决定了该方案每次只需处理一个方位时刻的 SAR 回波数据,完成数据在内存与显存之间的传输和 Kernel 函数的执行,得到“1 帧”SAR 图像,并等待与下一方位时刻下“1 帧”SAR 图像相干叠加直至得到最终 SAR 图像。将 SAR 回波数据按方位向分块交由多台 GPU 设备同时执行,可以实现数据高并行度;而利用 CUDA 编程模型中的异步执行流处理技术可以实现数据和任务的并行,进一步掩盖数据在内存

与显存之间的传输，从而减少甚至消除 Kernel 函数执行时对数据交互的等待时间。

3.2.1　多 GPU 并行处理技术

当进行 SAR 成像处理有多台 GPU 计算设备时，可以利用数据的并行度将 SAR 回波数据按方位向分块分配给不同的 GPU 进行处理。由于成像方案对 GPU 显存容量要求不高，所以对每个 GPU 计算设备而言，不需考虑 GPU 设备显存容量大小而直接按照 GPU 设备的计算能力大小将 SAR 回波数据进行分配，可以保证多 GPU 同时完成计算任务，减少计算能力强的 GPU 计算设备的等待时间，以实现计算资源的最大化利用。

当共有 N_a 方位向 SAR 回波数据需要处理时，对于第 i 个 GPU 计算设备，所对应的计算能力大小为 C_i，则所需处理的方位向数据个数 N_i 为：

$$N_i = \frac{C_i \times N_a}{\sum_{i=1}^{n} C_i} \tag{1}$$

3.2.2　异步执行流处理技术

CUDA 编程模型支持异步执行处理技术和流处理技术。异步执行处理技术允许内存与显存之间的数据传输和 Kernel 函数执行并行实现，但是这种并行处理方法对数据在内存和显存之间的传输和 Kernel 函数的执行有一定的限制：在数据从内存转入到显存完成之前 Kernel 函数一直处于等待状态；Kernel 函数执行完成之前也无法将处理结果从显存转入内存。流处理技术可以在数据在内存和显存交互的过程中，保持 Kernel 函数的执行，辅助实现异步执行处理。基于 GPU 的 BP 成像算法主要任务包括 SAR 回波数据从内存到显存的转存、距离压缩和后向投影、SAR 图像数据从显存到内存的转存，数据和任务的并行机制与异步执行流处理技术相结合，可以隐藏数据在内存与显存之间的交互传输，尽可能地保证 GPU 计算设备处于忙碌状态。

如图 3 所示，创建 4 个流，并将 SAR 回波数据按方位向分块分配给不同的流进行处理，每块数据的处理均包括“内存→显存的数据传输、Kernel 函数的执行、显存→内存的数据传输”3 项任务。由于流之间的相同任务不能并行执行、不同任务可以并行处理，因此 4 个流并行处理过程

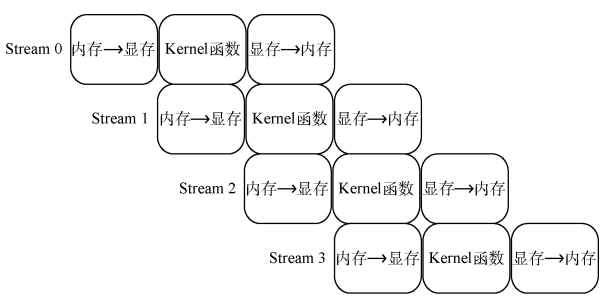


图 3　异步执行流处理技术

中，除了首尾少量任务不能并行执行外，其余绝大部分时间 4 个流都处于并行处理状态，从而掩盖了 Kernel 函数执行时对数据交互的等待时间，使得 GPU 计算设备始终处于忙碌状态。结合上述多 GPU 并行处理技术，当进行 SAR 成像处理有多台 GPU 计算设备时，每个 GPU 设备内部同样也可以实现异步执行流处理技术。

3.3　CUDA 特殊函数处理技术

基于 GPU 的 BP 成像算法主要包括距离向的脉冲压缩和方位向的后向投影两个过程。其中距离向的脉冲压缩包括距离向的 FFT、匹配滤波和插值重采样后 IFFT 3 个步骤。对于 FFT 和 IFFT 的实现，直接调用 CUDA 提供的 CUFFT 函数库实现；对于插值重采样，采用频域中段 N 倍补零的方法，既减小了线性插值带来的误差，也降低了 sinc 插值带来的巨大计算量，cudaMemcpyDeviceToDevice 函数可以在显存中高效的完成中段 N 倍补零操作，减少了一次数据在内存和显存之间的交互以及 CPU 端补零操作的耗时；BP 算法后向投影过程需要对 SAR 回波数据进行相位补偿，SFU 提供的三角函数 $\cos(x)$ 和 $\sin(x)$ 函数 1 个时钟周期可以执行 1 次运算，而 CPU 版本的三角函数 $\cos(x)$ 和 $\sin(x)$ 函数执行一次运算则需要 4 个时钟周期，因此使用 SFU 提供的特殊函数可以大大降低相位补偿的运算时间。需要注意的是，当实测数据中斜距波长比值较大时，此时需要补偿的相位因子 $4\pi R/\lambda$ 不能直接进行 $\cos(x)$ 和 $\sin(x)$ 操作，需将补偿的相位因子处理为 2π 小数倍的形式，即 $2\pi\left(2R/\lambda - \left|2\frac{R}{\lambda}\right|\right)$。

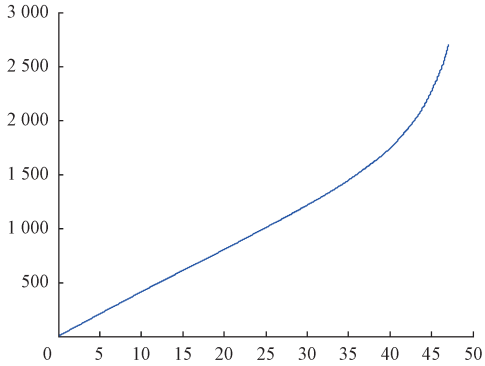
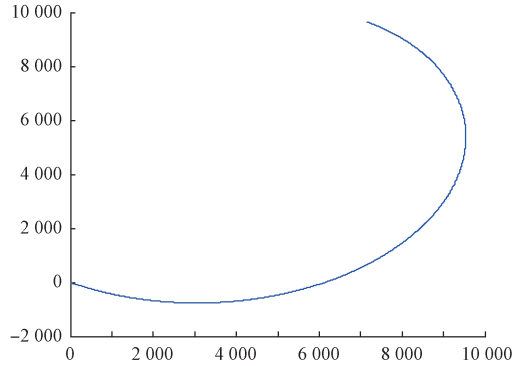
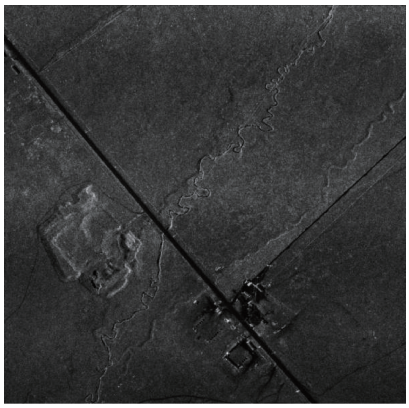
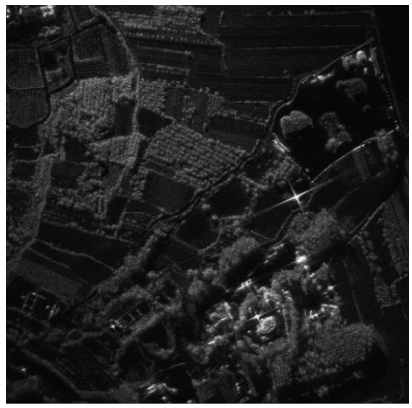
4　实验结果分析

本文提出的基于 GPU 的时域 SAR 成像算法选取 BP 算法，对成像处理算法本身未做任何更改，因此处理所能达到的精度与 CPU 实现方式相同且适用于多数 SAR 工作模式；该方案所需计算量与 CPU 实现方式相同，本文同样也研究了该方案在 GPU 上实现相对于 CPU 上实现的速度提升，最后给实时成像性能分析。本节测试所选用 GPU 为 NVIDIA GT 740 M，CPU 选用 INTEL Q6600 @ 2.40 GHz。

4.1　适用性分析

基于 GPU 的 BP 成像算法未对成像算法本身做任何改变，理论上应达到与 CPU 实现方式相同的精度。该方案不依赖于近似处理假设，适用于多种工作模式的 SAR 成像处理。为更有突出性地体现该方案对于多种 SAR 工作模式成像的适用性，本节选取了条带 SAR 和圆迹 SAR 作为研究对象。利用该方案对国内某研究所提供的条带 SAR 和圆迹 SAR 实测数据进行了成像处理，并给出了飞行轨迹、成像结果和分辨率成像处理指标，如表 1 所示。由表 1 可知，该方案对条带 SAR 和圆迹 SAR 进行成像处理均可得到聚焦良好的高分辨率 SAR 图像。

表 1 条带 SAR 和圆迹 SAR 实测数据成像结果

SAR 模式	条带	圆迹
飞行轨迹		
SAR 图像		
分辨率/m	0.32×0.24	0.25×0.25

4.2 加速比分析

对于 BP 成像算法而言,进行 SAR 成像处理的时间取决于距离及方位向点数、成像区域大小和成像网格大小(分辨率)。本节利用一段 C 波段机载 SAR 实测数据分别用该方案与常规 CPU 处理方式进行成像处理,并对处理消耗时间进行对比及分析。本节利用距离采样点数均为 131 072、方位采样点数为 2 048 的 SAR 实测数据进行成像处理。成像平台软硬件配置如表 2 所示。由于读取 SAR 回波数据与 SAR 图像写入操作所消耗时间对于两种实现方式相同,且不属于计算消耗时间,因此成像消耗时间并未将 SAR 回波数据的读操作和 SAR 图像的写操作所消耗时间计入。

表 2 成像平台的软硬件配置

硬件配置	CPU 型号(INTEL Q6600@2.40 GHz)
	内存大小(4 G DDR3 内存)
	显卡型号,显存大小(Geforce GT 740 M,2 G 显存)
软件配置	系统(windows7 旗舰版 64 位操作系统)
	VS 型号(Visual Studio 2010)
	CUDA 型号(CUDA 6.0)

利用距离采样点数为 131 072、方位采样点数为 2 048 的 SAR 实测数据,对成像区域大小为 600 m×600 m,成像间隔大小为 0.2 m×0.2 m 的区域成像,单核 CPU 处理所消耗时间为 12 642 s,GPU 处理所消耗时间为 84.584 s,加速比达到了近 150 倍。本文提出的成像方案对内存容量和显存容量均没有很高要求,因此选用笔记本电脑作为成像平台便足以满足成像硬件要求,使得 SAR 处理设备便携性大大提高。当选用拥有更强计算能力的 GPU 计算设备时,将会得到更快的成像处理速度。

4.3 实时性分析

基于 GPU 的 BP 成像算法每次只需读取 1 个方位时刻的 SAR 回波数据,进行成像处理得到该方位时刻的“1 帧”SAR 图像,并等待与下一方位时刻下“1 帧”SAR 图像相干叠加直至得到最终 SAR 图像,所以方案成像处理速度是否大于数据采集速度决定了方案是否满足实时性要求。基于 GPU 的 BP 成像算法处理速度与距离采样点数、脉冲发射频率(PRF)和成像区域处理点数相关,而与 SAR 系统波段、天线尺寸等因素无关。

利用距离采样点数为 131 072、方位采样点数为 2 048 的 SAR 实测数据,对有 3 000×3 000(9 M)个像素点的成

像区域进行成像时，PRF 在不超过 24 的情况可以使得成像处理速度大于数据采集速度，满足实时成像条件。但 PRF 不超过 24 的情况与实际 SAR 成像系统 800 的 PRF 相差很远，基于 NVIDIA GT 740 M 实现的 BP 成像算法对 9 M 个像素点很难实现实时成像。在此基础上，减少成像像素点数量或选用拥有更强计算能力的 GPU 计算设备、并增加 GPU 计算设备的个数，提高成像运算能力以达到实时成像的要求。

5 结　论

本文对基于 GPU 的 SAR 实时成像算法进行了深入研究。首先对 SAR 成像处理技术的频域算法和时域算法进行分析，选取了实现方便且适用于多模式 SAR 的 BP 成像算法作为研究对象，并对算法的 GPU 并行实现进行了论述。基于以上研究内容，提出了基于 GPU 的 BP 成像算法，针对 CUDA 编程模型与 BP 成像算法特点提出了优化加速方案。该方案利用 BP 成像算法数据的不同特征，与 CUDA 内存访问机制相结合，提高数据访问速度；然后针对方案中数据和任务的并行度，提出了多 GPU 并行处理技术和异步执行流处理技术，最大限度地利用计算资源；最后利用 GPU 硬件特点，使用 SFU 提供的特殊函数减少运算时钟周期，提高运算速度。使用本文方案，与传统 CPU 实现方式相比，拥有更高的成像效率。但是要实现该方案的实时成像，还需对成像区域大小有所限制或使用拥有更强计算能力的 GPU 计算设备，辅以多 GPU 进行实现。

参考文献

[1] 孟大地，胡玉新，石涛，等. 基于 NVIDIA GPU 的机载 SAR 实时成像处理算法 CUDA 设计与实现[J]. 雷达学报，2013(4)：481-491.

[2] 刘斌. 机载 SAR BP 算法成像的运动补偿及 GPU 并行化实现研究[D]. 成都：电子科技大学，2013.

[3] 班阳阳，张劲东，陈家瑞，等. 后向投影成像算法的 GPU 优化方法研究[J]. 雷达科学与技术，2014(6)：659-665.

[4] 姜晓龙，王建，宋千，等. 基于 GPU 的后向投影 SAR 成像算法[J]. 雷达科学与技术，2014，(4)：350-357.

[5] 桑德斯. GPU 高性能编程 CUDA 实战[M]. 北京：机械工业出版社，2011.

[6] 邹岩，杨志义，张凯龙. CUDA 并行程序的内存访问优化技术研究[J]. 计算机测量与控制，2015，17(12)：2504-2506.

[7] SHI J，MA L，ZHANG X. Streaming BP for non-linear motion compensation SAR imaging based on GPU[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing，2013，6(4)：2035-2050.

[8] HU K，ZHANG X，WU W，et al. Three GPU-based parallel schemes for sar back projection imaging algorithm [C]. 2014 IEEE 17th International Conference on Computational Science and Engineering (CSE)，IEEE Computer Society，2014：324-328.

[9] MOON K，LONG D G. A new factorized backprojection algorithm for stripmap synthetic aperture radar[J]. Positioning，2013(4)：42-56.

[10] 刘百玲，江海清，倪书爱. 基于 GPU 的 ISAR 成像算法实现[J]. 电子测量技术，2015，38(8)：76-78.

[11] 吴铮，张磊，李宁. 基于 GPU 的机载高分 SAR 运动补偿和自聚焦[J]. 国外电子测量技术，2015，34(8)：94-99.

[12] 庄晋升，汪丙南，向茂生. MEMS IMU 随机误差建模在 SAR 运动补偿中的应用[J]. 国外电子测量技术，2015，34(10)：88-94.

[13] 查正兴，鲁昌华，陶志颖，等. 增强型 Shearlet 域 SAR 图像去噪[J]. 电子测量与仪器学报，2014，28(6)：644-649.

作者简介

翟新刚，1990 年出生，2013 年毕业于吉林大学获工学学士学位，2013 年起至今就读于中国科学院电子学研究所，研究方向为 GPU 并行实现机载合成孔径雷达的实时成像。

E-mail：zhaixingang@163.com

韦立登，1973 年出生，副研究员，目前主要从事信号与信息处理方面的研究。

E-mail：wld@mail.ie.ac.cn

汪丙南，1984 年出生，助理研究员，主要研究方向为干涉合成孔径雷达信号仿真和处理方法。

E-mail：wbn@mail.ie.ac.cn

向茂生，1964 年出生，研究员，博士生导师，主要研究方向为干涉合成孔径雷达系统技术和方法。

E-mail：xms@mai.ie.ac.cn