

文章编号: 1672-2892(2012)05-0525-04

基于 GPU 的软件化雷达恒虚警概率算法实现

贾 可, 李世丹, 郭 燕, 王德生

(清华大学 电子工程系, 北京 100084)

摘 要: 高效实现恒虚警概率检测(CFAR)是新型雷达终端信号处理系统研制的重要部分。在基于图形处理器(GPU)的软件化雷达终端架构下, 采用统一计算设备架构(CUDA)技术, 并根据 GPU 的特点对算法实现进行优化, 实现了高效的软件化 CFAR 算法, 相比 CPU 软件实现大大缩短了数据处理时间, 能够满足雷达信号处理对实时性的需求, 同时验证了研发基于 GPU 的软件化雷达终端具有较高的可行性。

关键词: 图形处理器; 计算设备架构; 恒虚警概率检测; 雷达信号处理; 雷达终端

中图分类号: TN957.51

文献标识码: A

Software radar Constant False-Alarm Rate algorithm implementation based on GPU

JIA Ke, LI Shi-dan, GUO Yan, WANG De-sheng

(Department of Electronic Engineering, Tsinghua University, Beijing 100084, China)

Abstract: Efficient implementation of Constant False-Alarm Rate(CFAR) detector plays an important role in the development of signal processing system of new radar terminal. Under the Graphic Processing Unit(GPU) based software radar terminal architecture, this paper optimizes the CFAR algorithm implementation on GPU by using Compute Unified Device Architecture(CUDA) technology, which cuts data processing time tremendously compared to CPU implementation. The requirements of real-time for radar signal processing can then be satisfied, and the feasibility of the development of GPU based software radar terminal is verified.

Key words: Graphic Processing Unit; Compute Unified Device Architecture; Constant False-Alarm Rate; radar signal processing; radar terminal system

雷达信号处理是雷达终端的重要组成部分, 需要对大量原始采样的雷达回波数据进行脉冲压缩、杂波抑制、恒虚警概率(CFAR)检测等基本处理。以舰载导航雷达为例, 在 75 MHz 采样率, 量程 40 n mile 的情况下, 单条回波的采样数据为 36 Kbyte, 仅就恒虚警检测算法来说, 实现窗长为 32 的单元平均恒虚警(Cell Average-CFAR, CA-CFAR)检测器, 每秒数据计算量可达 20 亿次, 为满足实时性, 对信号处理机的 I/O 带宽及运算能力提出了很高的要求。传统雷达信号处理主要基于 FPGA 与 DSP 来实现^[1], 随着计算机运算能力、总线带宽的持续提高与加强, 雷达终端信号处理已可以在通用计算机上完全采用软件实现^[2]。采用纯软件实现的雷达终端设备的开发、生产、装备和维护都可得到极大简化, 有利于实现雷达终端系统设备的开发、升级、维护成本的最低化^[3]。为满足实时图像运算、高清晰度显示的市场需求而诞生的图形处理器(GPU), 已经逐渐发展成为高并行度, 多线程, 具有强大计算能力和极高存储器带宽的多核处理器^[4]。GPU 是专为密集、高并行度计算(如图像渲染)而设计的, 其能力更多用在数据处理而不是数据缓存和流程控制, 其浮点数运算和存储器带宽已远远超过 CPU。统一计算设备架构(CUDA)是一种基于 GPU 的通用并行计算架构, 它对图形硬件和应用程序接口进行封装, 使开发人员使用 C 语言对 GPU 进行编程, 开发 GPU 通用计算系统^[4-5]。以 GPU 为平台, 采用 CUDA 技术, 实现终端雷达信号处理系统, 能够满足计算能力和实时性的要求。本文以 CA-CFAR 的实现为例, 并与 CPU 实现性能进行比较, 验证了基于 GPU 的雷达信号处理的高实时性。

1 基于 GPU 的软件化雷达终端架构

基于 GPU 的软件化雷达终端主要由运行 GPU 信号处理及显示软件的通用计算机和雷达信息采集板构成。雷达信息采集板主要包括 AD 与 FPGA 这 2 部分, AD 部分对雷达信号进行采样, FPGA 完成打包、缓存等逻辑功能。来自雷达前端的回波及脉冲等信息输入雷达信息采集板, 采样后经预处理, 以信息帧包的形式, 通过总线接口传入宿主计算机^[3]。宿主计算机上运行着基于 GPU 的雷达信号处理及显示软件, 处理后得到的目标点迹数据与雷达一次视频输入显示器, 进行叠加显示^[6], 同时点迹数据还将通过数据接口送入存储设备及其他设备, 以进行记录或者进一步的数据处理, 其架构见图 1。

基于 GPU 的软件化雷达终端相比传统雷达终端结构更简单, 除去显示器等外设, 整套系统的硬件只由 1 块雷达信息采集板和 1 台通用计算机构成, 信号处理及显示完全由终端计算机上基于 GPU 的软件来实现。这种结构改变了传统雷达终端的开发、升级和维护方式, 新型号的开发和升级将以软件开发为主, 极大地提高了开发的灵活性, 缩短开发周期; 同时宿主计算机可方便地进行更新换代, 提高性能的同时对整个终端系统不会产生不良影响, 系统硬件升级变得非常简单可靠, 最终实现雷达终端系统开发、升级、维护的成本最低化^[3]。

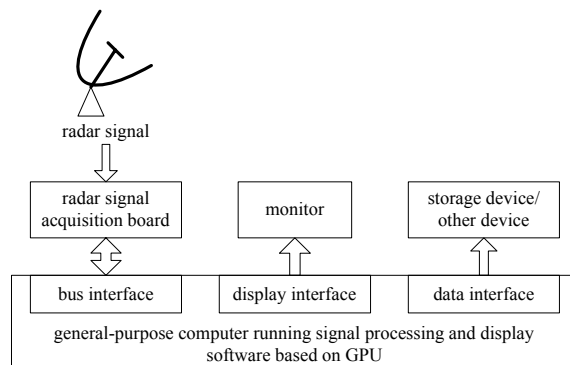


Fig.1 Software radar terminal architecture based on GPU

图 1 基于 GPU 的软件化雷达终端架构

2 CA-CFAR 基本算法

恒虚警概率(CFAR)检测器在干扰环境下对目标临近单元的采样数据进行实时估计并调整检测门限, 使雷达接收机能够保持恒定的预定虚警率 P_{fa} 。CFAR 的主要形式是 CA-CFAR^[7], CA-CFAR 检测器见图 2。

CA-CFAR 的自适应门限值由杂波和雷达回波附近的噪声决定。对位于待检单元(Cell Under Test, CUT)两边的参考单元(CFAR window cells)环境回波信号进行采样, 将测试单元的输出与由参考单元输出总

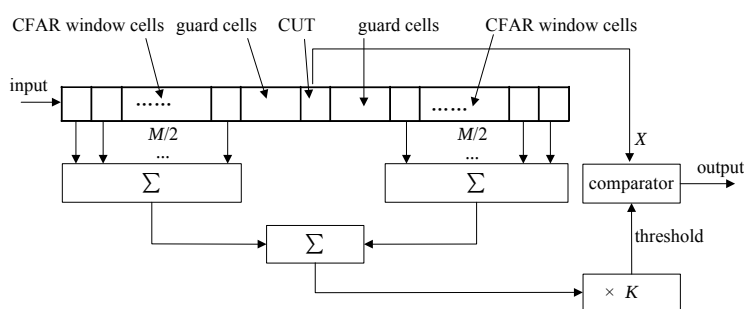


Fig.2 CA-CFAR algorithm^[8]

图 2 CA-CFAR 算法^[8]

和所得到的自适应门限相比较并进行判决。由于目标可能跨越多个距离单元, 将紧临 CUT 的单元设为保护单元(Guard Cells), 其采样不参与门限的计算。虚警概率 P_{fa} 的表达式为^[8]: $P_{fa} = \frac{1}{(1+K_0)^M}$ 。

3 基于 GPU 的 CFAR 算法实现

基于 CUDA 编程模型的程序由 2 部分构成, 在 CPU(称为 HOST 端)上运行的串行计算及流程控制代码, 在 GPU(称为 DEVICE 端)上运行的并行计算代码。DEVICE 端可开启极多的轻量级线程, 其运行的代码称为 Kernel, 每个线程执行一个 Kernel, 所有线程同时运行相同的 Kernel 并根据各自的线程 ID 接入不同的内存地址进行运算, 从而达到并行运算的目的。CUDA 模型采用一种可扩展的线程组织结构, 多个线程构成 1 个 Block, 多个 Block 构成 1 个 Grid, Block 内的线程是并行的, 而 Grid 内的 Block 则根据硬件进行调度执行, 这种可扩展性使得已编译好的 CUDA 程序能够在任意核心的 GPU 上执行^[4]。HOST 端与 DEVICE 端分别拥有独立的存储空间, DEVICE 端的显存(Global Memory)由 HOST 端代码进行管理, 包括分配、释放以及数据在 HOST 端与 DEVICE 端之间的复制等操作。

1 个 CUDA 程序的执行由 HOST 代码开始, 首先完成初始化、数据输入等工作, 随后对 DEVICE 端显存进行分配, 将内存中数据传入显存, 调用 DEVICE 端 Kernel 进行计算, 并将结果数据从显存传回内存中, 然后释放显存, 最后输出计算结果^[4]。

利用 CUDA 实现 CA-CFAR, 回波中的每 1 个数据均会被多次访问, 频繁地访问显存数据将造成极大延时, 为减少数据访问延时可使用 DEVICE 端的 Shared Memory。Shared Memory 是 GPU 的片上存储器, 其访问速度远快于显存, 在线程间不存在存储体冲突(Bank Conflicts)情况下, 其访问速度与寄存器一样快。对于需要频繁读数据的 Kernel, 将显存中的数据先读入 Shared Memory 中, 之后只访问 Shared Memory, 将带来极大的效率提升^[9]。利用 CUDA 实现 CA-CFAR 的算法流程见图 3。

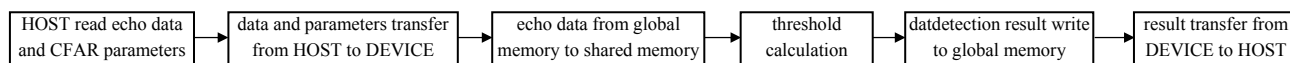


Fig.3 CA-CFAR algorithm implementation process using CUDA

图 3 CUDA 实现 CA-CFAR 的算法流程

计算门限值的直观做法是每个线程计算 1 个门限值, 将每个 CUT 两侧参考单元依次相加, 这样每个门限值的计算需要窗长 M 次加法。因 M 的取值一般为 2 的幂次, 对其进行进一步并行化处理, 每个线程仍计算 1 个门限值, 算法见图 4, 图中自左向右进行运算, 原始数据为 $X[k]$, 计算得到各门限值为 $Y[k]$, 箭头表示将两数值相加, 图中 CFAR 算法窗长 M 为 8。

不考虑数据起始与结束部分的补零, 对于第 n 次相加 ($1 \leq n \leq \log_2 M - 1$), 将每个数据 $X[k]$ 与 $X[k+2^{n-1}]$ 相加, 第 $\log_2 M$ 次相加须将每个待测单元跳过, 于是将每个数据 $X[k]$ 与 $X[k+2^{\log_2 M-1}+1]$ 相加, 这样得到的最终门限值 $Y[k]$ 即为原始数据 $X[k+2^{n-1}]$ 两侧单元相加得到的自适应检测门限值。改进后的算法在充分利用 GPU 并行性的同时减少了计算量, 实现每个门限值的计算量为 $\log_2 M$ 次加法。

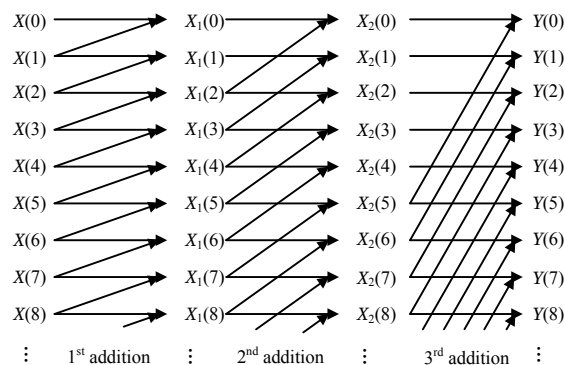


Fig.4 Optimized CA-CFAR threshold calculation method for GPU

图 4 针对 GPU 改进的 CA-CFAR 门限计算方法

4 结果及分析

在 GPU 与 CPU 上分别实现 CFAR 算法, 在确保运算结果相同的情况下, 对两者的运算时间进行对比, 以分析 GPU 算法实现的效率。开发计算机采用的显卡 GPU 为 nVIDIA GeForce9500GT, CPU 为 Intel Core2 Quad Q9550 2.83 GHz, 操作系统采用 WindowsXP SP3。门限值的计算主要为数据的累加, GPU 对整型与浮点型的加法运算能力相当, 故 GPU 与 CPU 处理的数据均为整型数据。

CPU 的 CFAR 算法实现采用滑窗法, 对每个单元 CFAR 门限的计算在前 1 个单元 CFAR 门限的基础上进行 1 个单元滑动后的加减, 这种实现方法以降低并行化的代价极大减少了计算量, 在 CPU 上具有最高的运算效率。

对 GPU 运算时间的测量采用 CUDA 的事件函数, 通过创建和销毁事件并记录时标来测量程序耗时, 其精确度可达 $0.5 \mu s$ ^[9]; 对 CPU 运算时间的测量采用 Windows 系统的高精确度计时器 QueryPerformanceFrequency() 函数和 QueryPerformanceCounter()^[10], 其精确度与系统和硬件都有关, 为微秒级。

在窗长为 32 的情况下, 两者分别对不同数据长度的单个主脉冲回波数据进行 10 000 次 CFAR 运算, 单次运算的平均运算时间对比见表 1, 对 GPU 时间的测量将数据传入传出的耗时与运算耗时均计算在内。由表 1 可以看到, 用 GPU 进行 CFAR 运算的计算用时要明显低于 CPU, 且随着数据量的增大, CPU 运算时间呈线性增长, 而 GPU 由于并行计算的特点, 运算时间增长较少, 在数据量达到 80 Kbyte 时运算效率接近 CPU 的 6 倍。

5 结论

本文针对雷达信号处理计算量大, 实时性要求高的特点, 在基于 GPU 的软件化雷达终端架构下, 利用 CUDA 技术实现了雷达信号处理软件的恒虚警检测部分, 实验结果表明其实现效率相比 CPU 有了很大提高, 证明了研发基于 GPU 的软件化雷达终端具有较高的可行性。

(下转第 533 页)

表 1 CPU 与 GPU 运算时间对比

Table1 Comparison of calculation time between CPU and GPU

data length/Kbyte	calculation time of CPU/ms	calculation time of GPU/ms
10	0.084 129	0.050 782
20	0.167 529	0.059 093
40	0.336 863	0.084 477
80	0.673 043	0.118 601