

基于 OpenCL 的 DRR 算法优化研究

田林琳, 李 莹

(沈阳工学院 信息与控制学院, 辽宁 抚顺 113122)

摘 要: 放射治疗计划系统 (TPS) 是为放疗患者制定放疗计划的特殊系统, 对放射治疗的精度和效果有重要影响, 是放射治疗的核心技术之一。数字重建放射影像算法 (DRR) 是 TPS 中的关键算法, 广泛用于实现射野验证、病人摆位等。针对 DRR 算法性能不能满足交互式 and 实时性的要求, 提出了一种使用 OpenCL 技术对 DRR 算法进行并行加速的计算方法。首先介绍了 DRR 算法在 TPS 系统中的重要性和 OpenCL 框架, 接着在 DRR 算法的优化过程中以 X 射线衰减理论公式为基础, 构建了 DRR 算法的串行版本作为算法优化的基准, 分析了基于光线跟踪的 DRR 算法的特点, 给出了算法的并行化方案, 并结合 OpenCL 的存储器特性对并行算法进行了性能调优。在 NVIDIA 平台对算法进行了实验和评测, 结果显示, OpenCL 并行优化版本相对其串行版本加速约 36 倍左右, 满足了系统的性能要求。

关键词: OpenCL; 放射治疗计划系统; 数字重建放射影像; 光线跟踪; 并行化

中图分类号: TP301.6

文献标识码: A

文章编号: 1673-629X(2018)04-0165-04

doi:10.3969/j.issn.1673-629X.2018.04.035

Research on DRR Algorithm Optimization Based on OpenCL

TIAN Lin-lin, LI Ying

(School of Information and Control, Shenyang Institute of Technology, Fushun 113122, China)

Abstract: Radiotherapy treatment planning system (TPS), as one of core technologies of radiotherapy, is a special system for radiotherapy patients, which has important influence on the accuracy and effectiveness of radiotherapy. Digitally reconstructed radiograph (DRR) is a key algorithm in TPS and widely used in field validation, patient placement and so on. Concerning the performance of DRR can't meet the interactive and real-time requirement, we propose a new approach to accelerate the DRR using open computing language (OpenCL). Firstly, we introduce the importance of DRR in TPS and OpenCL framework. Then in the optimization, according to the formula of X-ray attenuation, we develop the serial version of the DRR as the benchmark of optimization, and analyze the characteristics of DRR based on ray tracing. Finally the parallel scheme about the DRR is established and done performance tuning according to OpenCL memory architecture. The experiment and test on NVIDIA platform show that the parallel optimization of OpenCL is about 36 times faster than its serial version, which can meet the requirements of the system performance.

Key words: OpenCL; TPS; DRR; ray tracing; parallelization

1 概 述

放射治疗计划系统 (TPS) 是利用计算机软件技术、剂量计算方法、控制手段等, 解决治疗过程中的精确定位、剂量计算的快速精确、保证目标剂量的准确实施以及治疗计划的优化等问题^[1]。数字重建放射影像 (DRR) 是 TPS 系统中的关键功能, 是通过模拟 X 光线的衰减过程, 从 CT 断层数据中生成类似 X 光片的成像效果, 这对于放疗过程中的质量控制具有非常重要的意义^[2]。DRR 可在任意三维空间角度下实现对病患的“透视”模拟, 能够随意观察放射靶区、特定组织

或器官, 或靶区与周围器官间的空间关系, 因为没有治疗床的限制, 可利用 DRR 得到模拟定位机难以拍摄的图像。同时省略了使用模拟放射摄影图像的成像步骤, 既节省了临床费用, 也减少了对病患的辐射^[3]。

在 TPS 的临床应用中, DRR 具有重要作用, 为了达到临床应用的目标, 要求 DRR 能够准确、快速地反映放射治疗中的真实情况, 并能够满足动态性、交互性的需要^[4]。这也是三维图像可视化领域中, 人们一直试图解决的两个问题: 一是实时地进行图像绘制, 在 DRR 的实际使用中, 要求可以快速、多角度地生成图

收稿日期: 2017-05-04

修回日期: 2017-09-06

网络出版时间: 2017-12-05

基金项目: 国家自然科学基金 (61603262)

作者简介: 田林琳 (1981-), 女, 副教授, CCF 会员 (75539M), 研究方向为虚拟现实、并行计算、高性能计算。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20171205.1434.110.html>

像;二是真实地显示结果,这要求计算结果能够真实、客观地反映其解剖结构在三维空间上的细节与特征^[5]。

典型的 DRR 算法是以光线跟踪算法为基础,利用虚拟摄影机视图追踪光线至投影图像中的每一个像素,每一条光线所经过的体元数据进行累加,累加值与像素亮度值成正比,使用这种方法实现重建出的 DRR 图像显示精度和图像质量都非常好,满足了显示结果真实性的要求^[6-7]。但由于此算法需要跟踪每条从虚拟光源发出的光线,涉及到大量的光线与 CT 三维体数据进行求交测试的运算,运算量非常大。传统的基于 CPU 技术的实现方式通常不能满足交互式 DRR 算法的需求,随着 GPU (graphic processing unit) 在并行计算能力、存储容量和可编程能力等方面的发展,使 DRR 算法的性能提升成为可能^[8]。

英伟达 (NVIDIA) 公司于 2007 年提出的 CUDA 架构 (compute unified device architecture) 使得开发者能够以 GPU 的强大并行计算能力为基础,建立一种更快更高效的数据计算解决方案。但 CUDA 架构的一个最主要问题是通用性,人们只能使用 NVIDIA 系列产品对 GPU 进行开发。而作为一个软件系统,TPS 是运行在医院提供的计算机上,各医院的硬件环境千差万别,所以使用 CUDA 加速的局限性比较高。而基于异构平台的 OpenCL 通用计算规范,允许开发者以相同的代码在任何支持该规范的平台运行,包括 NVIDIA 和 AMD 显卡,甚至包括支持 OpenCL 的 CPU,这大大降低了程序对硬件平台的依赖性,具有较强的理论意义和实践价值。

使用 OpenCL 对 TPS 系统中的 DRR 算法进行并行加速,主要出于三个目的:

(1) 通过对 DRR 算法的加速,满足算法交互性和实时性的需要,这不仅能减少患者的等待时间,也提高了 TPS 的使用效率。

(2) OpenCL 作为通用计算标准,与 CUDA 架构相比,其性能缺乏评判,通过使用 OpenCL 技术在 NVIDIA 显卡上实现对 DRR 算法的加速,软件人员可以更好地审视其性能,以便于在今后的开发中进行平台选择。

(3) 通过将数据存储在不同的硬件存储器上的性能对比,开发人员可以深入了解不同存储器在并行优化中的影响,便于在使用 OpenCL 技术时进行数据的合理布局。

2 OpenCL 概述

OpenCL 是一个可以在异构平台上进行并行编程的开放框架标准,包括一组底层程序接口和一个高效、

快速、可移植的抽象层,为开发者提供了一个有效的并行开发环境、一组与平台无关的工具和丰富的中间层。

OpenCL 框架运行时使用主机端程序对所有支持 OpenCL 的设备进行统一管理。对于每个支持 OpenCL 规范的计算设备,其在内部又进一步划分为多个计算单元 (compute unit), 每个计算单元又划分为多个处理单元 (processing element), 每个处理单元以 SIMD 或 SPMD 的方式运行^[9]。

为了支持代码的可移植性,OpenCL 定义了一个抽象的内存模型,开发者可以根据该模型编写代码,硬件厂商可以将该模型映射到他们自己实际的硬件上^[10-11]。OpenCL 定义的内存空间如图 1 所示,这些内存空间与 OpenCL 程序密切相关。

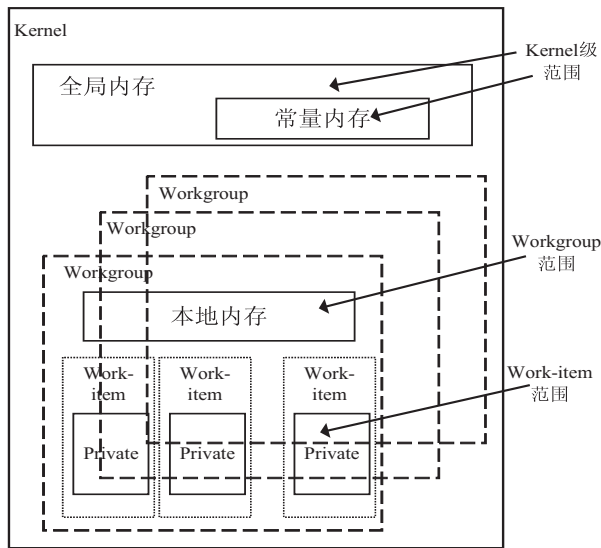


图 1 OpenCL 内存模型

OpenCL 内存模型按照访问权限划分,依次分为全局内存和常量内存 (Kernel 范围)、本地内存 (Workgroup 范围)、私有内存 (Work-item 范围)。其中纹理内存和常量内存的数据位于全局内存,但其拥有高速缓存^[12]。

3 DRR 算法的优化过程

3.1 DRR 算法原理

DRR 图像是通过模拟 X 光线衰减过程生成的,X 光线穿过的组织不同,被吸收的剂量也不同,所以采用 X 光线衰减模型进行模拟,有如下公式:

$$I = I_0 \times e^{-\int_0^L \mu(x) dx} \quad (1)$$

其中, I_0 为 X 射线的初始强度; μ_i 为组织 i 的线性衰减系数; L 为 X 射线的长度。

式 1 的离散形式如式 2 所示:

$$I = I_0 \times e^{-\sum \mu_i l_i} \quad (2)$$

其中, I_0 和 μ_i 同式 1; l_i 表示 X 射线穿过组织 i 的距离。式 2 为生成 DRR 图像的关键公式^[13]。

3.2 编写串行程序

编写一个能够正确运行、没有存储器资源泄露的 CPU 串行程序,作为并行优化的正确性和性能测试的基准。这样做的原因是:(1)由于在目前的 OpenCL 版本上,使用 C++ 只能实现一些 OpenCL 编程中主机部分的函数,包括 OpenCL 设备初始化、CPU 和 GPU 之间的数据通信、注销 OpenCL 计算环境等,而真正在 GPU 上执行的 kernel 函数和 device 函数都是以 C 语言函数的形式编写的,所以首先把 TPS 系统中 DRR 算法部分由 C++ 程序转换为 C 程序。(2)OpenCL device 上运行的代码不易调试,所以在串行程序中去除不易并行的因素,使得程序中的循环或者迭代之间没有或者较少的数据依赖,这会大大减少并行程序出错的概率,因此接下来在第 1 步串行程序的基础上改写以符合 OpenCL 的并行操作,例如函数中的静态变量和全局变量在程序运行过程中,会被多个线程同时访问,但其值一直保持不变,会使并行程序中线程之间存在依赖性,不利于程序的并行化,所以根据算法的实现过程改造为局部变量。还有将程序中代表 CT 体数据的数据指针由二级指针修改为一级指针,这样 CT 体数据成为一个连续的内存数据块,方便了数据从 CPU 端向设备端的复制和在绑定纹理时对体数据内存排列的要求。

3.3 确定并行化方案

支持 OpenCL 计算的 GPU 通常具有数个多处理器,每个多处理器都具有 SIMD 架构,支持在同一时钟周期内处理不同的数据,这使得 OpenCL 主要针对基于数据的并行计算方式^[14]。光线追踪算法中每一条光线的计算过程都是相互独立的,不同光线之间没有依赖关系,因此可以方便地进行并行化,只需将每一条光线设计成一个独立的计算线程就能实现基于光线追踪的 DRR 算法并行化。这是利用 OpenCL 实现基于光线追踪的 DRR 算法的可能性。图 2 是 DRR 算法的并行化方案。

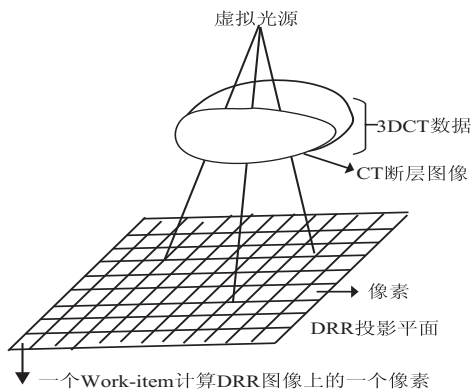


图2 DRR 算法的并行化方案

使用 OpenCL 进行并行程序设计,其中一个主要

特点是针对不同的存储器类型进行数据存储。GPU 中有全局内存、常量内存、纹理内存和共享内存。DRR 算法的输入数据 CT 体数据少则 80 ~ 150 张,多则 200 ~ 400 张甚至更多。以 150 张 CT 图像为例,其需要的内存为 $150 * 0.5 \text{ MB} = 75 \text{ MB}$,所以只有全局内存和纹理内存可供选择。纹理内存缓存在芯片上,因此它能够减少对内存的请求并提供更高效的内存带宽^[15]。纹理缓存是专门为那些在内存访问模式中存在大量空间局部性 (spatial locality) 的图形应用程序而设计的^[16]。文中对 CT 体数据存储在全局内存和纹理内存的性能进行了对比。

常量内存中的数据位于全局内存,但拥有局部缓存加速,常量内存的空间较小(只有 64 K),因此将虚拟光源的点坐标、CT 图像的层厚、图像大小、像素大小等在计算过程中恒定不变的少量参数放在常量内存中。

对于不同 CT 值数据的衰减表,大小是 4 096,类型是单精度浮点,可以选择将数据存储到全局内存、纹理内存或者常量内存,文中也对三种情况下的程序性能进行了对比。

3.4 并行化实现步骤

OpenCL 实现 DRR 算法的步骤主要有:

(1)调用 `clGetPlatformIDs` 获得可用的平台,设置使用 OpenCL 设备;

(2)调用 `clCreateContext` 在指定平台上创建上下文;

(3)调用 `clCreateCommandQueue` 函数,在已经创建好的上下文上创建命令队列;

(4)在 GPU 上分配三维 image 对象,并将 CT 体数据和 CT 衰减表从 CPU 读取至设备的全局内存或者纹理内存中;

(5)在 GPU 上为算法的其他参数分配设备内存或者常量设备内存,如虚拟光源的点坐标、CT 图像的层厚、图像大小、像素大小等,并将 CPU 端的数据复制到对应的设备内存中;

(6)创建一个网格,并且设置好 workgroup 和 workitem 的大小;

(7)创建内核对象,使用 `clCreateKernel` 创建一个 kernel 对象,调用 `clEnqueueNDRangeKernel` 启动 kernel 函数;

(8)kernel 函数执行,先判断当前的 workitem 要处理的像素是否在 DRR 图像范围内,如果不是,则不进行计算直接返回;否则执行 DRR 算法内核函数,在内核函数的执行过程中更新 DRR 图像关联的全局设备内存;

(9)kernel 函数执行完毕,把更新后全局内存数据

复制回 CPU 内存;

(10)在屏幕上描画 CPU 内存上的结果;

(11)释放 OpenCL 设备上的空间以及 CPU 上的内存。

3.5 并行参数设置

基于 OpenCL 的 DRR 算法中,用一个线程来计算 DRR 图像中一个像素的值,如 DRR 图像的分辨率为 $m * n$,则在一个 kernel 中就包括使用 $m * n$ 个线程,每个 workgroup 中的 workitem 设置为 $32 * 16$,则一个 kernel 中的 workgroup 数目为 $(m/32) * (n/16)$ 。

4 实验

4.1 实验环境

针对相同的算法复杂度,将 CT 体数据分别存储在全局内存和纹理内存并分别与串行程序进行了对比,另外对 CT 衰减表分别存储于全局内存、纹理内存和常数内存并与串行程序进行了对比,以评估 OpenCL 将数据存储在不同存储器上的性能。

使用的平台如下:

GPU:NVIDIA GeForce GTX760 2 G 显存;

CPU: Intel i5 4590 4 核 4 线程 3.3 GHz;

内存:DDR3 1 600 MHz 8 G。

4.2 结果分析

文中分别对分辨率为 $2\ 048 * 2\ 048$ 和 $1\ 024 * 1\ 024$ 的 DDR 图像进行了 CPU 和 GPU 的计算结果和性能对比,使用 GPU 运算的结果和 CPU 相比,运行结果误差是 $1e-2$,结果可以接受,如图 3 所示。

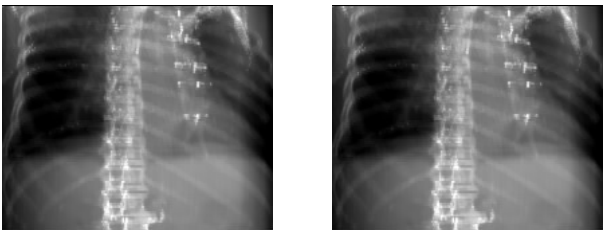


图 3 CPU 和 GPU 的运算结果对比

性能对比如表 1~3 所示。

表 1 体数据在全局内存时的性能对比

性能	2 048 * 2 048	1 024 * 1 024
CPU/s	12.182	4.215
GPU/s	1.674	0.412
时间比	7.277	10.231

表 2 体数据在纹理内存时的性能对比

性能	2 048 * 2 048	1 024 * 1 024
CPU/s	12.347	4.262
GPU/s	0.351	0.108
时间比	36.177	39.463

表 3 CT 衰减表在不同存储器上的性能对比

性能	2 048 * 2 048	1 024 * 1 024
CPU/s	12.274	4.301
全局内存/s	1.477	0.429
常数内存/s	0.983	0.328
纹理内存/s	0.362	0.112

由表 1~3 中的对比数据可见:经过 OpenCL 优化的 DRR 算法速度比 CPU 串行版本明显快很多;在 OpenCL 编程中使用纹理内存可以大大提高程序的加速比;在相同条件下,纹理内存的速度略大于常数内存,并且远大于全局内存;OpenCL 并程序序和串行程序的加速比与基于 CUDA 的 DRR 图像生成算法相近^[13]。

5 结束语

通过利用 DRR 算法的可并行性,使用 OpenCL 技术对基于光线追踪的 DRR 算法进行优化,并在 GPU 上进行了并行算法调优;通过对 OpenCL 中的各种存储器的优化使用,显著提高了程序的运行效率。目前,OpenCL 的使用仍然有很多局限,在诸如复杂的分支和逻辑判断等方面仍然不如 CPU 灵活,但随着 GPU 的快速发展,相信 OpenCL 的应用范围将更加广阔。

参考文献:

[1] 闫 锋.数字重建放射影像生成方法及其应用研究[D].合肥:合肥工业大学,2010.

[2] 戴建荣,胡逸民.图像引导放疗的实现方式[J].中华放射肿瘤学杂志,2006,15(2):132-135.

[3] 吴宜灿,李国丽,陶声祥,等.精确放射治疗系统 ARTS 的研究与发展[J].中国医学物理学杂志,2005,22(6):683-690.

[4] FERNANDO R,KILGARD M J. The Cg tutorial;the definitive guide to programmable real-time graphics[M]. [s. l.]: Addison-Wesley,2003.

[5] 汪 俊,吴章文,张从华,等.基于射线追踪的数字重建影像增强技术[J].生物医学工程学杂志,2005,22(1):125-128.

[6] HUANG Jianbing,CARTER M B. Interactive transparency rendering for large CAD models[J]. IEEE Transactions on Visualization and Computer Graphics, 2005, 11(5):584-595.

[7] LEVORY M. Volume rendering by adaptive refinement[J]. Visual Computer,1990,6(1):2-7.

[8] RUSSAKOFF D B,ROHLFING T,MORI K,et al. Fast generation of digitally reconstructed radiographs using attenuation fields with application to 2D-3D image registration[J]. IEEE Transactions on Medical Imaging,2005,24(11):1441-1454.

(下转第 173 页)

dows PC、Mac PC、Windows Web、IOS 或者 Android 等平台的文件格式,实现跨平台操作。文中发布为 Windows PC 的可执行文件“TBM_display.exe”。



图6 沉浸式TBM虚拟仿真系统

5 结束语

利用虚拟现实技术构建了TBM系统组成和工作过程演示的仿真系统,通过人机交互的方式,人员可以在三维空间中观察装备各个子系统的结构,对其中机构的设计及其工作原理有直观的理解;同时,由于对模型进行了渲染并添加了碰撞等物理场效果,使得仿真效果更加真实;最后,通过Unity SteamVR 插件将Unity 3D 中的虚拟场景内容和 HTC Vive 硬件设备相结合,开发了具有沉浸感和交互性的虚拟现实仿真系统,对TBM的设计、方案评估和施工可视化有积极的推进作用。

参考文献:

[1] 荆留杰,张娜,杨晨. TBM 及其施工技术在中国的发展与趋势[J]. 隧道建设,2016,36(3):331-337.

[2] 王梦恕. 中国盾构和掘进机隧道技术现状、存在的问题及发展思路[J]. 隧道建设,2014,34(3):179-187.

[3] 张友湖,夏超群. 基于 CATIA 的 TBM 主箱梁的数字建模与有限元分析[J]. 中国重型装备,2008(1):23-25.

[4] 王子昱,唐泽圣,王田苗,等. 基于虚拟现实的计算机辅助立体定向神经外科手术系统[J]. 计算机学报,2000,23(9):931-937.

[5] 刘相,刘玉庆,朱秀庆,等. 基于虚拟现实的航天员舱内导航训练方法[J]. 计算机辅助设计与图形学学报,2017,29(1):101-107.

[6] VAUGHAN N, GABRYS B, DUBEY V N. An overview of self-adaptive technologies within virtual reality training[J]. Computer Science Review, 2016, 22: 65-87.

[7] MARTIN-GUTIERREZ J, MORA C E, ANORBE B, et al. Virtual technologies trends in education[J]. Eurasia Journal of Mathematics Science and Technology Education, 2017, 13(2): 469-486.

[8] 张林鎰,辛献杰,崔冰,等. 面向汽车产品设计的虚拟现实服务平台研究[J]. 系统仿真学报,2014,26(10):2407-2411.

[9] 王大虎,刘海洋,王敬冲. 基于虚拟现实的采煤机培训系统开发[J]. 计算机仿真,2015,32(6):262-265.

[10] 蔡林沁,张优东,杨卓,等. 基于多智能体的井下安全事故虚拟现实仿真[J]. 系统仿真学报,2014,26(12):2914-2920.

[11] TORRES F. A learning evaluation for an immersive virtual laboratory for technical training applied into a welding workshop[J]. Eurasia Journal of Mathematics Science and Technology Education, 2017, 13(2): 521-532.

[12] 栾飞. 基于 Unity3D 的液压传动虚拟仿真教学系统开发[D]. 济南:山东建筑大学,2015.

[13] 于潇翔,彭月橙,黄心渊. 基于 Unity 3D 的道具系统研究与开发[J]. 成都理工大学学报:自然科学版,2014,41(4):523-528.

[14] 刘国柱. Unity3D/2D 游戏开发从 0 到 1[M]. 北京:电子工业出版社,2015.

[15] BUN P, GORSKI F, GRAJEWSKI D, et al. Low-cost devices used in virtual reality exposure therapy[J]. Procedia Computer Science, 2017, 104: 445-451.

[16] 杜晓刚,党建武,王阳萍. 基于 CUDA 的数字重建影像生成算法[J]. 计算机科学,2015,42(2):301-305.

[17] 刘磊. 基于 GPU 的医学图像三维重建及可视化技术研究[D]. 广州:南方医科大学,2008.

[18] 刘鹏,高军,雷勋祖,等. 一种基于光场的数字重建影像快速生成算法[J]. 南方医科大学学报,2007,27(10):1537-1539.

[19] NAGY Z, KLEIN R. Depth-peeling for texture-based volume rendering[C]//Proceedings of the 11th Pacific conference on computer graphics and applications. Washington, DC, USA: IEEE Computer Society, 2003: 429-433.

[20] 李森,李新亮,王龙,等. 基于 OpenCL 的并行方腔流加速性能分析[J]. 计算机应用研究,2011,28(4):1401-1403.

[21] 贾海鹏,张云泉,龙国平,等. 基于 OpenCL 的拉普拉斯图像增强算法优化研究[J]. 计算机科学,2012,39(5):271-277.

[22] NVIDIA Corporation. NVIDIA OpenCL JumpStart guide[M]. America: NVIDIA, 2009.

[23] MUNSHI A. The OpenCL specification[S]. America: Khronos OpenCL Working Group, 2009.

(上接第 168 页)

[9] 李森,李新亮,王龙,等. 基于 OpenCL 的并行方腔流加速性能分析[J]. 计算机应用研究,2011,28(4):1401-1403.

[10] 贾海鹏,张云泉,龙国平,等. 基于 OpenCL 的拉普拉斯图像增强算法优化研究[J]. 计算机科学,2012,39(5):271-277.

[11] NVIDIA Corporation. NVIDIA OpenCL JumpStart guide[M]. America: NVIDIA, 2009.

[12] MUNSHI A. The OpenCL specification[S]. America: Khronos OpenCL Working Group, 2009.