doi:10.11918/j.issn.0367-6234.2016.05.002

基于 GPU 的势能场骨架提取并行算法

赵丝喆, 王宽全, 袁永峰

(哈尔滨工业大学 计算机科学与技术学院, 150001 哈尔滨)

摘 要: 为解决势能场骨架提取方法计算效率低、提取过程耗时大的问题,同时为降低该方法的时间复杂度,提出了基于 GPU 的势能场骨架提取并行算法,并充分利用 CUDA 架构特有的常量存储器和共享存储器对普通并行算法进行改进.讨论了如何根据程序和显卡设备的固有属性来分配线程以达到最高的 GPU 占用率,从而得到最优的加速效果.对多组 3D 模型进行测试的结果表明,随着数据规模的增大,加速效果逐渐提升,处理 256×256×487 的体数据时,可获得 18 倍的加速比.

关键词: 图形处理器;并行计算;势能场;骨架提取;通用并行计算架构

中图分类号: P315.69 **文献标志码**: A **文章编号**: 0367-6234(2016)05-0018-05

Parallel method of skeleton extraction using potential field on GPU

ZHAO Sizhe, WANG Kuanquan, YUAN Yongfeng

(School of Computer Science and Technology, Harbin Institute of Technology, 150001 Harbin, China)

Abstract: For curve skeleton extraction algorithm, in order to improve the efficiency of potential field computation and save the time of extraction process, we presented a parallel potential field skeleton extraction method to reduce the time complexity, which was suitable for implementation on GPU, and then improved it by using constant memory and shared memory which was unique in CUDA. In order to achieve the highest GPU occupancy and the best speedups, we discussed how to assign threads according to the property of program and graphics device. The implementation was tested on several complex 3D models in CUDA framework. The results showed that our method had excellent performance especially on large data scale. When processing the volume data with the scale of 256×256×487, this improved method achieved speedups of 18x.

Keywords: GPU; parallel computing; potential field; skeleton extraction; CUDA

3D 物体的骨架类似于二维情形下的中轴线,可以形象地表述物体的拓扑特征,被广泛应用于计算机动画、机械设计、医学图像、虚拟导航和虚拟内窥镜等技术领域[1].目前,骨架提取的方法可以根据输入数据类型的不同来划分.文献[2]针对计算机动画中的三角网格面数据计算出数据的 voronoi 图,应用平均曲率对其进行边缘划分,进而得到离散的骨架点;文献[3]针对三维激光扫描技术获得的点云数据利用马尔可夫随机场模型的 3D 非刚性匹配技术追踪点云数据,得到的轨迹即为物体骨架;此外,还有 CT、MRI 等序列扫描图像合成的体数据,由于体数据可以完整保留物体的内部信息,在医学图像可视化和生物信息等方面都具有重要意义.

针对于体数据的骨架提取方法主要包括:拓扑细化法,如文献[4]中反复剥离物体的最外层体素,直至单连通即为骨架;距离场方法,如文献[5]对数据进行距离变换,抽取出局部距离极大点作为骨架点.但文献[6]表示以上两种方法所提取出的骨架易受噪声干扰.因此,Cornea ND 等人提出了势能场方法[7],综合考虑到所有表面点的影响,而不像距离场只考虑距离最近的单个表面点,因此该方法具备更好的鲁棒性.但实验表明,运用该方法作用于 256×256×487 的虚拟人体数据时,需要耗费 5 h,这在临床应用中是无法忍受的.因此,本文提出基于图形处理器(GPU)的势能场骨架提取并行算法,并加入 CUDA 架构特有的存储器结构来优化算法,讨论了如何根据显卡设备和程序的固有属性来分配线程以达到最高的 GPU 占用率,从而得到更优的加速效果.

收稿日期: 2015-02-03.
基金项目: 国家自然科学基金面上项目(61173086).
作者简介: 赵丝喆(1991—),女,硕士研究生;
王宽全(1964—),男,教授,博士生导师.
通信作者: 王宽全,wangkq@hit.edu.cn.

1　势能场骨架提取方法

物体的骨架需要具备以下特性：1）纤细性，凝练出物体的基本结构；2）连通性，若物体本身是连通的，则骨架也同样是连通的；3）中心性，骨架应处于物体的核心位置上.针对体数据而言，骨架指的是由单体素组成的、连通的、位于物体中心的一序列体素[8].

Cornea ND 提出的势能场方法假设物体表面遍布同种点电荷，作用于物体内部，从而形成静电斥力场.通过计算各点在势能场中的受力情况，选取特殊的点作为种子点，连接这些点形成 3D 物体的核心骨架.具体步骤如下：

1）将体数据中的各个离散体素点分类为外部点、表面点、边界点和内部点.外部点是体素值为 0 的点；表面点是指其 26 邻域中至少存在一个外部点；边界点是指其 26 邻域中至少存在一个表面点；其余的均为内部点.

2）势能场的计算. 内部点或边界点 P 会受到周围的表面点 C 所带来的斥力，该斥力与距离成反比，计算公式为

$$F_{PC}=\frac{\vec{CP}}{R^m}.$$

式中：$\vec{CP}$ 为点 C 指向点 P 的归一化矢量；R 为两点间距离；m 为调节参数，m 越大，远距离表面点的作用力越小，势能场的梯度也越大，当 $m=2$ 时，即为牛顿斥力.

3）选取场值为 0 且场方向发生改变的点为关键点.首先检测一个立方体区域内的 8 个顶点，若场值在 X,Y,Z 方向上均发生改变，则该区域中可能存在关键点，迭代划分立方体，直至得到一个不可分的体素点为止.接着计算该点势场力的雅克比矩阵，根据以下定义将关键点分类：

定义 1　雅克比矩阵特征值的实部和虚部均为负时，该点称为吸引点，其周围所有向量都指向该点.

定义 2　雅克比矩阵特征值的实部和虚部均为正时，该点称为排斥点，其周围所有向量都背离该点.

定义 3　雅克比矩阵特征值的实部和虚部有正有负时，该点称为鞍点，其周围向量有的指向该点，有的背离该点.

4）骨架生长和骨架连接.遍历所有鞍点，每个鞍点均生成一个骨架段.从鞍点出发，以正特征值对应的特征向量为方向，使用力跟随法按照一定的步长前进.连接所有鞍点形成骨架.

2　基于 GPU 的势能场骨架提取并行算法

通过实际运行发现，势能场的计算占据了整个提取过程 98% 的时间，因此，减少骨架提取时间的核心思想就是加快势能场的计算.

2.1　并行性分析

计算势能场时，需要针对每一个内部点或边界点，扫描所有表面点.因此，势能场的求解过程中存在两层循环：外层循环是遍历所有内部点或边界点，求得各个点的势场力；内层循环是遍历每个表面点，计算每个表面点对该内部点或边界点的影响.计算公式如下：

$$F=\sum_i F_{P_i}+\sum_j F_{P_j}=\sum_i\sum_k F_{P_iC_k}+\sum_j\sum_k F_{P_jC_k}. \quad (1)$$

其中，i 为内部点，j 为边界点，k 为表面点，势能场计算的时间复杂度是 $O((i+j)\times k)$.由式(1) 可知，势场力的计算具备独立性，点与点之间不互相影响，因此可以将原本的串行计算并行化.通常情况下，内部点和边界点的个数大于表面点的个数，外层循环的计算量更大，于是本算法将外层循环放至 GPU 中，为每一个内部点和边界点分配一个线程，将时间复杂度降至 $O(k)$，且 k 远小于 $(i+j)$.另外，根据定义可知边界点处于表面点和内部点之间，而计算平均值要比计算距离简单很多，于是本算法将边界点 26 邻域点的平均势场力作为该点的势场力，此过程同样放至 GPU 中.

2.2　算法流程

在基于 GPU 的并行系统中，CPU 和 GPU 各司其职，CPU 负责复杂的流控制等需要串行处理的部分，而密集型数据的并行计算部分则交由 GPU 完成[9].有别于原始的串行算法，本文中 CPU 只负责简单的体素点划分，而复杂的势场力计算则交由 GPU 完成.计算时将各个内部点和边界点平均分配给每个线程，多线程并行执行.具体的程序流程见图 1.

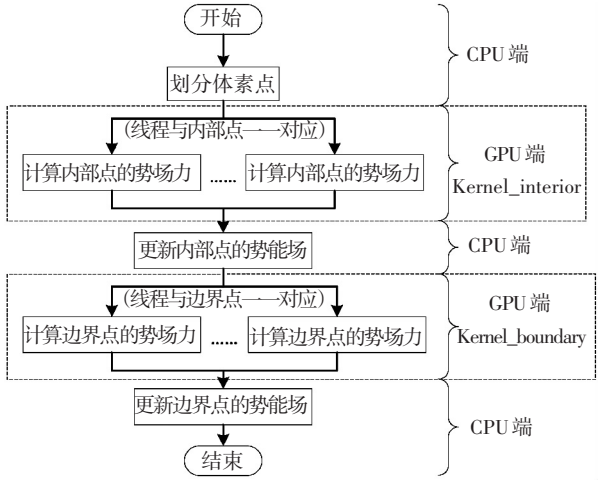


图 1　GPU 并行的势能场骨架提取流程图

2.3 算法改进

在CUDA架构中存在多种存储结构,按照存取速度由快到慢排列依次是:寄存器(Register)、常量存储器(Constant Memory)、共享存储器(Shared Memory)、纹理存储器(Texture Memory)、局部存储器(Local Memory)和全局存储器(Global Memory).

在普通的并行算法中,仅仅使用了寄存器和全局存储器,为了进一步加快程序的运行速度,提出了改进的并行算法.结合算法本身的特点,使用了访问速度可以与寄存器媲美的常量存储器和共享存储器.

首先,对于每个内部点而言,周围的表面点个数有限,数据量小于常量存储器的容量64 kB,并且表面点仅用于读取,并不对其改写,因此可以将表面点的信息存储在常量存储器中以加快内部点的势场力计算.另外,在程序执行过程中,需要频繁地访问全局存储器来修改势场力数组的值,这无疑会带来较大的延迟,因此在改进算法中将势场力数组移入到共享存储器中.

2.4 线程分配

对于串行程序而言,程序的执行时间可以表达为关于问题规模(即输入数据规模)的函数,而对于并行程序而言,执行时间不仅与问题规模相关,还与并行体系结构相关,而GPU的占用率(occupancy)就是其中一个重要的考虑参数[10],计算公式如下MP(Multiprocessor):

$$\text{occupancy}=\frac{\text{ActiveWarps per MP}}{\text{maxWarps per MP}}=\frac{(\text{ActiveThreadBlocks per MP})\times(\text{Warps per Block})}{(\text{maxThreads per MP})/(\text{Threads per Warp})}. \tag{2}$$

其中,$\text{Warps per Block}=\frac{\text{Threads per Block}}{\text{Threads per Warp}}$,而Active Thread Blocks per MP则取

$$\left(\frac{\text{maxSharedMemory per MP}}{\text{SharedMemory per Block}},\ \frac{\text{maxRegister per MP}}{(\text{Registers per Thread})\times(\text{Threads per Block})},\ \text{maxThreadBlocks per MP}\right)$$

的最小值.

maxRegister per MP,maxSharedMemory per MP和maxThreadBlocks per MP分别表示最大寄存器数量、最大共享存储器容量以及最大线程块数目,其值均取决于显卡的规格参数;而Registers per Thread和SharedMemory per Block则表示程序运行中实际用到的寄存器数量和共享存储器的容量,其值均由CUDA在启动线程时自动分配.

由此可知,对于不同的显卡设备,可先将显卡和程序本身的固有属性参数代入式(2),计算出合适的Threads per Block值即每块线程数,以达到最优的GPU占用率,从而得到更好的加速效果.

3 实验结果与性能分析

3.1 实验环境

开发平台为3.33 GHz Intel(R) Xeon(R) X5680双核CPU,24G内存,显卡分别为NVIDIA Quadro FX 5800和NVIDIA GeForce GTX 580,显卡的各项参数如表1所示.开发工具为Microsoft Visual Studio 2010和CUDA4.0.

表1 显卡的规格参数(MP: Multiprocessor)

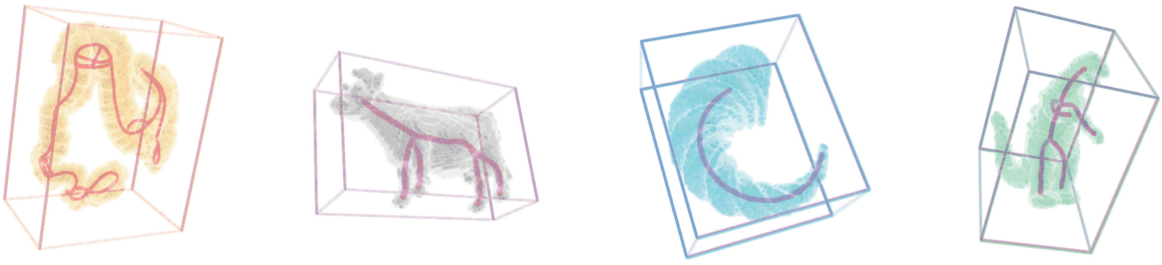
显卡	Architecture	Threads per Warp	MaxThreads per MP	MaxRegister per MP	MaxSharedMemory per MP(bytes)	MaxThreadBlocks per MP
FX 5800	GT200	32	1 024	16 384	16 384	8
GTX 580	GF100	32	1 536	32 768	49 152	8

3.2 提取结果

图2显示了本程序的骨架提取结果,采用的体数据来自于文献[6],分别是含有91 329个边界点,大小为204×132×260的结肠模型;含有6 555个边界点,大小为85×31×54的牛模型;含有9311个边界点,大小为87×74×45的螺旋模型;含有6 986个边界点,大小为54×87×75的恐龙模型.

3.3 加速比

在不同的输入数据规模下,普通并行算法和改进并行算法的加速比结果如表2所示.将Threads per Block和Blocks per Grid均设定为128,分别在显卡FX5800和GTX580上进行测试,均得到了较好的加速效果,说明本算法不受显卡设备的制约.算法的加速比随着数据规模的增大而逐渐提升,当数据规模达到256×256×487时,普通并行算法在两种显卡上分别可以达到11倍和15倍的加速比,而加入了常量存储器和共享存储器的改进算法又进一步将加速比提升了20%左右,分别达到了15倍和18倍.



(a)结肠模型　(b)牛模型　(c)螺旋模型　(d)恐龙模型

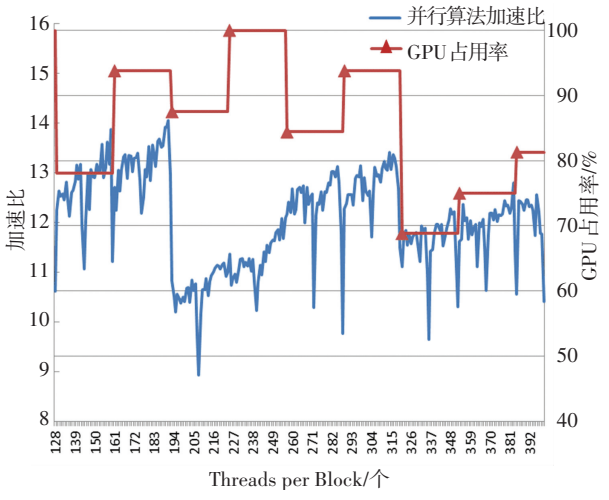
图 2　GPU 并行的势能场骨架提取结果

表 2　普通并行算法与改进并行算法的加速比

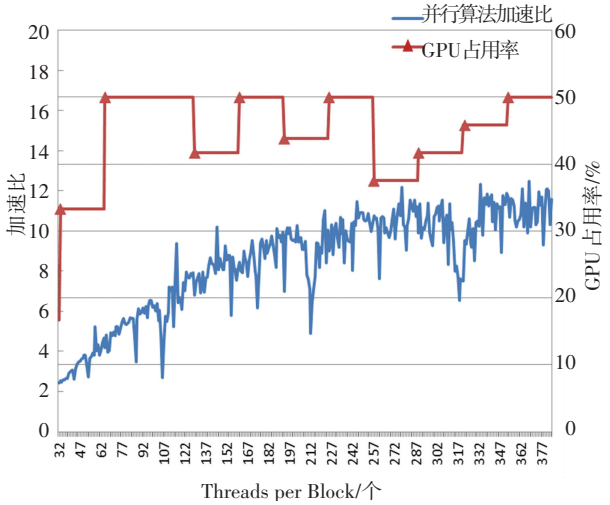
体数据规模	边界点个数	串行算法运行时间/s	并行加速比(FX 5800)			并行加速比(GTX 580)		
			简单并行	加常量存储器	加共享存储器	简单并行	加常量存储器	加共享存储器
16×16×487	10 971	3.1	3.0x	3.3x	3.6x	2.7x	2.4x	2.6x
32×32×487	22 254	24.8	5.3x	5.8x	6.2x	5.0x	5.3x	5.6x
64×64×487	45 464	213.6	7.2x	7.8x	8.8x	6.4x	7.0x	7.4x
128×128×487	101 619	1 800.9	8.8x	9.6x	11.1x	11.0x	12.1x	13.0x
256×256×487	206 010	1 6212.2	11.2x	12.4x	14.7x	15.2x	16.9x	18.7x

3.4　线程选择

分别对基于 GT200(计算核心 1.3)和 GF100(计算核心 2.0)架构的 FX5800 和 GTX580 进行了测试.由式(2)可知,影响 GPU 占用率的主要参数是 Registers per Thread, SharedMemory per Block 和 Threads per Block.运行程序前,在编译选项里面加入--ptxas-options=-v 命令,可以查看到本程序的 Registers per Thread 的值为 42, SharedMemory per Block 的值为 1548 字节.将参数代入式(2),再调节 Threads per Block 的值,就可以得到不同的 GPU 占用率,如图 3 所示.图 3(a)表明,对于 FX5800,GPU 的占用率对加速比产生了较大影响.当 Threads per Block 为 193 时,GPU 占用率从 93.8% 下降到 87.5%,而加速比也相应从 14.1 下降到了 10.5.随后当 Threads per Block 的值为 321 时,又再次出现了加速比随 GPU 占用率的降低而急速下降的情况.而图 3(b)表示,对于 GTX580 而言,这种关联并不明显,这是由于 GF100 架构中新增加了高速缓存功能,降低了访问冲突的发生.但仍可以发现,几个加速比的低峰值,均出现在 GPU 占用率较小的情况下.因此,在分配线程时应选择合适的 Threads per Block,使得 GPU 的占用率足够大,从而得到更好的加速比.



(a) FX5800



(b) GTX580

图 3　不同 Threads per block 下的 GPU 占用率和加速比

4 结 语

针对势能场方法提取骨架耗时长、计算复杂度高等问题,提出了一种基于GPU的势能场骨架提取并行算法.由于在提取过程中势能场的计算占据了98%的时间,对势能场的计算进行了并行性分析并提出并行算法,另加入了CUDA架构特有的存储器结构对普通并行算法进行了优化.此外,还讨论了如何根据显卡设备和程序的固有属性来分配线程以达到最高的GPU占用率,从而得到最优的加速效果.实验结果表明,当处理256×256×487规模的体数据时,可以获得18倍的加速比,有效地减少了骨架提取时间.

参考文献

[1] LIVESU M, SCATENI R. Extracting curve-skeletons from digital shapes using occluding contours[J]. The Visual computer, 2013, 29(9):907-916.

[2] TAGLIASACCHI A, ALHASHIM I, OLSON M, et al. Mean Curvature Skeletons[J]. Computer Graphics Forum, 2012, 31:1735-1744.

[3] Zhang Q, SONG X, SHAO X, et al. Unsupervised skeleton extraction and motion capture from 3D deformable matching [J]. Neurocomputing, 2013, 100:170-182.

[4] BERTRAND G, COUPRIE M. Powerful parallel and symmetric 3d thinning schemesbased on critical kernels [J]. J Math Imaging Vis, 2014, 48:134-148.

[5] ARCELLI C, DI BAJA G S, SERINO L. Distance-driven skeletonization in voxel images[J]. IEEE Trans Pattern Anal Mach Intell, 2011, 33(4):709-720.

[6] CORNEA ND, SILVER D, MIN P. Curve-skeleton properties, applications, and algorithms [J]. IEEE Transactions on Visualization and Computer Graphics, 2007, 13(3):530-548.

[7] CORNEA N D, SILVER D, YUAN Xiaosong, et al. Computing hierarchical curve-skeletons of 3D objects[J]. The Visual Computer, 2005, 21(11):945-955.

[8] LIU B, TELEA A C, ROERDINK J B T M, et al. Parallel centerline extraction on the GPU [J]. Computers & Graphics, 2014, 41:72-83.

[9] LEE C, RO W W, GAUDIOT J L, et al. Boosting CUDA applications with CPU-GPU hybrid computing [J]. International Journal of Parallel Programming, 2014, 42(2):384-404.

[10] JIMÉNEZ J. Three-dimensional thinning algorithms on graphics processing units and multicore CPUs [J]. Concurrency and Computation: Practice and Experience, 2012, 24:1551-1571.

(**编辑** 王小唯 苗秀芝)

封面图片说明

形状记忆聚合物SMP(shape memory polymer)是指具有某一特定的初始形状,在一定条件下变形固定后,通过热、电、磁等外部条件的刺激,能够恢复至其初始形状的聚合物,其密度低、可大尺寸成形、形变大.它具有形状记忆特性、自锁紧特性和大刚度变化特性,在航天航空、生物医学等领域的应用备受关注.

基于形状记忆聚合物复合材料的铰链和桁架等空间展开结构和锁紧释放机构,通过材料和结构自身的变形实现结构的展开、驱动和释放,无需机械、电机和火工品等驱动装置,具有冲击小、重量轻、结构简单、展开稳定和可靠性高等优点.

图片A为形状记忆聚合物可展开花,其中图片B(形状记忆聚合物复合材料铰链的形状回复过程),图片C(八角形智能扭转释放机构)分别引自: LAN X, WANG X H, LIU Y J, et al. Fiber reinforced shape-memory polymer composite, its application in a deployable hinge[J]. Smart Materials and Structures, 2009, 18: 024002; WEI H Q, LIU L W, ZHANG Z C, et al. Design and analysis of smart release devices based on shape memory polymer composites[J]. Composite Structures, 2015, 133: 642-651.

(图文提供: 刘立武, 赵伟, 兰鑫, 刘彦菊, 冷劲松. 哈尔滨工业大学航天学院)