基于GPU的图像特征并行计算方法

张 杰 柴志雷 喻 津
(江南大学物联网工程学院 无锡 214122)

摘 要 特征提取与描述是众多计算机视觉应用的基础。局部特征提取与描述因像素级处理产生的高维计算而导致其计算复杂、实时性差,影响了算法在实际系统中的应用。研究了局部特征提取与描述中的关键共性计算模块——图像金字塔机制及图像梯度计算。基于NVIDIA GPU/CUDA架构设计并实现了共性模块的并行计算,并通过优化全局存储、纹理存储及共享存储的访问方式进一步实现了其高效计算。实验结果表明,基于GPU的图像金字塔和图像梯度计算比CPU获得了30倍左右的加速,将实现的图像金字塔和图像梯度计算应用于HOG特征提取与描述算法,相比CPU获得了40倍左右的加速。该研究对于基于GPU实现局部特征的高速提取与描述具有现实意义。

关键词 图像金字塔机制,图像梯度计算,GPU,CUDA
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.10.060

Parallel Computation Method of Image Features Based on GPU

ZHANG Jie CHAI Zhi-lei YU Jin
(School of Internet of Things Engineering, Jiangnan University, Wuxi 214122, China)

Abstract Feature extraction and description are the foundation for many computer vision applications. Due to its high dimensional computation of pixel-wise processing, feature extraction and description are computationally intensive with poor real-time performance. Thus it is hard to be used in real-world applications. In this paper, the common computational modules used in feature extraction and description, pyramidal scheme and gradient computation were studied. The method used to compute these modules in parallel based on NVIDIA GPU/CUDA was introduced. Furthermore, computational efficiency was improved by optimizing memory accessing mechanism for global, texture and shared memory. Experimental results show that a 30x speed-up is obtained by GPU-based pyramidal scheme and gradient computation against that of CPU. By employing these GPU-based optimization techniques into HOG (Histogram of Gradient) implementation based on GPU, it obtains a 40x speed-up against that of CPU. The method proposed in this paper is of significance for implementing fast feature extraction and description based on GPU.

Keywords Image pyramidal scheme, Image gradient computation, GPU, CUDA

1 引言

特征提取与描述是许多视觉算法的先行条件,是目标识别、分类和匹配的基础。在特征提取的过程中,根据特征提取的范围的不同可将特征分为全局特征与局部特征,全局特征的关注点在于整幅图像而不在于图像中的前景,与之相反,局部特征关注的是图像的子区域,根据图像的各个子区域呈现出的不同特征值滤除图像中的背景信息来获得所需的前景信息。诸如梯度方向直方图(Histogram of Gradient, HOG)[1]、形状描述子(Shape Context)[2]、Shi-Thomas角点特征[3]等就是常见的局部特征。

通过对不同的局部特征算法进行综合分析可知,尽管其应用领域及构造方式不同,但可提取出其具有的共性计算操作,如梯度计算与图像金字塔机制。在构造描述子的过程中,需要利用到图像的梯度信息,HOG特征是通过图像的梯度信息构造梯度方向直方图来作为对特征区域的描述;Shape Context是通过以图像的梯度信息求取出的轮廓点作为特征点构造极坐标直方图来描述特征区域;KLT[4]的特征选择部分,利用梯度信息构造的Hessian矩阵求取出Shi-Thomas角点来作为特征点,并利用矩阵的较小特征值作为对特征区域的描述。图像金字塔也为这些算法广泛采用,它为多分辨条件下描述特征区域提供了依据。视觉算法的实现过程中需要在多分辨率的图像下提取特征并构造描述子以进行目标分类、识别与追踪等高层视觉操作。

在视觉算法中我们希望获得局部特征的过程尽可能快,但实际中图像局部特征提取与描述子的构造过程往往比较耗时。在实际应用中局部特征的高维性将会对算法的高效实现产生重要的影响。图像金字塔以及图像梯度是构造局部描述子时常用的共性操作,快速地实现梯度求取与图像金字塔操作是快速构造局部特征的关键。M. Strengert等人[5]高效实

到稿日期:2014-10-15 返修日期:2015-02-01 本文受国家自然科学基金资助项目:高可靠实时系统的计算平台(SoPC)研究(60703106),模糊支撑函数及其在图像特征表示中的应用(61170121)资助。

张 杰(1988—),男,硕士生,主要研究方向为计算机视觉、图像处理、基于GPU的机器视觉设计,E-mail:zhjkobe2013@live.com;柴志雷副教授,硕士生导师,主要研究方向为高性能视觉系统、嵌入式系统与结构;喻 津 硕士生,主要研究方向为高性能视觉系统、计算机视觉。

现了图像金字塔缩放中的双线性纹理插值方法以使其应用于双二次 B 样条插值图像缩放、任意宽度图像的高效滤波以及离散像素数据的插值。P. Dollár 等人[6]对物体检测中的多尺度特征提取的快速实现进行了研究，在不牺牲性能的前提下计算特征金字塔的部分，从而减少时间复杂度。

NVIDIA GPU 的并行计算架构 CUDA[7]在如今的计算机领域有着众多的应用，赵杰伊[8]等人采用基于 CUDA 的表面细分算法，通过合理地利用 GPU 的共享内存结构使得曲面细分过程变得高效，减小了显存空间的占有率并获得了 4 倍的速度提升。唐家维[9]等人对传统的 Apriori 算法中的支持度统计、候选集生成等方面合理地采用 CUDA 并行加速计算，以使得支持度统计与候选集生成在 10000 条事务的处理上效率分别提高了 16%以及 25%。合理地利用 CUDA 的并行加速技术会使得传统算法在 GPU 架构上得到最大限度的加速，这也是本文采用 GPU 并行技术加速图像特征求取的主要原因。

本文通过采用 NVIDIA CUDA 并行加速技术合理地设计和使用 GPU 的存储以实现图像金字塔以及图像梯度的高速计算。通过将 GPU 上图像数据存取的 3 种内存访问方式即全局内存、纹理内存、共享内存作为切入点，分析和设计图像金字塔以及图像梯度在这 3 种内存下的并行实现，从而在 GPU 上较为高效地实现这两种操作。在研究的过程中采用 GPU 并行实现图像梯度和金字塔，以获取多尺度特征的快速提取为目标，针对单纯的图像金字塔操作和多尺度特征求取方法进行寻优改进，从节约存储资源以及更加符合 GPU 的并行编程架构的角度出发，对计算做了改进，如改进图像金字塔的操作方法以使其更满足 GPU 下的并行实现。同时通过将文中的图像金子塔与图像梯度求取方法用于 GPU 下的 HOG 特征的构造，验证了图像金字塔与图像梯度求取并行计算方法的有效性。

2 梯度计算与图像金字塔机制

2.1 图像梯度计算

图像被看作是二维的离散函数，对图像求取梯度可以看作是对二维离散函数求导数。求取梯度实质上是利用一些模板或卷积核与图像进行卷积操作。本文采用二维高斯函数产生的卷积核作为卷积核，这是由于二维高斯函数有可分离的特性，使得卷积操作可以转换为图像与两个一维卷积核的操作，可减少运算量，提高运算效率[12]。

卷积操作是以图像中的每个像素点作为处理中心，将该点以及邻域内的点与卷积核的每个元素对应相乘，所有乘积之和作为卷积输出图像对应像素点的值。

在可分离的卷积操作中，求取水平与垂直梯度都需要将图像与水平卷积核以及垂直卷积核做两次卷积运算。相对于不可分离的卷积核，利用可分离的卷积核能降低计算复杂度。

2.2 图像金字塔机制

图像金字塔[10]是一系列以金字塔形状排列的分辨率逐级降低的图像组成的集合，图像金字塔的底部通常是待处理图像的高分辨率表示，而顶部是低分辨率的近似。在行人检测等机器视觉应用中，当目标物体的尺寸较小时，需要在高分辨率的图像中找出目标；如果物体的尺寸较大时，在低分辨率的图像中便可以找到目标；如果目标物体的大小变化多样化，以多分辨率方法进行研究是很有必要的。

在图像金字塔的构造过程中，如何从上一层分辨率的图像中采样建立下一层分辨率的图像是构成不同的图像金字塔的关键。不同的采样或插值方法将会构造出不同的图像金字塔，比如最近邻插值、双线性插值、双三次插值等[11]都是常用的插值采样方法。在本文的研究中，采取双线性插值操作建立图像金字塔是因为基于双线性插值得到的插值图像质量较高且具有较快的速度。

3 基于 CUDA 的 GPU 并行编程模式

3.1 CUDA 编程模型

CUDA 是 NVIDIA 推出的通用并行计算[13]架构，它使得 N 卡具有通用并行编程的能力。CUDA 可以使用标准 C 语言编写并行程序。

在采用 CUDA 进行并行程序的编写时，应用程序代码分为设备端代码与主机端代码两部分，在主机端代码中执行的是串行部分的代码，并行部分的代码将会以 kernel 的形式在设备端执行。图 1 为 CUDA 程序的执行过程。

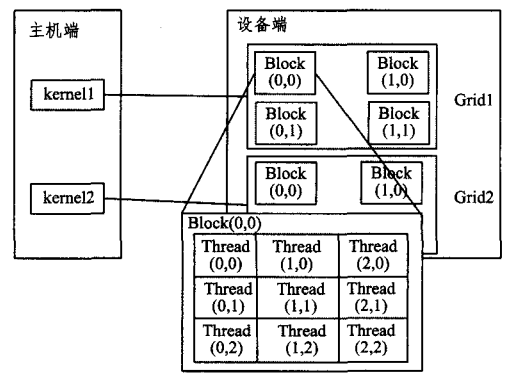

图 1 GPU 编程模型

从图中可以看到一个 kernel 对应一个网格(Grid)，每个网格对应着若干个线程块(Block)，每个线程块对应着若干线程(Thread)。在编写 CUDA 程序时需设置好 block 数目以及 block 中拥有的线程数目。

3.2 GPU 存储体系

当一个线程访问 GPU 中的数据时，可以访问多种存储类型[14]。

全局内存：在 GPU 中全局内存承载着与主机端之间的交互，从主机端拷贝过来的数据都是放在全局内存上，当在 GPU 中处理完后，目的数据需要从全局内存拷贝到主机端的内存中。

纹理内存：纹理内存能够绑定全局内存数据，绑定到纹理内存中的全局数据会获得纹理内存所提供的特性，纹理具有一组高速的纹理缓存，能够保存最近访问的数据，并从缓存中访问数据。

共享内存：共享内存使得线程块内的数据具有交互能力，同时它相对于全局内存具有更快的运算速度，在进行 CUDA 编程时合理地设计和利用共享内存会使得程序的性能得到一定程度的提升。

4 梯度计算及金字塔机制的并行计算

4.1 GPU 上的图像梯度求取

在 GPU 端求取图像梯度时，需将高斯模板连同图像数据拷贝到设备端。图像梯度的求取分为水平梯度的求取与垂

直梯度的求取，它们的求取过程中的卷积操作满足如下公式：

$$f(n)*k(n)=\sum_{m=-r}^{r} f(m+r)k(n+m) \quad (1)$$

式(1)为对图像进行卷积的过程，其中 $f(n)$ 代表图像，$k(n)$ 代表卷积核，r 代表卷积核的半径。

图 2 是对一幅图像进行二维卷积操作时的示意图，一个线程块要处理内框中 16 * 16 的像素，需要访问外框包围的所有像素数据。

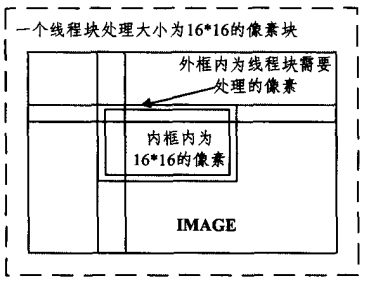

图 2　二维卷积计算

全局内存方式：

采用全局内存方式进行处理时，若卷积核的长度为 k，对于每个待处理的像素，进行卷积操作时，需要访问以它为中心与之水平相邻或垂直相邻的 $k-1$ 个像素。

纹理内存方式：

对于长度为 L 的卷积核，对某一像素点进行卷积操作时需要将该像素点及其周围的 $L-1$ 个像素点的像素值以 $tex2D(pixel\_x, pixel\_y)$ 的方式进行读取操作，然后与相应的卷积核进行卷积操作。

共享内存方式：

在共享内存的方式中，每一个线程块需要将处理图像块时的公用数据拷贝到共享存储器中，如果每一个线程将一个像素数据拷贝到共享内存中，那么处理图中的内框与外框之间的像素的线程将会在进行卷积操作时空闲，而且空闲线程数目会随着卷积核长度的增长而增加。为了减少空闲线程，每个线程必须载入多个像素到共享内存中，以增加图像块的宽度，使每个线程块处理多个目标像素的像素值。

在卷积过程中，对于水平梯度处理，可以增大每个线程处理的像素，这等同于增大处理的图形块的宽度，同理可知对于垂直梯度的处理，则需要增大图形块的处理高度。

每个线程块一次处理图像块中的一个部分，分几次即可完成整个图像块的处理，在这个过程中图像块中的每个线程对多个像素进行了处理，充分利用了共享存储器，从而提高了程序的性能。

图 3 为采用共享内存时 GPU 中的线程进行卷积操作的过程。

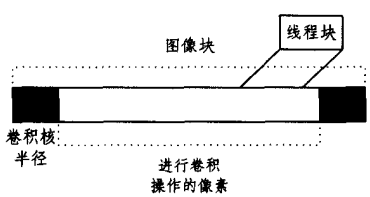

图 3　GPU 上的卷积操作

在垂直梯度的求取过程中，可设置好线程块中的线程使得每一个线程顺序访问共享内存的不同部分，为使同一线程在处理图像块中下一个像素点时尽可能地访问原来的共享内存区域，可设置处理的图像块的宽度为 32，即使垂直相邻的图像块中的像素相距一个 warp 大小，从而高效合理地利用内存。

4.2　GPU 上的图像金字塔操作

4.2.1　GPU 上的图像金字塔实现

将主机端的图像数据拷贝到设备端，针对每一层图像金字塔在设备端分配好相应的存储空间。图像金字塔的最底层图像数据与设备端拷贝过来的数据是相同的，之后的每一层图像均是由它的下一层金字塔图像经过双线性插值操作得到。

由于下层分辨率图像的每个像素点的值都是由上层分辨率图像的 4 个像素值决定的，因此在对该像素点操作时需要访问上级图像的 4 个像素值。当该像素位于不同的存储空间中时，将会产生对该像素的不同的访存方式。

对于金字塔的两层图像的处理如图 4 所示。

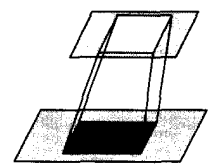

图 4　图像金字塔计算

假定某个线程块需要处理白色区域的像素值，那么在操作过程中该线程需要访问的区域为黑色区域，而在一个线程处理白色区域中的某个像素值时，需要访问黑色区域中的 4 个像素值。

全局内存方式：

当采用这种方式索引处理包含白色区域的图像的每个像素时，以全局内存的读取方式访问保存在黑色区域中的图片数据。

纹理内存方式：

当采用纹理内存的方式索引时，需要利用纹理绑定函数 *cudaBindArrayToTexture*() 把待求层的下一层金字塔图像绑定到纹理内存，然后利用 *tex2D*() 函数进行处理，该函数具有插值计算功能以及边界像素自动填充的功能。

对于待求层的图像上一点 (x,y)，将其按缩放比率 scaleRatio 缩放到该层的下一层图像中，坐标为 $(u,v)=(x*\text{scaleRatio}, y*\text{scaleRatio})$，这样就能得到像素点 (x,y) 的像素值 $Pixel(x,y)=tex2D(u,v)$。

共享内存方式：

采用共享内存的存取方式时，线程块要处理白色区域，需要将白色区域所在像素拷贝到线程块共用的共享内存中，在后续的计算过程中每个线程需要访问黑色区域中的 4 个像素的值。当一个线程块内的线程能够进行对多个像素的处理或者它有较忙碌的处理资源时，它将会在一定程度上发挥共享内存的优势。因此设计的线程处理方式如图 5 所示。

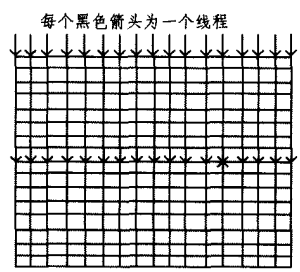

图 5　GPU 上的图像梯度

图中的每个线程需要处理并获得 8 个像素的像素值。

4.2.2 改进图像金字塔

使用传统的图像金字塔操作时，由于每一层的图像都是由上一层的图像插值得到的，此时在存储过程中需要存储每一层的图像数据。在包含图像金字塔的视觉算法计算过程中，金字塔的每层图像只是参与计算而无需把每层图像的结果拷回主机端，对于 GPU 中的宝贵的存储资源，上述传统方法有所欠缺。因此，在 GPU 中为了减少存储资源的浪费而使用改进的图像金字塔是很有意义的。

在改进的图像金字塔操作中，金字塔的任何一层图像都是由分辨率最高的最底层图像金字塔经过双线性操作得到。采用这种方式的图像金字塔操作使得每一层图像数据在参与完计算后被替代，增加了存储资源的利用率。

由图 6 可以看到在设备端存储好原始图像的数据后，可以将图像金字塔的每一层图像都存储在目标图像所在的存储空间中，图像 1 是对源图像的拷贝也是图像金字塔分辨率最高的底层图像，以后的各层图像都是由源图像进行双线性插值操作得到的。由于在视觉操作过程中都是分辨率高的图像参与完运算后再进行下一层图像的处理，因此在进行图像 2 的数据生成时，仅需改变存储资源中相应的位置便可。采用改进后的金字塔操作正是由于插值源均是源图像，因此每一层金字塔图像参与完计算后，它所占据的存储空间可由当前层替换掉。与之相反，传统的金字塔方法中图像 i 是由图像 $i-1$进行双线性插值得到的，因此在插值操作过程中图像 i 与图像 $i-1$ 在存储中不能有重叠部分。当采用纹理内存进行纹理绑定访问图像数据以参与运算时，采用传统的图像金字塔将会出现多次的绑定与解绑操作，会在一定程度上浪费时间。使用改进后的图像金字塔操作，绑定的图像始终为源图像，解绑是在金字塔中的所有图像参与完运算后再进行。

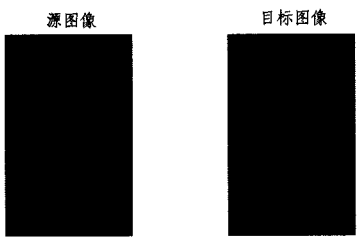

图 6 图像金字塔存储

5 实验结果与分析

本文采用的实验平台为操作系统：CentOS 6.4；CPU：Intel Core i3-240；GPU：NVIDIA Tesla K20X。

5.1 图像梯度计算

在实验中采用的卷积核是长度为 7 的一维高斯卷积核以及一维高斯导数卷积核，测试图像的分辨率分别为 320 * 240，640 * 480，1920 * 1080。

从表 1 中可以看到使用共享存储器的方式的计算速度相较于使用另外两种方式有一定的提高，证明了合理地利用共享存储器可使性能得到一定的提升。

表 1 GPU 上的图像梯度耗时

分辨率	全局内存	纹理内存	共享内存
320 * 240	0.460448ms	0.43212ms	0.354400ms
640 * 480	1.450272ms	1.13145ms	1.049344ms
1920 * 1080	8.885760ms	6.921941ms	6.379104ms

表 2 表明，采用 GPU 实现图像梯度算法相较于 CPU 的实现在处理速度上具有很大的提升。

表 2 图像梯度算法时间对比

分辨率	GPU 最小耗时	CPU 耗时	加速比
320 * 240	0.35440ms	5.987322ms	16.90
640 * 480	1.049344ms	29.123424ms	27.75
1920 * 1080	6.379104ms	225.101523ms	35.29

5.2 图像金字塔计算

图像金字塔的实现过程中，金字塔层数设置为 31 层，缩放比率设置为 1.05。

表 3 给出了不同分辨率图像在 GPU 上实现时使用不同内存的耗时对比。

表 3 GPU 上图像金字塔耗时

分辨率	全局内存	纹理内存	共享内存
320 * 240	1.960578ms	1.840992ms	1.750336ms
640 * 480	6.153132ms	5.723600ms	5.420384ms
1920 * 1080	37.413425ms	35.151711ms	33.099998 ms

图 4 给出图像金字塔在 GPU 与 CPU 上的运行时间对比。

表 4 图像金字塔算法时间对比

分辨率	GPU 最小耗时	CPU 耗时	加速比
320 * 240	1.750366ms	38.342311ms	21.90
640 * 480	5.420384ms	148.212205ms	27.30
1920 * 1080	33.099998ms	1077.413324ms	32.55

从表 4 可见，随着图像分辨率的增加，GPU 相对于 CPU 的加速比也在逐步提高，这是由于 SVM 需处理的线程块数量增多，相应的活动的线程块数量也变多了，对于访存操作的掩盖更加完善，同时其对存储资源的利用更加充分，并使 GPU 处于忙碌状态与空闲状态的时间比率增高，使得 GPU 相对 CPU 在计算上的加速优势得到展现。

5.3 基于 GPU 的 HOG 描述子求取

表 5 列出了将图像梯度和图像金字塔并行计算方法用于基于 GPU 的 HOG 描述子计算的结果。该描述子的计算基于图像金字塔操作，在金字塔的各个层级均需计算 HOG 描述子，在每次 HOG 描述子的计算过程中需要求取梯度信息，从而构造梯度方向直方图。320 * 240、640 * 480、1920 * 1080 分辨率的图像的耗时如表 5 所列。

表 5 HOG 描述子耗时对比

分辨率	GPU 耗时	CPU 耗时	加速比
320 * 240	18.963776ms	543.272644ms	28.64
640 * 480	60.45ms	2663.108482ms	44.05
1920 * 1080	328.668945ms	26172.694541ms	79.63

同样可以看到，随着图像分辨率的增加，GPU 相对于 CPU 的加速比也在提高。

5.4 算法准确性

为了直观地展示提出的并行计算的准确性，本文在 GPU 上实现了完整的基于 HOG 描述子及 SVM 分类器的行人检测算法。

图 7 是在 CPU 和 GPU 上的行人检测结果，从检测结果中可以看到两者的检测效果一样，只是由于 GPU 进行浮点运算时对于对齐、截断等操作的处理与 CPU 的处理方式存在差异，从而造成 round-off 误差积累，导致某些检测框的大小有着轻微的差异。

（下转第 324 页）

Processing,1997,6(1):103-113
[2] Gavrila D, Philomin V. Real-Time Object Detection for smart Vehicles[C]// IEEE international Conference on Computer Vision,2010:533-542
[3] Holzer S, Hinterstoisser S, Ilic S, et al. Distance transform templates for object detection and pose estimation[C]// CVPR. 2009:1177-1184
[4] Steger C. Occlusion Clutter, and Illumination Invariant Object Recognition[C]// IAPRS. 2002
[5] Dalal N, Triggs B. Histograms of Oriented Gradients for Human Detection[OL]. http://lear. inrialpes. fr/people/triggs/pubs/Dalal-os. pdf. cvpr
[6] Ferrari V, Jurie F, Schmid C. From images to shape models for object detection[J]. IJCV,2009,87(3):284-303
[7] Tombari F, Franchi A, Automation D, et al. BOLD features to detect texture-less objects[C]// 2013 IEEE International Conference on Computer Vision(ICCV 2013). 2013:1265-1272
[8] Hinterstoisser S, Cagniart C, Ilic S, et al. Gradient response maps for real-time detection of textureless objects[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012,34(5):876-888
[9] Cheng M M, Zhang Z, Lin W Y, et al. Bing: Binarized normed gradients for objectness estimation at 300fps[C]// CVPR. 2014: 3286-3293

(上接第 300 页)

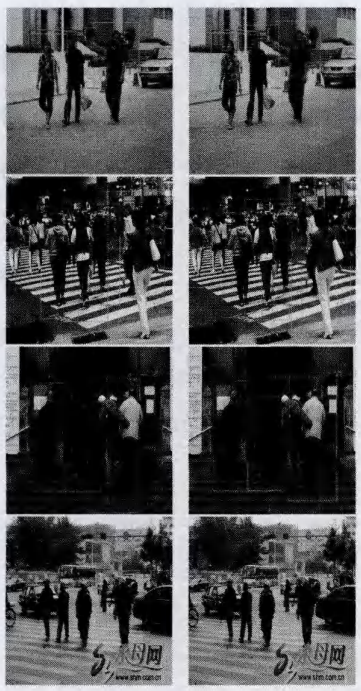

图 7 GPU 行人检测结果和 CPU 行人检测结果

结束语 图像金字塔和图像梯度的快速实现对于诸如 HOG、Shape Context、KLT 等视觉算法的快速实现是至关重要的，它们是视觉算法在进行譬如行人检测、字符识别、图像配准过程中常见的共性操作。合理高效地实现这两个操作对于包含它们的视觉算法是很有意义的。本文合理地设计和使用 GPU 的纹理内存、全局内存、共享内存，在 GPU 上分别进行图像梯度和图像金字塔的求取，力求找到快速处理这两种操作的方法。同时对传统的图像金字塔求取过程进行改进，使其更适合在 GPU 上实现，减少了图像金字塔需要耗费的存储空间。相对于 CPU 上的实现，随着图像分辨率的增加，在 GPU 上实现的加速比逐步增加。本文提出的并行计算方法对于局部特征在 GPU 上的实现也具有借鉴意义。

参 考 文 献

[1] Dalal N, Triggs B. Histogram of oriented gradients for human Detection[C]// CVPR. 2005:886-893
[2] Belongie S, Malik J, Puzicha J. Shape Matching and object recognition Using Shape Contexts[J]. IEEE Trans. on Pattern Analysis and Machine Intelligence,2002,24(4):509-522
[3] Shi J, Thomasi C. Good feature to track[C]// IEEE Conference on Computer Vision and pattern Recognition. 1994:593-560
[4] Lucas B, Kanade T. An iterative image registration technique with an application to stereo vision[C]// Proceedings of the International Joint Conference on Artificial Intelligence. 1982:674-679
[5] Strengert M, Kraus M, Ertl T. Pyramid Methods in GPU-Based Image Processing[C]// Proceeding of Vision, Modeling, and Visualization 2006. 2006:169-176
[6] Dollár P, Appel R, Belongie S. Fast Feature pyramids for object detection[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2014,36(8):1532-1545
[7] Nvidia. NVIDIA CUDA A Programming Guide version 4.0 [EB/OL]. http://www. nvidia. com/object/cuda-cn
[8] 赵杰伊，唐敏，童若锋. 基于 CUDA 的细分曲面阴影体算法[J]. 浙江大学学报(工学版)，2012,46(7):1301-1306
Zhao Jie-yi, Tang Min, Tong Ruo-feng. CUDA based shadow volume algorithm for subdivision surfaces [J]. Journal of Zhejiang University (Engineering Science),2012,46(7):1301-1306
[9] 唐家维，王晓峰. 基于 GPU 的并行化 Apriori 算法的设计与实现[J]. 计算机科学，2014,41(10):238-243
Tang Jia-wei, Wang Xiao-feng. Design and Implementation of Apriori on GPU[J]. Computer Science,2014,41(10):238-243
[10] Lindeberg T. Scale-space theory in computer vision[M]. London: Kluwer Academic Publishers,1994
[11] 王森，杨克俭. 基于双线性插值的图像缩放算法的研究与实现[J]. 动化技术与应用，2008,27(7):44-46
Wang Sen, Yang Ke-jian. An Image Scaling Algorithm Based on Bilnear Interpolation with VC++[J]. Techniques of Automation and Applications,2008,27(7):44-46
[12] 冯煌. GPU 图像处理的 FFT 和卷积算法及性能分析[J]. 计算机工程应用，2008,44(2):120-122
Feng Huang. Implementation and performance of FFT and convolution in image filtering on GPU[J]. Computer Engineering and Applications,2008,44(2):120-122
[13] 吴恩华. 图形处理器用于通用计算的技术、现状及其挑战[J]. 软件学报，2004,15(10):1493-1504
Wu En-hua. State of the Art and Future Challenge on General Purpose Computation by Graphics Processing Unit[J]. Journal of Software,2004,15(10):1493-1504
[14] 马安国，成玉，唐遇星，等. GPU 异构系统中的存储层次和负载均衡策略研究[J]. 国防科技大学学报，2009,31(5):38-43
Ma An-guo, Cheng Yu, Tang Yu-xing, et al. Research on Memory Hierarchy and Load Balance Strategy in Heterogeneous System Based on GPU[J]. Journal of National University of Defence Technology,2009,31(5):38-43