CUDA 技术及其在数字图像拼接中的应用

王亮亮，赵曙光
（东华大学 信息科学与技术学院，上海 201620）

摘 要：将 CUDA 技术应用于数字图像拼接领域，阐述了图像拼接的基本理论及其关键技术、多分辨率图像融合的关键算法以及 CUDA 技术的基本原理和开发方法，并编写了软件以实现图像快速拼接。采用对于尺度具有鲁棒性的 SIFT 算法进行特征点的提取与匹配，使用稳健的 RANSAC 算法求出图像间变换矩阵的值，并将图像映射到拼接平面，最后使用基于 CUDA 的 SIFT 算法实现了图像的无缝拼接。该方法提高了图像拼接的效率，克服了传统图像拼接方法因计算量大而等待时间长的缺点。实验结果表明，CUDA 在数字图像处理的实际应用中卓有成效，有广阔的应用前景。

关键词：CUDA；图像拼接；SIFT；多分辨率融合
中图分类号：TP391.41 **文献标识码**：A **文章编号**：1674-7720(2013)02-0034-03

CUDA and its application in digital image mosaic

Wang Liangliang, Zhao Shuguang
(College of Information Science and Technology, Donghua University, Shanghai 201620, China)

Abstract: This paper presents an application for digital image mosaic by using CUDA technology. The basic theory and key technology of image mosaic and CUDA are illustrated in this paper. Then, CUDA is used in the software to parallel the process of image mosaic. A scale-invariant feature extracting algorithm SIFT is used for feature extraction and matching. The transforming matrix is computed with RANSAC algorithm and the image is mapped to the mosaic plane. Finally, image mosaic is completed with CUDA based on SIFT method. This method achieves a seamless image mosaic and improves the efficiency of image process, which also overcomes long waiting time because of huge computations in traditional ways. Experiment results show that digital image processing combined with CUDA is much more efficient in practical applications and has bright potential applications in the future.

Key words: CUDA; image mosaic; SIFT; multiresolution spline mosaic

图像拼接是计算机视觉领域的一个重要分支，它是一种将多幅相关的部分重叠图像进行无缝拼接从而获得宽视角图像的技术。利用计算机进行匹配，将多幅具有重叠关系的图像拼合成为一幅具有更大视野范围的图像，这就是图像拼接的目的。CUDA 是英伟达(NVIDIA)公司倾力开发和推广的并行计算架构，该架构通过利用图形处理器(GPU)的处理能力，能够大幅提升计算性能。随着微软 Windows 7 与苹果 Snow Leopard 操作系统的问世，GPU 计算必将成为主流。本文基于 SIFT 算法，使用最新的 CUDA 并行计算技术重编算法并编制软件，不仅可以克服传统图像拼接技术中的局限性(如光照、尺度变化的影响等)，实现光照和尺度变化条件下的多视角无缝图像拼接，而且将提高图像拼接的速度和效率。

1 数字图像拼接的基本理论和方法

图像拼接的基本流程如图 1 所示，主要分为图像预处理、图像配准和图像融合与边界平滑 3 个步骤。图像预处理主要指对图像进行几何畸变校正和噪声点的抑制等，使参考图像和待拼接图像不存在明显的几何畸变。图像预处理主要是为下一步图像配准做准备，使图像质量能够满足图像配准的要求。图像配准主要指对参考图像和待拼接图像中的匹配信息进行提取，在提取出的信息中寻找最佳的匹配，完成图像间的对齐。图像拼接的成功与否主要是图像的配准。待拼接的图像之间可能存在平移、旋转和缩放等多种变换或者大面积的同色区域等很难匹配的情况，一个好的图像配准算法应该能够在各种情况下准确找到图像间的对应信息，将图像对齐。图像融合指在完成图像匹配以后对图像进行缝合，并对缝合的边界进行平滑处理，使缝合自然过渡。由于任何两幅相邻图像在采集条件上都不可能做到完全相同，因此，对于一些本应该相同的图像特性(如图像的光

照特性等),在两幅图像中就不会表现得完全一样。图像拼接缝隙就是从一幅图像的图像区域过渡到另一幅图像的图像区域时,由于图像中的某些相关特性发生了跃变而产生的。图像融合就是使图像间的拼接缝隙不明显,拼接更自然。

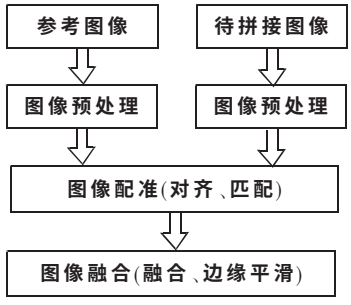


图 1 图像拼接流程图

2 CUDA 技术的基本原理和开发方法

图像处理的本质是大规模矩阵运算,特别适合并行处理。但 CPU 通用计算很难利用该特性。与此相反,GPU 在并行数据运算上具有强大的计算能力,特别适合作运算符相同而运算数据不同的运算,当执行具有高运算密度的多数据元素时,内存访问的延迟可以被忽略。

CUDA 编程模型将 CPU 作为主机(Host),GPU 作为协处理器(Coprocessor)或设备(Device),一个系统中可以存在多个设备。在这个模型中,CPU 与 GPU 共同工作,CPU 负责逻辑性强的事务处理和串行计算,GPU 则专注于执行高度线程化的并行处理任务。图 2 显示了 CUDA 的流程架构。

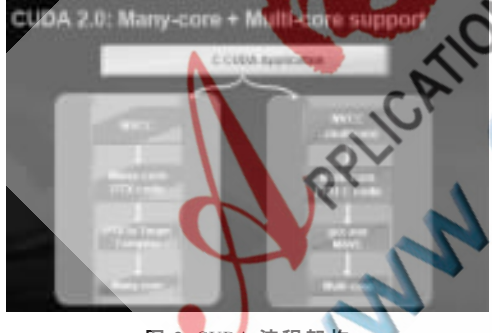


图 2 CUDA 流程架构

CUDA 对内存的操作与一般的 C 程序基本相同,操作显存则需要调用 CUDA API 中的存储器管理函数。一旦确定好程序中的并行部分,就可以将这部分计算交给 GPU。运行在 GPU 上的 CUDA 并行计算函数称为 Kernel,即内核函数,它并不是一个完整的程序,而是 CUDA 程序中可以被并行执行的步骤。内核函数必须通过 _global_ 函数类型限定符定义,且只能在主机端代码中调用。

CUDA 线程结构如图 3 所示。Kernel 以线程网格(Grid)为组织形式,每个 Grid 由若干个线程块(Block)组成,每个 Block 又由若干个线程(Thread)组成。在程序运行过程中,Kernel 是以 Block 为单位执行的,Grid 只是一系列可以被执行的 Block 的集合。不同 Block 是并行执行的,没有执行的先后顺序,而且相互无法通信。

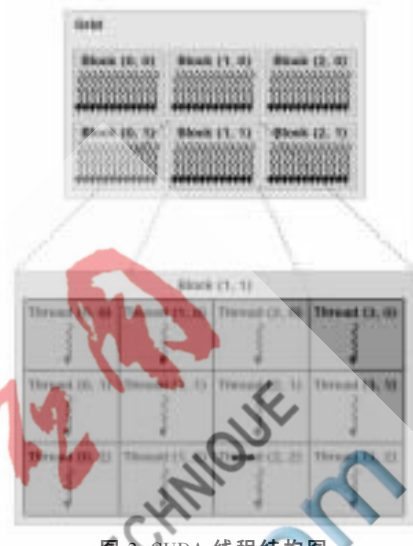


图 3 CUDA 线程结构图

CUDA 软件体系由 CUDA Library、CUDA Runtime API 和 CUDA Driver API 构成,如图 4 所示。CUDA 的核心是 CUDA C 语言,需要通过 nvcc 编译器进行编译。编译得到的仅是 GPU 端的代码,要在 GPU 上分配显存并启动 Kernel 就需要借助 CUDA Runtime API 或者 CUDA Driver API 来实现。CUDA Runtime API 和 CUDA Driver API 提供了实现设备管理、上下文管理、存储器管理、代码块管理和执行控制等操作的应用程序接口。CUDA Runtime API 在 CUDA Driver API 的基础上进行了封装,使得编程方便代码简洁,因此通常采用 CUDA Runtime API 进行项目开发。

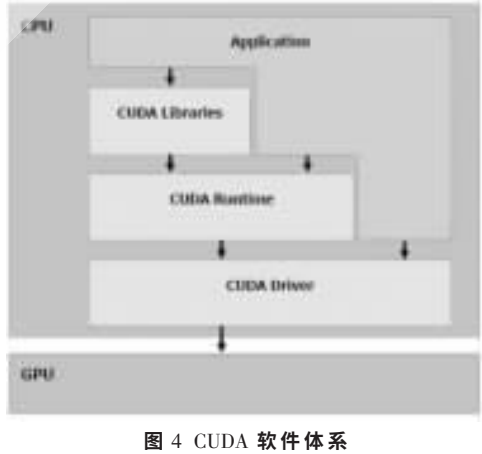


图 4 CUDA 软件体系

3 SIFT 特征匹配算法

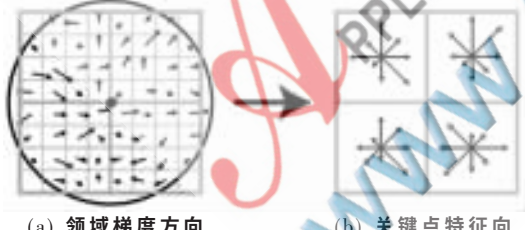
SIFT 算法首先在尺度空间进行特征检测,并确定关键点(Key Points)的位置和关键点所处的尺度,然后使用关键点邻域梯度的主方向作为该点的方向特征,以实现算子对尺度和方向的无关性。

SIFT 特征匹配算法包括两个阶段:(1)SIFT 特征的

生成，即从多幅待匹配图像中提取出对尺度缩放、旋转和亮度变化无关的特征向量；(2)SIFT 特征向量的匹配。

在实际的尺度不变特征点提取中，SIFT 算法将图像金字塔引入了尺度空间。首先采用不同尺度因子的高斯核对图像进行卷积以得到图像的不同尺度空间，将这一组图像作为金字塔图像的第一阶。接着对其中的 2 倍尺度图像(相对于该阶第一幅图像的 2 倍尺度)以 2 倍像素距离进行下采样来得到金字塔图像第二阶的第一幅图像，对该图像采用不同尺度因子的高斯核进行卷积，以获得金字塔图像第二阶的一组图像。以此类推，获得高斯金字塔图像。每一阶相邻的高斯图像相减，就得到了高斯差分图像，即 DoG 图像。通过拟和三维二次函数以精确确定关键点的位置和尺度，同时去除低对比度的关键点和不稳定的边缘响应点(因为 DoG 算子会产生较强的边缘响应)，以增强匹配稳定性、提高抗噪声能力。利用关键点邻域像素的梯度方向分布特性为每个关键点指定方向参数，使算子具备旋转不变性，生成 SIFT 特征向量。

接下来以关键点为中心取 8×8 的窗口。图 5(a)的中央黑点为当前关键点的位置，每个小格代表关键点邻域所在尺度空间的一个像素，箭头方向代表该像素的梯度方向，箭头长度代表梯度模值，图中圈代表高斯加权的范围(越靠近关键点的像素，其梯度方向信息贡献越大)。然后在每 4×4 的小块上计算 8 个方向的梯度方向直方图，绘制每个梯度方向的累加值，即可形成一个种子点，如图 5(b)所示。图 5(b)中，一个关键点由 2×2 共 4 个种子点组成，每个种子点有 8 个方向向量信息。这种邻域方向性信息联合的思想增强了算法抗噪声的能力，同时对于含有定位误差的特征匹配也提供了较好的容错性。



(a) 领域梯度方向　　(b) 关键点特征向

图 5 关键点邻域梯度信息生成的特征向量

4 基于 CUDA 的图像拼接软件的设计

4.1 Host 端实现

程序的 Host 端由 C++编写，负责控制整个程序的执行流程、所有供 CPU 和 GPU 所用的数据的分配管理以及 Device 端模块的调用。界面使用最基本的 Windows SDK 编写。

在数据初始化阶段，包含所有之后处理步骤中所需要的图像数据对象的生成，将输入图像作为高斯金字塔底层，通过系统中的 PYRAMID_LEVEL 宏指定金字塔的层数，在输入图像的尺寸基础上循环计算各层金字塔图像的分辨率，并对图像进行初始化。由于所有的图像数据需要在设备端处理，使用 cudaMallocPitch 函数分配数据地址空间，数据结构不再是 OpenCV 中的 IplImage，而是 GPU 可以识别的 uchar 数组类型。

使用 cudaMemcpy2D 函数将 IplImage 结构中的原始数据复制到相应高斯金字塔的最底层，也就是 uchar 数组的第一个元素，供 Device 端函数使用。随后进行 Kernel 函数调用，对于每一个需要处理的金字塔层，Host 端发起一次 Kernel 调用。例如：

```
reduce<<<(int)ceil ((float)(imageSize[l]/THREAD_NUM)),
HREAD_NUM >>>(lGaussianData [l+1],rGaussianData [l+1],
lGaussianData[l], rGaussianData[l], lLaplacianData[l], rLapla-
cianData[l], stride[l+1], stride[l], width[l+1], height[l+1],
width[l], height[l]);
```

4.2 Device 端实现

主要 Device 函数如下：

(1) reduce()函数对左、右图和掩码图像各完成一次 reduce 操作，生成下一层高斯金字塔图像。reduce 变换按照前文所述的方法对目标层的金字塔图像进行逐像素处理，每一个目标像素的颜色值按一定的权重值对原始图像中的一个 5×5 子块进行计算求得。

```
_global_ void reduce (uchar* lGaussianDataSrc, uchar*
rGaussianDataSrc, uchar* mGaussianDataSrc,uchar* lGaussian-
DataDst, uchar* rGaussianDataDst, uchar* mGaussianDataDst,
size_t strideSrc, size_t strideDst, int srcWidth, int srcHeight,
int gauWidth, int gauHeight);
```

(2) expand_and_minus()函数对左、右图像各完成一次 expand 操作和减法操作，生成下一层拉普拉斯金字塔图像。expand 变换相当于 reduce 变换的逆过程，它对目标层的金字塔图像进行逐像素处理，每一个目标像素的颜色值也是按 reduce 变换中所使用的权重值对原始图像中的一个 5×5 子块进行计算求得的。

```
_global_ void expand_and_minus (uchar*lGaussianDataSrcH,
uchar* rGaussianDataSrcH,uchar*lGaussianDataSrcL,uchar*rGau-
ssianDataSrcL,uchar* lLaplacianDataDst,uchar*rLaplacianDataDst,
size_t strideSrc,size_t strideDst,int srcWidth, int srcHeight, int
expWidth, int expHeight);
```

(3) blend ( ) 函数根据掩码图像的高斯金字塔以及左、右图像的拉普拉斯金字塔合成当前层的目标图像的拉普拉斯金字塔，所有像素值均以掩码图像的高斯金字塔为权重而求得。

```
_global_ void blend (uchar* mGaussianDataSrc, uchar*
lLaplacianDataSrc, uchar* rLaplacianDataSrc,uchar* sLaplacian-
DataDst, size_t stride,int lapWidth, int lapHeight)
```

(4) collapse()函数对图像的拉普拉斯金字塔分别完成一次 expand 操作和累加操作，本质上等同于 expand 操作，两者的基本算法是相同的。不同点在于 expand 模块

用于各层高斯金字塔的 expand 操作，从而生成各层拉普拉斯金字塔，而 collapse 函数则用于整个融合过程最后的图像重构步骤，将各层已经求得的拉普拉斯金字塔作扩展和累加操作，生成最后的拼接图像。

global void collapse (uchar* sLaplacianDataSrc, uchar* sExpandDataSrc, uchar* sExpandDataDst, size_t strideSrc, size_t strideDst, int srcWidth, int srcHeight, int expWidth, int expHeight)

本文借助于 SIFT 特征对于旋转和尺度的不变性以及对于噪声干扰良好的鲁棒性进行图像拼接与匹配，使用 CUDA 技术简单地对多分辨率融合算法进行了优化，提高了其执行效率和速度。编写了界面化的 Demo 程序，实现了基本的图像拼接功能。

参考文献

[1] 谭康.图像拼接技术与实现[D].南京:南京理工大学，2006.

[2] HARRIS C, STEPHENS M. A combined corner and edge detector[C]. Proceedings of the 4th Alvey Vision Conference, 1988:147-151.

[3] 张小洪，李博，杨丹.一种新的 Harris 多尺度角点检测[J]. 电子与信息学报，2007(7):1735-1738.

[4] LOWE D G. Object recognition from local scale-invariant features[C]. The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999(2):1150-1157.

[5] 骞森，朱剑英. 基于改进的 SIFT 特征的图像双向匹配算法[J]. 机械科学与技术，2007(9):1179-1182.

[6] Peng Xiaoming, Ding Mingyue, Zhou Chengping, et al. Improved approach for object location under affine transformation using the Hausdorff distance[J].Optical Engineering, 2003,42(10):2794-2795.

[7] 张毓晋. 图像工程(上册)图像处理(第 2 版)[M]. 北京：清华大学出版社，2006.

[8] LINDEBERG T. Detecting salient blob. like image structures and their scales with a scale-space primal sketch[J]. International Journal of Computer Vision,1993,11(3):283-318.

[9] NVIDIA. NVIDIA CUDA Programming Guide[Z].

[10] 张舒，褚艳利. GPU 高性能运算之 CUDA[M]. 北京：中国水利水电出版社，2009.

(收稿日期：2012-10-24)

作者简介：

王亮亮，男，1987 年生，硕士研究生，主要研究方向：信号与信息处理、数字图像处理。