引用格式：王浩,王含宇,杨名宇,等. Retinex图像增强在GPU平台上的实现[J].地球信息科学学报,2019,21(4):623-629. [Wang H, Wang H Y, Yang M Y, et al. Implementation of Retinex image enhancement algorithm on GPU platform[J]. Journal of Geo-information Science, 2019,21(4):623-629.] **DOI:**10.12082/dqxxkx.2019.180576

Retinex图像增强在GPU平台上的实现

王　浩[1,2],王含宇[1,2*],杨名宇[1,2],许永森[1,2]

1. 中国科学院航空光学成像与测量重点实验室,长春 130033;2. 中国科学院长春光学精密机械与物理研究所,长春 130033

Implementation of Retinex Image Enhancement Algorithm on GPU Platform

WANG Hao[1,2], WANG Hanyu[1,2*], YANG Mingyu[1,2], XU Yongsen[1,2]

1. Key Laboratory of Airborne Optical Imaging and Measurement, Chinese Academy of Sciences, Changchun 130033, China;
2. Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China

Abstract: With the advent of the era of UAV,the real-time requriements for massive data processing are getting higher.Achieve parallel processing of Retinex image enhancement algorithm on the GPU (Graphic Processing Unit) platform, which improves the processing speed of Retinex image enhancement algorithm for processing high resolution digital images.Firstly, by data combine-accessing and memory data interaction technology realize fast access of data, shorten the transmission time of data between different kinds of memory, and improve the efficiency of data access. Then, using kernel instruction optimization and data parallel computing technology, the multi-core programming of Retinex image enhancement algorithm on GPU platform is realized.Finally, the asynchronous execution mode of the host and the device is used to perform parallel calculation of the kernel data while data transmission, and the execution time of the algorithm on the GPU platform is further shortened by the parallel of the task level. With the powerful parallel computing power of the GPU, the processing speed of the Retinex algorithm is greatly improved. For images of different resolutions, the processing speed of the Retinex image enhancement algorithm is tens of times higher than that of the CPU platform. Processing an image with a resolution of 2048×2048 pixels requires only 38.04 ms, and the processing speed of the algorithm is 40 times higher than that of the CPU.

Key words: GPU; image enhancement ; Retinex algorithm ; parallel computing; UAV

***Corresponding author:** WANG Hanyu, E-mail: hanyu112@126.com

摘要：伴随着无人机时代的到来,对海量数据处理的实时性要求越来越高。本文在GPU(Graphic Processing Unit)平台上实现了Retinex图像增强算法的并行处理,提升了Retinex图像增强算法处理高分辨率数字图像的处理速度。首先,通过数据合并访问和内存数据交互技术实现了数据的快速访问,缩短了数据在不同种类内存间的传输时间,提升了数据访问的效率;然后,采用内核指令优化和数据并行计算技术,实现了Retinex图像增强算法在GPU平台上的多核程序设计;最后,采用主机端和设备端的异步执行模式,在数据传输的同时进行内核数据的并行计算,通过任务级的并行进一步缩短了算法在GPU平台上的

收稿日期：2018-11-13;**修回日期：**2019-03-20.

基金项目：国家重点研发计划项目(2017YFB0503001、2016YFC0803000)。[**Foundation items:** The National Key Research and Development Program of China, No.2017YFB0503001, 2016YFC0803000.]

作者简介：王　浩(1986-),男,吉林长春人,助理研究员,研究方向为航空数字图像处理。E-mail: wanghao7600@163.com

***通讯作者：**王含宇(1991-),女,吉林长春人,实习研究员,研究方向为航空光谱图像处理。E-mail: hanyu112@126.com

执行时间。研究表明，对于不同分辨率的图像，Retinex图像增强算法的处理速度相比于CPU平台均有数十倍的提高，如处理一帧分辨率为2048像元×2048像元的图像仅需要38.04 ms，算法的处理速度较CPU提高了40倍。

关键词：GPU；图像增强；Retinex算法；并行计算；无人机

1 引言

近十几年来，无人机行业蓬勃发展，在军事侦察、公安反恐、灾情评估、国土勘测、精准农业等领域发挥了重要的作用。无人机组网可以发挥协同作战的优势，在对地观测任务中能够获取范围更广、信息量更大的遥感数据。伴随着数据量的爆炸性增长，对无人机遥感图像处理算法实时性的要求也越来越高。图像增强是数字图像处理技术中的一种，是指按照某种特定的需求，突出图像中有用的信息，去除或削弱图像中无用的信息。图像增强的目的是使经过处理后的图像更适合人的视觉特性或易于机器识别[1-2]，图像增强可作为特征提取、特征匹配、图像测量、目标跟踪等的预处理算法[3-6]。常见的图像增强方法有直方图拉伸，直方图均衡，小波变换等。直方图拉伸和直方图均衡算法的实时性好，但增强后的图像会有部分细节信息丢失。小波变换可以在图像增强的同时抑制噪声，有着较好的增强效果。但需要根据图像亮度、对比度等信息人为调整输入参数，不能满足自适应的处理要求[7-9]。

Retinex算法[2]是基于人类视觉恒常性理论提出来的，Land于1986年最早提出了中心环绕Retinex算法。在此基础之上，又接连提出了单尺度Retinex算法和多尺度Retinex算法。随着算法的发展，又出现了具有色彩保持能力的多尺度Retinex算法和能够进一步增强图像细节的结合双边滤波的Retinex算法[10-11]。然而，Retinex算法需要对像素点进行高斯卷积和对数域下的差分运算，随着图像分辨率的增大和运算尺度的增加，算法的计算复杂度越来越大，不能满足实时性的要求。

近年来，GPU发展迅速，它可以提供强大的并行计算能力，同时呈现出嵌入式的发展趋势，为在嵌入式平台上应用提供了可能[12]。GPU在浮点数运算性能和数据带宽方面已经远远地超过了CPU，基于GPU平台的数字图像并行处理技术成为了新的研究热点。目前在GPU平台上已经实现了图像配准，图像分割，图像检索，目标跟踪，超分辨率重建等图像处理算法。相比于CPU平台，算法的处理速度有了十几倍甚至几十倍的提高[13-16]。

本文采用英伟达（NVIDIA）公司推出的GPU，应用CUDA并行应用程序开发环境，采用并行数据传输、数据内存交互、内核指令优化、数据并行计算、设备异步执行等方法，在GPU平台上实现了Retinex图像增强算法的并行处理。

2 Retinex算法在GPU上的实现

2.1 Retinex理论

Retinex理论认为一副图像可以表示成反射分量 $R(i,j)$ 和照度分量 $L(i,j)$ 的乘积，即：

$$I(i,j)=R(i,j)\cdot L(i,j) \tag{1}$$

式中：(i,j) 表示像素点的空间二维坐标；$I(i,j)$ 表示原始图像；$R(i,j)$ 表示反射分量，反映物体本身的颜色特性，对应图像中高频的部分，主要包含一些细节信息；$L(i,j)$ 表示照度分量，反映环境的亮度，对应图像中低频的部分。Retinex的思想就是从原始图像中剔除环境亮度的影响，求解出物体本身的颜色特性。由于对数域下更接近人眼的视觉特性，将式(1)转换为对数域下求解得：

$$\log R(i,j)=\log I(i,j)-\log L(i,j) \tag{2}$$

照度分量 $L(i,j)$ 利用中心环绕函数求解，

$$L(i,j)=I(i,j)\times F(i,j) \tag{3}$$

式中：$F(i,j)$ 表示中心环绕函数，将高斯函数作为中心环绕函数，高斯函数比较平滑，具有低通滤波的效果，能够将图像中的高频信息滤除，保留图像中的低频信息，可以将低频信息作为图像照度分量的估计。高斯函数的形式如下：

$$F(i,j)=Ke^{-r^2/c^2} \tag{4}$$

式中：K 表示归一化系数；r 表示空间距离；c 表示标准差。由于单尺度Retinex算法不能同时保证颜色的真实性和丰富的动态范围压缩，在此基础上又提出了多尺度Retinex算法（MSR）。MSR通过设定高斯函数中不同的标准差数值，在不同尺度下求取反射分量的值，再进行加权平均。MSR算法公式如下，在实际运用中，可以根据需要来选择中心环绕函数的个数[17]。

$$R(i,j)=\sum_{k=1}^{N}W_k\{\log I(i,j)-\log(I(i,j)\times F_k(i,j))\} \tag{5}$$

式中：N 表示中心环绕函数的个数；W_k 表示权重系数。

2.2 并行程序开发流程

CUDA是英伟达公司专门为其生产的GPU推出的并行应用程序开发环境。在CUDA编程模型中，CPU端称作主机端，GPU端称作设备端，二者协同工作。主机端主要负责逻辑控制以及对设备端进行调度等串行部分的工作，设备端主要完成大数据量的并行计算工作[18]。在CUDA的执行模型中，Thread（线程）是可以并行执行的最小单位，线程分布于多个Block（并行块）中，同一个block中的线程可以进行通信，一个multiprocessor（流多处理器）内包含多个Bock。Thread通过执行相同的核函数，完成对数据的并行计算。在CUDA的内存模型中，存储器被分为多种类型，包括全局存储器，纹理存储器，常量存储器，共享存储器等。它们的存储空间依次递减，存储器访问速度依次加快[19]。如何合理地使用好这几种类型的存储器，是提升GPU性能的关键。

在整个Retinex算法的执行过程中，各个像素点的计算过程没有结果间的依赖关系，符合GPU中多线程并行计算的条件。算法中对像素点进行高斯环绕滤波这个步骤是最费时间的，假设图像的分辨率为$n\times n$，如果用CPU实现，计算高斯环绕滤波的时间复杂度为$O(n^4)$。采用GPU的多线程对这部分进行并行计算，一个线程负责计算一个像素点的高斯环绕滤波，算法的计算时间复杂度降为$O(n^2)$。考虑到线程在执行的过程中需要保证一定的负载量，同时也为了保证计算结果的连续性，将算法的其他步骤也交给GPU中的线程来执行。Retinex算法在GPU上的实现流程图如图1所示。

2.2.1 主机与设备数据传输

在进行Retinex算法并行计算之前，首先将图像数据从主机端的内存拷贝到设备端的全局存储器中。全局存储器是GPU中主要的存储空间，它的特点是存储量大，传输延迟大。

本文采用CUDA提供的合并访问机制来提高全局存储器的访问效率[20]，合并访问需要满足对齐和连续2个条件。对齐是指所分配的内存地址是对齐的。由于warp是线程最基本的调度单位，一个warp内有32个线程，每个线程可以访问一个4字节的数据。因此，对齐的要求是使申请的首地址是128 B的倍数。在定义Retinex算法中的结构体时，采用_align_(4)去约定成员变量的对齐方式。连续是指一个warp内的线程可以访问到连续的地址，如果访问的地址空间不连续，warp内的线程就会分几次来访问该段地址空间，进而降低性能。在Retinex算法的设计中，尽量为变量分配连续的地址空间。

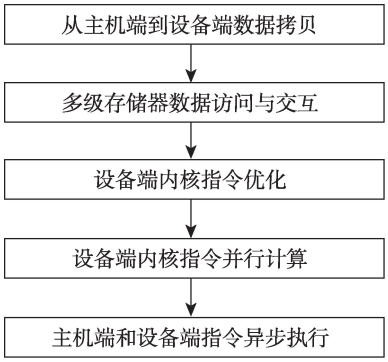

图1　Retinex算法在GPU上实现流程

Fig. 1　Flow chart of implementation of Retinex algorithm on GPU

2.2.2 存储器的使用

CUDA的存储模型中，包括多种类型的存储器。其中寄存器组和共享存储器都位于芯片的内部，因此它们具有最高的访问速度。常量存储器和纹理存储器都具有缓存机制，其访问速度都比全局存储器快。CUDA中存储器的数据访问关系如图2所示。

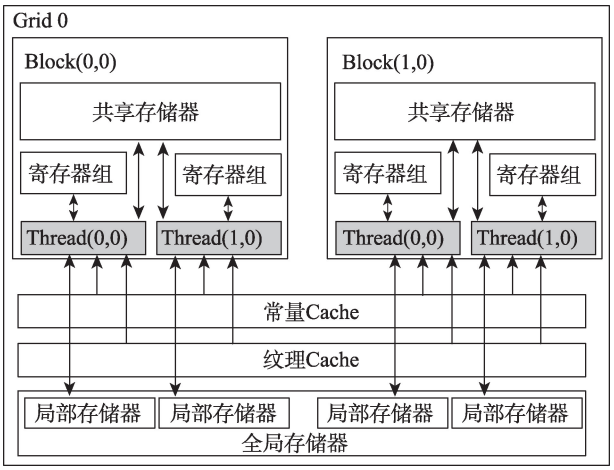

图2　CUDA存储器中的数据访问关系

Fig. 2　Relationship of data access in CUDA storage

Retinex算法中，在对每个像素点进行高斯环绕滤波时，需要重复读取各像素点的灰度值，像素点灰度值重复读取的次数和滤波网格的大小有关。在求取图像的反射分量时，也需要用原始图像的灰度值减去照度分量的灰度值，原始图像中的数据会被频繁用到。由于共享存储器具有很小的访问延迟，为了提高数据访问的效率，先把图像数据导入

到共享存储器中再进行计算。为了避免共享存储器中的bank冲突,使其空间大小的每个维度都能够被32整除。同时,使访问共享存储器的方式按照s_data[tid]=g_data[tid]的方式进行。

Retinex算法中,多数计算是在对数域下完成的,由于对数函数属于超越函数,计算十分耗费时间。为了提高计算效率,将对数运算制成一个查找表,需要对数计算结果时,查找此表即可。在求解照度分量和反射分量的时候需要大量地查找此表,需要不断地访问存储器。考虑到常量存储器具有缓存机制,当访问命中时,只有1个时钟周期的延迟。因此,将对数查找表存储在常量存储器中。常量存储器拥有64 KB的缓存,把0~255之间的数每隔0.1个单位求取对数运算的结果。存储对数表需要的存储空间为256×10×4=10 KB,满足常量存储器64 KB的最大空间使用要求。Retinex算法在GPU上的存储器使用过程如图3所示。

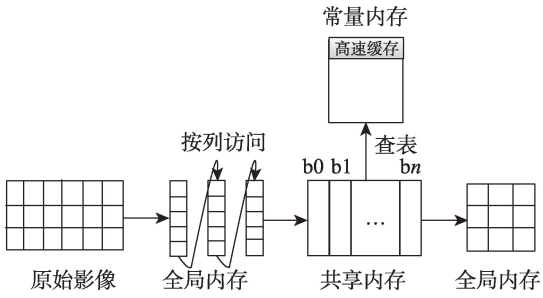

图3 Retinex算法的存储器使用

Fig. 3 Memory access of Retinex

2.2.3 内核指令优化

GPU的内核不具有复杂的逻辑控制单元,其只适合执行大量简单并行的指令,而不适合执行具有逻辑判断的分支指令和一些复杂的算术指令。GPU适合执行单精度浮点类型的指令,对于整数和双精度浮点类型的指令,GPU的计算能力比较弱[21]。

Retinex算法需要对每个像素点进行高斯卷积,计算高斯卷积时需要在循环体内进行乘积的累加操作。循环体存在条件分支,会降低内核程序的执行性能。本文使用CUDA提供的#pragma unroll等指令将循环体展开,从而避免条件分支。

在进行高斯函数卷积计算的过程中需要执行多次乘法操作,乘法指令的执行代价比移位指令的执行代价高。因此,本文用移位指令和加减指令的组合来代替乘法指令。同时,在并行计算的过程中使用单精度浮点类型的数据和指令来提高内核程序的执行效率。

2.2.4 数据并行计算

在CUDA的执行模型中,需要满足以下3方面的限制:① 一个Multiprocessor所能并行运行的Thread总数的限制;② 一个Multiprocessor所能并行运行的Block总数的限制;③ 一个Block所能并行运行的Thread总数的限制。一个Multiprocessor所能并行运行的Thread总数是确定的,为了获得Retinex增强算法最大的并行计算效率,合理地设计Block中所包含的Thread数目是关键。在对Retinex算法进行GPU实现的过程中,采用英伟达公司的计算能力为3.0的GPU。其每个Multiprocessor最多可以同时运行的Thread数目为2048。由于一个Warp一次调度32个Thread。因此,为了提高并行计算效率,一个Block所包含的thread数应该是32的倍数。同时考虑Block中寄存器等资源的限制,设计Block的尺寸为32×32=1024。每个Multiprocessor运行2048/1024=2个block,小于Blocks/Multiprocessor=16的限制,GPU中的所有内核都可以同时处于并行计算的状态。

在Retinex算法的内核程序设计中,通过索引来指导内核程序完成并行计算,索引通过下面的式子获得:

$$index = BlockIdx.x \times BlockDim.x + ThreadIdx.x \quad (6)$$

式中:*BlockIdx.x*代表*Block*在*Multiprocessor*中的索引;*BlockDim.x*代表*Block*的维度,即一个*Block*所包含的*Thread*数目;*ThreadIdx.x*代表*Thread*在*Block*中的索引。

2.2.5 异步执行模式

使用CUDA流可以实现任务级的并行。CUDA流是一系列顺序执行的指令,可以是主机端不同的线程发射的。流之间可以是乱序执行或并行执行的。例如,当GPU在执行核函数的同时,还可以利用CPU完成主机端和设备端的数据传输。利用CUDA的流机制来对Retinex算法的GPU程序进行优化。

主机端依次创建流0和流1,2个流均包含从主机向设备传输数据,并行计算,从设备向主机传输数据这3个任务,3个任务被平均分配给两个流执行。不同流中的任务在没有资源冲突的前提下可以并行执行。在Retinex算法的执行过程中,数据传输需要占用PCI-E总线,并行计算需要占用计算单元,二者使用不同的资源,因此数据传输和并行计算可以并行执行。用CUDA流机制实现Retinex

增强算法任务间并行的过程如图4所示。假设3个任务的执行时间分别为T1、T2、T3，使用异步模式之前3个任务执行的总时间为T1+T2+T3，使用异步模式之后任务执行的总时间缩短为T1/2+max（T1/2，T2/2）+ max（T2/2，T3/2）+T3/2。

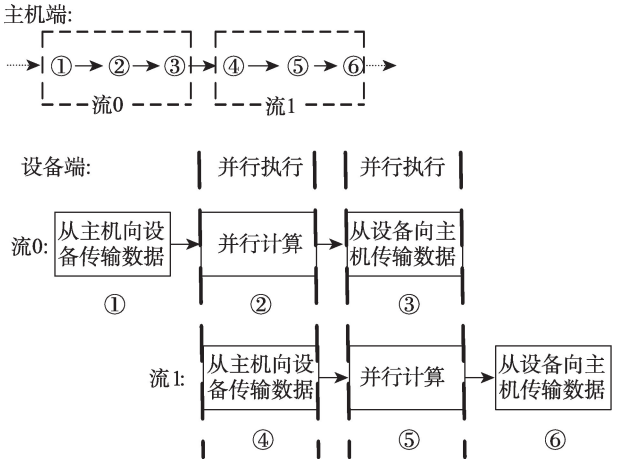

图4　Retinex算法任务间并行

Fig. 4　Tasks parallel of Retinex algorithm

3　实验及结果分析

实验平台的GPU采用NVIDIA公司的GeForce GTX 780 M，实验平台的CPU采用Intel i5处理器，内存4 G，测试程序在VS2010环境下编译完成。实验分别对分辨率为1024×1024（图5(a)）、2048×1050（图5(c)）、2048×2048（图5(e)）的低对比度图像在

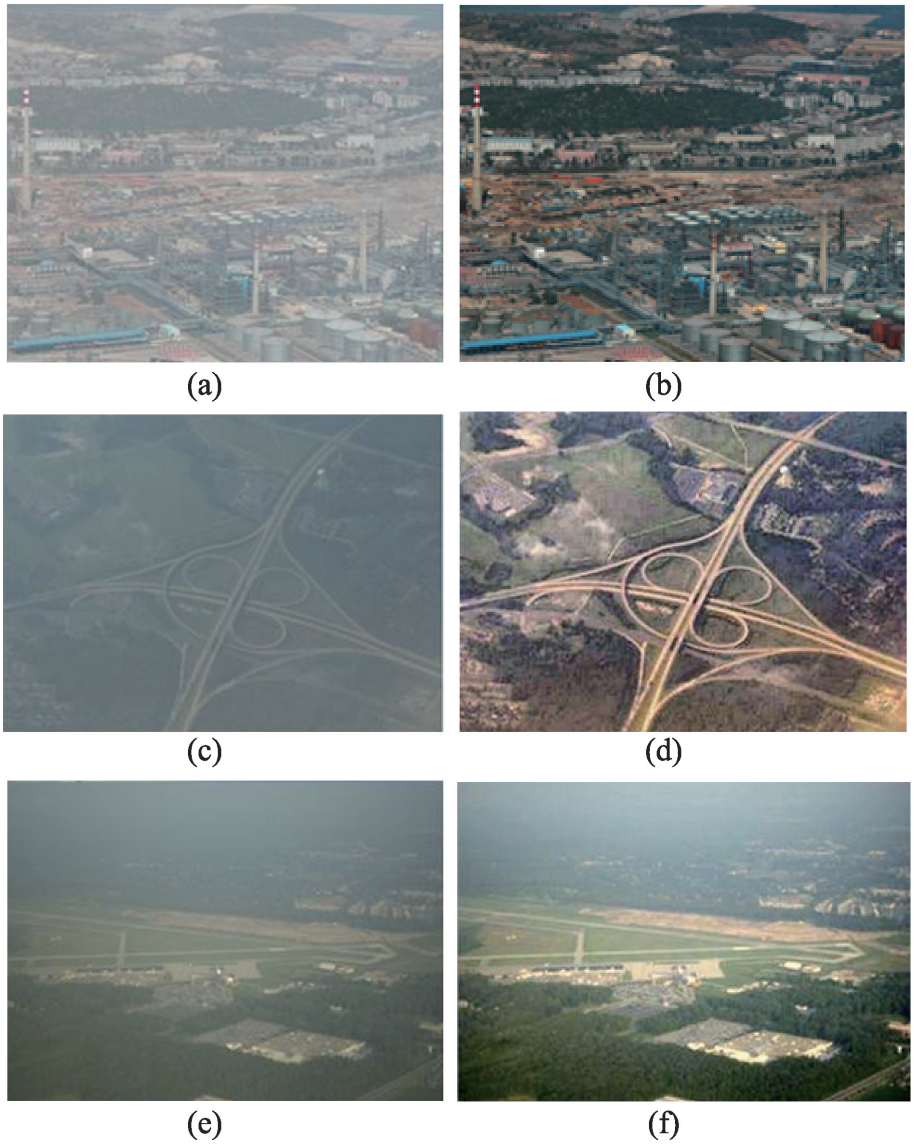

图5　图像增强效果

Fig. 5　Results of image enhancement

GPU平台上进行Retinex图像增强，并对增强后的图像进行质量评价。采用信息熵、图像均值、图像标准差这3个指标对图像进行质量评价。其中信息熵反映图像中包含的信息，其值越大表示图像包含的信息越多。图像均值反映图像的平均亮度，其值越大表示图像的平均亮度越大。图像标准差反映图像的对比度，其值越大表示图像的对比度越大。图5(b)、图5(d)、图5(f)分别为对应的增强后的图像。表1是对原始图像和增强后的图像的质量评价。

在CPU平台上，Retinex增强算法处理一帧图像的时间如表2所示。从表中的数据可以看出，在GPU平台上，Retinex算法的处理速度提高了数十倍。随着图像分辨率的增大，GPU的并行计算优势越来越明显。当图像的分辨率达到2048×2048时，在GPU平台上算法处理一帧图像的时间为38.04 ms，1 s可以处理26帧的图像。

表1 图像质量客观评价

Tab. 1 Objective evaluation of image quality

	信息熵	图像均值	图像标准差
原始图像1	6.7452	138.2679	45.1426
增强图像1	7.1960	157.8206	60.9217
原始图像2	5.8062	125.9326	40.8128
增强图像2	6.4915	136.2894	65.9286
原始图像3	6.2948	140.5862	51.8146
增强图像3	6.5042	150.6439	69.2931

表2 Retinex算法在不同平台上运行时间测试结果

Tab. 2 Results of time test for Retinex algorithm on different platform (ms)

图像大小	1024×1024	2048×1050	2048×2048
CPU处理时间①	395.80	816.73	1572.59
GPU处理时间②	13.62	23.15	38.04
①/②	29.06	35.28	41.34

4 结论

本文采用CUDA并行应用程序开发环境，在GPU平台上实现了Retinex图像增强算法，主要结论如下：

(1)本文充分发挥了GPU多线程并行计算的优势，采用并行数据传输、数据内存交互、内核指令优化、数据并行计算、设备异步执行等方法，在GPU平台上实现了Retinex图像增强算法。

(2)实验分别对1024像元×1024像元、2048像元×1050像元、2048像元×2048像元等高分辨率图像进行测试，在GPU平台上，Retinex算法的处理速度均有数十倍的提高。对于分辨率为2048×2048的图像，Retinex算法的处理速度提高了40倍，处理一帧图像的时间只需要38 ms。

本文提出的Retinex算法在GPU平台上的实现方法具有较强的通用性，对于其他图像处理算法在GPU平台上的实现有一定的借鉴意义。

参考文献(References)：

[1] 席诗琼,韩胜,耿卫东.一种指纹奇异点区域图像增强算法[J].液晶与显示,2018,33(9):801-807. [Xi S Q, Han S H, Geng W D. Fingerprint singularity region image enhancement algotithm[J]. Chinese Journal of Liquid Crystals and Displays, 2018,33(9):801-807.]

[2] 朱遵尚.图像增强技术研究[D].长沙:国防科学技术大学,2009. [Zhu Z S H. Research on image enhancement technique[D]. Changsha: National University of Defense Technology, 2009.]

[3] 施慧慧,王妮,滕文秀,等.结合Gabor小波和形态学的高分辨率图像树冠提取方法[J].地球信息科学学报,2019,21(2):249-258. [Shi H H,Wang N, Teng W X, et al. Tree canopy extraction method of high resolution image based on Gabor filter and morphology[J]. Journal of Geo-information Science, 2019,21(2):249-258.]

[4] 王竞雪,崔昊.局部点、线仿射不变性约束的近景影像直线段匹配[J].地球信息科学学报,2019,21(2):137-146. [Wang J X, Cui H. Line segment matching based on Local point-line affine invariance constraints for close-range image[J]. Journal of Geo- information Science, 2019,21(2):137-146.]

[5] 张雄星,王伟,刘光海,等.温度场纹影定量测量技术[J].中国光学,2018,11(5):860-873. [Zhang X X, Wang W, Liu G H,et al. Quantative measuring technique for the temperature of flow fields in schlieren systems[J]. Chinese Optics, 2018,11(5):860-873.]

[6] 刘林涛,董雪莹,刘俊,等.基于时空上下文和随机森林的人眼跟踪定位算法研究[J].液晶与显示,2018,33(5):443-449. [Liu L T, Dong X Y, Liu J, et al. Human eye locating and tracking using space-time context and random forest[J]. Chinese Journal of Liquid Crystals and Displays, 2018,33(5):443-449.]

[7] Wharton E, Agaianb S, Panetta K. A logarithmic measure of image enhancement[C]. Proceeding. SPIE 2006 Defense and Security Symp, Orlando, FL, 2006,6250: 62500P.

[8] 陈莹,朱明.多子直方图均衡微光图像增强及FPGA实现[J].中国光学,2014,7(2):225-233. [Chen Y, Zhu M. Multi-

ple sub-histogram equalization low light level image enhancement and realization on FPGA[J]. Chinese Optics, 2014,7(2):225-233.]

[9] 吴笑天,鲁剑锋,贺柏根,等.雾天降质图像的快速复原[J].中国光学,2013,6(6):892-899. [Wu X T, Lu J F, He B G, et al. Fast restotation of haze-degraded image[J]. Chinese Optics, 2013,6(6):892-899.]

[10] 洪平.基于RETINEX理论的图像去雾研究[D].上海:上海交通大学,2013. [Hong P. Research of restoration of haze image based on Retinex[D]. Shanghai: Shanghai Jiao Tong University, 2013.]

[11] 赵宏宇,肖创柏,禹晶,等.马尔科夫随机场模型下的Retinex夜间彩色图像增强[J].光学精密工程,2014,22(4):1048-1055. [Zhao H Y, Xiao C B, Yu J, et al. A Retinex algorithm for night color image enhancement by MRF[J]. Optics and Precision Engineering, 2014,22(4):1048-1055.]

[12] 马永杰,宋晓凤,李雪燕,等.基于嵌入式车流量实时检测算法研究与实现[J].液晶与显示,2018,33(9):787-792. [Ma Y J, Song X F, Li X Y, et al. Research and implementation of real-time vehicle flow detection algorithm based on embedded system[J]. Chinese Journal of Liquid Crystals and Displays, 2018,33(9):787-792.]

[13] 黄先楼.基于Normalized Cut的图像分割及其CUDA并行实现[D].北京:北京交通大学,2014. [Huang X L. Image segmentation based on normalized cut and CUDA parallel implementation[D]. Beijing: Beijing JiaoTong University, 2014.]

[14] 赵中兴.基于GPU的图像检索与重建技术研究[D].厦门:厦门大学,2014. [Zhao Z X. Research on image retrieval and reconstruction technology based on GPU[D]. Xiamen: Xiamen University, 2014.]

[15] 张平.基于CUDA的TLD视觉跟踪算法研究[D].北京:北京交通大学,2014. [Zhang P. Study of TLD visual tracking algorithm based on CUDA[D]. Beijing: Beijing Jiao Tong University, 2014.]

[16] 刘超.快速超分辨率重建方法研究与实现[D].杭州:杭州电子科技大学,2014. [Liu C. Study and implementation of fast super-resolution reconstruction methods[D]. Hangzhou: Hangzhou Dianzi University, 2014.]

[17] 陈志斌,张超,宋岩,等.灰度拉伸Retinex在大动态范围烟雾图像增强中的应用[J].红外与激光工程,2014,43(9):3146-3150. [Chen Z B, Zhang C, Song Y, et al. Application of Retinex with grayscale stretching in large dynamic range smoke image enhancement[J]. Infrared and Laser Engineering, 2014,43(9):3146-3150.]

[18] Lu M.Fast implementation of scale invariant feature transform based on CUDA[J]. Applied Mathematics & Information Sciences, 2013,7(2):717-722.

[19] Julien M, Yann D, Andre D. Real-time dense and accurate parallel optical flow using CUDA[C]. The 17th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision.Bory，Czech Republic, 2009:105-111.

[20] 李晔,于双元,罗四维.基于改进的压入与重标记算法的图割在GPU上的实现[J].计算机科学,2014,41(1):64-68. [Li Y, Yu S Y, Luo S W. Realization of graph cuts based on improved push-relabel algorithm on GPU[J]. Computer Science, 2014,41(1):64-68.]

[21] Fresse V, Houzet D, Gravier C. GPU architecture evaluation for multispectral and hyperspectral image analysis[C]. Design and Architectures for Signal and Image Processing (DASIP)，2010 Conference on. IEEE，2010:121-128.