第 42 卷 第 6A 期
2015 年 6 月

计 算 机 科 学
Computer Science

Vol. 42 No. 6A
June 2015

基于 CUDA 的加速双边滤波算法

曾炫杰 陈 强 谭海鹏 牛四杰 孙权森

(南京理工大学计算机科学与工程学院 南京 210094)

摘 要 双边滤波算法在去噪的同时能够很好地保持图像边缘细节信息,因此在图像处理领域中得到了广泛的应用。但双边滤波算法复杂度较高,无法满足实时处理的要求。因此首先对双边滤波算法进行分析研究,提出了基于 CUDA 的并行化双边滤波算法;其次,根据 CUDA 的特性对算法进行优化;最后,通过实验分析证明,该方法能够在不改变双边滤波的效果下,使加速比达到 75 以上。

关键词 CUDA,GPU 加速,快速双边滤波,并行计算

中图法分类号 TP391.4 文献标识码 A

CUDA-based Acceleration Algorithm of Bilateral Filtering

ZENG Xuan-jie CHEN Qiang TAN Hai-peng NIU Si-jie SUN Quan-sen

(School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China)

Abstract Bilateral filtering is a good de-noising algorithm while preserving image edge details, so it has been widely used in the field of image processing. The algorithm complexity of bilateral filtering is so high that it cannot meet the requirements of real-time processing. In this paper, by analyzing the bilateral filtering algorithm, a parallelization bilateral filtering algorithm based on CUDA was proposed, and according to the characteristics of CUDA, the proposed algorithm was optimized further. Experimental results and analysis prove that our method can achieve a speedup up to 75 or more under keeping the traditional bilateral filtering effect.

Keywords CUDA, GPU acceleration, Fast bilateral filtering, Parallel computing

1 引言

图像传感器在工作的过程中会受到各种各样的因素影响,如获取图像的环境传感元器件自身质量等,因此,数字图像在形成的过程中会受到多种噪声的污染。这些噪声会对图像后期处理造成很大的影响。如何在能够保持图像原本信息的同时滤除这些噪声就显得十分重要。传统的滤波器在去除噪声的同时也会对图像的边缘造成模糊。为了解决这个问题,Tomasi 和 Manduchi 在 1998 年提出了双边滤波算法[1]。与高斯滤波类似,双边滤波也采用了局部加权平均的思想,不同的是双边滤波在计算权重的过程中加入了像素值相似度这一项。因此双边滤波在去噪的过程中能够很好地保留图像的边缘信息,更加符合人眼的视觉习惯,在解决图像处理领域中的一些实际问题时取得了很好的效果,目前广泛应用在图像去噪、图像分割、图像复原、光流估计等领域。双边滤波算法通过实践证明是非常有效的,但它的计算时间较长,无法应用到很多需要实时处理的图像任务当中。如何快速实现双边滤波算法就变得十分重要。

目前,很多国内外的学者都对快速双边滤波算法进行了研究。其中 Durand 和 Dorsey 提出分段线性(Piecewise-linear)型双边滤波,将非线性的双边滤波器线性化,分段的灰度域核函数与图像相乘后,再与空间域核函数卷积,使用灰度域的分段线性近似和适当的采样加速双边滤波,算法速度得到很大的提升,但是滤波精度不够理想[2]。Porikli 提出一种通过改变图像直方图的表示方式来加速双边滤波的方法,同时在任意位宽的图像、任意大小的滤波核的情况下其时间复杂度都为 $O(\log n)$[3]。Weiss 利用迭代掩模理论在逐点滤波的过程中对掩模只进行细微的调整,它是一种利用近似的思想来实现双边滤波算法,算法运算速度快,且掩模大小对计算复杂度影响不大[4]。Pham 和 Vliet 提出了可分离型高斯滤波,用一维的双边滤波器分别对垂直和水平方向进行滤波,该算法在窗口较小的时候运行速度比较快[5]。Paris 和 Durand 提出了基于快速傅里叶变换的加速方法,该方法通过二维快速傅里叶变换减少计算量[6]。Shin Yoshizawa 等人利用快速高斯变换的方法实现了双边滤波的加速,而且实验表明加速效果优于 Paris、Porikli、Pham 和 Vliet 等人的方法[7]。Kunal N. Chaudhury 等人提出了基于三角函数关系的值域核算法,通过余弦函数来逼近高斯值域函数,实现了对双边滤波的加速[8]。

也有一些文章针对国外快速双边滤波算法进行改进。张志强等人[9]针对 Paris 和 Durand 所提算法提出改进方法,通过对双边滤波器线性化和图像矩阵进行映射,再由快速傅里叶变换完成线性卷积,然后将计算结果逆映射还原为图像矩阵,最后依据图像的原始坐标和灰度值的差异进行像素补值。

本文受中央高校基本科研业务费专项基金(30920140111004),中国航天科技集团公司航天科技创新基金(CASC05131418),国防 973(61321001)资助。

曾炫杰(1989—),男,硕士生,主要研究方向为图像处理与计算机视觉、GPU 高性能计算,E-mail:lonju@163.com。

该方法避免了插值过程，提高了效率。改进的双边滤波器在滤波精度与传统双边滤波器相仿的同时，运算时间为传统双边滤波器的 4% 左右，但是相对于原始算法信噪比比较低。李俊峰等人[10]根据灰度差值划分图像的像素层，然后在像素层上选择标识点，并利用标识点计算像素层的滤波值，最后通过线性插值获得各像素点的滤波值，并输出滤波图像。

CUDA(Compute Unified Device Architecture)即统一计算设备架构，是 NVIDIA 公司推出的一种利用 GPU(Graphic Processing Unit)图形处理器进行通用并行计算的架构。相较于 CPU(Central Processing Unit)中央处理器，GPU 专为密集型、高度并行化的计算而设计，且在浮点运算方面能力更强，这样巨大的优势使得 CUDA 在矩阵计算、图像视频处理、机器学习、计算机可视化等领域都有着良好的应用前景。在图像处理领域，近些年来有很多国内外学者都尝试利用 CUDA 来优化自己的算法。在医学图像方面，Anders Eklund 认为 GPU 可以大大加速并行计算，而且价格低廉，高效节能，非常适合用于解决需要大量计算的医疗成像问题[11]；在视频图像处理方面，Yu-Shan Pai 等人利用 CUDA 来加速 LDPC 解码，最终可以达到 46.52 的加速比[12]。在目前对双边滤波进行加速的研究中都没有考虑到双边滤波算法中存在大量可以并行计算的部分以及利用 CUDA 在并行计算方面巨大的优势。本文主要利用 CUDA 对双边滤波进行并行化加速，同时根据架构本身的特性对算法进行优化，最高加速比能达到 75 以上。

2 双边滤波

在双边滤波器中，输出像素的值由其邻域内像素值通过加权平均来获得。令 $f(x,y)$ 表示一幅图像 f 在 (x,y) 处的像素值，则双边滤波的表达式为

$$f(x,y)=\frac{\sum_{(i,j)\in R(m,n)} f(i,j)w(x,y,i,j)}{\sum_{(i,j)\in R(m,n)} w(x,y,i,j)} \tag{1}$$

式中，$R(m,n)$ 表示输出像素点周围大小为 $(2N+1)\times(2N+1)$ 的邻域，$w(x,y,i,j)$ 表示邻域内每个像素点的权重系数。在双边滤波中，权重系数是由值域高斯函数 $w_r(x,y,i,j)$ 和空域高斯函数 $w_d(x,y,i,j)$ 这两个部分的乘积所组成。

$$w(x,y,i,j)=w_d(x,y,i,j)\times w_r(x,y,i,j) \tag{2}$$

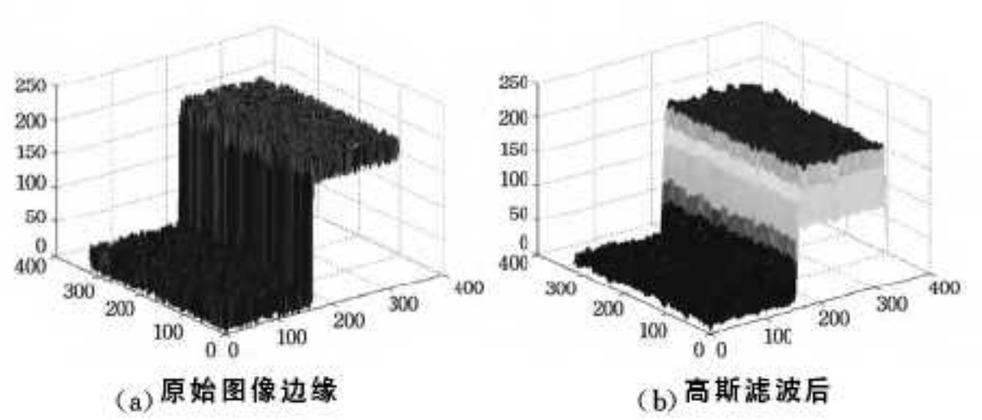
$$w_d(x,y,i,j)=e^{\left(-\frac{(x-i)^2+(y-j)^2}{2\sigma_d^2}\right)} \tag{3}$$

$$w_r(x,y,i,j)=e^{\left(\frac{-\|f(x,y)-f(i,j)\|^2}{2\sigma_r^2}\right)} \tag{4}$$

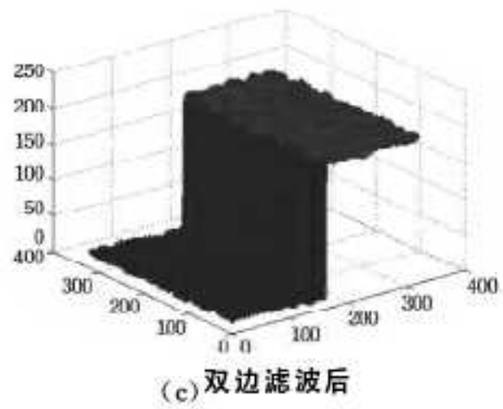
从式(3)、式(4)可以得出空域高斯函数与像素点亮度值的变化无关。在图像低频区域中，像素点亮度值的变化不明显，值域高斯函数近似等于 1，在这个区域内双边滤波相当于高斯低通滤波；而在图像的高频区域中，像素点亮度值变化剧烈，值域高斯函数很小，相应的权重系数很小，因此双边滤波很够很好地保存图像边缘等高频信息部分，这从图 1、图 2 中得到了直观体现。

(a)原始图像边缘　(b)高斯滤波后　(c)双边滤波后

图 1　双边滤波和高斯滤波对图像缘边的影响



(a)原始图像边缘　(b)高斯滤波后



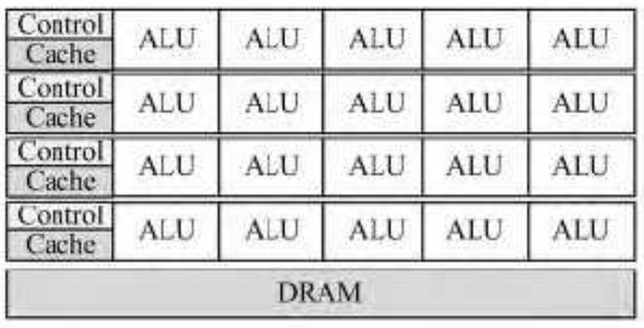
(c)双边滤波后

图 2　灰度图像 3D 显示

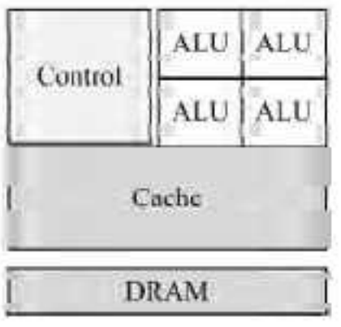
3 基于 CUDA 的双边滤波

3.1 GPU 计算优势

GPU 最初用于 3D 图形处理，帮助 CPU 分担计算量。经过近些年的发展，GPU 在通用计算领域所表现出来的能力已经得到了越来越多的关注。相比于 CPU，GPU 的优势在于：在浮点计算能力方面，GPU 是由数以千计的更小、更为高效、高度并行计算(如图像渲染)所设计的核心组成，这些核心都是用于数据处理，而非 CPU 中的数据缓存和控制，如图 3 所示。在并行计算方面，GPU 中可以运行大量的线程，非常适合处理同一个程序在多个数据上面的执行的并行计算问题。



(a)CPU



(b)GPU

图 3　与 CPU 相比 GPU 有更多的运算单元

另外 GPU 的造价和功耗相对较低，在一定程度上能够满足那些需要计算海量数据而无法使用昂贵的巨型计算机的用户的需求。

3.2 CUDA 编程模型

GPU 拥有很强的通用计算能力，但是单纯地将其应用于计算机领域仍需要解决很多技术难题，例如用于与 GPU 进行交互的接口十分复杂，并且 DRAM 内存存在较大的局限性。此外，研究人员为了能够使用 GPU 还需要掌握复杂的 Shading Language(着色语言)和计算机图形学。这些都限制了 GPU 在通用计算方面的应用。为了解决上述问题，NVIDIA 公司设计出了 CUDA，这种模型能够大幅度提升 GPU 的计算速率，同时让更多的研究人员使用 GPU 进行通

用计算。

CUDA 模型采用单指令流多数据流(Single Instruction Multiple Data,SIMD)执行模式,CPU 作为控制中心,负责指令控制和处理,GPU 视为一个计算设备,是 CPU(也可以称为主机)的协处理器,在 GPU 中可以并行地运行大量的线程,它可以用来承担高度并行的计算。CPU、GPU 都拥有自己独立的存储空间。

一个 CUDA 程序包含在 CPU 中执行的部分和在 GPU 中执行的部分,如图 4 所示,两个部分之间的工作方式是串行的。在 CPU 中执行的部分称为普通函数,承担着数据的初始化、输入、内存分配、数据传递、结果显示与保存。在 GPU 中执行的部分称为核函数(kernel),是这个程序中能够大量并行执行的部分。在 CUDA 模型中,CPU 把所有需要计算的数据准备完成后复制到 GPU 内存中,再由 GPU 中的核函数完成对数据的计算,最后再将计算结果从 GPU 的内存复制到主机内存中。

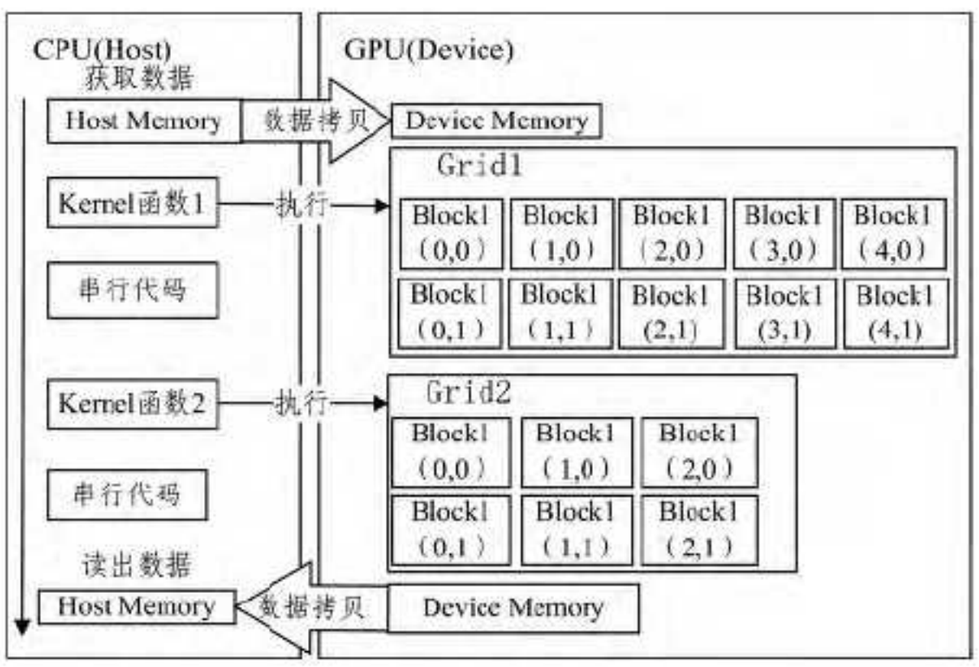


图 4 CUDA 编程模型

3.3 并行化双边滤波

在对双边滤波器的分析中,根据输入图像矩阵 f 和卷积模板 G 来计算滤波后的图像 f',其中 l 代表图像的像素点个数。

$$f'=\sum_{i=1}^{l} f * G_i \tag{5}$$

由于值域高斯函数的卷积模板对于每一个像素都不同,因此需要对每一个点都要重新计算一次。模板的计算在 CPU 中占用了大量的时间。假设一幅图像的大小为 $M\times N$,滤波窗口大小为 K,那么完成双边滤波运算一共需要 $M\times N\times(2K+1)^2$ 次循环。通过对算法的研究,我们发现在整个双边滤波算法中像素点之间的数据依赖性非常小,可并行化程度高,因此非常适合在 GPU 上运算。在并行化算法中不需要循环语句来逐个对像素点进行计算,只需要设计一个内核程序,根据图像的大小分配足够的线程,再利用转换函数把每一个线程的 ID 号对应到每一个像素点上,一个线程完成一个像素点的计算,通过大量线程的并行来完成算法。式(6)中 i 代表线程的 ID。

$$f_i'=f_i * G_i \tag{6}$$

具体算法流程如下:

1. CPU 获取图像数据,在主存中初始化变量 Input 和 Output,分别用来保存输入的图像数据和输出的图像数据;

2. 申请 GPU 内存中的 Global Memory 空间,用来保存需要计算的数据;

3. 将主存中 Input 复制到 GPU 中;

4. 通过图像大小计算需要的线程数,为了让并行化算法达到最高效率、线程数与图像大小一致,一个线程处理一个像素点的计算。然后在 CPU 中调用核函数实现双边滤波算法,实现方法如式(5)与式(6)所示;

5. 将计算结果从 GPU 中复制到主存内;

6. 保存计算结果到 Output 中。

3.4 基于 CUDA 的并行算法优化策略

如果只是单独地把并行化后的算法放到 CUDA 架构上运行,虽然也能得到比较大的计算速度的提升,但是它的没有考虑到架构本身的一些特性,会损失很大一部分的计算性能。本文针对 CUDA 的一些特性,提出几点优化策略。

数据访问优化:

在 CUDA 存储器架构中,全局存储器空间最大,几乎相当于整个显存,图像数据一般放在全局存储器中。但是它的访问的速度是最慢的,延迟在几百个时钟。共享存储器的容量很小,一般 16kB。它的访问速度非常快,一到两个时钟就可以完成读写。在双边滤波算法中,高斯空域函数的模板不变,可以先在 CPU 完成模板的计算,再复制到 GPU 的共享存储器中,在计算中读取共享存储器中的数据可以提高计算性能。

数据传输优化:

在计算数据量增大而算法复杂度不变的情况下,GPU 加速的效率会逐渐降低。这是因为在 GPU 进行计算之前必须先等待主存中的数据复制到 CPU 中,之后才开始计算。在图像数据较大的情况下,数据复制的时间会大于核函数在 GPU 中计算的时间。以彩色图像为例,由于彩色图像有 3 个颜色通道,在同样的长度和宽度下,彩色图像所需要的存储空间是灰度图像的 3 倍左右,这导致了 CPU 到 GPU 的传输时间会相对较长,从而影响整体算法加速的效果。在实际计算中,3 个颜色通道是可以分开进行计算的,把 3 个颜色通道作为 3 个图像矩阵数据,利用 CUDA 事件流的机制[13]在数据拷贝的同时,在 GPU 中进行计算,可提高计算/拷贝率。具体模型如图 5 所示。其中 Stream0、Stream1、Stream2 分别对应图像 R、G、B 3 个通道的计算。

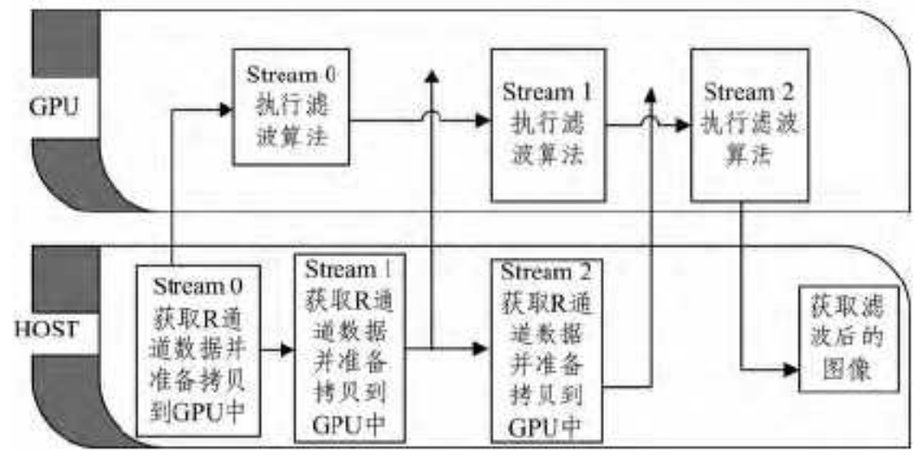


图 5 CUDA 事件流

指令优化:

与 CPU 不同,目前 GPU 的单精度运算性能要远远超过它的双精度性能,虽然双精度能够获得更可靠的运算结果,但在双边滤波算法中并不需要像素值的计算结果非常的精确。因此,采用单精度类型来计算能够在允许的误差范围内完成给定的计算任务,同时能提高计算性能。CUDA 采用 SIMT(单指令多线程)架构,在 SMIT 中指令按照代码的顺序来执行,由于没有分支预测功能,当算法当中有大量的逻辑判断时,就会导致计算性能大幅下降。因此可以通过 #pragma unroll 来告诉编译器可以展开循环语句,减少了每次循环的

条件判断，从而提升计算性能（对于一幅 $M\times N$ 的图像而言，能够省略 $M\times N$ 次条件判断）。由于 GPU 不具有分支预测等复杂的流控制单元，因此对于分支语句而言计算效率会比较差，可以通过编程手段来减少条件判断语句。

4 实验结果

为了验证本文所提方法的性能优势，分别从算法效果和时间性能方面与标准双边滤波[1]以及 Fast O(1) 双边滤波[8]做对比实验。实验环境：硬件平台 PC 机，Xeon E5530 CPU，16GB 内存，NVIDIA FX4800 GPU。软件环境是 Windows7，Visual studio 2010，CUDA 5.0。为了获得最准确的结果，文中所有测试都在空闲的系统中执行

4.1 算法效果

实验选择 4 幅 512×512 8 位灰度图像进行测试，图像分别是 lenna、man、boat、couple。为了测试算法的去噪效果，对这 5 幅图像添加标准差为 0.1 的高斯噪声。不同的算法对同一幅图像 σ_r、σ_s 分别取(3,50)和(3,70)，对不同算法的结果分析对比。本文通过用峰值信噪比(PSNR)来定量分析去噪效果。如式(7)所示，其中 f 表示噪声图像，f' 表示原图像，N 表示图像的像素大小。

$$PSNR(f,f')=10\log_{10}\left(\frac{N\times(255)^2}{\sum_{i=1}^{N}(f(i)-f'(i))^2}\right) \tag{7}$$

图 6、图 7 为不同 σ_r、σ_s 下标准双边滤波、快速双边滤波以及 Fast O(1) 双边滤波后的结果对比，括号中的为滤波参数，从左到右依次表示 σ_s、σ_r。图 6 中(a)表示原始噪声图像，图(b)—图(d)分别表示在 $\sigma_s=3$、$\sigma_r=50$ 下标准双边滤波和本文算法以及 Fast O(1) 双边滤波的结果。图 6(e)—图 6(g)分别表示在 $\sigma_s=3$、$\sigma_r=70$ 下标准双边滤波、本文算法以及 Fast O(1) 双边滤波的结果。从定量分析和定性分析中可以看出，本文算法和标准双边滤波算法的去噪效果是一致的，同时去噪性能方面要好于 Fast O(1) 双边滤波。

(a)带噪声图像 PSNR:19.65dB；(b)标准双边滤波(3,50) PSNR:27.83dB；(c)Fast O(1) 双边滤波(3,50) PSNR:27.36dB；(d)本文算法(3,50) PSNR:27.83dB

(e)标准双边滤波(3,70) PSNR:28.49dB；(f)Fast O(1) 双边滤波(3,70) PSNR:27.75dB；(g)本文算法(3,70) PSNR:28.49dB

图 6 算法效果

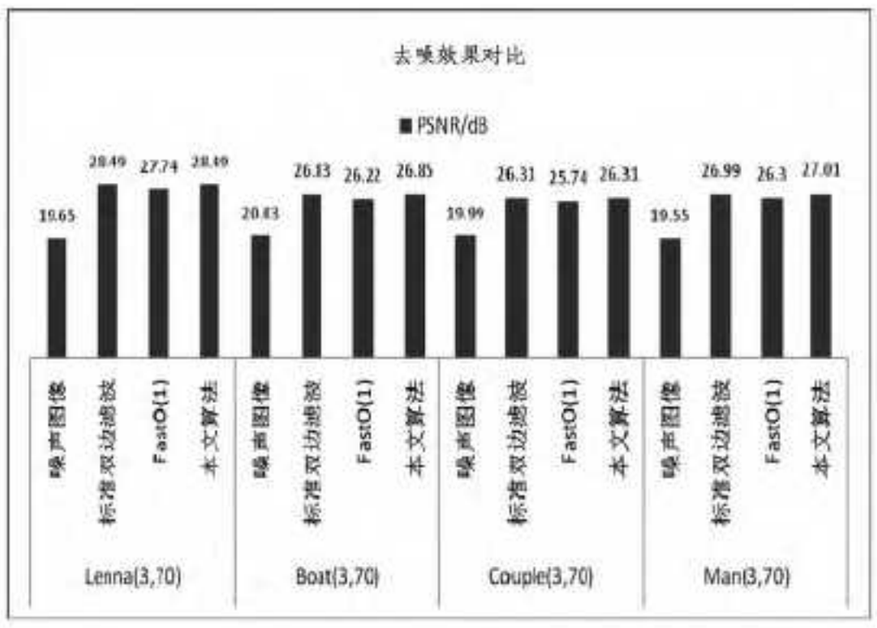


图 7 算法在各个图像中去噪性能对比

4.2 时间性能

双边滤波算法的执行时间受到掩模窗口和图像大小的影响。为了证明本文算法在时间性能上要远远优于标准双边滤波算法，同时也好于 Fast O(1) 双边滤波，实验分别对大小为 512×512 和 1024×1024 的图像在掩模窗口等于 3、5、7 下进行时间性能测试，最终时间取 10 次计算结果的平均值，实验结果如表 1 所列。

表 1 计算时间对比

图像大小	512×512			1024×1024		
掩模窗口	7	3	5	7	3	5
标准双边滤波(ms)	3274	7749	15172	12280	29603	55961
Fast O(1) 双边滤波(ms)	最大:1230 最小:236 平均:449			最大:4674 最小:873 平均:1840		
本文算法(ms)	73.442	186.8	344.5	293.2	734.7	1350
优化后本文算法(ms)	40.765	101.7	189.6	158.2	393.5	734.3
加速比(优化前/优化后)	45/80	42/76	44/79	43/77	40/75	42/76

Fast O(1) 双边滤波是利用余弦函数来逼近高斯值域函数，从而实现对标准双边滤波的加速。因此，算法的执行时间与掩模窗口无关，但是算法的循环次数在一定范围内会随着 σ_r 的变小而增大，时间也会相应变长。当 $\sigma_r>40$ 时算法的循环次数基本不再随着 σ_r 的变化而变化，而当 $\sigma_r<20$ 时算法已经不再具有优化的效果，如图 8 所示。为了与 Fast O(1) 双边滤波进行时间性能对比，我们选取不同的 σ_r 值(5,10,20,30,40,50,60,70,80,90,100)下 Fast O(1) 双边滤波的计算时间，

最后统计算法的最大、最小以及平均计算时间。

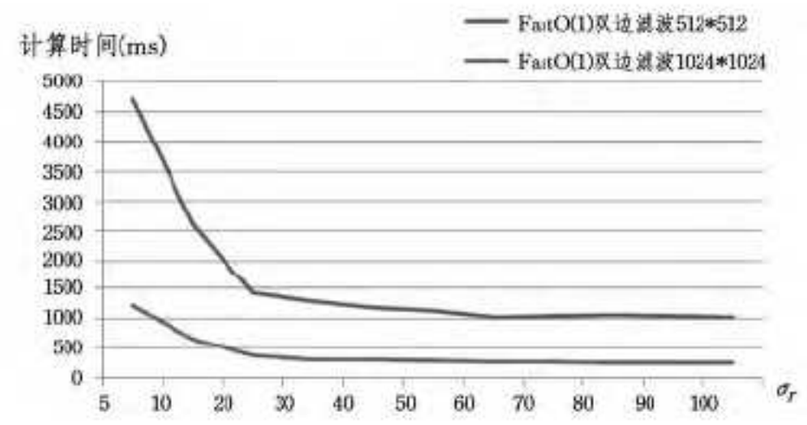


图 8 Fast O(1) 双边滤波时间性能

从表 1 中可以看出，本文算法的最大计算时间都小于 Fast O(1) 双边滤波的计算时间的最小值，因此在时间性能方面要好于 Fast O(1) 双边滤波，同时也远远好于标准双边滤波。

当图像大小增加时，本文的加速效果会有所减弱，这是由于随着图像像素点数的增加，需要 GPU 启动更多的线程来投入到计算中，启动越多的线程投入到计算中意味着更多的时间开销。未经过优化的快速算法加速效果能达到 40 倍左右，经过优化后的算法可以达到 75 倍左右。

结束语 本文针对以往对双边滤波的加速算法没有利用到 CUDA 在并行计算方面巨大的优势的情况，从硬件和编程架构的角度出发，通过将标准双边滤波算法并行化，再根据 CUDA 本身的架构特性，对并行的算法进行优化。与现有的快速双边滤波算法相比，本文的快速算法能够在去噪性能不变的情况下，利用 GPU 的大量线程完成双边滤波的计算，节省了大量的计算时间。实验结果表明，本文实现的快速双边滤波算法在 512×512 和 1024×1024 图像上运算的速度比原始的标准双边滤波算法快 75 倍左右。

参 考 文 献

[1] Tomasi C, Manduchi R. Bilateral filtering for gray and color images[C]// Sixth International Conference on Computer Vision, 1998. IEEE, 1998: 839-846

[2] Durand F, Dorsey J. Fast bilateral filtering for the display of high-dynamic-range images[C]// ACM Transactions on Graphics (TOG). ACM, 2002, 21(3): 257-266

[3] Porikli F. Constant time O(1) bilateral filtering [C]// IEEE Conference on Computer Vision and Pattern Recognition, 2008. IEEE, 2008: 1-8

[4] Weiss B. Fast median and bilateral filtering[J]. ACM Transactions on Graphics (TOG). ACM, 2006, 25(3): 519-526

[5] Pham T Q, Van Vliet L J. Separable bilateral filtering for fast video preprocessing [C] // IEEE International Conference on Multimedia and Expo, 2005 (ICME 2005). IEEE, 2005: 4

[6] Paris S, Durand F. A fast approximation of the bilateral filter using a signal processing approach[M]// Computer Vision-ECCV 2006. Springer Berlin Heidelberg, 2006: 568-580

[7] Yoshizawa S, Belyaev A, Yokota H. Fast gauss bilateral filtering [J]. Computer Graphics Forum. Blackwell Publishing Ltd, 2010, 29(1): 60-74

[8] Chaudhury K N, Sage D, Unser M. Fast O(1) bilateral filtering using trigonometric range kernels[J]. IEEE Transactions on Image Processing, 2011, 20(12): 3376-3382

[9] 张志强，王万玉. 一种改进的双边滤波算法[J]. 中国图象图形学报, 2009, 14(3): 443-447

[10] 李俊峰，杨丰，黄靖. 一种改进的增维型双边滤波的快速算法[J]. 电路与系统学报, 2013, 1(18): 137-143

[11] Eklund A, Dufort P, Forsberg D, et al. Medical image processing on the GPU-Past, present and future[J]. Medical image analysis, 2013, 17(8): 1073-1094

[12] Pai Y S, Shen Y C, Wu J L. High efficient distributed video coding with parallelized design for LDPCA decoding on CUDA based GPGPU[J]. Journal of Visual Communication and Image Representation, 2012, 23(1): 63-74

[13] Sanders J, Kandrot E.: CUDA By Example an Introduction to General-Purpose GPU Programming[M]. 聂学军，译. 北京：机械工业出版社, 2011: 135

(上接第 162 页)

3)提出一种基于采样矩阵优化的 ND-GSM 模型的相干斑噪声抑制算法。对真实 SAR 图像的仿真实验结果验证了本文方法是有效的，去噪性能得到明显提高，且明显优于小波等方法去噪的效果，边缘和纹理等细节信息保持得更好。

参 考 文 献

[1] Portilla J, Strela V, Wainwright M J, et al. Imagedenoising using scale mixtures of Gaussians in the waveletdomain [J]. IEEE Transactions on Image Processing, 2003, 12(11): 1338-1351

[2] Zhang Lei, Shi Han-qing, Du Hua-dong, et al. Estimation of seasur facewind direction using Space borne SAR images and wavele tanalysis[J]. Journal of remote Sensing, 2014, 1: 215-216

[3] Velisavljevic V, Beferull-Lozano B, Vetterli M, et al. Directionlets: anisotropic multi-directional representation with separable filtering[J]. IEEE Trans. on Image Proc., 2006, 15(7): 1916-1933

[4] 胡贺军，高清维，卢一相，等. 基于方向波域混合高斯模型的 SAR 图像去噪[J]. 计算机应用与软件, 2013, 30(7): 283-286

[5] Do M N, Vetterli M. The Contourlet Transform: An Efficient Directional Multiresolution Image Representation [J]. IEEE proc., 2005, 14(12): 2091-2106

[6] 刘帅奇，胡绍海，肖扬. 基于小波 Contourlet 变换与 Cycle Spinning 相结合的 SAR 图像去噪[J]. 信号处理, 2011, 27(6): 837-842

[7] 张弛. Directionlet 变换在图像去噪中的应用研究[D]. 合肥：安徽大学, 2010

[8] 关辉. 基于 Directionlet 变换的 SAR 图像噪声抑制及边缘检测[D]. 西安：西安电子科技大学, 2010

[9] 岳春宇，江万寿. 基于最大后验和非局域约束的非下采样轮廓波变换域 SAR 图像去噪方法[J]. 测绘学报, 2012, 42(1): 59-64

[10] 张弛. 基于平稳小波域 GSM 模型的 SAR 图像去噪[J]. 黑龙江科技学院学报, 2010, 20(3): 222-226

[11] 侯青松，郭英，王布宏. 基于格理论的非均匀稀疏线阵旁瓣结构的分析方法[J]. 电子学报, 2010, 38(6): 1459-1463

[12] 刘杰，朱启兵，李允公. 基于新阈值函数的二进小波变换信号去噪研究[J]. 东北大学学报, 2006, 27(5): 536-539

[13] 张冬翠. 基于 Directionlet 变换的图像去噪和融合[D]. 西安：西安电子科技大学, 2010

[14] 武晓弱，郭宝龙，李雷达. 一种新的结合非下采样 Contourlet 与自适应全变差的图像去噪方法[J]. 电子与信息学报, 2010, 32(2): 360-365

[15] 侯建华，田金文，柳健. 小波域局部维纳滤波器估计误差分析及图像去噪[J]. 光子学报, 2007, 36(1): 189-191

[16] 张明，李开成，胡益胜. 基于 Bayes 估计的双小波维纳滤波电能质量信号去噪算法[J]. 电力系统保护与制, 2011, 39(4): 52-57