快速鲁棒特征算法的CUDA加速优化

刘金硕[1] 曾秋梅[1] 邹 斌[1] 江庄毅[1] 邓 娟[2]
(武汉大学计算机学院 武汉 430072)[1] (武汉大学国际软件学院 武汉 430072)[2]

摘 要 提出一种基于统一计算设备架构(Compute Unified Device Architecture,CUDA)的快速鲁棒特征(Speed-up Robust Feature,SURF)图像匹配算法。分析了SURF算法的并行性,在图像处理单元(Graphics Processing Unit,GPU)的线程映射和内存模型方面对算法的构建尺度空间、特征点提取、特征点主方向的确定、特征描述子的生成及特征匹配5个步骤进行CUDA加速优化。实验表明,相比适用于CPU的SURF算法,文中提出的适用于GPU的SURF算法在处理30MB的图片时性能提高了33倍。适用于GPU的SURF算法拓展了SURF算法在遥感等领域的快速应用,尤其是大影像的快速配准。

关键词 快速鲁棒特征,CUDA,特征提取,影像匹配
中图法分类号 TP391 **文献标识码** A

Speed-up Robust Feature Image Registration Algorithm Based on CUDA

LIU Jin-shuo[1] ZENG Qiu-mei[1] ZOU Bin[1] JIANG Zhuang-yi[1] DENG Juan[2]
(School of Computer,Wuhan University,Wuhan 430072,China)[1]
(College of International Software,Wuhan University,Wuhan 430072,China)[2]

Abstract This paper proposed a speed-up robust feature image registration algorithm based on Compute Unified Device Architecture (CUDA). We analyzed the parallelism of SURF algorithm, and optimized it by CUDA from thread mapping and memory model of the Graphics Processing Unit (GPU), aiming at the five steps of SURF algorithm including building scale space, extracting feature points, determining key direction of feature points, generating vector descriptor and feature matching. The experimental results show that the GPU implementation achieves 33 times faster than the CPU implementation while processing an image of 30MB. The GPU implementation extends the application of SURF algorithm in fast processing of remote sensing image, especially in the quick image registration.

Keywords Speed-up robust feature, CUDA, Feature extraction, Image matching

1 引言

2006年,Bay提出了基于加速鲁棒特征的SURF算法[1,2],这是局部特征描述器的一种方法。SURF特征提取方法通过引入积分图像和模板近似进行快速计算,并通过快速Hessian来检测特征点。在特征描述阶段,通过计算哈尔小波变换确定特征点的主方向并生成描述子。SURF算法所提取的特征点对光照变化、视角变化、旋转变换、图像模糊具有不变性和鲁棒性,被广泛应用到图像配准领域。

图像配准(Image Registration)[3,4]是将两幅或者多幅包含同一场景或者事物等内容的图片进行校准、匹配、重叠,使之能够将包含相同信息的部分尽量匹配,以便提供更多的视觉信息的过程。图像匹配是遥感、医学、计算机视觉、模式识别等很多领域中的一个基本问题[5]。提取图像特征是实现图像配准的一个重要环节。

基于特征的图像配准方法在进行图像配准时能取得较好的效果。文献[6-8]通过大量实验对几种具有代表性的局部特征算法进行了性能评估,结论表明:SURF算法是鲁棒性最好的局部特征算法。SURF特征提取和匹配属于计算密集型问题,计算量大,处理时间长,对其进行快速处理是很多领域的需要。针对SURF算子特征匹配搜索时间长的问题,李慧[9]等提出了基于加速分割检测特征(feature from accelerated segment test,FAST)和SURF的遥感影像自动配准方法,该方法能获得更多的匹配点对,也具有更高的几何匹配精度,但在尺度性方面略逊色于SURF算法。杜振鹏[10]等将基于KD-Tree搜索的SURF特征图像匹配算法与基于全局最近邻搜索的SURF算法进行实验比较,认为基于KD-Tree索引的SURF算法大大减少了计算量,提高了SURF算法的匹配速度。然而与特征匹配时间相比,特征检测时间所占的比重更大。因此对算法的特征检测过程进行加速,减少整个匹配过程的时间,将使算法更具有实时性。Nico Cornelis[11]等通过采用texture mipmap的数据存储结构,在图像级和尺度级对

到稿日期:2013-06-27 返修日期:2013-10-16 本文受国家自然科学基金(61303214),地震行业重大专项(201008007)资助。
刘金硕(1974—),女,博士,副教授,主要研究方向为图像处理、高性能计算;**曾秋梅**(1990—),女,硕士生,主要研究方向为图像处理、高性能计算;**邹 斌**(1989—),男,硕士生,主要研究方向为图像处理;**江庄毅**(1991—),男,硕士生,主要研究方向为高性能计算;**邓 娟**(1976—),女,博士,副教授,主要研究方向为图像处理、高性能计算。

SURF 算法的多个步骤进行 CUDA 并行处理，加速了 SURF 特征提取和匹配的效率。

本文对 SURF 算法实现了 CUDA 并行加速，计算设备选择 GPU 是因为 SURF 算法中的计算属于图像处理部分，尤其是卷积滤波这种单指令多数据是 GPU 并行加速的优势所在。而且在特征匹配过程中利用 CUDA 提供的基础数值线性代数运算库 BLAS 的 GPU 实现 CUBLAS 可以提高匹配效率。实验证明，本文提出的算法能够对大图像实现快速的特征提取和匹配，而且算法的适用性更广。

本文第 1 节主要介绍 SURF 算法及其优化工作的研究现状；第 2 节简要介绍 SURF 算法的流程和 CUDA 架构；第 3 节详述对 SURF 算法每个步骤实现 CUDA 并行加速的优化策略；第 4 节通过实验对 CPU 的 SURF 算法和使用 CUDA 优化后的算法的性能进行分析比较；最后是对本文工作的总结及未来展望。

2 研究背景

2.1 SURF 算法流程

SURF 算法的基本流程如图 1 所示。算法首先计算输入图像的积分图像，再采用盒子型滤波器(box filter)近似高斯滤波的方法，利用积分图像快速完成图像卷积以构建尺度空间，并在尺度空间内计算每个像素点近似的 Hessian 矩阵行列式值，然后采取非极大值抑制(Non-Maxima Suppression，NMS)方法提取特征点，再根据特征点邻域的小波响应确定特征点的主方向并生成特征描述向量，最后进行特征匹配。

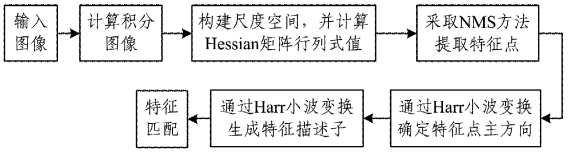

图 1 SURF 算法流程

SURF 算法采用的关键技术有：

(1)使用积分图像完成图像的卷积运算。

(2)利用 Hessian 矩阵快速检测特征点。

(3)基于分布的描述符信息。

2.2 CUDA 编程

CUDA[12]是由 Nvidia 公司推出的专门用于 GPU 通用计算的平台，它是一个完整的 GPGPU 解决方案，提供了硬件的直接访问接口，而不必像传统方式一样必须依赖图形 API 接口来实现 GPU 的访问。GPU 通用计算通常采用 CPU＋GPU 异构协同工作模式，CPU 作为主机，GPU 作为高性能计算设备。CPU 负责执行复杂逻辑处理和事务处理等不适合数据并行的计算，其中工作包括：①启动运行在 GPU 上的 CUDA 并行计算函数之前所需的数据准备，②设备初始化工作，③在 CUDA 并行函数之间进行一些串行计算。而 GPU 负责计算密集型的大规模数据并行计算[13]。在 GPU 中，整个可伸缩流处理器阵列可以被看成是由多个流多处理器组成的多单指令流多线程(Multiple SIMT，MSIMT)系统[14]。

一个完整的 CUDA 程序由一系列运行在 CPU 端的串行代码和分配给 GPU 端的 CUDA 并行计算函数共同组成(见图 2)。一个 CUDA 并行计算函数存在两个层次的并行，即线程网格(Grid)中的线程块(Block)间并行和线程块中的线程(Thread)间并行。两层并行模型是 CUDA 的关键特性和创新。CUDA 采用 CUDA C 语言作为编程语言，其简单的编程风格、高效的多线程并行处理模式使得人们在面对计算密集型任务时能够更好地利用 GPU 庞大的并行计算资源。

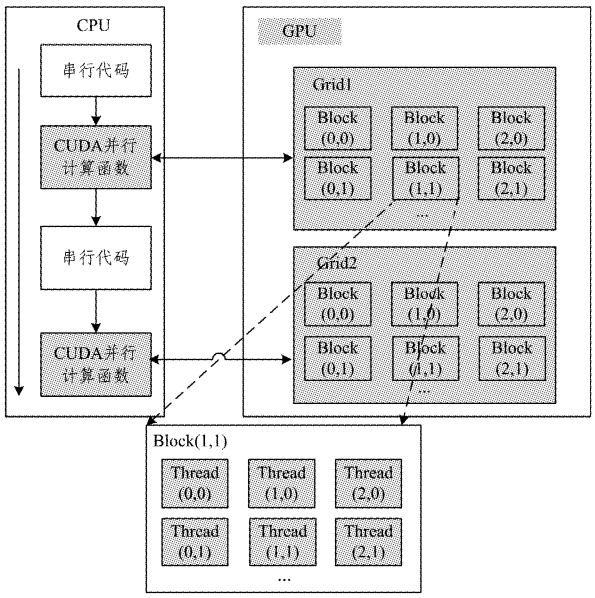

图 2 CUDA 编程模型

3 SURF 匹配算法的 CUDA 并行化

利用 CUDA 技术实现 SURF 算法，算法中有 5 个步骤是由 GPU 以并行计算的方式进行，如下：

(1)创建图像金字塔；

(2)关键点检测；

(3)计算特征点方向；

(4)计算特征描述向量；

(5)特征匹配。

整个实现过程是由 CPU 和 GPU 协同处理的，具体处理流程如图 3 所示。

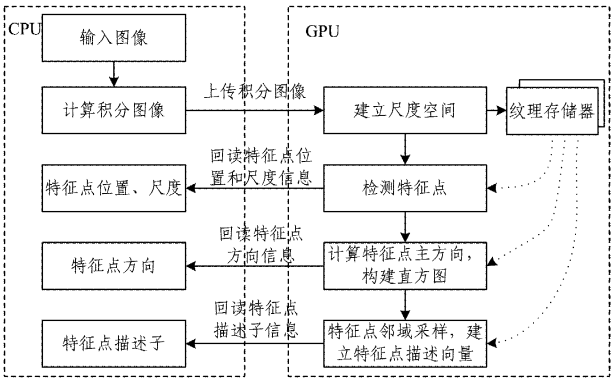

图 3 SURF 算法 CUDA 并行化实现流程图

图像输入后，首先在 CPU 端计算输入图像的积分图像，然后将其上传到 GPU 的全局内存中，随后绑定为纹理内存。在 GPU 上首先构建图像金字塔，并将图像金字塔即尺度空间数据存储在纹理存储器内，以便于后续检测特征点及生成特征描述子等操作时对图像金字塔进行随机访问。然后对尺度空间内所有像素进行并行计算，检测特征点，构建方向直方

图确定特征点的主方向，在特征点邻域内采样，建立特征点描述向量。GPU 并行处理的每一步骤生成的中间结果如特征点位置、所在尺度、方向及特征点描述子等信息都需要回读到 CPU，以便在 CPU 端输出或显示。

3.1 构建尺度空间优化

构建尺度空间时，先将在 CPU 端计算好的对应于原图像的积分图像上传到 GPU 端的全局内存，随后将其绑定为纹理内存，如图 4 所示。与全局内存不同，纹理内存没有严格的合并访问限制，在随机存取的环境下能够保持较高的性能。再将用于近似 x 方向、y 方向和 xy 方向高斯滤波的盒子型滤波器的模板存储在常量内存。对图像进行滤波卷积时，每个像素点可以分派给一个内核线程处理，每个线程可以并行计算在某一尺度的某一像素点 x 的 x 方向、y 方向和 xy 方向的卷积值，结果记为：$D_{xx}(x,\sigma)$，$D_{yy}(y,\sigma)$，$D_{xy}(x,\sigma)$，再根据式(1)计算每个像素点的 Hessian 矩阵行列式值。

$$\det(H_{approx})=D_{xx}(x,\sigma)D_{yy}(x,\sigma)-(0.9D_{xy}(x,\sigma))^2 \quad (1)$$

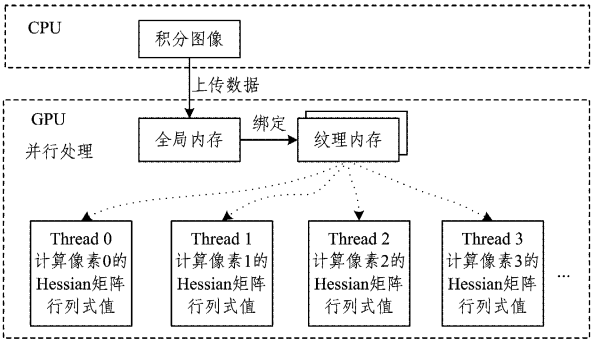

图 4 构建尺度空间 CUDA 并行化

3.2 特征点提取优化

采用 NMS 方法检测特征点时，每个线程负责一个像素点是否为特征点的判断，可在 GPU 内对尺度空间所有像素进行并行处理，如图 5 所示。每个线程先将该像素点与所在尺度空间的 8 个邻近像素进行 Hessian 矩阵行列式值比较，若它比这 8 个像素都大或都小，则再将该像素点与相邻尺度空间的 $9*2$ 个像素点进行比较；否则直接退出线程，无需进行后续的比较。为了得到更为精确和稳定的特征点，需要在尺度空间和图像空间对特征点进行插值运算。假定一个尺度空间有 100 个像素点，则硬件条件允许的情况下最多可以分派 100 个并行线程，从而充分利用现有的 GPU 的多核计算能力。

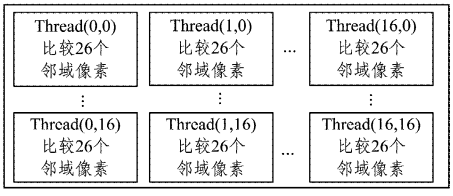

图 5 GPU 并行处理检测特征点

3.3 主方向的确定优化

在 GPU 上计算特征点的主方向时，将特征点列表上的每个特征点映射为一个线程块，该线程块中每个线程对应该特征点的一个邻域像素点，利用 CUDA 技术加速优化流程如图 6 所示。首先计算其 x 方向和 y 方向的小波响应值，记为 $harrX$ 和 $harrY$，通过式(2)求得向量的方向 $angle$，每个线程计算的 $harrX$、$harrY$ 和 $angle$ 值保存在共享存储器中，以提高后续线程计算时的访存效率。再根据 $angle$ 将该向量划分到 42 个 60°的滑动窗口范围内，每个线程并行处理一个滑动窗口内采样点的统计，将属于该滑动窗口内的采样点的 $harrX$ 值和 $harrY$ 值分别累加得到 $sumX$ 和 $sumY$，通过式(3)和式(4)计算累加向量的长度 $metric$ 及方向 $orientation$，并保存到 2 个分组为 42 的直方图 $metric$ 和 $orientation$ 上，完成直方图统计。直方图 $metric$ 和 $orientation$ 放在线程块的共享内存中，共享内存是 GPU 芯片内部的高速缓存，带宽及访问延迟都大幅优于全局内存。最后由一个线程遍历存储在共享内存区的直方图，求得统计值最大的方向输出。

$$angle=\arctan\frac{harrX}{harrY} \quad (2)$$

$$metric=sumX*sumX+sumY*sumY \quad (3)$$

$$orientaion=\arctan\frac{sumX}{sumY} \quad (4)$$

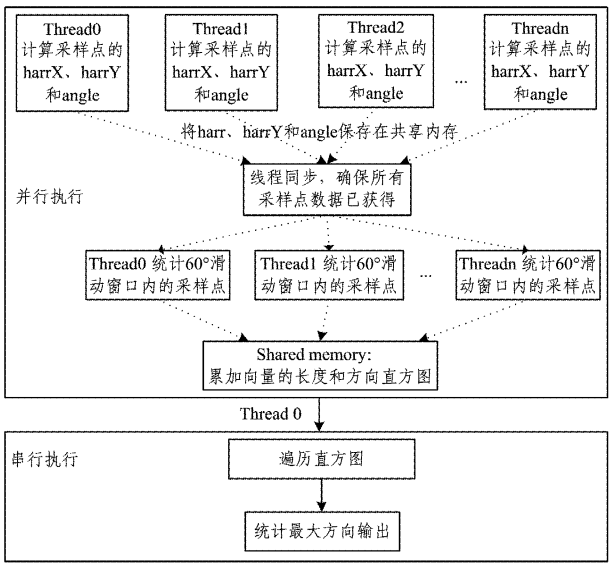

图 6 计算特征点的主方向 CUDA 加速优化流程图

3.4 描述子生成优化

由于每个特征点所在尺度不一，计算关键点描述子时其采样的邻域窗口也不等大，而 CUDA 要求每个线程块的大小一致且有大小限制。以尺度 $s=1.6$ 为例，统计窗口的大小为 $(20s)^2=1024$；但 CUDA 每个线程块目前支持的最大线程数目为 512，无法实现像素与线程的一一对应。本文采取的方法是固定线程块的大小为 $20*20$，在每个线程内计算采样坐标，实现在邻域窗口内的均匀采样。如图 7 所示，X 代表特征点，每个特征点对应一个线程块，在特征点对应尺度的图像上 $20s*20s$ 的邻域，每 $5s*5s$ 子域内分派 $5*5$ 个线程对其均匀采样，线程块内的每个线程对应一个采样点，每个线程并行计算采样点的 x 方向和 y 方向的小波响应及响应的绝对值，记为 d_x，$|d_x|$，d_y，$|d_y|$，累加到存放在共享内存的描述子上。这样，每个子域形成一个 4 维向量 $(\sum d_x,\sum|d_x|,\sum d_y,\sum|d_y|)$，从而在 $20s*20s$ 邻域中获得长为 $4*(4*4)=64$ 的特征描述向量，最后将特征描述向量从共享内存拷贝到全局内存输出。

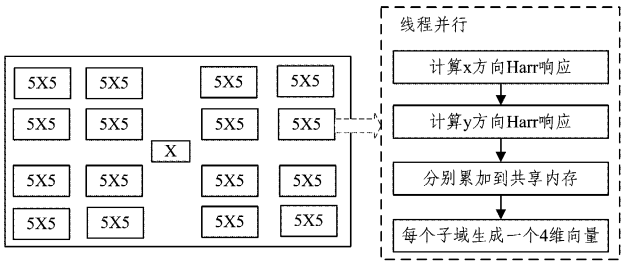

图 7　特征点描述子生成的 CUDA 并行化

3.5　特征匹配优化

将影像 I 的 m 个特征点的特征描述向量组合为 $m*64$ 的矩阵 A，影像 J 的 n 个特征点的特征描述向量组合为 $64*n$ 的矩阵 B，计算 AB^{T} 即得两幅影像间特征的相关系数矩阵，如图 8 所示。CUDA 提供了基础数值线性代数运算库 BLAS 的 GPU 实现 CUBLAS。首先使用 cublasSgemm 函数完成矩阵乘法，再开启 m 个线程为第一幅影像的特征点搜索最佳匹配点。

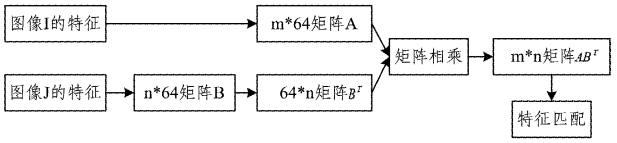

图 8　图像 I 与图像 J 的特征匹配

4　实验结果

本文的实验平台采用 Window 7 操作系统，CPU 为 Intel Core i3-530，主频为 2.93GHz，系统内存为 4GB；GPU 为 Nvidia Geforce GTS450，GPU 的显存为 1024M；CUDA 版本为 4.2。实验对两组不同分辨率的影像分别使用 CPU 的 SURF 算法与 GPU 的 SURF 算法进行特征提取和特征匹配，同时对两种方法的实验数据进行整理，表 1 列出了两种方法实现 SURF 算法各个流程时的耗时及特征点匹配数目的对比数据，表中符号“+”表示处理一组中两幅影像时的不同耗时。例如：由表 1 第 5 行可以看出计算构建尺度空间和提取特征点的耗时。第一组影像大小为 3240 * 2230 和 5120 * 3840 像素，仅使用 CPU 串行算法的耗时分别为 24.014s 和 66.080s；采用了 GPU 并行算法以后的耗时分别为 1.101s 和 2.854s。第二组影像大小为 4000 * 2551 和 4000 * 2551 像素，仅使用 CPU 串行算法的耗时分别为 33.344s 和 32.727s；采用了 GPU 并行算法以后的耗时分别为 1.565s 和 1.358s。由于 SURF 方法计算积分图时主要采取的是串行计算的方式，因此串并行计算时间几乎相等，其余步骤的加速比均超过了 20。实验采用的第一组图像的左图是右图的子影像，第二组图像的右图是左图经过旋转及仿射变换之后的图像。图 9、图 10 是两组影像的匹配结果示意图。

实验结果表明：得益于 GPU 并行处理的强大运算功能，适用于 GPU 的 SURF 算法在处理 30MB 的图像时，算法的整体处理效率提高了 33 倍。在处理大影像数据时，GPU 的 SURF 方法的计算速度远远超过了 CPU 的 SURF 方法；并且当图像越大，计算量越大时，适用于 GPU 的 SURF 算法在速度提升方面的效果越显著，说明其能够满足实时应用的需要。

表 1　SURF 算法在 CPU 和 GPU 上运行时间的比较

	第一组影像		第二组影像	
图像大小(像素)	3240 * 2230 和 5120 * 3840		4000 * 2551 和 4000 * 2551	
串行算法与并行算法对比	串行算法	并行算法	串行算法	并行算法
计算积分图(s)	0.117+0.303	0.117+0.303	0.147+0.147	0.147+0.147
构建尺度空间和提取特征点(s)	24.014+66.080	1.101+2.854	33.344+32.727	1.565+1.358
确定主方向和生成描述子(s)	32.767+57.958	0.968+1.876	71.416+64.717	2.986+2.747
特征匹配(s)	104.515	2.463	259.300	4.973
特征点匹配数目(个)	77	58	748	584
总时间(s)	285.754	9.682	461.798	13.923
加速比	29.514		33.168	

图 9　第一组图像特征匹配结果示意图

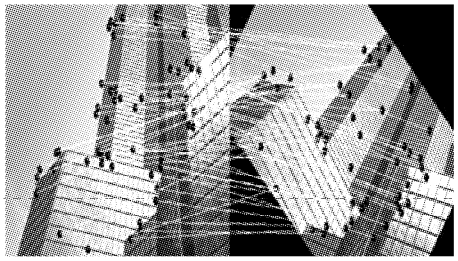

图 10　第二组图像特征匹配结果示意图

结束语　本文实现了对 SURF 算法的 CUDA 并行加速，该算法充分利用了 GPU 的并行计算能力及 CUDA 内存管理机制。实验结果表明，相比适用于 CPU 的 SURF 算法，本文提出的适用于 GPU 的 SURF 算法在处理 30MB 的图片时性能上提高了 33 倍，能够对大影像实现快速的特征提取和匹配。在下一步研究中需要改进的重点是考虑采用 KD-Tree 的索引方法加速特征点的匹配过程，进一步提高配准效率。

参 考 文 献

[1] Bay H, Tuytelaars T, Gool V L. SURF: Speeded Up Robust Features[C]//Proc of European Conference on Computer Vision 2006. Austria: Graz, 2006: 404-417

[2] Bay H, Ess A, Gool V L. Speeded Up Robust Features (SURF)[J]. Computer Vision and Image Understanding, 2008, 110(3): 346-359

[3] Sinha N S, Frahm J, Pollefeys M, et al. GPU-based Video Feature Tracking and Matching[R]. EDGE Workshop on Edge Computing Using New Commodity Architectures, 2006

[4] Sinha N S, Frahm J, Pollefeys M, et al. Feature tracking and matching in video using programmable graphics hardware[J]. Machine Vision and Applications, 2011, 22(1): 207-217

[5] 林晓帆，林立文，邓涛，基于 SURF 描述子的遥感影像配准[J]. 计算机工程，2010，36(12)：216-218

（下转第 43 页）

据符合前两种情况，相比于其他几种算法，C-BFD 算法较大程度地减少了时隙碎片的产生。

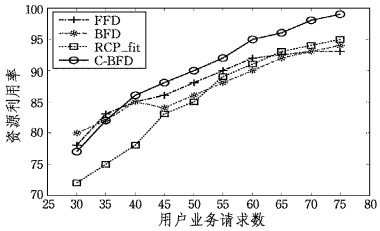

图 4　系统资源利用率和业务请求数的关系

(2)申请阻塞率

申请阻塞率可以表示为 N_{rej}/N_{req}，N_{rej} 为系统拒绝掉的业务申请数，N_{req} 为总共收到的业务申请数。如图 5 所示，随着用户业务申请到达数的不断增加，申请阻塞率都呈现增长的趋势。由于 C-BFD 算法采取对业务进行组合并优先分配，最大限度地减少时隙碎片的产生，从而降低了因时隙碎片而导致的业务请求阻塞，故 C-BFD 算法要明显优于其他算法。

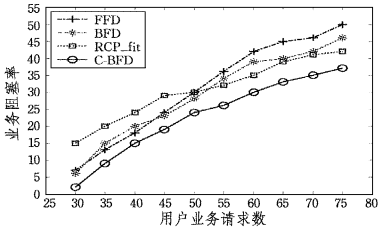

图 5　业务阻塞率和业务请求数的关系

(3)空闲信道数

如图 6 所示，随着业务申请数量的增加，空闲信道数也会减少，但是不同的分配算法在时隙碎片上面的性能不一样，明显可以看出 C-BFD 算法在减少时隙碎片方面有更好的表现。

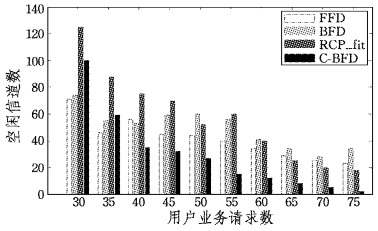

图 6　信道碎片和业务请求数的关系

结束语　根据 GEO 卫星通信系统中 MF-TDMA 信道结构以及分配约束，针对现有信道分配算法在时隙碎片方面的不足，在一维装箱算法基础上提出了基于优先组合的 C-BFD 方法，并给出了理论分析。仿真结果显示，相比于传统算法，该算法提高了系统资源利用率，降低了业务阻塞率以及信道空闲数。

参 考 文 献

[1] 叶晓国，肖甫，孙力娟，等. 卫星移动通信网络切换算法的研究[J]. 计算机科学，2009，36(9)：79-82

[2] 郭庆，王振永，顾学迈. 卫星通信系统[M]. 北京：电子工业出版社，2010：15-16

[3] European Telecommunications Standards Institute. GMPRS-101. 201v2. 2. 1General Packet Radio Service: Introduction to the GMR-1 Family [EB/OL]. http://pda. etsi. org/exchange-folder/ts_1013760102v020201p. pdf，2005-01-02

[4] European Telecommunications Standards Institute. GMPRS-101. 201v2. 2. 1General Packet Radio Service: Radio interface physical layer specifications [EB/OL]. http://pda. etsi. org/exchangefolder/ts_101376-55v02 0201p. pdf，2005-03

[5] 许楠，郝学坤，许众. MF-TDMA 卫星通信系统信道分配时间优化方法[J]. 无线电通信技术，2012，38(2)

[6] Park J-M，Savagaonkar U R，Chong E，et al. Allocation of QoS connections in MF-TDMA satellite systems: a two-phase approach[J]. IEEE Transactions On Vehicular Technology，2005，54(1)：177-190

[7] 董启甲，张军，张涛，等. 高效 MF-TDMA 系统时隙分配策略[J]. 航空学报，2009，30(9)：1718-1726

[8] 余国松. 与装箱相关的几类问题[D]. 杭州：浙江大学，2009

[9] Park J-M，Savagaonkar U R，Chong E，et al. Allocation of QoS connections in MF-TDMA satellite systems: a two-phase approach[J]. IEEE Transactions on Vehicular Technology，2005，54(1)：177-190

[10] Scalable Network Technologies，Inc. QualNet 4. 0 Programmercs Guide[M]. Los Angeles，USA，2007：1-5

(上接第 27 页)

[6] Murillo A C，Guerrero J J，Sagues C. Surf features for efficient robot localization with omnidirectional images [C]//IEEE International Conference on Roboties and Automation. Roma，Italy，2007：3901-3907

[7] Arturo G，Oscar M，Monica B，et al. A comparative evaluation of interest point detectors and local descriptors for visual SLAM [J]. Machine Vision and Application，2010，21(6)：905-920

[8] Valgren C，Lilienthal A. SIFT SURF and Seasons: Longterm outdoor Localization Using Local Features[C]//Proc of the 3rd European Conf on Mobile Robots. Germany，2007：253-260

[9] 李慧，蔺启忠，刘庆杰. 基于 FAST 和 SURF 的遥感影像自动配准方法[J]. 国土资源遥感，2012，2(6)：28-34

[10] 杜振鹏，李德华. 基于 KD-Tree 搜索和 SURF 特征的图像匹配算法研究[J]. 计算机与数字工程，2012，40(2)：96-98

[11] Cornelis N，Gool V L. Fast Scale Invarint Feature Detection and Matching on Programmable Graphics Hardware[C]//Computer Vision and Pattern Recognition Workshops 2008. AK，2008：1-8

[12] Randima F. GPU Gems: programming techniques，tips and tricks for real-time graphics[M]. Addison-Wesley Professional，2006

[13] GPGPU. General-purpose computation on GPUs[OL]. http://www. gpgpu. org，2010-07-04

[14] 钱悦. 图形处理器 CUDA 编程模型的应用研究[J]. 计算机与数字工程，2008，36(12)：177-180