

基于 GPU 的变型 SIFT 算子实时图像配准

袁修国 彭国华 王 琳

(西北工业大学理学院应用数学系 西安 710129)

摘 要 针对 SIFT 变型算法描述向量维数过高、实时性差的问题,分别在建立高斯尺度金字塔、关键点的亚像素定位等方面进行改进与并行化。利用 CUDA 设备构架在 GPU 硬件上实现多线程,一方面避免了 PCA 方法造成的关键点信息流失,另一方面使得配准速度达到了工程中的实时性要求。在 VS2005 平台上通过 C 语言实现混合 CUDA 编程,结果表明该方法使得配准速度和点对匹配正确率都有较大提升。

关键词 旋转不变特征变换(SIFT),梯度定位与方向直方图(GLOH),图形处理器(GPU),计算统一设备构架
中图分类号 TP391 **文献标识码** A

GPU-based Real Time Image Registration with Variant SIFT

YUAN Xiu-guo PENG Guo-hua WANG Lin

(Department of Applied Mathematics, College of Science, Northwestern Polytechnical University, Xi'an 710129, China)

Abstract Improvements and parallelization in the aspects of creating differences of gaussian and locating sub-pixel key-points were made against the problem that the dimension of descriptors from variant SIFT-GLOH (Gradient Location Orientation Histogram) is too high to meet the need of real time. The algorithm was implemented in multi-threads on the GPU hardware with CUDA (Compute Unified Device Architecture). On one hand, the information loss of keypoints caused by PCA was avoided. On the other hand, the speed of registration sacrificed the need of real time in engineering. The programs were compiled by C language & CUDA on VS2005 platform. The result shows that the ratio of correct matching point pairs and registration speed are both promoted greatly.

Keywords SIFT, GLOH, GPU, CUDA

1 引言

基于特征的图像配准算法是图像配准技术的重要组成部分,它包括基于点特征、边缘特征、线段、轮廓、闭合区域、矩统计量等。普遍应用的有点特征和边缘特征,但是,边缘特征的图像配准技术忽略了图像的绝大部分信息,当图像出现几何扭曲时,其描述能力大大下降。点特征算法与边缘特征算法相比虽然更多地忽略了图像原本的信息,但是,近年来许多基于关键点的特征描述符算法广为应用,它不仅考虑了某关键点的灰度信息,还全面考虑了关键点领域范围内的图像信息,其中最具代表性的是 SIFT (Scale Invariant Features Transform) 算法,该算法经实践证明具有很高的匹配精度与鲁棒性^[1],但无论是 SIFT 算法,还是对 SIFT 算子所做改进的众多算法,特征点描述符向量都是上百维的,其时间复杂度都很难达到实时要求。本文将 GPU (graphics processing unit) 应用于 SIFT 变型算法 GLOH (Gradient Location and Orientation Histogram),使得计算时间大大缩减,且有更高的精度。

2 SIFT 算法及其发展

2.1 基于点特征算法的历史

2.1.1 点提取

基于点描述符的图像配准算法最早要追溯到 1988 年的

Harris 角点提取算法^[2],该算法建立在二元函数的二阶导数之上,在图像未发生尺度变换时具有很强的鲁棒性,但是当图像出现尺度变换时,提取到的特征点在数目和位置上都有很大差距。Mikolajczyk 和 Schmid 在 Harris 算法和尺度算法的基础上建立了鲁棒的尺度不变特征算子 Harris-Laplac 和 Hessian-Laplace^[3]。2004 年 David Lowe 在前人基于图像尺度的点提取算法基础上总结出了一种尺度不变特征变换 (SIFT) 算子,利用 DoG (Differences of Gaussian) 尺度空间作点提取。Herbert Bay 等人根据 Mikolajczyk 与 Schmid 的点描述算法提出了一种新的基于 Hessian-Laplace 算子的加速鲁棒特征 (Speeded Up Robust Features, SURF) 描述符^[4]。

2.1.2 点描述符

当特征点被提取之后,点描述符特征向量的确定十分关键,它关系到算法的复杂度和匹配精度。Yan Ke 和 Rahul Sukthankar 在文献^[5]中提出了一种 Principal Components Analysis (PCA)-SIFT 算法,用于描述符向量的降维,实验证明该算法具有更高的实时性和更强的鲁棒性。Mikolajczyk 在研究比较了许多改进算法之后,提出了一种变型的 SIFT 算法 GLOH。该算法可以得到更高的匹配精度,但是在时间复杂度上须付出更大代价^[6]。本文将在该算法基础上利用 GPU 硬件将其并行化,使其在高匹配精度的基础上获得更好的实时性。

到稿日期:2010-04-22 返修日期:2010-08-19 本文受国家自然科学基金(60672135)资助。

袁修国(1986—),男,硕士生,主要研究方向为图像配准、机器视觉与人工智能,E-mail:zvgtjfs0519@163.com;彭国华(1962—),男,博士,教授,博士生导师,主要研究方向为图形图像处理、机器视觉、CAGD&CG。

2.2 SIFT 算法步骤描述及其改进

由于拥有许多优秀特点,SIFT 算法被提出后迅速在各个领域得到应用,但是在一些工程中仍然难以达到高精度、高实时的要求,以下将从算法精度方面叙述对其所做的改进。

2.2.1 构建尺度空间

Koendetink 与 Lindeberg 等人证明了高斯核是唯一的线性核。一幅二维图像的尺度定义为高斯函数对图像做卷积:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

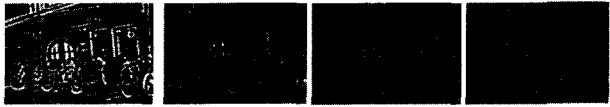
$$\text{式中, } G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

尺度空间金字塔的构建过程如下:先采用不同尺度因子(本处选取 5 个等比尺度)对初始图像进行卷积,得到一系列不同尺度图像,作为金字塔的第一层。再将上一层中尺度为该层初始图像尺度二倍的图像进行下采样,将其作为本层的初始图像;重复上述步骤,可以得到一个金字塔组(此处作三层金字塔构建),各层不同尺度图像间按照如下公式进行差分可以得到 DoG 图像层,图 1 中的(a)、(b)分别显示了金字塔二三层的差分图像。

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2 \nabla^2 G$$



(a) 尺度金字塔第二层



(b) 尺度金字塔第三层

图 1

2.2.2 关键点的提取、剔除及亚像素定位

依照金字塔层数与尺度关系,遍历所有像素点将其同 3 维邻域的 26 个像素进行比较,可以到 6 个尺度的关键点。将尺度空间函数 $D(x, y, \sigma)$ 利用泰勒展开式在像素级关键点 (x_0, y_0, σ_0) 做二次展开:

$$D(x, y, \sigma) = D(x_0, y_0, \sigma_0) + \frac{\partial D}{\partial d} d + \frac{1}{2} d^T \frac{\partial^2 D}{\partial d^2} d$$

式中, $d = (x - x_0, y - y_0, \sigma - \sigma_0)$ 为真实点向量相对于像素级关键点 (x_0, y_0, σ_0) 的偏移量,求得 $D(d)$ 的极值响应偏移量为:

$$d_0 = -\frac{\partial D}{\partial d} \left(\frac{\partial^2 D}{\partial d^2} \right)^{-1}$$

将 d_0 带入展开式,与阈值比较判断是否剔除该关键点。

也可通过计算关键点处的 Hessian 矩阵:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

来判断关键点是否可取,可知:

$$\text{Tr}(H)^2 = (\lambda_1 + \lambda_2)^2$$

$$\text{Det}(H) = \lambda_1 * \lambda_2$$

将各关键点的 $\frac{\text{Tr}(H)^2}{\text{Det}(H)}$ 值同计算好的阈值做比较可剔除边缘响应点。文献[7]认为,将尺度空间函数 $D(x, y, \sigma)$ 在关键点邻域用泰勒展式逼近,首先略去了二次项,而后在计算离

差值时利用差分表示导数和二阶导数,这样并不能得到足够精确的离差值,且没有充分利用该关键点的邻域信息。因此该文利用尺度空间的 27 个采样点作三维插值函数近似,其拟合函数如下:

$$D'(x, y, \sigma) = a_{11}x^2 + a_{12}y^2 + a_{13}\sigma^2 + a_{21}xy + a_{22}x\sigma + a_{23}y\sigma + a_{31}x + a_{32}y + a_{33}\sigma + a_0$$

然后将 27 个采样点坐标带入上式求解关于 a 的线性方程组 $Aa = D$,得到方程 $D'(x, y, \sigma)$ 。并由系数向量 a 得到该点的 Hessian 矩阵和梯度信息:

$$H = \begin{bmatrix} 2a_{11} & a_{21} & a_{22} \\ a_{21} & 2a_{12} & a_{23} \\ a_{22} & a_{23} & 2a_{13} \end{bmatrix}, \nabla D'(0) = \begin{bmatrix} a_{31} \\ a_{32} \\ a_{33} \end{bmatrix}$$

然后通过设置阈值剔除不恰当关键点。

2.2.3 直方图统计与描述符

计算关键点邻域各像素的梯度幅值与梯度方向:

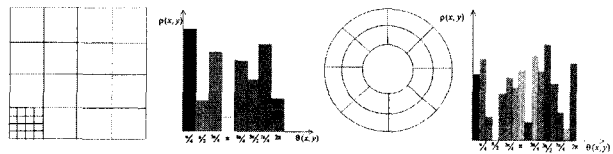
$$\rho(x, y) =$$

$$\sqrt{(D(x+1, y) - D(x-1, y))^2 + (D(x, y+1) - D(x, y-1))^2}$$

$$\theta(x, y) = \arctg((D(x, y+1) - D(x, y-1)) / (D(x+1, y) - D(x-1, y)))$$

在以特征点为中心的领域内,用高斯窗函数对梯度幅值 $\rho(x, y)$ 作加权,尺度为该特征点所在层尺度的 1.5 倍,这样可以避免边缘处的误匹配。将 16×16 个像素分为 4×4 的块,共 16 块,对每块作方向直方图统计,以 $\pi/4$ 为步长分为 8 个方向,各个方向上的幅值为该方向上所有幅值的和,如图 2(a)所示,每一块可得到一个长度为 8 的向量,共有 16 块,按位置连接可得到一个长度为 128 的向量,就此得到一个关键点的描述符向量。再将此向量作标准化处理,去除光照、对比度变化所带来的影响。

文献[11]所描述的 GLOH 算法是 SIFT 算子的一个变型,目的在于增加 SIFT 算法的鲁棒性和匹配精度。其直方图统计方法如下:以关键点为圆心、半径为 15 像素的范围作为关键点邻域。建立在半径方向 6, 11, 15 像素格和角度方向 $\pi/4$ 的划分可以得到 17 格区域(如图 2(b)所示,中心区域不做角度划分)。在每个区域内做步长为 $\pi/8$ 的直方图统计,可以得到一个长度为 272 的向量。同理,将此向量作标准化,以去除光照、对比度影响。



(a) SIFT 统计

(b) GLOH 统计

图 2

2.2.4 点对匹配

假设 $V_1(i)$ 和 $V_2(j)$ 分别是第一幅图像的第 i 个关键点描述符和第二幅图像的第 j 个关键点描述符,第二幅图中到 $V_1(i)$ 距离最小的描述符为 $V_2(\hat{j})$,次小距离描述符为 $V_2(\hat{j}')$,若二者满足:

$$||V_1(i) - V_2(\hat{j})|| < k ||V_1(i) - V_2(\hat{j}')||$$

可肯定 $V_1(i)$ 与 $V_2(\hat{j})$ 为一个匹配点对。

3 GPU 算法描述

3.1 GPGPU 的发展

2003 年 GPU 可编程性的实现标志着 GPGPU(GPU 通用算法编程)的出现。很多人开始利用其并行性解决一般代数问题。Moreland 等利用 GPU 实现了 FFT, 这为 GPU 应用于图像处理开拓了很好的前景^[8]。Hopf-Matthias 等人利用 GPU 实现了三维卷积, 这为高维图像算法的实现提供了很好的契机^[9], 并且在文献^[10]里将 GPU 与小波相结合来实现算法。同时, GPU 的应用在代数方面也发挥出其巨大的潜能。Hillesland 等将最速下降法和共轭梯度法求解带有简单约束和规则化的非线性最小二乘优化问题映射到最新的图形硬件上, 并将其应用到复杂的图像建模问题上^[11]。

2007 年 NVIDIA 公司发布 GPU 编程的官方开发平台 CUDA, 其核心有 3 个重要抽象概念: 线程组层次结构、共享存储器、屏蔽同步(barrier synchronization), 可将其作为 C 语言的最小扩展级公开给程序员。

GPU 用于解决可表示为数据并行计算的问题——在许多数据元素上可并行执行的程序, 具有极高的计算密度(数学运算与存储器运算的比率)。CUDA 编程模型公开了 GPU 的并行功能, 最新一代的 NVIDIA GPU 基于 Tesla 架构, 支持 CUDA 编程模型, 可显著加速 CUDA 应用程序。由于所有数据元素都执行相同的程序, 因此它对精密流控制的要求不高。由于在许多数据元素上运行, 且具有较高的计算密度, 因而它不必使用较大的数据缓存。

3.2 GPU 执行细节

以下将从 5 个方面来说明 GPU 的执行细节, 其流程如图 3 所示。

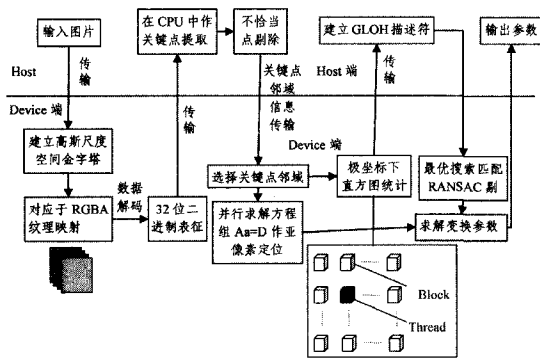


图 3 GPU 执行流程

3.2.1 尺度金字塔的建立

高斯尺度空间金字塔的构建可以利用高斯卷积的可分性通过 GPU 作并行加速。

$$I'(x, y) = I(x, y) * g_x(t)$$

$$I''(x, y) = I'(x, y) * g_y(t)$$

$$I''(x, y) = I(x, y) * g_{x,y}(t)$$

由于在 x 和 y 方向的可分性, 使得卷积可以在不同的 Block 间作并行化。

3.2.2 关键点提取

我们将图像灰度值、梯度以及高斯差分值得保存在 RGBA 纹理里, 在一个四通道的 RGBA 像素里正好存放这样 4 个尺度的数据, 最大值提取就可以在向量内部进行比较计算, 而向量内部运算是相对省时的。然后, 对于每个像素, 数据被解码为四字节即 32 位的二进制数据。到此, 数据将从 Device 端

传输回 Host 端作关键点提取, 并计算 2×2 的 Hessian 矩阵作阈值处理。

3.2.3 关键点的亚像素定位

为了充分利用 GPU 的优越性, 线性方程组的并行求解在关键点的亚像素定位上得到应用, 只需将主机端关键点邻域的少许信息传输到设备端, 然后通过求解关于三维插值函数的系数线性方程组得到关键点的精确定位。借助 CU-BLAS 函数库求解关于 a_i 的线性方程组 $Aa_i = D_i$ 。

其中 a_i 为插值函数的待定系数, D_i 为三元函数的差分响应值。

3.2.4 极坐标下的直方图统计

我们将关键点半径 15 个像素的圆邻域作为关键点描述符的邻域信息。在 CUDA 编程过程中, 首先将这些信息存储在一个网格中, 网格由二维的块构成, 而块由二维或者三维的线程构成, 在同一个块中, 各线程共享内存。于是, 同一个块的线程之间构成并行关系, 不同的块之间再次构成并行关系。CUDA 的并行接口由标识符号 $\langle\langle \text{dimGrid}, \text{dimBlock} \rangle\rangle$ 进行调用, 其参数分别表示网格和块的维度。在此, 先利用高斯窗函数对整个邻域的梯度幅值做一个加权, 而后构建圆邻域的区域划分, 每个区域相应于块, 而像素相应于线程。然后将 2π 做 16 个划分, 通过统计各个块内的像素梯度信息得到一个 17×16 的向量。该向量的标准化, 同样可以通过四通道的 RGBA 纹理映射来进行加速处理。

3.2.5 特征匹配

假设有 n 个特征向量, 相当于在两个 $n \times 272$ 的矩阵之间做一个最优搜索, 可以在这两个矩阵之间构造一个统计量, 如 SSD, Hausdorff 距离等, 由于向量维数较大, 在此以欧氏距离计算。同理, 将向量分块用 Block 表示, 向量的各个分量作为 Thread, 只需做一个欧氏距离指标的并行循环搜索, 并用随机抽样方法作不恰当匹配去除即可。

4 数据分析

4.1 本文采用实验平台

操作系统: 64 位 WindowsXP;

内存(RAM): 16.0GB;

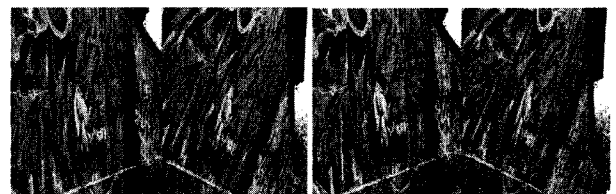
CPU: Intel(R) Xeon(R) CPU E5410 @2.33GHz;

GPU: NVIDIA Quadro FX1700 着色器 32 联合, 核心频率 460MHz, 显存 512MB, 带宽 12.8GB/s, 驱动版本 nvidia-dmkm7.15.11.7626;

编译器平台: Microsoft VS2005;

数据集: 牛津大学 Visual Geometry Group, Katholieke University, Inria Rhone-Alpes 和 Center for Machine Perception 所提供的共享数据集。

本文运用多对图片进行测试, 利用计算统一设备构架 CUDA2.0 进行混合 C 语言编程。图片 Graff 的点匹配如图 4 所示。



(a) 关键点提取

(b) 点对匹配

图 4

4.2 GPU 运算能力

表 1 显示了在不同分辨率图像数据条件下 GPU 相对于 CPU 所表现出的运算优势,总体来看,当图片分辨率较小时, GPU 的加速效果并不明显,随着数据量的增大, GPU 的加速比显著增长,在处理 800 万像素的图片时,加速比达到了 180 倍。但是当数据量大到一定程度时,加速比的增长率开始下降,这是由硬件条件和 GPU 的块线程数目局限所决定的。

表 1 CPU 与 GPU 运算时间比较

图像分辨率	CPU 运行时间(ms)	GPU 运行时间(ms)	加速比(倍)
256×256	1048.2	54.31	19.3
800×640	20865	181.75	114.8
1018×1008	36958	254.53	145.2
3246×2448	115290	627.60	183.7

4.3 数据集实验

图 5 显示了以 15 个不同图片对为例,本文算法同 SIFT 算法和 PCA-SIFT 算法的匹配率比较。首先,本文由于以 GPU 的运算能力为代价采用了不降维情况下的 GLOH 算法,使得匹配精度大大提高,在上述实验条件下,本文算法的正确匹配率明显高于 SIFT 算法和 PCA-SIFT 算法。其次,本文采用三维二次函数的亚像素定位方法,使得参数估计阶段的参数输出有更高的精度。

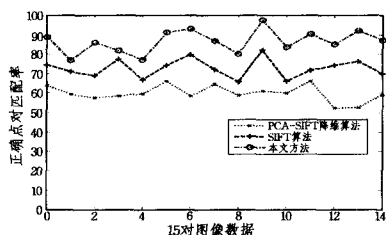


图 5

结束语 通过实验数据可以发现, GPU 的并行硬件架构优势相对于 CPU 的串行模式十分明显,使得图像配准的实时应用有了更好的硬件基础。同时, GPU 的高效并行模式为配准精度提供了余地,使得算法复杂度有了更多的空间。

GPU 的纹理存储技术和块网格并行结构,以及块内线程的共享内存机制使得并行程序的效率大大提高。但是,块内线程数量有限是其并行化的瓶颈,所以在串行算法并行化的过程中须考虑这一点。

基于局部特征描述符的点匹配算法在很多方面仍值得改进,鉴于 GLOH 算法的高匹配率,可以在关键点提取时,适当

放宽阈值,以获取更多的点对。可以选取关键点附近更大的邻域,然后通过主成分分析稍作降维,以获取更大的精度等等。 GPU 对于图像配准的应用可以推广到机器视觉的其他方面,这也将是作者以后的研究方向。

参考文献

- [1] Lowe DG. Distinctive image features from scale-invariant keypoints[J]. International Journal of Computer Vision, 2004, 60 (2):91-110
- [2] Harris C, Stephens M J. A combined corner and edge detector [C]// Proceedings of Fourth Alvey Vision Conference, 1988. Manchester: ACCV, 1988: 147-151
- [3] Mikolajczyk K, Schmid C. Indexing based on scale invariant interest points [C]// Proceedings of Eighth IEEE International Conference on Computer Vision, 2001. Vancouver: ICCV, 2001: 525-531
- [4] Bay H, Tuytelaars T, Gool L V. SURF: Speeded Up Robust Features [C]// Ninth European Conference on Computer Vision, 2006. Graz: ECCV, 2006: 404-417
- [5] Ke Y, Sukthankar. R. PCA-SIFT: A more Distinctive Representation for Local Image Descriptors [C]// Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. Washington, DC: CVPR, 2004: 506-513
- [6] Mikolajczyk K, Schmid C. A performance evaluation of local descriptors [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005, 27(10): 1615-1630
- [7] Loncomilla P, Ruiz-del-Solar J. Improving SIFT-Based Object Recognition for Robot Applications [J]. Image Analysis and Processing, 2005: 1084-1092
- [8] Moreland, Kenneth, Angel E. The FFT on a GPU [C]// Proceedings of Graphics Hardware. July 2003: 112-119
- [9] Hopf M, Ertl T. Accelerating 3 D convolution using graphics hardware [C]// Proceedings of IEEE Visualization. 1999: 471-474
- [10] Hopf M, Ertl T. Hardware accelerated wavelet transformations [C]// Proceedings of EG/ IEEE TCVG Symposium on Visualization. 2000: 93-103
- [11] Hillesland K E, Molinov S, Grzeszczuk R. Nonlinear optimization framework for image based modeling on programmable graphics hardware [J]. ACM Transactions on Graphics, 2003, 22(3): 925-934
- [12] Zhou Dengwen, Shen Xiaoliu. Image denoising using block thresholding [J]. Image and Signal Processing (CISP '08), 2008, 3 (5): 335-338
- [13] Do M N, Vetterli M. The contourlet transform: an efficient directional multiresolution image representation [J]. IEEE Transactions on Image Processing, 2005, 14(12): 2091-2106
- [14] 贾建, 项海林. 基于剪切不变的递归 Contourlet 变换图像去噪 [J]. 计算机科学, 2009, 36(5): 254-256
- [15] 付仲凯, 王向阳, 郑宏亮. 一种新的非下采样 Contourlet 域图像去噪算法 [J]. 计算机科学, 2009, 36(11): 286-289

(上接第 294 页)

- [9] Selesnick I W. 2-D Double-density Complex Wavelet Transform [J/OL]. <http://taco.poly.edu/selesi/DoubleSoftware/index.html>
- [10] Gilboa G, Sochen N, Zeevi Y Y. Texture Preserving Variational Denoising Using an Adaptive Fidelity Term [C]// Proc. VLISM. Nice, France, Oct. 2003
- [11] Chen G Y, Bui T D, Krzyzak A. Image denoising with neighbour dependency and customized wavelet and threshold [J]. Pattern Recognition, 2005, 38: 115-124