Turbopix 算法的 CUDA 并行实现 *

徐佳栋，袁红星，吴少群，余辉晴
(宁波工程学院 电子与信息工程学院，浙江 宁波 315016)

摘 要：过分割是计算机视觉领域流行的图像预处理方法。针对其运行速度慢的缺点，对广泛采用的 Turbopix 算法提出 CUDA 并行优化的方法。通过每个线程执行一个超像素扩张的任务分配，实现了水平集函数的并行演化；利用纹理存储空间和常数存储空间的优化策略，改善了数据访存的效率。实验结果表明，在 GT 240M 平台上，平均加速比达到了 15 以上。

关键词：过分割；超像素；Turbopix；CUDA
中图分类号：TP391 **文献标识码**：A **文章编号**：1674-7720(2013)12-0035-03

CUDA-based parallel implementation of Turbopix algorithm

Xu Jiadong, Yuan Hongxing, Wu Shaoqun, Yu Huiqing
(School of Electron and Information Engineering, Ningbo University of Technology, Ningbo 315016, China)

Abstract: Over-segmentation is a popular method for image pre-processing in computer vision. In order to overcome its intensive computational requirement, a CUDA based parallel implementation for widely used Turbopix algorithm is proposed. The parallel level-set evolution is implemented via superpixel expansion in each thread. The texture and constant memory optimization techniques are also adopted to improve performance. Experimental results in a GT 240M graphics card show that a speed up of more than 15X on average is achieved.

Key words: over-segmentation; superpixel; Turbopix; CUDA

超像素是图像中局部区域内连通、亮度相近、边缘描述性较好的像素集合。将图像划分成超像素后，其图像单元更加符合人们期望的结构粒度[1]。由于能够更有效地表示图像结构，超像素被广泛应用于图像内容标记[1]、虚拟漫游[2]、图像分割[3-6]等领域。将图像表示由像素转换成超像素的过程称为过分割(Over Segmentation)。常用的过分割算法有均值漂移[7]、分水岭[8]、N-Cuts[9]和 Turbopix 算法[10]。均值漂移和分水岭算法的优点是计算复杂度较低，缺点是平滑区域存在着严重的欠分割(Under Segmentation)现象。N-Cuts 和 Turbopix 算法通过紧致性约束(Compactness Constraint)有效解决了该问题。N-Cuts 综合考虑全局和局部信息，利用图论算法对图像内容进行划分。图像中的像素对应图的节点，像素间的邻域关系对应图的边，边的权重反映了相邻像素间的相似性。但是 N-Cuts 算法是 NP 难题。SHI J 等人提出的谱估计算法[9]复杂度为$O(N^{3/2})$，其中N为像素数目。FEZENSZWALB P 等人基于图节点聚类准则将 N-Cuts 算法的复杂度降为$O(N\log N)$[11]，但该方法不能控制生成的超像素数目。VEDALDI A 等人在 Mean-Shift 的基础上提出了计算速度更快的 Quick-Shift 算法[12]；LUCCHI A 等人提出了计算复杂度为$O(N)$的简单线性迭代聚类算法 SILC(Simple Linear Iterative Clustering)[13]，但这两种方法在平滑区域都存在欠分割现象。在计算机视觉领域曲线演化算法[14-15]的启发下，LEVINSHEIN A 等人提出的 Turbopix 算法[10]具有与 N-Cuts 算法同等或更优的性能，但显著降低了计算时间，其计算复杂度和像素数目近似成线性关系。Turbopix 算法通过自适应局部结构的种子膨胀实现超像素的分割，其核心思路是将复杂的超像素分割难题简化为易解的几何流(Geometry Flow)问题。

虽然 Turbopix 算法与 N-Cuts 算法相比计算速度有显著提高，但对于中等分辨率的图像，其计算时间仍需要十几秒。为了进一步提高过分割的计算速度，本文在分析 Turbopix 算法并行性的基础上，提出在 GPU 上 CUDA 并行实现的方法。

1 Turbopix 算法分析

Turbopix 算法的基本思路是设计一个几何流，使得曲

* 基金项目：王伟明助创基金；宁波市智慧产业人才基地第四批核心引导课程；宁波市自然科学基金(2012A610043)

线在演化过程中逼近超像素的边界。定义一个连续平滑的标量函数 $\psi:R^2\times[0,\tau)\rightarrow R^2$，并将超像素的边界定义为 ψ 的零水平集，边界曲线上点的运动速度为 S。则函数 ψ 的演化方程[10]为。

$$\psi_t=-S\parallel\nabla\psi\parallel \tag{1}$$

通过式(1)的不断演化逐渐扩张超像素的边界。为便于计算机求解，将式(1)离散化为：

$$\psi^{n+1}=\psi^n-S_I S_B\parallel\nabla\psi^n\parallel\Delta t \tag{2}$$

其中，ψ 定义为图像平面上像素到最近的超像素边界点之间的有符号欧几里得距离(为便于表示，下文中省去 ψ 的演化时刻上标)。若 $I(x,y)$ 已分配到某个超像素，则 $\psi(x,y)>0$，否则 $\psi(x,y)\leqslant 0$。S_I 表示与局部图像结构相关的速度分量，其定义如式(3)所示。S_B 表示像素与超像素边界邻近关系决定的速度分量，位于未分配区域骨架处的所有像素点 $S_B=0$，其他区域像素点 $S_B=1$。

$$S_I(x,y)=[1-\alpha\kappa(x,y)]\phi(x,y)-\beta[N(x,y)\cdot\nabla\phi(x,y)] \tag{3}$$

其中，$\kappa(x,y)$ 表示超像素边界在点 (x,y) 处的曲率，略去 (x,y) 后其定义如式(4)所示；$\phi(x,y)$ 表示像素的局部相似性，高梯度区域值较小，低梯度区域值较大，以保证超像素的边界收敛于边界处，其定义如式(5)所示；$N(x,y)$ 表示点 (x,y) 处的法线矢量，其计算公式为 $N(x,y)=\nabla\psi(x,y)/\parallel\nabla\psi(x,y)\parallel$。

$$\kappa=\frac{\psi_{xx}\psi_y^2-2\psi_x\psi_y\psi_{xy}+\psi_{yy}\psi_x^2}{(\psi_x^2+\psi_y^2)^{\frac{3}{2}}} \tag{4}$$

其中，$\psi_?$ 表示函数在?分量上的偏导。

$$\phi(x,y)=e^{-\frac{\parallel\nabla I(x,y)\parallel}{G_\sigma^*\parallel\nabla I(x,y)\parallel+\gamma}} \tag{5}$$

算法的具体计算步骤如下。

(1)计算像素的局部相似性函数 $\phi(x,y)$ 及其梯度。

(2)在图像 I 上均匀放置 K 个种子。

(3)扰动种子位置，使其偏离梯度较大的区域，以避免初始阶段超像素边界扩张过慢。

(4)有符号欧几里得距离函数 ψ 和分配值图 A 的初始化。$A(x,y)$ 若为非负值，则表示像素 $I(x,y)$ 所属超像素的序号；否则表示未分配到任何超像素中。

(5)统计演化前已分配的像素个数 C_1。

(6)将演化时刻 n 初始为 0。

(7)超像素扩张：计算速度图 S_I 和 S_B，根据式(2)更新 ψ，并由 ψ 更新分配值图 A。

(8)统计演化后已分配的像素个数 C_2。

(9)演化时刻 n 递增 1。

(10)判断终止条件：如果 $(C_1-C_2)/$图像总的像素数$>$某个常数，将 C_2 值赋给 C_1 并跳到步骤(7)，否则跳到步骤(11)。

(11)后处理：将未分配的像素划分到最邻近的超像素中。

(12)从 ψ 中提取零水平集，作为超像素的边界。

2 Turbopix 算法的 CUDA 并行实现

2.1 局部相似性函数及其梯度计算的 CUDA 优化

由式(5)可知计算局部相似性函数 ϕ 的关键步骤是输入图像及其平滑版本梯度幅值的计算。首先介绍输入图像梯度幅值的 CUDA 优化。由于计算每个像素点处的梯度幅值任务是彼此独立的，因而适合于 CUDA 并行实现。为加快图像数据的访问，将其绑定到纹理存储器，通过纹理存储器的缓存特性提高数据访问速度。对于式(5)中 $G_\sigma^*\parallel\nabla I(x,y)\parallel$ 的计算，首先利用 CUDA SDK 自带的递归高斯滤波器对输入图像进行预处理，然后计算其梯度。采用类似的方法实现相似性函数 ϕ 的梯度计算。

2.2 种子初始放置与扰动的 CUDA 优化

根据 Turbopix 算法，首先将种子在图像平面上等距均匀放置，然后扰动种子位置，使其偏离高梯度区域。从相似性函数 ϕ 的定义可知，在高梯度区域其值较小，反之则较大。因此，本文 CUDA 优化的思路是，将图像划分成 $\lceil w/K\rceil\times\lceil h/K\rceil$ 个单元，每个单元的大小均为 distx×disty；然后将种子放置在每个单元的中心位置；最后将种子移到每个单元最大处。为减少计算量，每个单元内部搜索 ϕ 局部最大值范围限定在中心位于单元中心处的边长为 $2r$ 的正方形之内，r 的取值应为小于 $\lceil w/K\rceil$ 的某个正数。

2.3 距离函数与分配值图初始化的 CUDA 优化

距离函数 ψ 与分配值图 A 的初始化都涉及 2.2 节计算的种子位置，因而将两者的初始化操作合并。ψ 初始化为像素到最近种子位置的有符号欧几里得距离。实现思路是：比较像素到 K 个种子的欧几里得距离，取最小值作为该像素处的 ψ 初始值，同时将最小距离对应的种子序号作为分配值图 A 在该像素处的初始值。为加快访问种子位置的速度，将 2.2 节计算的种子位置拷贝到一个常数存储空间内。

2.4 已分配像素个数统计的 CUDA 优化

根据分配值图 A 的定义可知，已分配像素的个数为 $\sum_{x,y}[A(x,y)\geqslant 0]$。这是一个并行规约的优化问题，本文采用张舒等人提出的 CUDA 实现方法[16]。

2.5 超像素扩张的 CUDA 优化

超像素扩张首先通过式(2)更新距离函数 ψ，然后由 ψ 更新分配值图 A。在此涉及速度函数 S_I 和 S_B 以及 ψ 的一阶和二阶偏导等的计算。其中分配值图 A 的更新方程为：

$$A^{n+1}(x,y)=\begin{cases}A^n(NP(x,y)), & \psi^{n+1}(x,y)\leqslant 0\\ 0, & \text{否则}\end{cases} \tag{6}$$

其中，$NP(x,y)$ 表示距离像素 $I(x,y)$ 最近的超像素边界点。

2.6 后处理的 CUDA 优化

当超像素边界扩张终止时，还有部分像素处于未分配状态。后处理就是将这些未分配的像素划分到距离其最近的超像素中。

2.7 超像素边界提取的 CUDA 优化

超像素的边界为距离函数 ψ 的零水平集，1 表示边界，0 表示非边界。这实际上就是分配值图中非负值和负值之间的边界。

3 实验与结果讨论

本文实验的硬件配置为 Intel Core2 Duo 2.20 GHz CPU，2 GB 内存；GeForce GT 240 M 1.21 GHz GPU，16 KB 共享内存，436 MB 全局内存。软件配置为 Windows 7+Visual Studio 2008 +CUDA SDK 4.0 +NVIDIA Driver for Windows7(275.33)。

将 Turbopix 算法的 CPU 实现[10]与本文的 CUDA 实现进行对比，测试图像为参考文献[10]提供的 lizard。图 1 给出了在不同超像素个数下，本文 CUDA 优化实现的平均加速比。在该实验中，测试图像 lizard 的分辨率为 518×344。从图 1 可以看出，经过 CUDA 优化后平均加速比达到了 15 以上。

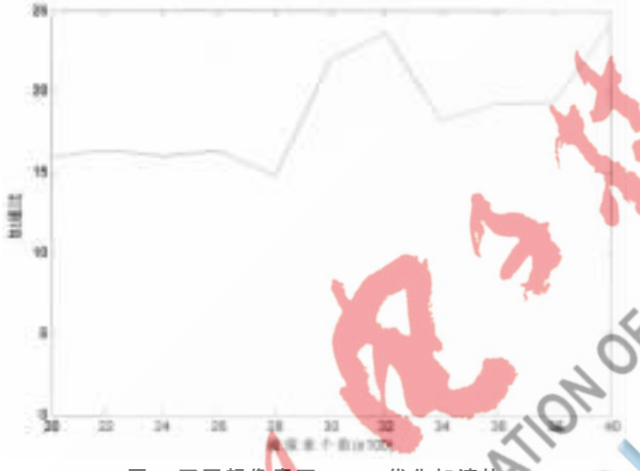
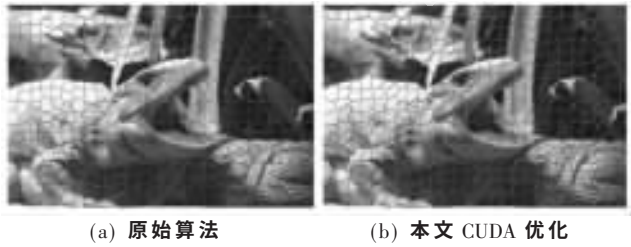


图 1 不同超像素下 CUDA 优化加速比

图 2 给出了超像素个数为 500 时，原始算法[10]和本文 CUDA 优化得出的过分割结果。从图 2 可以看出，本文 CUDA 优化输出的结果与原始算法结果存在差异。原始算法过分割的结果更接近目标的边界，而本文优化的结果则在各超像素大小上更趋于一致。这种差异存在的主要原因是这两种实现的 S_B 速度计算方法不一样。原始算法将位于未分配区域骨架处的所有像素点对应的 S_B 设为 0，其他区域处则设为 1。本文为便于 CUDA 实现，根据分配值图确定 S_B 的值。



(a) 原始算法　　(b) 本文 CUDA 优化

图 2 过分割结果比较

通过研究 Turbopix 算法的原理，本文提出了在 GPU 上高速并行实现的方法。该方法利用超像素边界扩张的数据独立性，提出了原始算法关键步骤 CUDA 并行优化的思路，并通过纹理存储器和常数存储器优化了内存访问的效率。未来工作是改进速度分量的计算方法，使本文实现结果与原始算法结果更加一致；进一步优化 CUDA 实现方式，使加速比能有更大的提高。

参考文献

[1] 刘靖，李翠华，杨敦旭. 一种基于超像素的户外建筑图像布局标定方法[J]. 厦门大学学报(自然科学版)，2010，49(2)：175－180.

[2] 袁淑娟，高秀芬. 基于图像精确过分割的虚拟现实场景构建[J]. 计算机工程与设计，2009，30(17)：4044－4046.

[3] 韩守东，赵勇，陶文兵，等. 基于高斯超像素的快速 Graph Cuts 图像分割方法[J]. 自动化学报，2011，37(1)：11－20.

[4] 刘陈，李凤霞，张艳. 基于图割与泛形信息的对象分割方法[J]. 计算机辅助设计与图形学学报，2009，21(12)：1753－1760.

[5] 方浩，仇丽英，卢鹏鹏. 基于区域划分和再融合的全幅视觉图像分割[J]. 北京理工大学学报，2009，29(9)：799－802.

[6] 刘陈，王欣欣，李凤霞，等. 一种快速保边的图像对象分割方法[J]. 北京理工大学学报，2010，30(2)：183－187.

[7] COMANICIU D, MEER P. Mean shift: a robust approach toward feature space analysis[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002,24(5): 603－619.

[8] VINCENT L, SOILLE P. Watersheds in digital spaces: an efficient algorithm based on immersion simulations[J].IEEE Transactions on Pattern Analysis and Machine Intelligence, 1991,13(6):583－598.

[9] SHI J,MALIK J. Normalized cuts and image segmentation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000,22(8):888－905.

[10] LEVINSHTEIN A, STERE A, KUTULAKOS K N, et al. TurboPixels: fast superpixels using geometric flows[J].IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009,31(12):2290－2297.

[11] FELZENSZWALB P,HUTTENLOCHER D. Efficient graph－based image segmentation[J].International Journal of Computer Vision, 2004(1):167－181.

[12] VEDALDI A, SOATTO S. Quick shift and kernel methods for mode seeking[C]. Proceedings of the 10th European Conference on Computer Vision, Marseille, France, 2008: 705－718.

[13] LUCCHI A, SMITH K, ACHANTA R. A fully automated approach to segmentation of irregularly shaped cellular structures in EM images[C]. Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention, Beijing, China, 2010:20－24.

[14] CASELLES V, CATTE F, COLL T, et al. A geometric

model for active contours in image processing[J]. Numerische Mathematik, 1993,66(1):1-31.

[15] MALLADI R, SETHIAN J, VEMURI B. Shape modeling with front propagation:a level set approach[J].IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995,17(2): 158-175.

[16] 张舒,褚艳利,赵开勇,等.GPU 高性能计算之 CUDA[M]. 北京:中国水利水电出版社, 2009.

(收稿日期:2013-02-21)

作者简介:

徐佳栋,男,1992 年生,本科,主要研究方向:电路设计。

袁红星,男,1980 年生,博士,高级工程理师,主要研究方向:信号与信息处理、3D 视频信号处理。

吴少群,女,1981 年生,硕士,讲师,主要研究方向:数字信号处理。