大尺度图像编辑的泊松方程并行多重网格求解算法

杜振龙[1] 李晓丽[1] 郭延文[2] 杨小健[1] 沈钢纲[1]
(南京工业大学电子与信息工程学院 南京 210009)[1]
(南京大学软件新技术国家重点实验室 南京 210000)[2]

摘 要 随着获取设备的发展,大尺度、高分辨率数字图像已逐步进入人们的生活,大尺度图像的梯度域编辑显得更为重要,求解大规模未知数的泊松方程是大尺度图像梯度域编辑的关键。传统多重网格算法的迭代、约束和插值操作单独进行,内存和外存间通讯量大,算法效率低,为此提出了一种面向大尺度图像梯度域编辑的并行多重网格求解泊松方程的算法。该算法利用多重网格的迭代、约束和插值过程的内存数据访问局部性和更新相关性,构造滑动工作窗口,使迭代、约束和插值操作并行运行,提高了多重网格算法求解泊松方程的计算效率。全景图拼接实验表明,所提算法的运行效率高于超松弛迭代、高斯塞德尔迭代和传统多重网格算法。

关键词 泊松方程,并行多重网格,大尺度图像编辑

中图法分类号 TP391 **文献标识码** A

Parallel Multigrid Approach for Solving Poisson PDE in Gigapixel Image Editing

DU Zhen-long[1] LI Xiao-li[1] GUO Yan-wen[2] YANG Xiao-jian[1] SHEN Gang-gang[1]
(College of Electronics and Information Engineering, Nanjing University of Technology, Nanjing 210009, China)[1]
(State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing 210000, China)[2]

Abstract With the development of image acquisition technology, gigapixel images are being produced and emerged into the modern society, and how to efficiently compile these gigapixel images within gradient domain is the research focus of image processing and computer graphics. To solve the Poisson PDE with large-scale unknowns is crucial to gigapixel image editing in gradient domain. Traditional multigrid approach separately performs iteration, restriction and prolongation, bears heavy communication load between RAM and external memory. In the paper, a parallel multigrid approach for solving poisson PDE was proposed, which exploits the locality and relevance of memory accessing and updating among the different stages to parallelly perform the iteration, restriction and prolongation in the sweeping window. Experiments of image stitching show that the presented method has the higher efficiency than the algorithms of successive over-relaxation, gauss-seider iteration and traditional multigrid.

Keywords Poisson PDE, Parallel multigrid, Gigapixel image editing

1 引言

泊松方程属于椭圆型偏微分方程,广泛应用于机械、物理、信息等领域的科学问题处理。自 Pérez 等[1]把泊松方程应用于图像编辑之后,基于泊松方程的梯度域图像编辑[2-4]成为图像处理的热点,例如,利用泊松方程进行图像拼接[2]、蒙太奇[3]、抠像[4]等图像编辑操作,取得了照片级的图像编辑效果。然而,随着数字设备的发展,获取图像的分辨率越来越高,编辑图像所要处理的图像尺寸也越来越大,基于泊松方程的图像编辑方法的时空复杂度也随着问题规模的增大而越来越高。因此,研究基于泊松方程的大尺度图像编辑的快速求解具有重要的现实需求和意义。

泊松方程的迭代式[5]求解在高频部分收敛很快,而在低频部分收敛则极慢[5],容易陷入局部光滑区域,不适合求解大规模未知数的泊松方程。多重网格式[6]求解以多分辨率方式在多个频段层次分别平滑误差余量,其效率优于迭代式求解算法,在大尺度图像的梯度域编辑上运用较多。但传统多重网格的迭代、约束和插值操作单独运行,在求解大规模未知数方程时存在内存和外存间通讯量大的问题,影响了计算效率。

本文在传统多重网格算法的基础上,探讨了多重网格计算过程迭代、约束和插值的内存访问和更新效率问题,提出了一种利用运算过程内存局部一致性和数据访问相关性的并行多重网格求解方法。所提出的并行多重网格求解泊松方程算法并行进行所构造的滑动窗口中的迭代、约束和插值操作,有

到稿日期:2012-10-19 返修日期:2012-12-27 本文受国家自然科学基金项目(61073098),教育部高等学校博士点基金(20113221120003),江苏省六大人才高峰基金(2012-WLW-023),江苏省自然科学基金(BK2009081),江苏省科技支撑计划项目(SBE201077457),江苏省高校自然科学基金(09KJB520006,11KJD520007),南京大学软件新技术国家重点实验室开放基金(KFKT2008B15),东南大学计算机网络和信息集成教育部重点实验室(K93-9-2010-04)资助。

杜振龙(1971—),男,博士,副教授,主要研究方向为可信媒体认证、计算机图形学、机器学习、云计算等,E-mail:duzhl@njut.edu.cn。

效降低了内、外存之间的通信量，提高了算法的运行效率。

本文所提方法的创新性在于提出了一种并行多重网格算法，它能够求解大规模未知数的泊松方程，适合处理大尺度图像的梯度域编辑操作。

2 泊松方程

二元函数 $U(x,y)=R^2\to R^2$（R 为实数域）的拉普拉斯型偏微分方程的一般形式如式(1)所示。

$$\Delta U(x,y)=\nabla\cdot\nabla U(x,y)=f(x,y) \tag{1}$$

式中，Δ 为拉普拉斯算子，∇ 为偏微分算子，$f(x,y)$ 是定解条件。当 $f(x,y)=0$ 时，式(1)为纽曼型泊松方程；当 $f(x,y)=$ 常数时，式(1)为狄雷克利型泊松方程。

在图像域 $\Omega=R\times R$，梯度域图像编辑的目的是使函数 $U(x,y)$ 与给定的梯度场 $\vec{G}(x,y)$ 最为接近，即 $\min\|U-\vec{G}\|$，该泛函等价于求解泊松方程，如式(2)所示。

$$\Delta U=\nabla\cdot\vec{G}=div\vec{G} \tag{2}$$

式中，div 为散度算子。

对式(2)采取五点差分格式离散[8]（见图 1），则式(2)的离散形式表示为：

$$U(x+1,y)+U(x,y+1)+U(x-1,y)+U(x,y-1)-4U(x,y)=div\vec{G}(x,y)$$

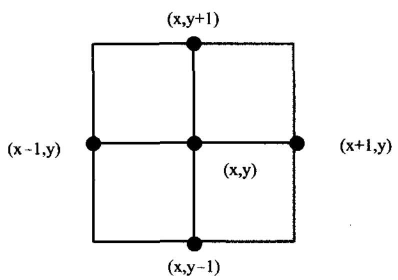

图 1　五点差分格式

Ω 域中每个像素 (x,y) 存在一个线性方程，所有方程联立得到线性方程组，用矩阵 $LU=B$ 表示，其中 $L=(a_{ij})_{n\times n}$、$U=(u_i)_{n\times 1}$ 和 $B=(b_i)_{n\times 1}$ 分别为拉普拉斯矩阵、未知数矩阵和边界条件矩阵。

泊松方程的数值求解方法有迭代法[5]（包括高斯-塞德尔迭代、雅可比迭代、共轭梯度迭代）、多重网格方法[5]和离散余弦变换等求解方法。当 Ω 域中包含的待求解未知数个数为 10^4 时，矩阵 L 的大小为 $10^4\times10^4$。大尺度图像（未知数个数大于 10^6）的 L 大小为 $10^6\times10^6$。尽管 L 为带状矩阵，但其可采用稀疏矩阵存贮。然而大规模的未知数会引起泊松方程求解迭代收敛极慢和求解不稳定的问题。

3 多重网格求解泊松方程

多重网格法是求解泊松方程的一种常用方法，通过迭代光滑当前层网格的高频误差分量，并把低频分量约束到下一级粗网格继续光滑。其实质是在粗网格通过迭代光滑高频分量，然后再将其约束到更粗一层网格，继续进行迭代光滑，直到误差足够光滑。每层粗网格的迭代为上一层细网格提供更加准确的误差校正结果。这种逐层多分辨率方式能迅速光滑不同频谱的高频分量，有效地加速了收敛速度。

设离散网格步长为 h，则泊松方程的 h 层离散形式可表示为式(3)。

$$L_hU_h=B_h \tag{3}$$

设 U_h 和 $\overline{U}_h$ 分别为式(3)的准确解和近似解，两者间的误差量 V_h 及式(3)的误差残量 d_h 分别定义为 $V_h=U_h-\overline{U}_h$，$d_h=L_h\overline{U}_h-B_h$。

多重网格算法由 V 循环过程构成，包括 3 个步骤：迭代近似求解 $\overline{U}_h$；将误差残量 d_h 约束到粗网格 H 上，如式(4)所示，并求解方程 $L_HV_H=-d_H$；将误差校正结果用插值方式返回到细网格 h，如式(5)所示，并在最细网格把误差校正结果添加到方程的近似解，即 $\overline{U}_h^{new}=\overline{U}_h+\overline{V}_h$（$\overline{V}_h$ 为 V_h 的近似解）。多重网格求解过程用到约束算子 R 和插值算子 P，其定义分别如式(4)、式(5)所示。

$$d_H=Rd_h \tag{4}$$

$$\overline{V}_h=P\overline{V}_H \tag{5}$$

一个多重网格 V 循环包括一个从细网格到粗网格的“粗化”过程和从粗网格到细网格的“细化”过程。“粗化”从最细网格开始迭代求解，把误差残量 d 约束到粗网格进行迭代“磨光”，一直到最粗网格，并在最粗网格求解未知数个数最少的方程组 $L_0V_0=d_0$。“细化”过程通过插值把误差校正从粗网格返回到细网格，直至最细网格，得到满足条件解 $\overline{U}^{new}$。“粗化”和“细化”分别对应图 2 的约束和插值操作。

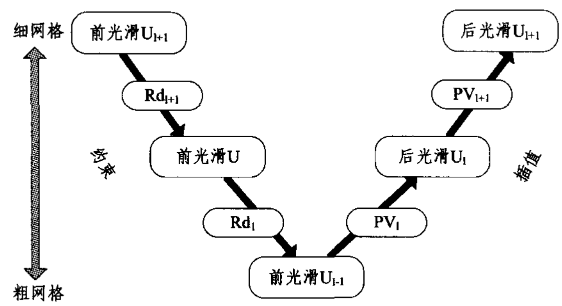

图 2　多重网格 V 循环过程

多重网格 V 循环的约束和插值都要进行迭代，即进行误差“磨光”操作。在约束操作之前的迭代也叫前光滑，在插值之后的迭代也称为后光滑，如图 2 所示。

4 并行多重网格求解泊松方程

传统多重网格采用 V 循环求解，即先进行近似迭代求解，再通过算子 R 把误差残量 d 逐层约束到粗网格，并通过算子 P 把误差校正 V 以插值方式传回细网格，算法如下所示。

算法 1　多重网格 V 循环求解算法

输入：k，N；

1. 近似求解 LU=B，得到 $\overline{U}_{old}$；

2. 约束

Do h=N−1，…，2，1

　2a. 迭代 k 次近似求解 $L_hU_h=B_h$，得到 $\overline{U}_h$；

　2b. $\overline{V}_h\leftarrow U_h-\overline{U}_h$，$d_h\leftarrow L_h\overline{U}_h-B_h$；

　2c. $d_h=Rd_{h+1}$

　End Do

3. 求解 $L_0V_0=d_0$；

4. 插值

　Do h =2，…，N−1，N

　　4a. 迭代 k 次近似求解 $L_hV_h=d_h$；

　　4b. $V_h=PV_{h-1}$

4b. 若 $h=1, \bar{U}_h \leftarrow \bar{U}_{old}+V_h$；

End Do

其中，k 为迭代次数，N 为多重网格层数。

上述算法的近似求解(2a 和 4a)、约束(2c)和插值(4b) 3 步操作单独进行，每一层网格的数据 V 和 d 在内存中需要装载两次，一次用于约束运算，另一次为插值运算。受内存容量所限，运算过程只能将部分运算数据载入内存，未载入数据只能在运算过程根据运算需要逐次载入内存。因此传统多重网格算法求解超大规模未知数的泊松方程时，会产生内存与外存的大量通讯，导致运算时间显著增加、运算效率降低。

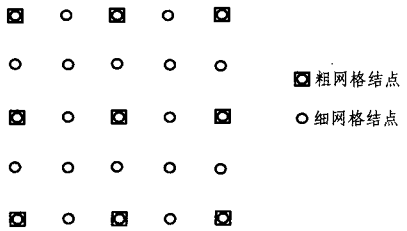

图 3 粗、细层网格节点关系

图 3 给出了粗、细相邻两层网格间节点的位置关系示意图，由图可知，矩阵 L、B、U 和 $\bar{U}$ 在内存中只需存贮一次，在不同层网格用到时只需根据位置进行索引。V 和 d 在每一层网格都会产生，在不同层间进行数值传递时要用到，且约束阶段计算 d 在插值阶段也要用到，因此，V 和 d 在计算过程要驻留内存。N 层网格计算中用到的 V 和 d 的未知数个数计算如式(6)所示，$|\cdot|$ 为计算矩阵的未知数个数。对于大尺度图像的梯度域编辑操作，当 $N=5$ 时，V 或 d 的未知数个数为 2×10^7，L、V 及 d 3 个主要矩阵的未知数数量之和达到 5×10^7。

$$\sum_{h=1}^{N}|V_h|=\sum_{h=1}^{N}|d_h|\approx10^7\times(2-2^{1-N}) \tag{6}$$

多重网格 V 循环求解包含的大部分运算，属于矩阵-矩阵或矩阵-向量型运算，易于实现并行化。本文通过求解 $\bar{U}_h$、误差量 V_h 和误差校正 d_h 计算过程中内存数据的局部一致性来构造工作集窗口 W，沿图像列方向移动工作集 W，并更新 V_h 和 d_h，使载入内存的数据得到充分利用。

本文所提出的并行多重网格算法的并行特性体现在两个方面：不同层网格间的约束操作或插值操作传递；约束操作和插值操作之间。不同层网格间的并行是指在迭代过程中，更新所需的数据满足后，执行相应的更新运算。约束操作和插值操作间的并行是在约束操作完成后，插值更新所需的数据满足后进行插值操作，当插值至最细网格时即得到方程的解。

如图 4 所示，当 $k=2$、$N=2$，细网格的第 7、6 行执行约束 $d_2\rightarrow d_1$ 时，第 5、4 行可执行 $d_1\rightarrow d_0$ 约束。当第 1 行的数据更新后，则可计算粗网格的 V_0，此时粗网格第 2、3 行可计算 $V_1\rightarrow V_0$ 的插值操作。

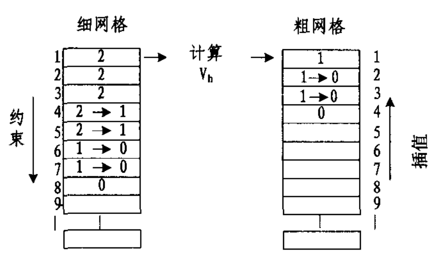

图 4 并行多重网格的并行方式

构造工作集 W 以图像行像素为单位，由当前处理行 i、相邻已处理行和待处理行组成。利用已处理行是为了更新当前行数据；引入相邻待处理行是在当前行数据更新后计算误差量和误差校正量。

表 1 列出了约束、插值阶段的数据窗口和计算相关量的窗口尺寸。以约束阶段为例，设定 i_h 为当前处理行，在细网格层 $h-1$ 的 i_h 后 $2k+1$ 行更新后，才可进行 h 层的前光滑，因而前光滑窗口为 $[i_h-1, i_h+2k+1]$。同时光滑后的行才能计算误差余量，因此误差余量窗口为 $[i_h-3, i_h+1]$。把这 3 个窗口取并集即得到约束阶段的窗口。插值阶段的数据窗口设置与约束阶段的窗口设置相似。

表 1 数据窗口 W

约束阶段	
$i_{h-1}+2k+1<\lfloor(i_h-1)/2\rfloor$	
数据窗口	$i_{h-1}+2k+1<\lfloor(i_h-1)/2\rfloor$
从 h−1 层的约束	$[i_h+2k+1]$
前光滑	$[i_h-1, i_h+2k+1]$
误差余量	$[i_h-3, i_h+1]$
插值阶段	
$i_{h+1}+2k+1<2i_h-1$	
数据窗口	$[i_h-1, i_h+2k+1]$
向 h+1 层插值	$[i_h+2k+1]$
后光滑	$[i_h-1, i_h+2k+1]$

并行多重网格算法如下所示。

算法 2 并行多重网格求解算法

输入：k，N，L(图像行数)；

1. 计算 k−1 行 $\bar{U}_h$ 作为预处理数据；
2. Do l＝K，K+1，…，L−K

 2a. 计算 $[i_l-3, i_l+2k+1]$ 行的 $\bar{U}_h$；

 2b. 计算 $[i_l-3, i_l+1]$ 行的 d_h；

 2c. 计算 $[i_l-3, i_l+1]$ 行的 V_h；

End Do

3. 计算 l＝K+1，…，L 的 V_N，并更新 U。

算法 2 的迭代、约束和插值操作并行进行，其充分利用了内存数据的局部一致和访问相关性。

5 实验

本文在同方 E2180、内存 2G 的双核台式机上实现了泊松方程的并行多重网格数值求解算法，并把算法应用到 1280×720、1024×768 分辨率的图像拼接实验中。

实验中迭代次数为 2，P 和 R 的取值分别如下。

$$P=\begin{vmatrix}\frac{1}{4} & \frac{1}{2} & \frac{1}{4}\\ \frac{1}{2} & 1 & \frac{1}{2}\\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4}\end{vmatrix}$$

$$R=\frac{1}{4}P^{\mathrm{T}}=\begin{vmatrix}\frac{1}{16} & \frac{1}{8} & \frac{1}{16}\\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8}\\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16}\end{vmatrix}$$

式中，P、R 取值根据双线性插值设置。

多重网格的划分规则是：相邻的粗网格划分步长是细网

(下转第 67 页)

gorithms for Manycore GPUs[C]// Proceedings of the 2009 IEEE International Parallel and Distributed Processing Symposium. US: IEEE Computer Society, 2009: 1-10

[8] Chhugani J, Nguyen A D, Lee, V W, et al. Efficient Implementation of Sorting on Multi-core SIMD CPU Architecture[C]// Proceedings of the VLDB Endowment. US: VLDB Endowment, 2008: 1313-1324

[9] Senggupta P, Harris P, Zhang P, et al. Scan Primitives for GPU Computing[C]// Proceedings of the 22nd ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics hardware. Switzerland: Eurographics Association, 2007: 97-106

[10] Gray J, Sundaresan P, Englert S, et al. Quickly Generating Billion-record Synthetic Databases[C]// Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data. NY USA: ACM Press, 1994: 243-252

[11] NVIDIA Corp. NVIDIA's Next Generation CUDA™ Compute Architecture: Fermi™ [EB/OL]. http://www.nvidia.com, 2009-05-15

[12] NVIDIA Corp. Tuning CUDA Applications for Fermi[EB/OL]. http://developer.nvidia.com, 2011-05-15

(上接第 61 页)

格划分步长的 2 倍，即 $H=2h$；最细一级网格是原始输入图像的尺寸。

图 5、图 6 示出用本文所提出的并行多重网格算法求解泊松方程实现的梯度域图像拼接结果。图像拼接前先采用文献[7]的配准算法对所有图像进行配准。图 5 用 5 张 1080×720 的图像拼接生成 8100×3680 的全景图，图 6 用 5 张 1024×768的图像拼接生成 7963×3580 的全景图。并行多重网格算法在图 5、图 6 实验中的内存占用、内存/外存通讯量、迭代次数和运算时间如表 2 所列。

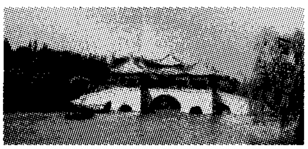

图 5　图像拼接实验 1(数据 1)

图 6　图像拼接实验 2(数据 2)

表 2　不同计算方法比较

算法		内存占用(兆)	I/O通讯(兆)	迭代次数	时间(秒)
超松弛迭代 SOR	数据 1	56	7.53	676	9.656
	数据 2	51	7.3	615	8.433
伽可比迭代	数据 1	56	7.52	1365	19.499
	数据 2	51	7.34	1127	15.451
多重网格	数据 1	224	5.95	232	3.315
	数据 2	204	5.53	209	2.866
并行多重网格	数据 1	184	3.53	165	2.357
	数据 2	180	3.36	160	2.194

从表 2 可知，并行多重网格算法在内存占用方面低于传统多重网格算法，在内存/外存通讯量、运算时间及迭代次数等方面都明显优于超松弛迭代、伽可比迭代和传统多重网格算法。

结束语　泊松方程是工程中常用的偏微分方程，其快速、精确的求解对于问题解决非常重要。基于泊松方程的梯度域图像编辑是目前图像处理和图形学的研究热点，大尺度图像的梯度域编辑操作需要求解超大规模未知数的泊松方程，传统多重网格算法在 PC 机上运算效率低。提出了一种面向大尺度图像梯度域编辑的并行多重网格求解泊松方程的算法，利用多重网格的迭代、约束和插值操作之间的内存局部性和更新相关性并以并行方式运行多重网格算法。该方法能够在 PC 机上高效地实现大尺度图像的梯度域编辑操作。

传统多重网格数值求解算法单独完成迭代、约束和插值操作，没有很好地利用内存数据的局部性，算法存在改善空间。文中所提出的泊松方程并行多重网格求解算法充分利用内存空间数据的局部性和更新相关性来并行完成迭代、约束和插值操作，提高了泊松方程的求解效率。

所提出的并行多重网格求解算法适合于求解纽曼边界条件、结构化数据的泊松方程。扩展本文工作，使之适合大尺度图像的复杂梯度域编辑操作，即适合求解狄雷克利边界的泊松方程，是未来的工作方向。

参考文献

[1] Pérez P, Gangnet M, Blake A. Poisson Image Editing[J]. ACM Transactions On Graphics, 2003, 22(3): 313-318

[2] Levin A, Zomet A, Peleg S. Seamless Image Stitching in the Gradient Domain[C]// Tomas P, Jiri M, eds. Proceedings of 8th European Conference on Computer Vision (ECCV'2004). Springer Verlag Publishing House, 2004, 1: 377-389

[3] Agarwala A, Dontacheva M, Agarwala M, et al. Interactive Digital Photomontage[J]. ACM Transaction on Graphics, 2004, 23(3): 294-302

[4] Press W H, Teukolsky S A, Vetterling W T, et al. Numerical Recipes in C[M]. US: Cambridge University Press, 2002: 871

[5] Kazhdan M, Hoppe H. Streaming Multigrid for Gradient-Domain Operations on Large Images[J]. ACM Transaction on Graphics, 2008, 27(3)

[6] Chow E, Falgout R D, Hu J J, et al. A Survey of Parallelization Techniques for Multigrid Solvers[C]// Frontiers of Parallel Processing for Scientific Computing. US: the Society for Industrial and Applied Mathematics, 2005

[7] Szeliski R. Image Alignment and Stitching: A Tutorial[J]. Foundations and Trends in Computer Graphics and Computer Vision, 2006, 2(1): 1-104

[8] 廖臣，祝大军，刘盛纲. 五点差分格式求解泊松方程并行算法的研究[J]. 电子科技大学学报, 2008, 37(1): 81-83