

基于矩阵分块化的 GVF Snake 模型

唐克伦¹,唐永亮¹,曾伟²,贾海洋¹,张洋¹,刘炎¹

(1. 四川理工学院机械工程学院,四川 自贡 643000; 2. 自贡凯迪碳素有限公司,四川 自贡 643000)

摘要:针对 GVF Snake 模型对高清图像计算速度较慢问题,通过对 GVF 力场的迭代计算过程、计算机 Cache 体系结构与原理的分析,提出了基于矩阵分块化的 GVF 力场的算法;确定了分块方式和最优分块大小。计算机仿真实验表明,这种方法能够大幅提高 GVF 外力场迭代运算速度。

关键词:GVF Snake 模型;高清图像;二级缓存;矩阵分块

中图分类号:O35

文献标识码:A

主动轮廓模型^[1](Active Contour Model),又称 Snake 模型,是 Kass^[2]等人于 1987 年提出并成功用来跟踪人脸嘴部运动。它是与 Marr 的计算视觉分层理论完全不同的计算过程,充分应用了计算视觉系统的高层与底层信息。之后它被广泛的应用到轮廓提取、区域分割、目标跟踪等多个领域。但 snake 模型也存在诸多不足,如对初始轮廓敏感、收敛凹型轮廓效果差、噪声敏感等。

针对传统 Snake 模型存在的诸多不足,许多学者对经典的 Snake 模型进行了改进。1997 年, Xu 和 Prince 提出了 GVF Snake 模型^[3],主要是针对外力场进行了改进,比较成功地解决了凹型轮廓检测,并扩大了可逼近半径,提高了 Snake 模型的应用范围。Shin-h young Kim 等人从 Snake 控制点入手,使控制点的数量能自适应轮廓不同区域,即在大曲率的范围内形成更多的控制点,而在直线区域内减少控制点的数量,加快了算法的效率与精度^[4]。

GVF Snake 模型相比较于其它算法有许多优点,特别是提取的轮廓非常精确,凹陷部位也能较好的提取出来,利于下一步的图像处理;其存在最大的问题就是算法的速度慢,特别是对于千万级像素的高清图像。通过对 GVF 外力场的迭代计算过程、CPU cache 体系结构的

分析,提出了基于矩阵分块化的 GVF 外力场的算法;确定了分块方式和最优分块大小。计算机仿真实验表明,这种方法能够大幅提高 GVF 外力场迭代运算速度。

1 GVF Snake 模型

传统的 Snake 模型有对初始轮廓敏感和无法收敛到凹陷的图像边缘等不足,限制了 Snake 模型的应用。针对传统 Snake 模型的缺陷, Xu 等人提出了一种新的外力场:梯度矢量流(Gradient Vector Flow Field, GVF)作为参数主动轮廓的外力场来改进 Snake 模型。GVF 方法相当于对图像边界进行了扩散,成功的解决了凹陷轮廓提取中对初始轮廓敏感问题。

GVF Snake 模型是把传统 Snake 模型中基于边界信息的外部能量函数替换为一个新的静态力场: $F_{ext} = [u(x, y), v(x, y)]$, u, v 分别表示图像灰度在 x, y 两个方向的变化。是由下述能量函数最小化得到 $E = \iint [(\mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) - |\nabla f|^2 |V - \nabla f|^2)] dx$ (1)

其中, u_x, u_y, v_x, v_y 分别表示 u 和 v 在 x 和 y 两个方向上的导数; f 是图像函数的梯度函数; ∇ 是梯度算子; μ 是扩散因子,其值的大小取决于图像中的噪声:噪声越大, μ 越小。在目标轮廓附近, ∇f 较大, GVF 外力场主要是能量

收稿日期:2012-04-12

基金项目:国家“十一五”科技支撑计划(2008BAD4B15);四川省教育厅青年基金项目(10ZB097);四川理工学院博士启动基金(2010ZY019);四川理工学院人才引进项目(2009XJKL002);四川理工学院研究生创新基金(Y2011001);过程装备与控制工程四川省高校重点实验室项目(GKYJ201101)

作者简介:唐克伦(1972-),男,四川泸县人,教授,博士,主要从事主动轮廓提取、计算力学方面的研究,(E-mail) cagd@tom.com

函数的第二项控制。而在远离目标轮廓的时候,很小,GFV 外力场主要由向量场的偏导数的平方和控制,既能能量函数的第一项控制,这时矢量场缓慢的向远离目标轮廓扩散。由变分法可知,GFV 外力场满足下面的 Euler 方程组^[5]:

$$\begin{cases} \mu \nabla^2 u - (u - f_x)(f_x^2 + f_y^2) = 0 \\ \mu \nabla^2 v - (v - f_y)(f_x^2 + f_y^2) = 0 \end{cases} \quad (2)$$

其中, ∇^2 是 Laplace 算子, $f_x = \frac{df}{dx}$, $f_y = \frac{df}{dy}$ 。由此可建

立迭代公式^[6]:

$$\begin{cases} u_i(x, y, t) = \mu \nabla^2 u(x, y, n) - (u(x, y, n) - \\ f_x(x, y, n)) \times (f_x^2(x, y, n) + f_y^2(x, y, n)) \\ v_i(x, y, t) = \mu \nabla^2 v(x, y, n) - (v(x, y, n) - \\ f_y(x, y, n)) \times (f_x^2(x, y, n) + f_y^2(x, y, n)) \end{cases} \quad (3)$$

在方程组(3)中, n 是迭代次数; $f_x(x, y, n)$ 和 $f_y(x, y, n)$ 是梯度图像 f 对 x, y 的偏导数, 在 GFV 外力场的迭代计算的过程中是常量, 为了减少迭代计算量可以在迭代计算开始前先计算出来。利用式(3)通过一定次数的迭代计算, 生成 GFV 外力场, 成功的改进了传统的 snake 模型, 改善了对初始轮廓的敏感性和不能收敛到凹陷轮廓问题。但是也产生了新的问题, 由于 GFV 力场是通过一定次数的迭代计算产生, 为了能够得到较好的 GFV 外力场, 迭代计算的次数都很高, 而且随着摄像技术的发展, 千万级像素的相机的出现, 工业对图像分割和轮廓的提取提出了更高的要求。不仅要实现高精度的图像提取与目标分割, 同时也要求实现高速甚至实时的图像处理, 因此研究者开始关注主动轮廓提取速度问题。

2 Cache

2.1 Cache 原理

程序运行期间, 在一个较短的时间间隔内, 程序产生的地址往往集中在很小的范围。指令地址也通常是连续分布的, 循环程序段、子程序要多次重复执行。因此 CPU 对存储器地址访问有时间集中分布(某存储单元被访问, 则可能被再次访问)和空间集中分布(某存储单元被访问, 其邻近单元也可能被访问)的倾向, 这种现象称为程序访问的局部化^[7]。由于程序访问的局部化以及 CPU 和主存在速度上的不匹配, 人们设计了 Cache 来提高计算机系统的性能, 并取得了非常好的效果。Cache 是主存与 CPU 之间的一级或多级存储器, 其特点是速度接近 CPU, 但容量小, 用来存储 CPU 近期使用的数据与指令。当 CPU 送出主存访问地址时, 首先检查标记确定访问的主存是否在 Cache 中, 若访问的字块

在 Cache 中, 则“命中”, 直接从 Cache 中读取数据或者指令。如果访问的字块不在 Cache 中, 则输出“未命中”, 通过一定的替换策略把字块从主存中加载到 Cache 中, 同时把该字块送入 CPU 或者再从 Cache 中读取该数据或者指令。由于内存的工作频率比 Cache 的工作频率慢得多, 如果发生 Cache 未命中时会使 CPU“长时间”处于临时等待状态, 严重降低计算机性能。

2.2 Cache 的加速比

在计算机系统中设置 Cache, 主要目的是提高储存系统的访问速度。Cache 系统的加速比 S_p 公式如下^[8]:

$$S_p = \frac{T_m}{HT_c + (1 - H)T_m} = f\left(H, \frac{T_m}{T_c}\right) \quad (4)$$

其中, T_m 是主存储器的访问周期, T_c 是 Cache 的访问周期, H 是 Cache 的命中率。由式(4)知: Cache 系统的加速比 S_p 是命中率 H 和主存储器的周期 T_m 与 Cache 访问周期 T_c 的函数。对于一个给定的计算机, T_m 与 T_c 都是常量。所以提高 Cache 系统加速比的最好途径是提高 Cache 的命中率 H 。

Cache 的命中率 H 主要与下面几个因素有关^[8]:

- (1) 程序在执行过程中地址流分布情况。
- (2) 当发生 Cache 块失效时, 所用的替换算法。
- (3) Cache 容量的大小。
- (4) 所采用的 Cache 预取算法等。

通过对影响 Cache 命中率 H 主要因素的分析, 提高 Cache 命中率 H 的方法可以分为两类; 一种是通过改进硬件的方法(如增加 Cache 容量, 采用硬件 Cache 预取算法), 其特点是提高 Cache 命中率效果好, 但成本高; 另一种是改进软件设计, 特点是成本低, 增加软件的设计难度, 效果根据不同程序有很大差别。通过改进软件设计方法提高 Cache 的命中率, 主要改进软件在执行过程中地址流的分布情况, 保持程序针对不同计算机硬件(主要是 Cache 容量的不同)其对存储器访问的时间局部性和空间局部性不被破坏。通过分析 GFV 力场算法程序在执行过程中地址流分布情况, 发现其算法程序对存储器访问有很好的空间局部性与时间局部性, 但随着图像像素的不断提高, 其空间和时间的局部性可能由于 Cache 容量不足而被破坏, 大幅降低 Cache 命中率。针对这种情况提出分块的方法来保持其空间局部性和时间局部性, 优化其算法程序在执行过程地址流分布。

3 图像矩阵的分块

3.1 分块方式

通过上面对 GFV 外力场的分析, 可以得出 GFV

外力场的计算过程可分为:图像数据前处理;迭代计算;计算结果处理三部分,其中迭代计算部分用的时间最多,是提高主动轮廓提取速度的主要方向。下面为基于 MATLAB 平台的 GVF 外力场迭代计算部分的代码:

```
for i = 1 to ITER    % ITER 表示迭代次数
    u = u + μ * del(u) - S * (u - f_x)    (5)
    v = v + μ * del(v) - S * (v - f_y)    (6)
end
```

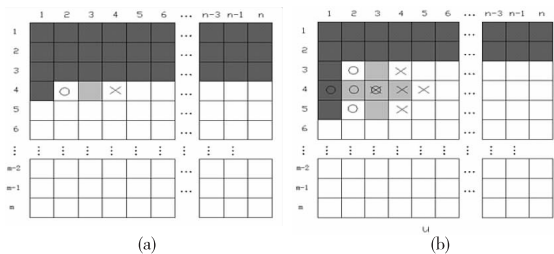
其中,del 是 Laplace 算子, $S = f_x^2(x, y, n) + f_y^2(x, y, n)$ 。式(6)和式(5)具有相同原理,为了分析的简单,仅对式(5)做详细的分析。把式(5)按照程序中算式的优先级及可能产生中间变量的原则进行更详细的划分,结果如下:

$$B = S * (u - f_x) \quad (7)$$

$$u = u + \mu * \text{del}(u) \quad (8)$$

$$u = u - B \quad (9)$$

其中, B 表示式(5) 计算过程中必须产生的中间变量。分析式(7) 知,在一次迭代运算过程中,读取了矩阵 u 和 f_x 的每个元素,计算对应位置元素的差值,再读取了矩阵 S 每个元素与 $(u - f_x)$ 对应位置的元素相乘,再把运算结果缓存到 B ;图 1(a) 表示一次 for 循环过程中对 B 的访问情况(S, u, f_x 的访问情况跟 B 相同)。式(8) 表示计算矩阵 u 的拉普拉斯算子,即 $l_{i,j} = (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) - 4u_{i,j}$,在一次循环过程中对矩阵 u 的访问情况如图 1(b)。



注:黑色表示已访问,表示上一次被访问,灰色表示正在访问,表示下一次被访问的,白色表示未访问

图 1 每次循环对矩阵访问图

通过上面对每次迭代运算过程中矩阵的访问情况的分析,在矩阵访问时在行上有很好的空间局部性(假定矩阵是按行操作),且计算过程中是循环计算,故 GVF 外力场算法有良好的程序访问的空间局部性和时间局部性,Cache 的命中率非常高。但是随着图像尺寸的不断变大,而现在大部分 CPU 的 Cache 都很小(512 kb - 4 Mb),迭代计算过程中的变量与常量不能完全装入 Cache 中,导致 Cache 的命中率非常的低,严重的影响计

算系统的性能,降低 GVF 力场计算速度。

为了能完全发挥计算机的性能,提高 GVF 力场计算速度,满足工业对图像轮廓提取与分割的要求,根据 GVF 外力场迭代计算过程中程序访问按行(或按列)上具有良好的空间局部性和时间局部性,提出基于行(或者列)的矩阵分块法,保持了算法的空间局部性和时间局部性,使得 Cache 有很高的命中率,提高了 GVF 力场原有算法对高清图像,特别是千万级像素图像的迭代计算速度。

3.2 最优分块

在矩阵分块化的 GVF 算法中,为了提高 Cache 命中率,充分发挥计算机性能,不仅要根据 GVF 力场计算过程中对存储的访问情况确定分块类型;还应在保证计算质量前提下,降低矩阵分块和计算结果的组装的额外开销而对矩阵分块大小进行优化,最大可能地提高 GVF 力场的计算速度。在应用原有 GVF 算法计算高清图像的 GVF 力场时,影响 Cache 的命中率的主要因素是图像矩阵太大,无法完全装载到 Cache 中,导致迭代运算时 Cache 替换太快,破坏 GVF 算法的时间局部性,大幅降低 Cache 命中率。对原始图像矩阵进行分块计算就是使不同 Cache 容量的计算机在迭代计算过程中能完全装入循环过程中所需的矩阵数据,使计算过程中会被密集使用的数据不会在密集使用期间被替换出 Cache,保证 Cache 命中率。对于特定的计算机 Cache 的容量大小会确定,进行优化的主要方向是尽量减少一个循环的循环过程中密集使用数据的量,在保证 Cache 完全装入循环过程中所需的矩阵数据的前提下,使分块矩阵最大化。方程组(3)中的两个方程是独立的,可以分别求解,可以增加分块的大小。最优的分块大小,根据不同的计算机系统,分块大小会不一样,这里给出最优分块的估计式

$$N \leq \alpha \frac{1024^2 C}{LV} \quad (10)$$

其中, N 表示分块后子矩阵的元素数量; α 是 Cache 占有率因子,表示循环过程中变量数据在 Cache 中所占比例,因 Cache 要存储 CPU 最近使用的数据与指令,而不同程序中数据和指令的比例可能差异很大,通过对大量程序的分析一般的取值范围 0.4 - 0.7; C 表示计算机系统中 Cache 的容量(单位 MB); L 表示一个矩阵元素的长度(单位 B); V 表示循环过程中矩阵变量的个数。

通过以上的分析,确定分块类型与分块大小后。得出基于分块化的 GVF 力场的算法流程如图 2 所示:

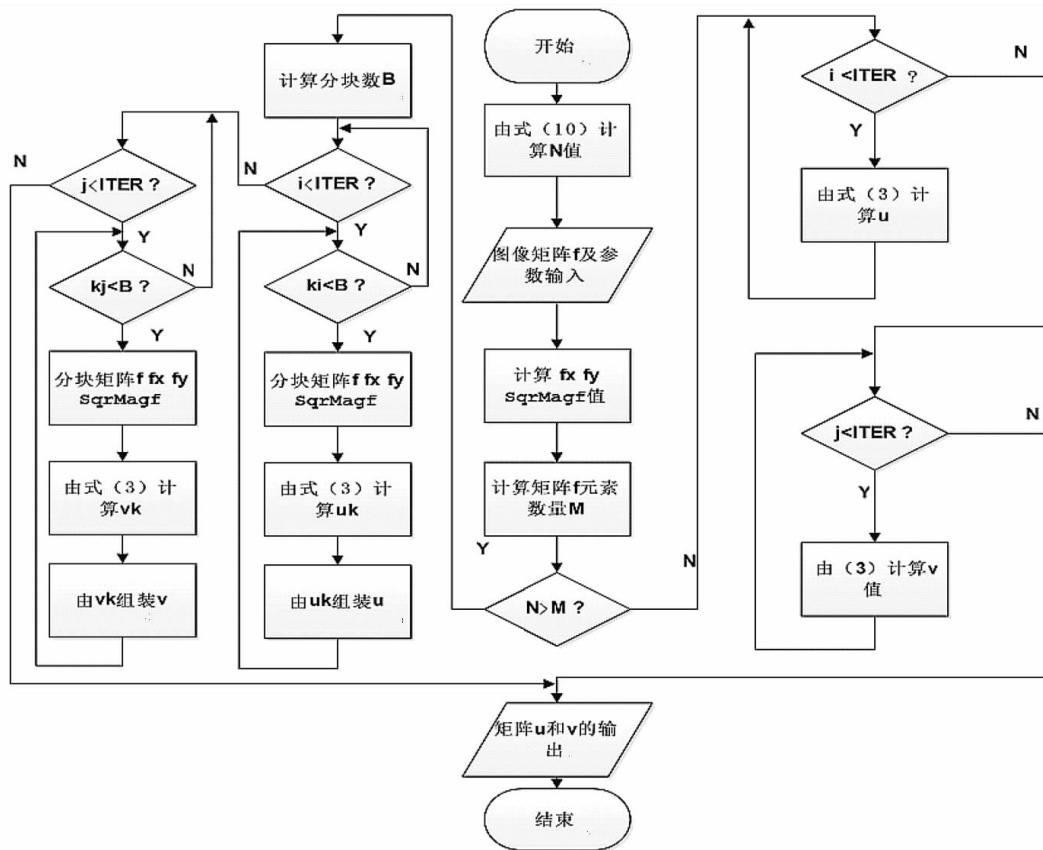
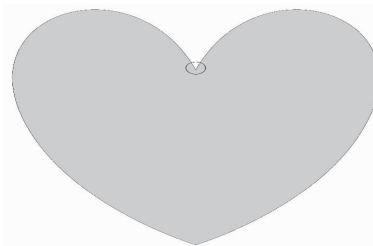


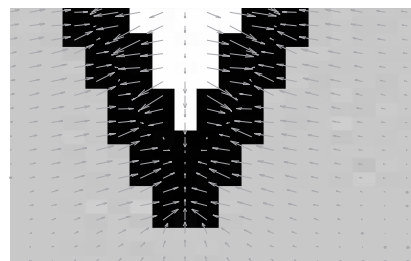
图2 基于分块化的GVF算法流程图

4 实验及结果分析

计算机仿真实验是在如下的软硬件环境下实现的。计算机型号:Lenovo A6800K;操作系统:Windows 7 Home Basic; 仿真软件:MATLAB 2010 a;CPU:Intel E7500 @ 2.93 GHz(3 MB, 12 - Way, 64 byte lines); Memory:2 GB(DDR3 1333 MHz)。图3(a)是实验的原始图像f(3992 × 3240 pixel),图3(b)是实验原始图像的GVF矢量图。由式(10)及实验平台信息(L = 8, V = 4, C = 3, α取0.6)得 $N \leq \alpha \frac{1024^2 C}{LV} = 0.6 \times \frac{1024^2 \times 3}{8 \times 4} = 58982$; 故在本实验中最优分块数 $B = (3992 \times 3240) / N \approx 220$ 附近。由图4 Block-time实验曲线知最优分块数是222,证明式(10)能够很好估计最优分块数目,满足工程应用。在式(4)中,取H = 1时及计算机硬件配置可得最大Cache系统加速比为 $Max(S_p) = \frac{T_m}{T_c} = \frac{2.93GHz}{1333MHz} = 2.1980$,由图4得实验的Cache系统加速比为 $S_{p实} = \frac{T_1}{T_{222}} = \frac{377.4570}{191.7640} = 1.9683$, Cache效率 $P = S_{p实} / Max(S_p) \times 100\% = 89.55\%$,很好地发挥了Cache的功效。图4的



(a) 实验的原始图像(3992 × 3240 pixels)



(b) 局部GVF矢量图

图3 实验图像及局部矢量图

Block-time实验曲线说明在分块的初始阶段,由于分块产生的额外运算量增加计算时间,但随着分块大小的逐步变小,Cache的命中率增加,计算时间逐渐减少,当分块数达到222时,计算时间达到最优,从无分块时的377.457秒减少到191.7640秒,计算时间减少185.693秒。

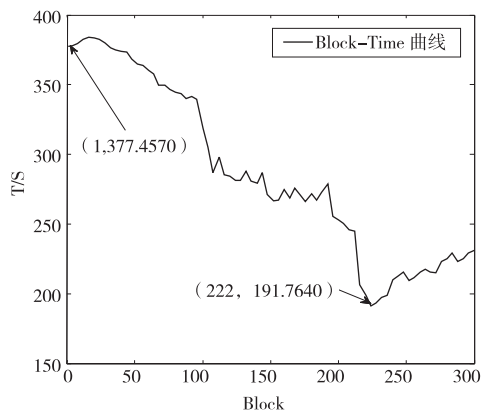


图 4 Block-time 实验曲线

5 结束语

针对 GVF Snake 模型的外力场在对千万像素的高清图像进行轮廓提取和图像分割时,计算速度较慢问题。提出使用分块的方法,提高了 Cache 的命中率,充分发挥 Cache 的效能,同时也有可能解决由于图像矩阵过大,导致一些计算机由于内存不足(内存溢出)而无法计算的问题;对原始图像矩阵进行分块计算方法也是的对其计算任务的平衡分解一种方法,有利于在此基础上研究其并行算法,进一步充分发挥现在主流多核计算机的性能。计算机仿真实验表明,对图像矩阵进行分块的

方法,能够大幅提高 GVF Snake 模型外力场对高清图像的计算速度。

参考文献:

- [1] 王文哲,唐克伦,牟宗魁.主动轮廓模型综述[J].机械设计与制造,2009(8):257-259.
- [2] Kass M, Witkin A, Terzopoulos D.Snake: Active Contour Model[J].Int.J.Computer Vision, 1987(1):321-331.
- [3] Xu C, Prince J L.Gradient Vector Flow: A New External Force for Snake [J].Comp Vis Patt Recog(CVPR), 1997,2(3):66-71.
- [4] Kim Shin-Hyoung,Alattar Ashraf,Jang Jong Whan.Snake Based Objects Tracking in Stereo Sequences with The Optimization of The Number of Snake Points [C]//Zou Guangyu.Proceeding of IEEE International Conference Image processing,USA, October 8-11,2006:193-196.
- [5] 王文哲,唐克伦,牟宗魁.分段迭代 B-Snake 模型[J].四川理工学院学报:自然科学版,2009,22(5):96-101.
- [6] 唐克伦.三坐标机与立体视觉的系统集成与信息融合的关键技术研究[D].重庆:重庆大学,2008.
- [7] 洪志全.计算机系统结构教程[M].北京:机械工业出版社,2010.
- [8] 张晨曦.计算机系统结构实践教程[M].北京:清华大学出版社,2010.

GVF Snake Model Based on Matrices Dividing

TANG Ke-lun¹, TANG Yong-liang¹, ZENG Wei², JIA Hai-yang¹, ZHANG Yang¹, LIU Yan¹

(1. School of Mechanical Engineering, Sichuan University of Science & Engineering, Zigong 643000, China;

2. Zigong K/D Carbon Co., Ltd, Zigong 643000, China)

Abstract: In order to promote the contour extracting velocity in a high-definition image through GVF snake models, the process in GVF iterative computation, the architecture and theory of computer cache system are analyzed, a new algorithm for GVF based on partitioned matrix is advanced, and the corresponding partition size and partition mode are determined. The simulation result indicates that the new algorithm would shorten the computing time significantly.

Key words: GVF snake model; high-definition image; computer cache system; partitioned matrix