基于多核的并行粒子滤波运动目标跟踪

王爱侠 李晶皎 王 青 王 骄
(东北大学信息学院 沈阳 110819)

摘 要 粒子滤波中大量的粒子计算使得算法的实时性较差。由于粒子滤波本身具有可并行化的特点,因此利用OpenMP多线程库派生出多个线程,将算法过程由单线程串行执行转变为多线程并行执行。用多核并行计算技术实现粒子滤波运动目标的跟踪。实验结果表明:基于多核的并行计算技术提高了粒子滤波算法的计算效率。

关键词 多核,并行计算,粒子滤波
中图法分类号 TP242 **文献标识码** A

Target Tracking Based on Multi-core Parallel Particle Filtering

WANG Ai-xia LI Jing-jiao WANG Qing WANG Jiao
(College of Information Science & Technology, Northeastern University, Shenyang 110819, China)

Abstract The particle filter's real-time capability is poor because of a large number of the particles' calculation of the algorithm. Because of the particle filter's characteristics of parallelism, we used OpenMP multithread implementation to derive multiple threads, in order to change the algorithm from the single thread serial implementation to multithread implementation in parallel. In the multiple processors platform, the parallel-computed technology and particle filter algorithm were used to implement the tracking of the moving object. The results show that the method can improve the algorithm's execution speed.

Keywords Multi-core, Parallel calculating, Particle filtering

基于粒子滤波(PF)的运动目标跟踪在视频监控、物体识别、人机界面等应用领域中有着广泛的应用。为了提高跟踪准确率和速度,已经提出各种改进方法,有基于粒子群优化的粒子滤波(PSOPF)[1]、基于人工鱼群算法的粒子滤波(AFSAPF)[2]、基于马尔科夫链和蒙特卡罗的粒子滤波(MCPF)[3]、基于均值变换的粒子滤波[4,5]、自适应粒子滤波[6]、基于扩展卡尔曼的粒子滤波(EKFPF)[3]、基于EM的混合高斯粒子滤波器[7]等方法[8-11]。这些方法都围绕着如何解决粒子退化导致跟踪准确率低的问题来研究,但是由于粒子滤波需要采样大量的粒子用于计算估计值,使得算法消耗大量的计算时间,影响了系统的实时性,这些改进方法中没有关注算法的实时性。通过观察发现,粒子滤波本身具有可并行化的特点,通过OpenMP多线程库,在多核计算机上可以利用并行计算技术实现粒子滤波算法,提高算法的实时性。

1 粒子滤波算法

粒子滤波算法是通过蒙特卡罗方法随机抽取一组带有权值的粒子来逼近后验概率密度分布,并序贯预测和更新状态。设非线性离散时间动态系统由式(1)、式(2)表示:

$$x_t=f(x_{t-1})+w_{t-1} \tag{1}$$

$$y_t=h(u_t,x_t)+v_t \tag{2}$$

式中,f 和 h 分别为系统状态转移函数和测量函数;x_t,y_t 分别为系统状态、观测向量;w_t,v_t 分别为系统噪声和观测噪声。

假设系统状态符合一阶马尔可夫过程:

$$p(x_k|x_{0,k-1})=p(x_k|x_{k-1}) \tag{3}$$

且观测值独立于给定的系统,则给定系统状态空间模型,就可以由式(4)迭代计算系统后验概率:

$$p(x_k|y_{1,k})=\frac{p(y_k|x_k)p(x_k|y_{1,k-1})}{p(y_k|y_{1,k-1})} \tag{4}$$

通常很难直接从后验概率密度进行抽样。粒子滤波算法引入一个容易抽样的已知概率密度分布函数 $q(x_k|y_k)$ 进行近似,以样本均值代替积分运算,从而获得状态最小方差的估计过程。每个粒子有自身的权值和数值。时刻 k 的权值递推公式为:

$$\begin{aligned}w_k&=\frac{p(y_{1,k}|x_{0,k})p(x_{0,k})}{q(x_k|x_{0,k-1},y_{1,k})q(x_{0,k-1}|y_{1,k})}\\&=w_{k-1}\frac{p(y_k|x_k)p(x_k|x_{k-1})}{q(x_k|x_{0,k-1},y_{1,k})}\end{aligned} \tag{5}$$

式中,q 为建议概率密度分布。

则后验概率密度可以近似表示为:

$$p(x_{0,k}|y_{1,k})\approx\sum_{i=1}^{N}w_k^i\delta(x_{0,k}-x_{0,k}^i) \tag{6}$$

式中,δ 为 delta 函数,$x_{0,k}^i$ 是 k 时刻从 $q(x_k|y_k)$ 上随机抽取的第 i 个粒子。

2 粒子滤波的并行处理

因为每个粒子都是相互独立的,各自的系统传播过程、系统观测过程以及粒子权值计算过程均与其他粒子无关,仅仅

到稿日期:2011-09-16 返修日期:2011-12-27 本文受国家自然科学基金项目(60970157)资助。
王爱侠(1974—),女,博士生,讲师,E-mail:wangaixia@ise.neu.edu.cn;李晶皎(1964—),女,教授,博士生导师。

在粒子权值归一化以及粒子重采样过程中需要利用所有粒子的权值。所以可以将并行化的思想引入到粒子滤波中，使用OpenMP多线程编程语言实现并行化的粒子滤波算法(PPF)，以达到减少算法执行时间、提高执行速度的目的。

粒子滤波共分为6个过程：粒子初始化、粒子状态预测、权值计算、跟踪估计、重采样和目标模型更新。这些过程之间具有数据继承性，因此从整体来看算法是串行的，不可并行化处理。但每个过程中粒子与粒子之间是独立的，没有数据依赖性，这为粒子滤波算法并行化提供了可能。粒子滤波串行执行和并行执行过程对比如图1所示，在粒子滤波算法的过程内使用了大量的采样粒子，串行算法中(见图1(a))粒子是一个接一个地进行相关操作，这种串行计算在粒子数量多的情况下会消耗大量的计算时间。假设在某阶段处理一个粒子需要时间为 c，那么处理 m 个粒子需要时间为 $c*m$。而采用多线程并行后(见图1(b))，虽然算法整体上看依旧是顺序执行，但是在粒子滤波每个过程中，粒子的操作已经采用了多线程的并行化处理方法，并非像左侧一个个地串行执行，而是根据分配的线程数(图中线程数为2)并行执行。假设在某阶段处理一个粒子需要的时间为 c，那么在 n 个线程并行执行的情况下处理 m 个粒子需要时间为 $c*m/n$，从而节省了算法的执行时间。图1为粒子滤波算法串行和并行执行过程的对比。

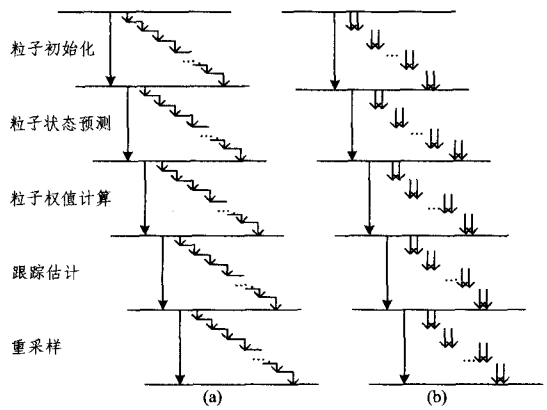

图1　粒子滤波算法串行和并行执行过程对比

(1)粒子初始化

粒子初始化的目的是在跟踪目标区域内随机分布一定数量的粒子，为每个粒子的结构体赋初值，包括粒子的位置、速度和权值。所有粒子相互之间并没有依赖关系，所以先赋值哪个粒子对初始化的结果以及之后的算法执行过程并没有影响，这种粒子独立性使得初始化过程具备了并行化的条件。在串行的初始化过程中，使用了循环操作来对每个粒子赋值。结合OpenMP编程语言，可以在串行的循环语句前添加相应的OpenMP编译指导语句实现并行化，形式如下：

```
#pragma omp parallel for num_threads(2)
  for(i=0;i < PARTICLE_NUM;i++)
{  …对一个粒子的结构变量进行初始化赋值…}
```

上面是使用OpenMP语言将粒子初始化过程并行化的方法。将线程数设置成2，可以将原来的串行执行变成两个线程同时并行执行，从而提高了执行效率。

(2)粒子状态预测

粒子状态预测主要是根据状态转移方程和上一时刻的粒子状态预测出此刻每个粒子的状态，即位置和速度。每个粒子的预测状态值只和上一时刻该粒子的状态有关，与其它粒子上一时刻和此刻的状态没有任何关系，即粒子与粒子之间是独立的。因此，粒子状态预测过程具备了串行执行转为并行执行的条件。并行化的粒子状态预测形式如下：

```
#pragma omp parallel for num_threads(2)
  for(i=0;i < PARTICLE_NUM;i++)
{…一个粒子通过状态转移方程预测状态… }
```

(3)粒子权值计算

粒子权值计算过程是以上一步粒子状态预测过程中得到的状态预测值为基础，结合此刻的观测值来计算出粒子区域直方图与目标模型直方图的Bhattacharyya系数，进而得到粒子权值。具体的做法是，每个粒子以上一步得到的预测状态的位置坐标为中心，做出与目标跟踪区域同样大小的粒子区域，然后计算粒子区域的颜色直方图，最后与目标模型直方图做相似性对比，求出Bhattacharyya系数。这个过程中每个粒子的操作都以自身在上一步得到的状态结果为基础，与其它粒子的状态无关。虽然每个粒子在计算Bhattacharyya时都会与目标模型直方图做对比，但是每个粒子对目标模型只是做读取操作，不存在数据竞争问题，因此可以做并行化处理。

通过以上分析可以看出，粒子滤波算法的前3步完全可以进行并行化处理。在串行的处理过程中，可以使用循环操作串行地对每个粒子进行操作。使用OpenMP语言将粒子处理过程并行化后，各处理器处理的粒子序列为：

处理器1　$x_1, x_1+p, x_1+2p, x_1+3p\cdots$；

处理器2　$x_2, x_2+p, x_2+2p, x_2+3p\cdots$；

处理器3　$x_3, x_3+p, x_3+2p, x_3+3p\cdots$；

⋮

⋮

处理器P　$x_p, x_{2p}, x_{3p}, x_{4p}$

这样使得在各个处理器上的粒子初始化是串行初始化序列的子序列，在数据上没有重复性，并且处理的粒子数和串行程序处理的粒子数一致。

(4)跟踪估计

粒子得到新的权值，通过权值归一化后的加权粒子集来计算跟踪估计值。在求跟踪估计值时，需要将每个粒子的位置乘以该粒子的归一化权值，然后再将所有粒子累加到一起得到跟踪估计值。这种操作可以用OpenMP中的归约编译语句实现并行化处理。并行化的跟踪估计过程形式如下：

```
#pragma omp parallel for reduction(+:sum) num_threads(2)
  for(i=0;i < PARTICLE_NUM;i++)
  {归一化累计加权粒子集进行估计}
```

(5)重采样

重采样过程是对产生退化问题的粒子做相应的替换更新操作，该过程会先判断每个粒子是否退化，只对退化的粒子做替换操作，对未退化的粒子不做任何操作。这个过程的粒子之间依然没有相关性，每个粒子自成一个体系，可能是退化粒子也可能不是退化粒子，完全由自身的权值大小和退化阈值决定，不影响其它粒子，所以重采样过程具备了并行化处理的条件。

并行化的重采样形式如下：

```
#pragma omp parallel for num_threads(2)
  for(i=0;i < PARTICLE_NUM;i++)
  {判断粒子是否退化，退化的粒子则被重采样}
```

3 测试与分析

3.1 测试环境

(1) 硬件环境：

CPU：Intel Core2 6300(双核) 1.86GHz

内存：2G

硬盘：160G

(2) 软件环境：

操作系统：Microsoft Windows XP Professional

开发环境：Microsoft Visual Studio 2008

辅助开发工具：OpenCV 2.0

intel parallel studio

(3) 视频文件：

IBM Research People Vision Outdoor Sequences(链接：http://www.research.ibm.com/peoplevision/performance-evaluation.html)中的 PetsD1TeC2.avi 的前 23 秒的视频段。所有测试选取的线程数为 2。

本节共分 3 次测试，目的是从算法整体角度比较串行和并行的执行时间。3 次测试选取的粒子数不同，分别是 100、1000 和 2000。每次测试以第 1 节中介绍的基本粒子滤波跟踪算法作为基础，分别执行串行算法和利用 OpenMP 语言改写的并行算法 10 次，将执行时间取平均值后进行比较。为了保证测试是在同一环境同一条件下进行的，每次测试都是对同一跟踪目标进行跟踪。选择的跟踪起始帧和终止帧相同，其都是起始帧为第 50 帧，终止帧为第 179 帧。跟踪目标的中心坐标相同，都为(79,231)。跟踪区域的大小相同，其都是长宽为(14,10)的矩形区域。测试时间从粒子初始化过程开始，包括整个粒子滤波过程，直到测试视频结束，其中并未包含视频图像的输出操作，测试时间为纯粒子滤波算法的执行时间。测试输出包括粒子数、起始终止帧数、跟踪目标中心、跟踪区域大小和执行时间。

图 2 中分别记录了不同粒子数下的跟踪效果。算法在 3 种粒子数下都较好地跟踪了运动目标，但从跟踪轨迹上可见，粒子数少的情况下，估计轨迹比较曲折，如图 2(a)所示的 100 个粒子中轨迹有很大的波动；在粒子数多的情况下，估计轨迹比较平滑，如图 2(c)所示的 2000 个粒子中，跟踪轨迹始终比较稳定，很少有波动较大的地方，近似一条直线，与实际中跟踪目标的运行轨迹基本一致。

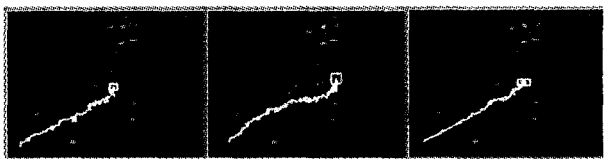

图 2 粒子数为 100、1000、2000 的跟踪效果

表 1 中分别记录了串行和并行算法在不同粒子数量情况下各执行 10 次的时间，然后计算出平均值，最后计算并行效率提高值。表 2 记录了粒子数为 100 的情况下，本文提出的并行粒子滤波算法和文献中各种算法执行时间的对比，每个实验执行 10 次，表中记录的是平均值。

表 1 并行和串行执行时间对比

	100 个粒子(测试 1)		1000 粒子(测试 2)		2000 粒子(测试 3)	
实验次数	串行时间	并行时间	串行时间	并行时间	串行时间	并行时间
1	2.322531	2.015210	13.666871	11.449055	25.161916	21.097425
2	2.304229	2.015590	13.652909	11.465244	25.131270	20.673801
3	2.318738	2.037170	13.710481	11.437541	25.130302	20.875400
4	2.296101	2.028666	13.673243	11.412015	25.133843	20.874914
5	2.320989	2.027016	13.708301	11.538984	25.133734	20.634475
6	2.320424	2.046817	13.672164	11.540720	25.137076	20.799571
7	2.298075	2.029663	13.708301	11.470101	25.133606	20.739002
8	2.326742	2.049481	13.685039	11.413307	25.135702	20.720399
9	2.294621	2.030412	13.708609	11.468578	25.133629	21.063819
10	2.324034	2.032834	13.680875	11.443216	25.142051	20.779521
平均值	2.312648	2.031286	13.686679	11.463876	25.137313	20.825833
提高率	12.17%(1.139)		16.24%(1.194)		17.15%(1.21)	

表 2 100 个粒子的条件下各种粒子滤波算法执行时间的对比

算法	pF	pPF	PSOPF	AFSAPF	MCPF	EKFPF
平均执行时间	2.3126	2.0313	4.4641	2.2361	3.6066	5.888

从表 1 中可以看出，算法取 2000 个粒子并行后效率提高 17.15%，取 1000 个粒子并行后效率提高 16.24%，取 100 个粒子并行后效率提高 7.84%。可见随着粒子数量的增加，速度提高率有了很大提升，原因是在使用 OpenMP 编译语句时，派生线程需要一定的执行时间，如果粒子数量少，并行化部分的执行时间就短，再加上派生线程耗费的时间，可能会使并行后的执行时间略小于串行执行时间，甚至多于串行执行时间。但是随着粒子数增多，并行部分的执行时间变长，其远远大于派生线程需要消耗的时间，并行处理后，整体的并行执行时间较串行执行时间就会有明显的提高。

从表 2 可以看出，在相同实验条件下，并行粒子滤波算法(PPF)执行时间更短，具有一定的速度优势。其他粒子滤波算法也可以按照本文提出的并行方法进行并行化，以提高整个算法的执行速度。

总之，通过实验中可以看出，并行粒子滤波算法相对于其他串行程序较大地降低了执行时间，提高了计算效率，而且粒子数量越大，并行效果越好，是一种有力提高粒子滤波执行效率的方法。

结束语 由于粒子滤波采用了蒙特卡罗方法，需要采样大量的粒子来近似估计目标跟踪值，大量粒子的计算消耗了系统的执行时间，使得粒子滤波的实时性较差。由于粒子滤波中的每个粒子相互之间没有数据依赖性，因此粒子滤波具备了并行化的特性。在基于颜色特征的粒子滤波跟踪算法的基础上，使用 OpenMP 多线程编程语言对粒子滤波过程进行

并行化处理，在基本粒子滤波的过程中添加了相应的OpenMP编译语句，实现了算法过程的并行化处理。测试结果表明，该方法能够有效提高基本粒子滤波的执行速度，对于文献中各种粒子滤波算法的提速有一定的借鉴性。

参考文献

[1] 方正，佟国峰，徐心和. 基于粒子群优化的粒子滤波定位方法[J]. 控制理论与应用，2008，25(3)：533-537

[2] 朱林户，李德芳，柳宏杰，等. 一种集群智能粒子滤波算法[J]. 西安电子科技大学学报：自然科学版，2008，35(3)：536-541

[3] 相威，汪立新，林孝焰. 几种改进的粒子滤波算法性能比较[J]. 计算机仿真，2009，26(4)：120-124

[4] 刘志明，韦巍. 基于均值变换的 Particle Filter 实时跟踪算法[J]. 模式识别与人工智能，2006，19(6)：825-830

[5] 石华伟，夏利民. 基于 Mean Shift 算法和粒子滤波器的人眼跟踪[J]. 计算机工程与应用，2006，19：26-28

[6] Bhandarkar S M，Zheng Wen-long. Face detection and tracking using a Boosted Adaptive Particle Filter[J]. Journal of Visual Communication and Image Representation，2009，20(1)：9-27

[7] 李静，陈兆乾，陈世福. EM-GMPF：一种基于 EM 的混合高斯粒子滤波器算法[J]. 计算机研究与发展，2005，42(7)：1210-1216

[8] Li Liu-bai，Ma Yan，Liang Yuan-yuan. Particle-Filter Tracking of Motion vector to locate objects and Pattern matching over particles Based on Mpeg2[J]. Advanced Materials Research，2011，225-226(1/2)：350-355

[9] Khan Zulfiqar Hasan，Gu Irene Yu-Hua，Backhouse A G. Robust Visual Object Tracking Using Multi-Mode Anisotropic Mean Shift and Particle Filters[J]. IEEE Transactions on Circuits and Systems for Video Ttechnology，2011，21(1)：74-87

[10] Yin Shi-min，Na Jin Hee，Choi Jin Young，et al. Hierarchical Kalman-particle filter with adaptation to motion changes for object tracking[J]. Computer Vision and Image Understanding，2011，115(6)：885-900

[11] Khan Z H，Gu Yu-hua，Backhouse Andrew G. A Robust Particle Filter-Based Method for Tracking Single Visual Object Through Complex Scenes Using Dynamical Object Shape and Appearance Similarity[J]. Journal of Signal Processing Sytems for Signal Image and Video Technology，2011，65(1)：63-79

(上接第 295 页)

从实验结果可以看出，本文算法的特征提取时间复杂度较高，主要原因是 NSDFB 所需的时间复杂度较高，同时在相同的参数设置下，双密度 Contourlet 变换方向子带细节较多，因此本文算法计算复杂度较高；文献[7]NSCT 变换特征提取时间复杂度较高的主要原因也是由于采用了 NSCT 变换。

结束语 本文提出一种新的双密度 Contourlet 变换，理论证明该变换在 $L_2(Z^2)$ 空间是一个框架算子，它可以直接应用于图像处理领域。与传统 Contourlet 变换相比，双密度 Contourlet 变换含有更为丰富的方向细节信息，具有较低的平移敏感性。实验分析表明，双密度 Contourlet 变换的方向子带系数分布具有非高斯性、高峰度和长拖尾等特点，可以采用广义高斯分布进行描述。基于纹理图像的检索实验表明，本文算法的检索精度高于文献[7]的 Contourlet 变换、NSCT 和 WBCT 算法，且检索精度比传统 Contourlet 变换算法提高了 5.3%。然而，本文算法仅提取了图像的纹理特征，因此利用双密度 Contourlet 变换提取其他底层特征(如形状和空间特征)并进行综合检索是本文进一步的研究方向。

参考文献

[1] Minh N D，Vetterli M. Wavelet-Based texture retrieval using generalized gaussian density and kullback-leibler distance[J]. IEEE Transactions on Image Processing，2002，11(2)：146-158

[2] Han Ju，Ma Kai-kuang. Rotation-invariant and scale-invariant gabor features for texture image retrieval[J]. Image and Vision Computing，2007，25(9)：1474-1481

[3] Quellec G，Lamard M，Cazuguel G，et al. Wavelet optimization for content-based image retrieval in medical databases[J]. Medical Image Analysis，2010，14(2)：227- 241

[4] 焦李成，张向荣，侯彪，等. 智能 SAR 图像处理与解译[M]. 北京：科学出版社，2008

[5] Do M N，Vetterli M. The contourlet transform：an efficient directional multiresolution image representation [J]. IEEE Transactions on Image Processing，2003，14(12)：2091-2106

[6] Cunha A L，Zhou Jiang-ping，Do M N. The Nonsubsampled contourlet transform：theory，design，and applications[J]. IEEE Trans Image Processing，2006，15(10)：3089-3101

[7] Rao C S，Kumar S S，Chatterji B N. Content based image retrieval using contourlet transform[J]. ICGST-GVIP Journal，2007，7(3)：9-15

[8] Qu Huai-jing，Peng Yu-hua，Wan Hong-lin，et al. Texture image retrieval based on contourlet transform and active perceptual similarity learning[C]// Proceedings of the 4th international conference on Advanced Data Mining and Applications. 2008：355-366

[9] 练秋生，李芹，孔令富，等. 融合圆对称轮廓波统计特征和 LBP 的纹理图像检索[J]. 计算机学报，2007，30(12)：2198-2204

[10] Selesnick I W. The double-density dual-tree DWT[J]. IEEE Trans. on Acoustics，Speech，and Signal Processing，2004，52(5)：1304-1314

[11] 尚赵伟，张明新，沈钧毅，等. 基于双密度小波变换的纹理图像检索[J]. 西安交通大学学报，2005，39(10)：1081-1084

[12] Eslami R，Radha H. Wavelet-based contourlet transform and its application to image coding [J]. IEEE International Conference on Image Processing，2004，5：3189-3192

[13] Moulin P，Liu J. Multiresolution Gray-Scale and Rotation Gaussian and complexity priors Invariant Texture Classification with Local Binary Patterns[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2002，24(7)：971-987

[14] Sun Jun-ding，Zhang Xi-min，Cui Jiang-tao，et al. Image Retrieval Based on Color Distribution Entropy[J]. Pattern Recognition Letters，2006，27(10)：1122-1126