文章编号:1001-506X(2010)02-0428-05

多核并行计算技术在景象匹配仿真中的应用

李 妮[1], 陈 铮[2], 龚光红[1], 彭晓源[1]
(1. 北京航空航天大学自动化科学与电气工程学院, 北京 100191;
2. 中国船舶系统工程部, 北京 100036)

摘 要: 随着计算机技术向多处理器以及多核的架构发展,以 OpenMP 和线程构建模块(thread building blocking, TBB)为代表的多核处理器并行计算平台的应用得到重视。对多核并行计算技术在建模仿真领域的应用进行了初步探讨,以景象匹配仿真计算为例,基于 OpenMP 和 TBB 并行计算平台构建了一个景象匹配评估平台,对不同的景象匹配算法效率进行评估。对多核并行计算技术及其在景象匹配仿真算法中的应用进行了详细介绍,并通过仿真实验验证多核并行计算技术能够大大提高景象匹配评估平台的运行效率,从而有效地支持对导弹真实景象匹配制导系统相关算法进行验证和研究。

关键词: 多核; 并行计算; 景象匹配; 建模仿真
中图分类号: TP 391.9 **文献标志码**: A

Application of multi-core parallel computing technology in scene matching simulation

LI Ni[1], CHEN Zheng[2], GONG Guang-hong[1], PENG Xiao-yuan[1]
(*1. School of Automation Science and Electrical Engineering, Beihang Univ., Beijing 100191, China;*
2. Systems Engineering Research Inst., China State Shipbuilding Corporation, Beijing 100036, China)

Abstract: Computer technology is developing towards multi-processor architecture and multi-core architecture. More and more focus is put on applications of multi-core parallel computing platform, such as OpenMB and thread building blocking (TBB). Application of multi-core parallel computing technology in modeling and simulation (M&S) area is concerned. A scene matching evaluation platform that helps to evaluate efficiency of different algorithms is implemented based on OpenMP and TBB. The multi-core parallel computing technology and its application in scene matching algorithms are introduced in detail. Simulation results shows that the multi-core parallel computing technology can greatly improve the simulation efficiency and effectively support verification and study of scene matching algorithms in real missile guidance systems.

Keywords: multi-core; parallel computing; scene matching; modeling and simulation

0 引 言

并行计算是指同时对多个任务或多条指令、或对多个数据项进行处理。当前计算机技术的发展从集中式机群到分布式 PC,再向多处理器以及多核的架构发展。因此,并行计算的发展方向也随着计算机技术的发展经历着相应的变化,特别是多核并行计算技术及其应用引起了越来越多的重视[1-2]。OpenMP 和线程构建模块(thread building blocking, TBB)是当前比较有代表性的支持多核处理器的并行计算平台。

在建模仿真领域,通过采用构建分布式交互仿真系统的并行计算方式以满足大规模仿真应用中各类仿真模型解算对计算资源的需求[3]。但在单节点仿真应用或者分布式体系结构下的成员节点中也可以利用多核并行计算技术充分调度计算资源,满足仿真模型解算的实时性要求。本文以景象匹配仿真计算为例,对多核并行计算技术在建模仿真领域的应用进行初步探讨。

景象匹配技术被广泛应用于飞行器的定位,如在导弹末制导过程中的精确定位等。景象匹配仿真是对真实导弹景象匹配过程的模拟,可对景象匹配的性能进行考察或验

收稿日期:2008-08-03; **修回日期**:2009-04-30。
作者简介:李妮(1980-),女,讲师,博士,主要研究方向为先进分布仿真、虚拟样机及仿真支撑平台技术。E-mail:lini@buaa.edu.cn

证[4]。景象匹配的性能是由多方面因素决定的,对性能影响最主要的因素是计算平台的解算效率和匹配算法的综合应用。

本文构建了一个景象匹配评估平台对景象匹配算法效率进行评估,并将 OpenMP 和 TBB 并行计算平台分别应用于景象匹配算法的实现。通过仿真实验验证多核并行计算技术能够大大提高景象匹配评估平台的运行效率,从而有效地支持对导弹真实景象匹配制导系统相关算法进行验证和研究。本文以下部分将对多核并行技术和景象匹配评估平台功能、结构进行介绍,对 OpenMP 和 TBB 在景象匹配仿真中的应用及仿真运行结果进行分析。

1　多核并行处理技术简述

Open MP 应用程序接口是针对共享内存多处理器体系结构的可移植并行编程模型,能够支持并行计算时对线程和变量的灵活设置和控制。对比于操作系统平台上的多线程编程的步骤,应用 OpenMP 的过程要更为简便[5]。

TBB 是 Intel 推广的支持多核处理器的 C++线程并行编程模型,它相对于 OpenMP 的最大优势就在于其面向对象特性的实现,而且具有支持复杂的并行模式、可扩展的线程嵌套并行等特点。但 TBB 应用灵活性的增加也使得用户应用的难度增大,需要基于 TBB 提供的并行算法模板类(如 parallel_for,parallel_reduce 等)编写程序,以支持复杂的并行模式[6-7]。

2　景象匹配评估平台的功能结构

景象匹配评估平台通过封装虚拟样机景象匹配常用算法,提供图形化的算法实施方案描述工具,实现实时图和基准图的预处理和匹配过程图像效果可视化,并集成了对算法实施方案在空间域和时间域上进行分析的功能[8]。

景象匹配评估平台的功能结构如图 1 所示,以下为主要功能模块。

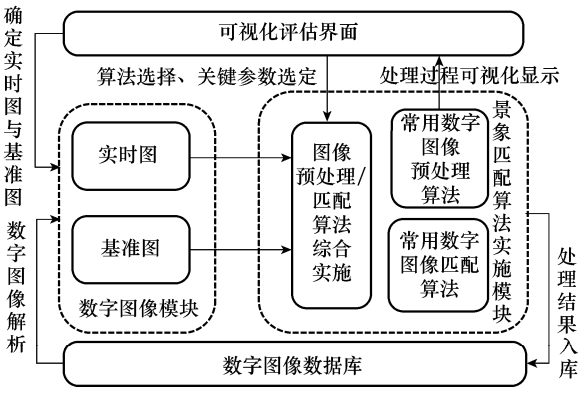

图 1　景象匹配评估平台的功能结构

(1) 景象匹配算法库。对导弹虚拟样机景象匹配的常见算法分别基于 OpenMP 和 TBB 进行封装。图像预处理算法的输入接口为被处理图像、算法参数,输出接口为经过处理的数字图像;图像匹配算法的输入接口为被匹配实时图、基准图、匹配参数,输出接口为匹配结果(匹配点的位置、实时图与基准图在匹配点的合成图像),存储格式为 C++源代码。

(2) 数字图像数据库。用于存储未经处理的实时图、基准图,以及经过处理的数字图像。图像存储格式为 8 位灰度 bmp 图片。

(3) 图像预处理与匹配算法实施模块。根据用户需求调用相应的数字图像、预处理/匹配算法、设置算法参数,建立算法实施方案并执行。该模块的输入为进行匹配的数字图像,输出为数字图像的当前处理结果以及最后的匹配结果。该模块还调用相应的辅助分析工具,如灰度直方图统计工具,算法执行时间统计工具等。

(4) 可视化评估界面。包含算法库、图像库浏览界面,提供图形化的算法实施方案创建界面,支持算法实施过程的图像效果显示以及匹配结果的显示。此外,还提供了灰度直方图统计、图像对比显示等功能。可视化评估界面如图 2 所示。

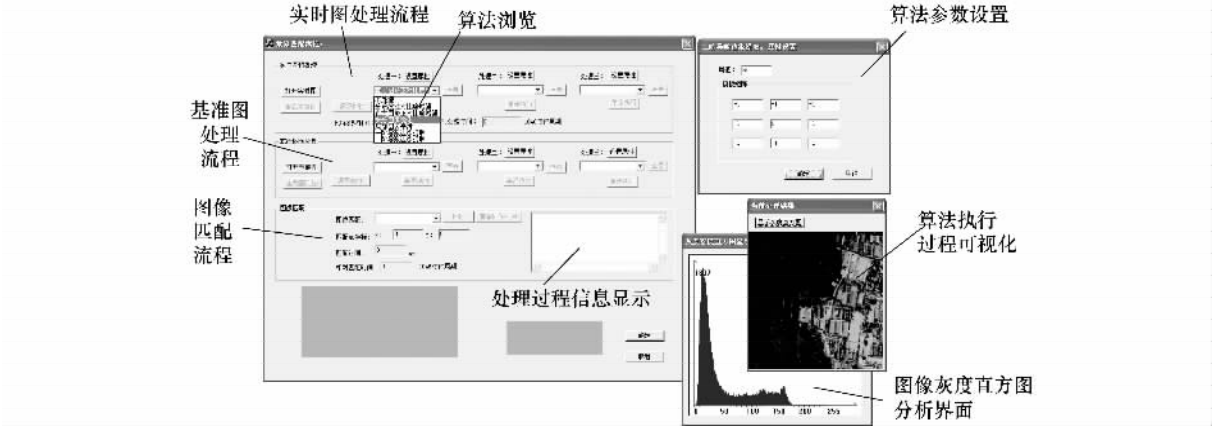

图 2　景象匹配评估平台的用户界面

3 多核并行处理技术在景象匹配算法中的应用

3.1 景象匹配算法

景象匹配中的匹配算法通常采用基于图像的灰度特征来进行匹配，可视化评估环境对匹配算法中的模板匹配算法提供了支持。匹配算法的基本原理是在基准图中提取子图并且与匹配模板进行相似度比较的过程，与匹配模板相似度最高的子图也就是匹配点所在的位置[9]。

本文采用式(3)表示相似度测度，其中 $T(m,n)$ 表示匹配模板中像素点 (m,n) 的灰度值，$S^{i,j}(m,n)$ 表示子图 $S^{i,j}$ 中像素点 (m,n) 的灰度值，也就是搜索图 S 中像素点 $(i+m,j+n)$ 的灰度值

$$D(i,j)=\frac{\sum_{m=1}^{M}\sum_{n=1}^{M}[S^{i,j}(m,n)\times T(m,n)]}{\sqrt{\sum_{m=1}^{M}\sum_{n=1}^{M}[S^{i,j}(m,n)]^2}\cdot\sqrt{\sum_{m=1}^{M}\sum_{n=1}^{M}[T(m,n)]^2}} \tag{1}$$

对于基准图上的任意一点 (i,j) 按照式(3)进行相似度计算，当 $D(i,j)$ 取得最大值时，其像素点就是最佳匹配点。由于模板匹配算法要对子图和模板图中每个像素点的灰度值都进行相似度计算，当子图和模板图的尺寸增大时像素点增多，匹配效率随着计算量的增大而降低。而各个像素点的计算是相互独立的，在多核计算机上运行时非常适合应用多核并行处理技术实现计算的并行化，大大提高景象匹配评估平台的运行效率。

3.2 多核并行处理技术的应用

3.2.1 OpenMP 在模板匹配算法中的应用

通过在模板匹配算法中增加 OpenMP 编译器指令实现算法的并行化。以下面计算最大相似度 dbMaxR 的 4 重循环的程序片段为例进行说明，省略了①、②、③处的代码。

```
dbMaxR=0.0;
 for(j=0;j<nImageHeight-nTemplateHeight+1;j++)
  {for(i=0;i<nImageWidth-nTemplateWidth+1;i++)
    {①;
      for(n=0;n<nTemplateHeight;n++)
      {for(m=0;m<nTemplateWidth;m++)
          {②;}
      }
          ③;}  }
```

由于程序中对各个像素点的计算相互独立，循环体②处的代码不存在循环依赖关系，因此只需要简单应用带 parallel 参数和 for 参数的编译器指令即可。需要注意的是对变量的定义，在并行化后 OpenMP 会把在 parallel 的范围以外声明的变量，当成是所有线程共用。因此 i,j,m 和 n 要在 parallel 的范围内进行声明，否则多个线程可能会同时对 i,n 和 m 进行自加的修改操作，导致循环的次数比预期的少。而 dbMaxR 要在 parallel 的范围以外声明，否则在②、③处计算得到的值不能正确反映所有线程对 dbMaxR 值的影响。并行化处理后的程序片段如下所示

```
    double dbMaxR=0.0;
#pragma omp parallel for num_threads(4)
    for (int j=0;j<nImageHeight-nTemplateHeight+1;
j++)
    {for(int i=0;i<nImageWidth-nTemplateWidth+1;
i++)
        {①;
            for(int n=0;n<nTemplateHeight;n++)
            {for(int m=0;m<nTemplateWidth;m++)
                {②;}
            }
                ③;}}
```

在预编译指令中可以指明计算所需的多线程数目，也可以调用 OpenMP 运行时例程的接口函数 omp_set_num_threads(4)设置 4 个线程。在不指定线程数目的缺省情况下，对于双核的计算机，OpenMP 会自动分配 2 个线程完成计算任务。

3.2.2 TBB 在模板匹配算法中的应用

应用 TBB 要根据需要选择不同的并行计算模板，并把需要并行处理的程序改写成一个符合 TBB 并行计算模板调用规范的类。parallel_for，parallel_reduce 是经常应用的两个并行算法模板类，其中 parallel_for 模板类会把一个循环划分为多个块，然后在不同的线程中并行处理，因此应用时要保证不能存在循环依赖；当存在循环依赖关系时，parallel_reduce模板类提供构造函数分离机制和重载 join() 函数的方法进行循环依赖处理。

同样以计算最大相似度 dbMaxR 的 4 重循环的程序片段为例进行说明。parallel_for 模板类适用于不存在循环依赖和无返回值的程序并行化处理。而在③处的代码是对不同位置计算处理的相似度进行比较，需要找到最大相似度并返回相应的像素点位置。因此，采用 parallel_reduce 模板类实现并行处理，具体包含如图 3 所示的步骤。

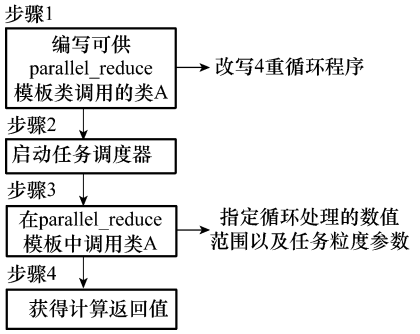

图 3 应用 TBB 步骤

首先编写可被 parallel_reduce 模板调用的类 A,类 A 必须包括接口 operator,join 以及 2 个构造函数。其中:

(1) 接口 void operator()(const blocked_range<size_t>& r)中是并行处理的主体程序,要基于以上 4 重循环的代码进行改写。将最外层循环的参数修改成 TBB 定义的 blocked_range 模板类,使之能够支持循环体内任务的并行划分。改写后的代码如下所示

```
void operator()( const blocked_range<size_t>& r )
    {for( size_t j=r.begin();j! =r.end();j++)
     {for(int i=0;i<nImageWidth-nTemplateWidth+1;
i++)
      {①;
          for(int n=0;n<nTemplateHeight;n++)
          {for(int m=0;m<nTemplateWidth;m++)
              {②;}
          }
              ③;}  }   }
```

需要注意的是循环体内①、②、③处的代码要尽量使用局部变量或者类 A 的成员变量,而不能使用全局变量,否则不仅不能够加快运行速度,反而会造成运算时间的增倍。

(2) 接口 void join(const A& y)能将多个线程中的运行结果进行合并,这里可用于获取多个线程中计算得到的最大相似度值和对应的像素位置。

(3) parallel_reduce 模板提供了 2 个构造函数,可以从整个任务空间中分离并构建子任务,以支持 join 接口中的合并操作。

TBB 定义了任务的概念,在步骤 2 中由任务调度器对象 task_scheduler_init 实现多任务的分配和并行计算,支持对多线程的划分。在步骤 3 中应用 parallel_reduce 模板类调用类 A,并在模板类参数中指定最外层循环处理的数值范围以及任务粒度参数。任务粒度参数决定了任务划分的粒度,如果粒度太大,不能充分提高运行效率;如果粒度太小,过度的并行化任务分配造成的开销反而降低了运行效率。可以使用 TBB 提供的自动分配函数 auto_partitioner()帮助用户设置合适的任务粒度参数。

4　仿真运行实例及结果分析

下面对不采用多核并行处理技术和分别采用 OpenMP 以及 TBB 支持并行计算的仿真运行实例和运行结果进行对比分析[10]。

采用两组不同大小的基准图和实时图,实时图所包含的景象与基准图中的景象采集角度和比例相同。第 1 组基准图为 256×256 像素的 8 位灰度图片,实时图为 140×140 像素的 8 位灰度图片;第 2 组基准图为 512×512 像素的 8 位灰度图片,实时图为 278×278 像素的 8 位灰度图片。并且设置一组 OpenMP 和 TBB 并行处理的多线程数目,分别为 1,2,4,8,16,32。仿真运行所用的机器配置为 Intel 酷锐双核 1.66 GHz CPU,512 MB 内存。在仿真过程中选择相同的预处理算法和参数设置,运行结果如表 1 和表 2 所示。

表 1　第 1 组图片匹配运行时间　　ms

	1	2	4	8	16	32
不使用多核并行				11 111.95		
采用 OpenMP	10 185.07	5 131.69	5 101.69	5 173.37	5 138.22	5 139.13
采用 TBB	9 644.68	4 910.76	4 917.39	5 106.95	5 024.64	5 073.72

表 2　第 2 组图片匹配运行时间　　ms

	1	2	4	8	16	32
不使用多核并行				150 804.82		
采用 OpenMP	157 783.52	79 547.21	79 064.38	79 237.35	79 132.11	79 144.38
采用 TBB	150 093.15	76 088.08	75 899.6	80 566.92	80 873.10	81 297.88

两组图片匹配过程运行所需时间如图 4 所示。

由此可见,无论是采用 OpenMP 还是 TBB 支持多核并行处理,都能够使得运行时间缩短 50%左右。并行线程的增加能够在一定范围内提高运行效率,但随着线程数目增多,多线程所需的开销增大到一定程度后反而会降低运行效率。由于本实例是在双核的机器上运行,本身硬件条件的限制也使得运行效率只能提高 1 倍左右。

通过操作系统任务管理器提供的功能,在执行过程中可以清楚地看到 CPU 资源的使用情况。如图 5 所示,图中的曲线表示 CPU 使用记录:图 5(a)是不采用并行处理的 CPU 使用记录,在匹配过程中两个处理核心的资源调度不够完全,匹配过程中的 CPU 使用率在 50%左右;图 5(b)是采用并行处理的 CPU 使用记录,此时对两个处理核心资源都进行了充分的调度,匹配过程中的 CPU 使用率达到 100%,因而才能够大大提高执行效率。

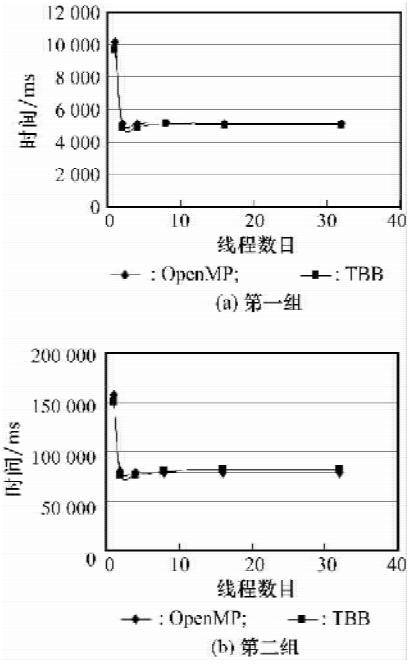

图 4　运行时间曲线

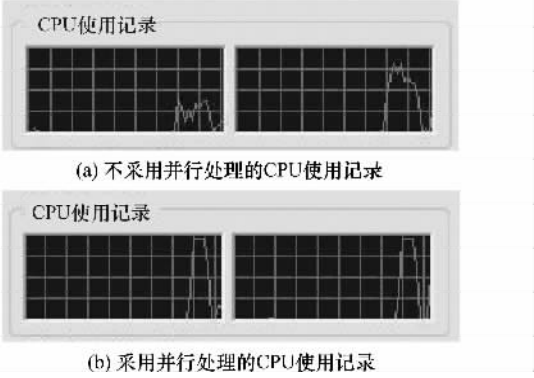

图 5　CPU 使用记录对比

5　结　论

多核架构计算机及多核并行计算技术的快速发展为建模仿真应用中的多核并行计算打下了基础，能够更加充分地调度 CPU 计算资源，提高仿真计算的效率。本文基于多核并行计算技术构建了景象匹配评估平台，使得景象匹配计算效率提高了 1 倍。但同时我们也看到无论是 OpenMP 还是 TBB 主要侧重于对多重循环计算的并行支持，要适用于建模仿真领域所有仿真模型的应用中还需要进一步的研究工作。

参考文献：

[1] 张林波，迟学斌，莫则尧，等. 并行计算导论[M]. 北京：清华大学出版社，2006：1－59.

[2] Gorder P F, Multicore processors for science and engineering[J]. *Computing in Science & Engineering*，2007，9(2)：3－7.

[3] 龚光红，王行仁，彭晓源，等. 先进分布仿真技术的发展与应用[J]. 系统仿真学报，2004，16(2)：222－230.

[4] 郭勤. 景象匹配技术发展概述[J]. 红外与激光工程，2007，36(增刊)：57－61.

[5] OpenMP application program interface. http：//www. openmp. org/mp-documents/spec30. pdf，2008.

[6] Intel threading building blocks tutorial. http：//www. threading-buildingblocks. org/documentation. php，2007.

[7] 胡斌，袁道华. TBB 多核编程及其混合编程模型的研究[J]. 计算机技术与发展，2009，19(2)：99－101.

[8] 陈铮. 虚拟样机可视化关键技术研究及实现[D]. 北京航空航天大学，2006.

[9] 杨枝灵，王开. 数字图像获取、处理及实践应用[M]. 北京：人民邮电出版社，2003：97－577.

[10] Kayi A, Yao Y, El-Ghazawi T, et al. Experimental evaluation of emerging multi-core architectures[C]// *IEEE International Parallel and Distributed Processing Symposium*，2007：1－6.