协同分布式图形硬件的混合并行体绘制

曹 轶[1)] 莫则尧[2)] 王弘玺[2)] 袁 斌[2)]

[1)](北京应用物理与计算数学研究所,计算物理实验室, 北京 100088)

[2)](北京应用物理与计算数学研究所,高性能计算中心, 北京 100088)

摘 要 由于一般的共享存储并行机缺乏图形硬件,其上产生的 3 维科学计算数据,无法采用硬件加速的并行体绘制来就地进行数据可视化。为此基于本地并行机和分布式图形工作站,给出了一种混合并行绘制模型。该模型的工作原理是先将源数据存留在并行机,然后通过并行机的多处理器发布远程绘制命令流,进而通过操控工作站的图形硬件完成绘制;后期图像合成在并行机上执行,以发挥共享存储通信优势。通过负载平衡优化,并行绘制流水线有效实现了绘制、合成与显示的重叠。实验结果显示,该方法能以 1 024×1 024 图像分辨率,交互绘制并行机上的大规模数据场。

关键词 并行体绘制 分布式绘制 远程可视化 3 维纹理硬件

中图法分类号:TP391.41 TP338.8 **文献标识码**:A **文章编号**:1006-8961(2008)07-1379-06

Hybrid Parallel Volume Rendering with Distributed Graphics Hardware

CAO Yi[1)], MO Ze-Yao[2)], WANG Hong-Kun[2)], YUAN Bin[2)]

[1)](*National Key Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, Beijing* 100088)

[2)](*High Performance Computing Center, Institute of Applied Physics and Computational Mathematics, Beijing* 100088)

Abstract For lack of graphics module, general shared memory Multiprocessors can not be used to visualize the three-dimensional scientific computational datasets generated on themselves using hardware-accelerated volume rendering techniques. An algorithm of the hybrid parallel rendering based on multiprocessors and distributed graphics workstations is presented. The datasets are processed on both local multiprocessors and distributed graphics workstations. The following image composition is performed by multiprocessors to utilize efficiently communicating capabilities. Through load balancing optimization, parallel rendering pipeline can overlap rendering, compositing and display process. Experimental results show that the interactive rendering abilities can be achieved when handling large datasets resided in multiprocessors with 1 024×1 024 image resolution.

Keywords parallel volume rendering, distributed rendering, remote visualization, 3D texture hardware

1 引 言

基于多处理器多图形硬件并行环境,采用对象空间划分任务方法,3 维纹理体绘制算法[1]可以并行处理大规模体数据。这种并行绘制算法同时适于分布式和共享存储环境,可以解决交互绘制需求。在笔者的工作中,并行体绘制是研究 3 维科学计算数据的有力手段,这些数据很多产生于运行在共享存储并行机的数值程序,仅其单时刻、单物理量、单精度的数据量就达数百兆字节。但由于并行机缺乏图形硬件,因而体绘制只能脱离并行机,使用图形终端或机群完成处理,这意味着源数据要经过网络再分配。由于很多数据属于临时实验结果,大量复制会带来内存和磁盘存

基金项目:国家杰出青年科学基金项目(60425205);国家重点基础研究发展计划(973)项目(2005CB321702);国家自然科学基金项目(605333020);中国工程物理研究院科学技术基金项目(20040659)

收稿日期:2006-10-19;**改回日期**:2007-01-05

第一作者简介:曹 轶(1975~),男。助理研究员。2006 年于中国工程物理研究院获计算数学专业硕士学位。主要研究领域为科学计算可视化、并行可视化。E-mail:cao_yi@iapcm.ac.cn

储压力,也考验了科研者的耐心,为此科研者希望利用并行计算环境,直接可视化计算数据,因而针对并行机的体绘制仍具有重要的研究意义。

绘制命令流方式可实现远程绘制,其数据加载和绘制位于网络互联的不同机器,且数据无须迁移。X-windows[2]最早提供了 2 维图形绘制命令流;GLX (the OpenGL extension to the X window system)[3]扩展了 X 协议对 OpenGL 的支持。WireGL[4]、Chromium[5]和 AnyGL[6]支持并行分布式绘制。Stegmaier 等人使用 GLX 实现了 PC 机间的远程绘制[7,8]。Ma 等人利用配置了多图形硬件的并行机,在远程终端完成了实变数据场的绘制[9]。金哲凡等人在分布式 PC 上研究开发了针对几何模型的并行绘制系统[10]。

但以前的工作多是针对图形硬件配置完善的机器而开展的研究,对实际科研需求考虑较少,且忽略了缺乏图形硬件的并行机资源。虽然本文的并行机缺乏图形硬件,但却配有很多具备体绘制能力的图形工作站。为此本文给出了一种混合并行绘制模型,以便用并行机的多处理器来发布远程绘制命令,进而操控工作站的图形硬件协同完成绘制;而图像合成则在具有通信优势的并行机上执行,并通过并行流水线和负载平衡优化改进了绘制性能。实验结果表明,该绘制模式能以 1 024 × 1 024 的图像分辨率,交互绘制并行机上的大规模数据场。

2 GLX 与 OpenGL 远程绘制

基于 GLX 的 OpenGL 远程绘制的特点是:源数据存留在本地,而经网络发给绘制端的是简短的绘制命令。其原理描述如图 1 所示:数据加载端负责建立 GLX Context,并设置 OpenGL 为间接绘制模式。绘制时则使用 GLX 将 OpenGL 命令打包成 X 协议命令流,定向发往远程绘制端的 X 服务器;然后 X 服务器将绘制命令解码,交给本地 GLX 进程,并使用绘制端的图形硬件完成绘制;绘制图像需要通过 X 远程模式显示。本文之所以采用 GLX 产生和解释绘制命令流,是因为使用的并行机与图形工作站都基于 X-windows 环境,并支持 OpenGL1.2 版本和 GLX 协议,因此并行机能获得工作站的 3 维纹理绘制能力。另外,GLX 可以适应原程序的 OpenGL 代码,无须调整程序接口和结构。虽然 WireGL 等也支持远程并行绘制,但在代码互用度和性能方面不易满足实际应用需要。

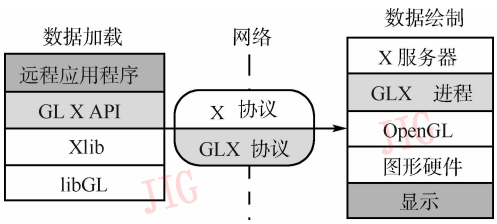

图 1 基于 GLX 的 OpenGL 间接绘制模式

Fig. 1 The GLX indirect OpenGL rendering mode

3 协同分布式图形硬件的混合绘制

混合并行绘制过程需要使用 $n+1$ 个并行机处理器和 $n+1$ 个图形工作站,其中 n 对处理器和工作站负责具体绘制。如图 2 所示,处理器 0 为 Master 节点,负责分配任务,并收集结果;显示端工作站负责与用户的交互和显示。本文使用绘制和交互两种 X 服务器程序,这与文献[7]类似,不同的是二者都需要 GLX 支持。其中绘制服务器驻留于第 n 个图形终端工作站上,用于接收来自并行机处理器的绘制命令,以调度本地图形硬件执行绘制和图像读取;交互服务器驻留在显示端工作站,负责将用户的指令提交给并行机,并反向接收图像,予以显示。

混合绘制过程描述如下:启动绘制后,处理器 0 连接交互服务器,先建立远程显示窗口;其余 n 个处理器连接对应工作站的绘制服务器,以创建远程绘制环境。绘制阶段,处理器 0 收到交互服务器的 X11 Events,首先得到视角等绘制参数,同时广播给其他处理器;然后 n 个处理器根据绘制参数发布远程绘制命令流,诸多绘制服务器在接到绘制命令后,随即利用工作站图形硬件完成绘制;接着并行机多处理器读取远程图像,并存入并行机本地内存;当 n 个处理器同步后,即执行图像合成,处理器 0 收集结果,并发送远程显示命令;最后交互服务器在接收图像后则刷新帧缓存,最终予以显示。

混合绘制引入 3 次同步动作:①多处理器读取远程图像,此步要通过 glWaitGL 确认远程绘制执行完毕,以保证结果正确。glWaitGL 比 glFinish 的绘制效率更高,因其避免了网络间的往复确认;②其余两次是 MPI_Barrier 同步,发生在多处理器并行合成图像前后,这是 Binary-swap 合成算法[11]特点决定的。下面给出混合绘制算法概述:

For(i = 0, 1, …, n)处理器 P_i并行执行 {

(1)并行初始化

①处理器 P_0联接交互服务器,建立远程显示窗口;

②处理器 P_i(i = 1, …, n)连接第 i 个绘制服务器(i = 1, …, n),建立远程绘制环境,内容如下:

XopenDisplay(“第 i 个绘制服务器:0”);建立和绑定 GLX Context 和 GLX Pbuffer;加载子数据块到主存;调用 glTexImage3D,加载数据到第 i 个绘制服务器的纹理内存;

(2)并行绘制

①处理器 P_0 发送绘制参数至第 i 个处理器 P_i(i = 1, …, n);

②处理器 P_i(i = 1, …, n)接收绘制参数,同时利用第 i 个绘制服务器的图形硬件进行体绘制,并将结果存入 Pbuffer;

③glXWaitGL 绘制同步,处理器 P_i(i = 1, …, n)读取 Pbuffer 中的绘制图像;

④MPI_Barrier 同步,执行并行图像合成,将结果发给处理器 P_0;

⑤MPI_Barrier 同步,处理器 P_0从处理器 P_i(i = 1, …, n)收集图像,使用交互服务器远程显示图像;}

3.1　XOpenDisplay 与 MPI 进程

每个并行机处理器与对应的图形工作站之间都存在远程连接,它独立维系一个 GLX 远程绘制环境。并行机处理器上的 MPI(message passing interface)进程,通过调用 XopenDisplay(〈第 i 个绘制服务器〉:〈0〉)执行与工作站上的绘制服务器的连接,冒号(:)代表用 TCP(transmission control protocol)方式连接。为了连接成功,还需事先启动各工作站的 X-Display,赋予 MPI 进程的访问权限,这可以通过 xauth 或 xhost 命令在工作站端设置。XOpenDisplay 成功执行后,会返回绘制服务器的 Display 指针,MPI 进程使用该指针建立 GLX Context,用于向绘制服务器发布远程绘制命令。MPI 进程退出时,则需用 XcloseDisplay 断开与绘制服务器的连接,以释放占用的工作站图形资源。

3.2　远程 OffScreen 绘制

在工作站产生的临时图像,由于需送回并行机进行图像合成,没有显示需要,因此可采用 OffScreen 绘制模式。它是基于 PBuffer(Pixel Buffer)[12] 进行体绘制,PBuffer 是 OpenGL 的附加缓冲区,其支持 SGI(silicon graphics, inc)硬件加速的 OffScreen 绘制。相对于使用 frameBuffer 绘制模式,Pbuffer 的优点是:无须建立绘制窗口,绘制可全硬件加速,支持高像素质量和图像分辨率。但建立 PBuffer 需要 GLX_SGIX_pbuffers 和 GLX_SGIX_fbconfig 两个 OpenGL 扩展函数的支持,另外,还要使用 glXChooseFBConfigSGIX 设置 Pbuffer 属性,其包括像素精度和 PBuffer 大小等参数。GLXMakeCurrent 用来绑定 GLX Context 和 PBuffer,此操作可使得本地进程产生的绘制指令都会被送往远程图形工作站的 Pbuffer 进行绘制。当该进程结束时,Pbuffer 资源的释放顺序为,首先解绑 GLX Context 和 Pbuffer,然后分别释放。

3.3　硬件加速体绘制的影响因素

远程绘制虽可避免数据经网络的迁移,但采用 3 维纹理的体绘制算法,还需从并行机将纹理数据下载到图形工作站的纹理内存,如果体数据较大,就会受网络延迟影响。但该迁移相比数据迁移,仍具有以下几点优势:①纹理数据经过量化处理,比单、双精度数据的容量缩减了;②下载的纹理数据只存储于工作站图形硬件内存,不占用其他资源;③对单时刻数据,下载过程只在初始化执行一次,因而不影响交互绘制;④对于时变数据场,通过并行迁移模式和流水线技术,迁移代价可在一定并行度后被降解。因此纹理硬件加速的体绘制仍适用于本文的混合绘制模式。

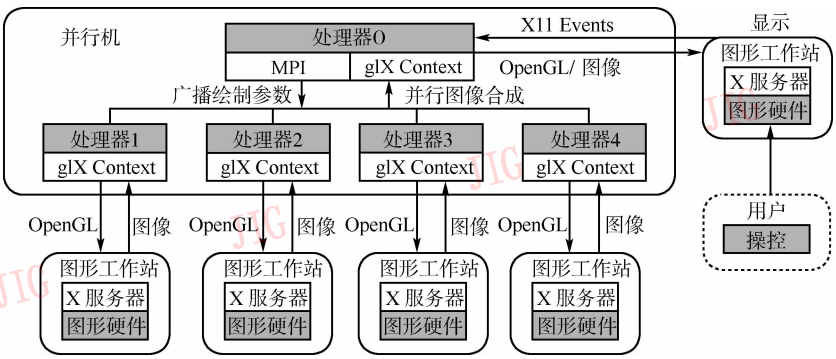

图 2　混合并行绘制模式的示意图

Fig. 2　The hybrid parallel rendering model

4 混合并行绘制模式的改进

4.1 基于并行绘制流水线的加速

混合并行绘制遵循 Sort-Last 并行绘制流程，要顺序完成绘制与图像读取、图像合成和显示 3 个任务模块。它们分别由不同的硬件负责，可利用并行流水线有效重叠这 3 个模块，以更充分利用并行机和工作站图形硬件资源。并行流水线需要并行机提供 $2\times n$ 个处理器，分成 n 组，每组 2 个处理器分别负责绘制和图像合成。设远程绘制时间为 T_r，读图像时间为 T_{br}，图像合成时间为 T_c，显示时间为 T_d，并行机的绘制处理器向合成处理器发送图像的时间为 T_1^S（上角 S 代表 send），显示处理器收集图像的时间为 T_2^S，则流水线运行时间可近似表示为 $\max\{T_r+T_{br}+T_1^S, T_c+T_2^S, T_d\}$，而非流水的运行时间为 $T_r+T_{br}+T_c+T_2^S+T_d$。其中之所以未将绘制与读图像分为两个共享 GLX Context 的独立模块，是因为 GLX 无法提供绘制状态跟踪机制，使得读图像进程很难判断正确的运行时机。

从表 1 可见，对并行流水线影响最大的是模块工作负载的不均衡。工作负载比重的计算如下：$W=L_1/L$，其中 L_1 为模块计算量，L 为总计算量，且绘制与读图像负载被分别列出。并行机处理器个数达到 16 时，模块负载差异甚至接近 7 倍，这严重阻塞了流水线的流动，需进行负载平衡优化。

表 1 流水线各模块负载的并行扩展性

Tab. 1 Scalabilities of pipeline's different workload

项目组成	不同并行机处理器个数 M 时的负载比率(%)			
	2	4	8	16
绘制	50.4	33.7	14.5	8.3
读图像	30.2	39.2	50.8	54.4
合成	2.8	5.4	7.5	8.7
显示	16.6	21.7	27.2	28.5

4.2 并行流水线的负载平衡优化

4.2.1 读图像与图像合成的负载优化

读取图像的速度取决于远程命令 glReadPixels 的执行时间，它包括了访问图形硬件和图像的网络传输时间。针对 SGI 图形硬件的最快数据读取类型为 GL_UNSIGNED_BYTE，像素排列格式为 GL_RGBA，否则 OpenGL 还需要对图像数据进行类型转换和格式重排。例如：如果以 GL_RGB 和 GL_ALPHA格式代替 GL_RGBA 格式，则针对本地硬件的读取速度就会降低一个量级。但影响更大的却来自网络传输，由于每个工作站上需读取的图像都与最终的图像大小（$N\times N$）相当，其数据传输量固定为 N^2，因此增加分布式工作站数量，也不会加速该过程，具体见图 3 中未优化的 RB 项（RB 为读取图像时间）。图像合成采用的 Binary-Swap 算法的时间复杂度为 $N^2\times(1-1/M)\times T_{over}$，$M$ 为并行机处理器个数，T_{over} 为图像合成“over”的计算时间。可见随着 M 的增加，图像合成时间也趋向恒定。考虑并行算法本身的处理开销，图像合成时间还会随着 M 递增，见图 3 中未优化的 CP 项（CP 为并行合成时间）。为减小 RB 和 CP 中的通信负载，可以使算法只对图像中包含有效图像的矩形区域进行标记，这样进行图像读取和合成时，就可避免矩形区域以外的图像数据参与计算，以此减轻通信量。优化结果如图 3 所示，当 M 为 16 时，读图像负载最多可以减少 6 倍，并行合成负载可减少 30%。

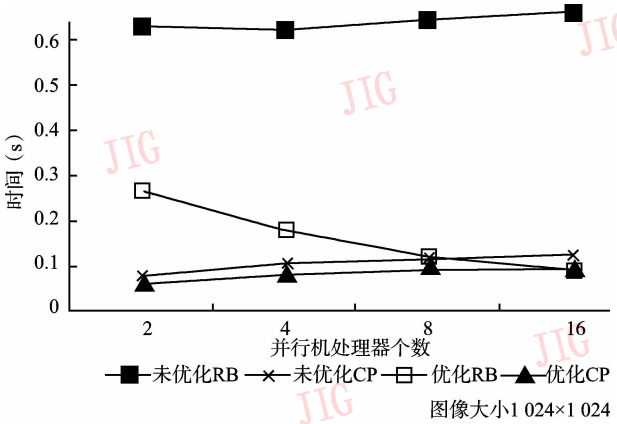

图 3 读图像与图像合成的负载优化

Fig. 3 The workload optimization of RB and CP

4.2.2 使用图像压缩技术加速远程图像显示

由于 GLX 远程显示只受图像大小和网络因素的影响，因此数据压缩是提升其速度的重要途径。文献[7]在 GLX 协议层上使用 VNC 软件层完成图像压缩；文献[8]、[9]采用多种压缩算法来解决图像传输瓶颈。而远程显示则通常要求压缩算法既拥有好的压缩比，又能快速进行压缩、解压缩操作。由于 LZO（Lempel-Ziv-Oberhumer）算法满足该要求，且属于无损压缩，因而在本文中被采用，但由于 GLX 本身不支持数据压缩，因此需改造图 2 中的远程显示模式。改进方法是：在显示端建立客户程序，用以取代原来的交互服务器，客户程序与处理器 0 通过中间件方式进行数据通信。数据压缩在处理器 0 上

进行,然后采用 Push-Push 的事件模式将压缩数据送往客户显示程序。客户程序收到压缩数据后首先完成数据解压,然后在本地使用图形硬件显示图像,而用户操控信息则由命名服务方式送回处理器 0。这里采用事件模式和命名服务可以提高通信效率和避免通信等待延迟。表 2 展示了不同图像分辨率下各显示部分的优化后时间与压缩比,表 3 比较了不同远程显示模式的性能差异。

表 2　显示优化的详细计算负载

Tab. 2　Detail workload of optimized display mode

项目组成	不同图像尺寸时的计算负载			
	128^2	256^2	512^2	$1\,024^2$
压缩时间 (s)	0.000 8	0.002 9	0.012 9	0.053 3
传输时间 (s)	0.000 9	0.002 3	0.006 9	0.023 3
解压缩时间 (s)	0.000 3	0.001 2	0.005	0.026 8
显示时间 (s)	0.010 4	0.010 4	0.010 4	0.010 4
压缩比率 (%)	85.57	89.59	90.96	91.88

表 3　两种显示模式的显示性能比较

Tab. 3　Performance comparison of two display mode

显示模式	不同图像尺寸时的显示速度 (fps)			
	128^2	256^2	512^2	$1\,024^2$
Remote GLX	96.0	32.0	12.1	3.0
Remote Compression	80.6	59.5	28.4	8.8

5　实验与分析

数值实验是在 1 台 SGI 共享存储并行机和 16 台 SGI 图形工作站上进行的,实验环境为基于 100Mbps TCP/Ethernet 的局域网环境,数值模拟数据 p2_3d_20 的网格规模为 221 × 602 × 221,填充后规模为 256 × 1 024 × 256,绘制图像分辨率为 1 024 × 1 024。性能测试是所有机器均空载下进行的,并行机多处理器间的通信延迟为 0.021μs,处理器间传输 1 024 × 1 024 分辨率的 4MB 图像数据大致需要 0.04s,体绘制结果见图 4。

未改进的混合并行绘制,其各计算部分随并行规模变化的构成见图 5,图中显示,纹理数据下载和体绘制计算两部分的时间开销可明显通过并行手段大幅降低。改进后的混合并行绘制的详细性能见表 4,表中的时间数据采用 4.2 节中的负载优化方法得到,绘制速度采用并行流水线技术测得。由表 4 可以发现,改进后的混合绘制,其读图像、图像合成和显示的负载时间都得到了控制,并行绘制性能有了显著的提高。

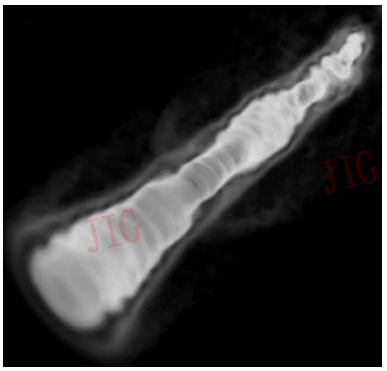

图 4　3 维激光等离子体交互作用数值模拟 p2_3d_20 数据

Fig. 4　3D numerical simulation of laser plasma interaction p2_3d_20 dataset

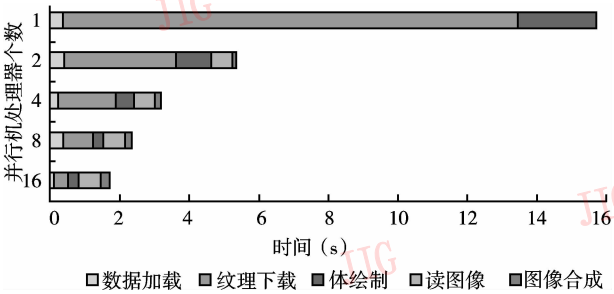

图 5　未改进的混合并行绘制各计算部分的时间开销

Fig. 5　Unimproved hybrid rendering performances

表 4　改进后混合绘制的详细性能列表

Tab. 4　Improved hybrid rendering performances

项目组成	不同并行规模下的绘制性能				
	1	2	4	8	16
绘制时间 (s)	2.256 9	1.009 8	0.518 1	0.176 9	0.097 6
读图像时间 (s)	0	0.241 6	0.158 4	0.098 8	0.068 9
合成时间 (s)	0	0.042 4	0.058 1	0.069 1	0.070 5
显示时间 (s)	0.01	0.113 8	0.113 8	0.113 8	0.113 8
绘制速度 (fps)	0.441	0.787	1.449	3.450	5.870

图 6 列出了逐级累积优化后的绘制性能,优化次序依次为读图像与图像合成(RB + CP)优化,以及在此基础上的并行流水线优化和远程显示(DP)优化。远程显示优化代表最终的优化结果,并与理想的优化值进行了比较(见图 6)。由图 6 可见,当并行机处理器个数小于 4 时,执行 RB + CP 优化的远程绘制时间仍占总时间的 50% 以上,性能提升不明显。而进一步采用流水线后则不仅隐蔽了远程显示时间,且性能提升了近 30%,但绘制仍占据主导地位。此时再进行 DP 优化,已无助于整体性能的提升。当并行机处理器个数到达 8 个时,由于绘制时间有了显著降低,

此时绘制与读图像时间已经接近图像合成与显示时间的总和,因此可充分发挥出流水线优势,提高绘制速度达 2 倍多,已达到交互绘制需求。但此后随着绘制负载不断减小,远程显示已成为性能瓶颈,进而影响到并行绘制的可扩展性。当采用 LZO 压缩技术后,则将显示端的网络传输量减少了 90%(见表 2),若进一步 DP 优化后,则绘制速度可提高 3 倍。当并行规模达到 16 时,累积优化后达到了 5.87fps 的最高绘制速度。平均而言,并行机处理器个数为 8 时,优化措施最具性价比,此时 3.45fps 的绘制速度已经可以满足流畅交互的需求了。图 7 给出了采用不同传输方程的 p2_3d_20 数据体绘制结果。

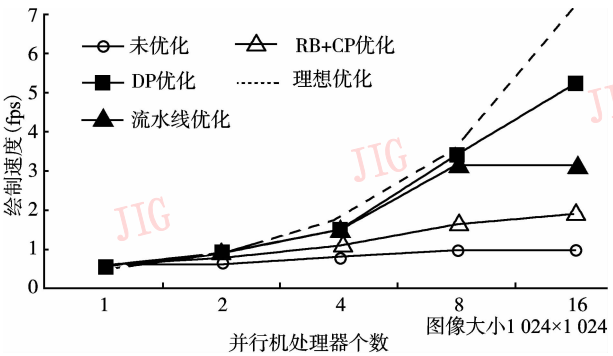

图 6 逐级累积优化后的绘制并行扩展性

Fig. 6 Rendering scalabilities with multi-pass optimizations

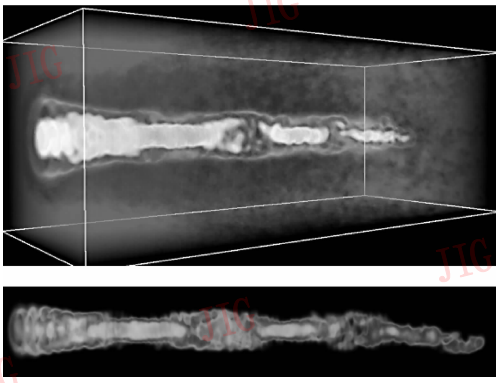

图 7 不同传输方程设置下的 p2_3d_20 数据的体绘制结果

Fig. 7 Volume rendering of p2_3d_20 dataset with different transfer function

6 结 论

由于没有图形硬件的传统并行机无法使用硬件并行体绘制处理大规模 3 维科学数据,为此本文介绍了一种协同分布式硬件的混合并行绘制模型,变不可绘制为可绘制,并充分利用现有硬件资源来满足实际科研需求。该绘制模型不仅兼顾了共享存储并行机和图形工作站的优势,而且通过并行流水线和负载平衡优化,弥补了远程模式的缺陷。实验结果表明,该绘制模型可以 1 024 ×1 024 的图像分辨率,交互绘制并行机上的大规模数据场。

参考文献(References)

1 Wilson O, Van Gelder A, Wilhelms J. Direct Volume Rendering via 3D Textures[R]. Technical Report UCSC-CRL-94-19, Jack Baskin School of English, University of California at Santa Cruz, CA, USA, 1994.

2 Gettys J, Karlton P. The X window system, version 11[A]. In: Software—Practice and Experience[C], New York: John Wiley & Sons, 1990: 35 ~ 67.

3 Womack P, Leech J. OpenGL Graphics with the X Window System, Version 1.3[EB/OL]. 1998. http://www.opengl.org.

4 Humphreys G, Buck I, Eldridge M, *et al*. Distributed rendering for scalable diaplays[A]. In: Proceedings of the ACM/IEEE Conference on Supercomputing 2000[C], Dallas, TX, USA, 2000: 30 ~ 38.

5 Humphreys G, Houston M, Ng R, *et al*. Chromium: A stream-processing framework for interactive rendering on clusters[J]. ACM Transactions on Graphics, 2002, **21**(3): 693 ~ 702.

6 Yang Jian. AnyGL: A lager scale hybrid distributed graphics system [D]. Ph. D Dissertation, Hangzhou: Zhejiang University, 2002. [杨 建. AnyGL: 一个大规模混合分布图形系统[D]. 杭州: 浙江大学,2002.]

7 Stegmaier S, Magallon M, Ertl T. A generic solution for hardware-accelerated remote visualization[A]. In: Proceedings of EG/IEEE TCVG Symposium on Visualization VisSym'02 [C], Barcelona, Spain, 2002: 87 ~ 94.

8 Stegmaier Simon, Diepstraten Joachim, Weiler Manfred, *et al*. Widening the remote visualization bottleneck[A]. In: Proceedings of IEEE International Conference on Image and Signal Processing and Analysis 03[C], Stuttgart University, Germany, 2003: 174 ~ 179.

9 Ma K L, Camp D M. High performance visualization of time-varying volume data over a wide-area network status[A]. In: Proceedings of the ACM/IEEE Conference on Supercomputing[C], Dallas, Texas, USA, 2000: 59 ~ 68.

10 Jin Zhe-fan, Lin Hai, Shi Jiao-ying. Research and implementation of a data distributed sort-first parallel rendering system[J]. Journal of Computer Research and Development, 2004, **41**(2): 376 ~ 382. [金哲凡,林海,石教英. 数据分布型 sort first 并行图形绘制系统的研究与实现[J]. 计算机研究与发展,2004, **41**(2): 376 ~ 382.]

11 Ma K L, Painter J, Hansen C, *et al*. Parallel volume rendering using binary-swap compositing [J]. IEEE Computer Graphics and Applications, 1994, **14**(4): 59 ~ 68.

12 OpenGL on Silicon Graphics Systems[EB/OL]. http://techpubs.sgi.com/library/manuals/2000/007-2392-003/pdf/007-2392-003.pdf, 2000.