基于 GPU 的体绘制算法研究

杨金柱,赵大哲,栗 伟,耿 欢,王艳飞
(东北大学医学影像计算教育部重点实验室,辽宁沈阳 110004)

摘 要: 本文给出了基于 GPU 的体绘制算法的流程,针对医学影像多组织标定问题,提出了基于二维表的纹理传输函数方法.针对 CPU 与 GPU 的资源分配与任务协同问题,提出了基于 CPU 的代理几何体生成算法,并利用该算法生成纹理坐标与 GPU 共同完成体绘制.最后对大量的医学影像数据进行了实验,实验证明本文提出的算法可以很好解决多组织标定与重建速度优化问题,使重建速度达到了毫秒级,完全满足临床需要.

关键词: 体绘制; GPU; 纹理传输函数; 代理几何体
中图分类号: TP391.41 **文献标识码:** A **文章编号:** 0372-2112 (2010) 2A-202-05

The Research Volume Rendering Algorithm Based on GPU

YANG Jin-zhu, ZHAO Da-zhe, LI Wei, GENG Huan, WANG Yan-fei
(*Key Laboratory of Medical Image Computing of Ministry of Education, Northeast University, Sheyang, Liaoning* 110004, *China*)

Abstract: This paper present the process of GPU-based volume rendering algorithm. Aiming at multi-organization for medical imaging calibration problem, we propose a two dimensions table based texture transfer function. For the problems of CPU and GPU resource allocation and tasks coordination, we propose a GPU based acting geometry algorithm to generate texture coordinates and cooperating with GPU to finish Volume Rendering. Finally, experiments have been carried out with a large number of medical imaging data and the result indicates that the algorithm proposed in this paper can be a good solution for multi-organization. The algorithm makes the calibration and reconstruction speed so efficient that the speed reaches a millisecond-level. The speed fully satisfies the clinical needs.

Key words: volume rendering; transfer function of texture; acting geometry

1 引言

体绘制技术由 Drebin 和 Levy 在 80 年代末提出,这是近年来发展迅速的一种三维数据场可视化方法,体绘制不生成中间图元,直接由三维数据场产生屏幕的二维图像[1].体绘制方法从提出以后,经过研究者 20 多年的完善与改进,从理论、方法及实现上都已经形成比较完善的体系,但仍然存在一些实际的问题,如绘制速度,传输函数的设计等.针对体绘制的速度问题,在体绘制方法领域提出了许多的加速方法,如空间跳跃、多分辨率、自适应采样和接近云等方法,这些加速优化方法确实大大加快了体绘制的速度,但是对于以惊人速度增长的影像数据而言,加速后的算法在实时绘制上也显得力不从心,很难在普通 PC 机上实现大规模数据的实时绘制.近些年来,计算机图形处理器(Graphic Processing Unit, GPU)以大大超过摩尔定律的速度高速发展,极大地提高了计算机图形处理的速度和图像质量,并促进了计算机图形相关应用领域的发展.目前,图形处理器的功能越来越强大,速度也越来越快,其性价比越来越高,为可视化工作提供了很好的基础.基于图形硬件的直接体可视化方法最早是由 Cullip 等人在 1993 年提出来的[2],但是一直以来这些方法都只能在大型的图形工作站上实现,成本太高,随着消费级的图形处理器的高速发展,越来越多的体绘制研究转向基于图形处理器的领域.GPU 的应用领域也不断扩大,在数值计算、流体模拟、场景绘制、图象处理、数据库操作等领域都有广范的应用.如 kruger 等人提出基于 GPU 的片断程序实现基本代数运算的框架[3].Guthe 等人通过定义一个剪裁纹理,在 GPU 片元上实现 B 样条曲面[4].在医学图象方面,Moreland 等人利用 GPU 渲染多遍纹理功能实现快速傅立叶变换[5],Yagel 使用纹理映射算法加速滤波反投影算法[6],在医疗领域已经开始利用 GPU 来进行三维实时超声绘制[7,8].

本文主要针对 GPU 体绘制技术进行了研究,描述

收稿日期:2007-01-23;修回日期:2008-12-03
基金项目:

了 GPU 实时体绘制的基本流程,针对医学影像分割后数据标注重建的问题,将标注与传输函数相结合,提出了一种二维表表示的纹理传输函数,对 GPU 与 CPU 任务协同中代理几何体的生成方法算法进行了研究,利用 CPU 计算代理几何体,生成纹理坐标,降低了 GPU 的运算量,合理的进行了计算资源的分配.

2　GPU 体绘制算法流程

传统意义上的直接绘制算法由于绘制时需要对整个三维体数据场进行采样,绘制速度很慢,在 PC 机上对一般尺寸的体数据集也难以进行实时绘制[9].本节重点介绍基于现代 GPU 的实时体绘制算法流程,该技术利用了现代 GPU 的灵活的程序模型和三维纹理化能力,实现对大规模体数据的实时绘制.

基于 GPU 的体绘制方法大体流程如图 1 所示,可以分为三个步骤:初始化、更新和绘制.初始化通常只运行一次,更新和绘制则需要运行很多次,每当绘制程序接受用户输入,例如观察或者渲染参数改变时,更新和绘制步骤就需要重新运行一次,具体流程如下:

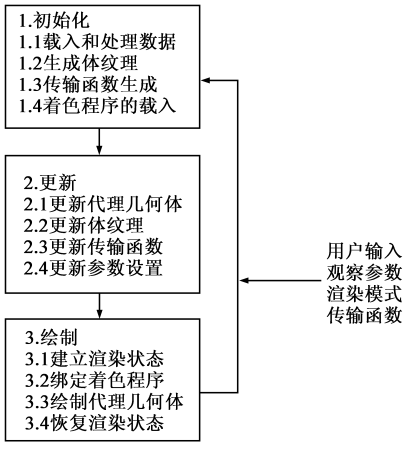


图1　基于GPU的体绘制渲染实现步骤

首先把体数据载入 CPU 的显存,某些情况下,在把数据压缩和生成纹理内存之前需要对它们进行处理,例如用户可以选择计算梯度或者减少该阶段的采样数据.也可以在算法之外做一些对数据的处理,包括增强、滤波的预处理操作.之后将体数据载入显存,生成三维纹理,算法生成默认的传输函数,载入指定的着色器程序.

在初始化和用户输入发生变化是,都需要重新计算代理几何体,然后把他们存储在顶点数组之中.当输入的体数据发生改变时,需要更新三维纹理,代理几何体由多边形组构成,而多边形则是垂直于视线方向把体切成薄片形成的.多边形切片的生成在本步骤中非常重要,下文中将详细介绍.对于用户输入改变,如视角变换、传输函数变更和渲染模式改变,还有具体环境下的配置参数,都需要在更新阶段将新的数据传递给算法.

绘制阶段,在按照排序的顺序画出多边形切片之前,需要适当的设置渲染状态,包括禁用光照和剔除,建立透明混合状态等.为了在不透明的几何体中进行绘制,必须激活深度检测,禁用写入深度缓冲的功能.根据用户配置绑定不同的着色程序,按照一定的顺序绘制代理几何体,光栅化过程中执行片元着色程序,在 GPU 上完成颜色映射,图像合成,如果开启光照,还需要实时的计算光照效果,输出最终的像素点颜色.绘制完成后,需要恢复正常的渲染状态.

3　基于二维表的纹理传输函数

在初始化步骤中,主要完成算法的一些必须的初始化操作,包括数据的载入和数据处理、三维纹理的生成、默认传输函数的生成以及着色程序的载入.

3.1　数据载入与体纹理生成

体数据有不同的大小和类型,载入的数据多数都不能满足各向比例相同的特性,为了完成体绘制,需要对这些数据进行插值,还原体数据原有的比例.有多种算法可以选择,本文使用三线性插值算法,该插值算法兼顾质量和速度,通常在 CPU 上实现,本文将该算法在 GPU 上重新实现,是 GPU 通用计算的一个应用,GPU 上实现带来效率上的提升,速度提高很多.经过数据读取和处理,体数据被缓存在系统内存中,基于 GPU 的体绘制算法需要将数据载入图形硬件缓存,也就是显存中,数据在显存中以纹理形式存在,纹理包括一维纹理、二维纹理和三维纹理.对于体数据包括三个维度,体现数据的三维空间性质,将体数据以三维纹理形式载入显存,供 GPU 处理使用.

3.2　基于二维表的纹理传输函数

传输函数就是将体数据的信息映射成颜色、不透明度等光学属性,并赋予每个体素.这是一种函数映射关系,根据作为输入的信息不同,可以分为一维传输函数和多维传输函数.传输函数设计的好坏,对体绘制的结果会产生十分重大的影响.设计出色的传输函数,可以将原始数据集中的不同物质、不同结构区分开,在显示人们感兴趣的重要结构的时候,隐藏不重要的信息.目前,广泛应用的是一维传输函数.之所以称之为一维,是因为它只以数据的标量值作为函数的输入,作为区分不同物质的依据.

如果在绘制时,对每个采样点都要根据传输函数计算映射的光学属性,无疑需要耗费大量的计算时间.因此在初始化阶段,预先将传输函数采样转化为查找表,绘制时只要以传输函数的自变量作为索引,到查找表中取出颜色、不透明度等光学属性即可.而查找的过

程,需要在 *GPU* 上完成.将查找表作为纹理装载至显存中,在片元着色器中查找纹理的方法即可取得需要的映射值[10].

对于医学可视化系统,通常需要对检测出的不同组织、器官加以标注,以不同的颜色或者绘制效果显示出来,这就需要满足对于同一个数据标量值,其对应的光学属性可能不相同,对于多组织器官的标定问题,传统的一维传输函数无法满足要求,本文提出一种基于二维表的纹理传输方法.

一维传输函数本身可以只使用一维纹理,使用数据标量值作为自变量,对应结果是 *RGBA* 的光学属性.在设计传输函数时考虑了体数据的多器官的标注问题,本文将标注与传输函数相结合,如图 2 所示,使用一个二维表来表示传输函数,纹理的 x 轴表示数据的标量值,y 轴表示对应的标注号,设计最多可以标注 8 个不同的组织,每个组织对应一个传输函数.将这个二维表生成二维纹理载入显存,颜色和透明度都需要转化到 0 ~ 255 之间.结果图像渲染时,在片元着色器中完成查找,根据采样点的数据标量值和标注值作为纹理坐标,查询传输函数二维纹理,即可获得采样点的光学属性.

4 GPU 与 CPU 任务协同中的代理几何体生成算法

在完成初始化和用户输入改变时,需要更新一些数据,包括代理几何体、纹理、传输函数和设置参数等,其中最为关键的是代理几何体的生成.产生代理几何体目的是为了生成纹理坐标,目前比较常用的是过图形 API 自动生成纹理坐标的方法.自动生成纹理坐标的方法需要耗费 GPU 运算资源,体绘制中需要大量的片元着色器运算,这将降低这 GPU 的使用效率,需要对大规模医学影像计算任务在 CPU 和 GPU 之间进行合理的分配,进行协同计算处理.而 GPU 的纹理坐标的生成对于 CPU 来说是一件相对轻松的工作,因此本文将在 CPU 上进行代理几何体的生成,并产生纹理坐标提供给 GPU.

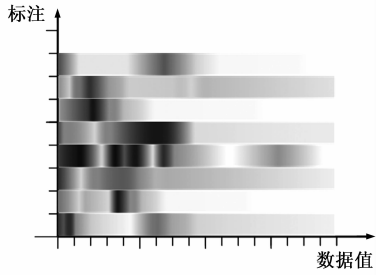


图2 传输函数的设计

对于任意的视线方向,需要生成与之垂直的一系列纹理多边形.首先求出一系列垂直于视线方向的平面切割体数据包围盒所形成的截面,这些截面是一些多边形,即切片多边形,这些多边形组就是代理几何体.

长方体的体数据包围盒与任意一个平面的切片形状有四种可能,如图 3 所示:

(a)三角形(图 3(a)),平面与体数据的三条棱相交.

(b)四边形(图 3(b),(c)),平面与体数据的四条棱相交.

(c)五边形(图 3(d)),平面正好经过体数据的五条棱相交.

(d)六边形(图 3(e)),平面与体数据的六条棱相交.

计算多边形切片的步骤如下:

Step 1: 使用观察模型矩阵,把体包围盒顶点转变为观察坐标.

Step 2: 计算垂直于视线的切面的平面方程.

假定视点为 $vp(x,y,z)$,视线的负方向为 $vec(x,y,z)$,垂直于视线的平面方程为:

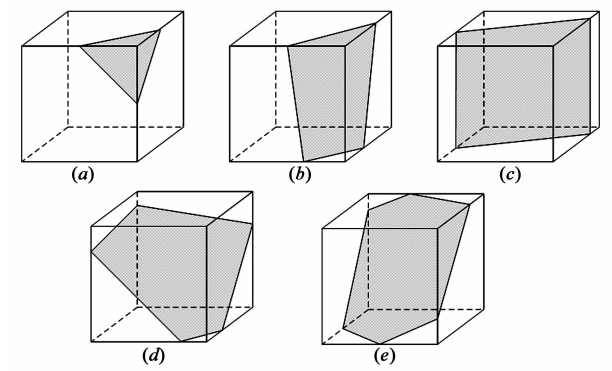


图3 面与体的相交

$Ax + By + Cz + D = 0$,则有:

$$\begin{cases} A = vec.x \\ B = vec.y \\ C = vec.z \end{cases} \tag{1}$$

取 D 的初始值为零,即初始切片平面通过体数据的原点,可以通过调整 D 的值,得到一组互相平行的平面.

Step 3: 按从前到后的顺序处理每个平面,计算平面与棱的交点.

体数据与平面的切片多边形,可以通过平面与包围盒的各条棱的交点来确定.设置数组 pts[8]存储长方体的各顶点坐标,整型的二维数 edges[12][2]存储每条棱的端点在 pts[8]中的索引.

对于每条棱判断其与已知平面是否相交,用函数表示已知平面方程为:$F(x,y,z) = Ax + By + Cz + D$.根据点与平面的关系,若点 pts 在平面上方,则 $F(pts) > 0$;若点在平面下方,则 $F(pts) < 0$;若点在平面上,则 $F(pts) = 0$.设棱的两个端点分别为 $pt1$、$pt2$,判断该棱与平面是否有交点:

若 $F(pt1) \cdot F(pt2) > 0$,则两点在平面同一侧,棱与

平面无交点;

若 $F(pt1)\cdot F(pt2)<0$,则两点在平面两侧,棱与平面相交;

若 $F(pt1)=0, F(pt2)=0$,则两点都在平面上,平面经过该棱;

若 $F(pt1)=0, F(pt2)\neq 0$,则点 $pt1$ 在平面上;

若 $F(pt1)\neq 0, F(pt2)=0$,则点 $pt2$ 在平面上.

若棱与平面相交,需要求出它们的交点. 令 $t=\dfrac{F(pt2)}{F(pt2)-F(pt1)}$,则交点 $edgeInt$ 的坐标计算如下:

$$\begin{cases} edgeInt.x = t\times pt1.x + (1-t)\times pt2.x \\ edgeInt.y = t\times pt1.y + (1-t)\times pt2.y \\ edgeInt.z = t\times pt1.z + (1-t)\times pt2.z \end{cases} \quad (2)$$

Step 4:将多边形转化为三角网格

为了提高渲染效率,需要将非三角形的切面转化为由三角形组成的切面. 对一个面上的相交点求平均值,计算代理多边形的中心点,把相交点投影到 $X-Y$ 平面上,使用第一个顶点或者 X 轴作参照,计算他们围绕中心的角度,从而对多边形顶点按顺时针或者逆时针排序. 把切片多边形转化为以中心为公共点的三角形扇面或者三角形带.

在绘制阶段,需要建立渲染状态,其中透明混合模式的设置决定最终的图像的渲染结果,透明混合是指使用指定的混合函数,把源颜色和目标颜色进行混合产生组合颜色值,GPU 着色程序负责查询传输函数,完成由矢量值到 RGBA 光学属性的映射,合成最终结果.

5　实验结果与评价

本文所述基于 GPU 的实时体绘制算法是在普通 PC 机上完成的,具体的软硬件系统配置如下:

(1)硬件系统

CPU:Intel® Pentium® 4 HT 3.06GHz

内存:DDR2,1536MB

显卡:NVIDIA GeForce 8800 GTS,640MB

硬盘:80GB,7200RPM,ATA100

(2)软件系统

操作系统:Windows XP Professional SP2

开发环境:Visual C++ 6.0

图形 API:OpenGL 2.0

着色语言:Cg 1.5

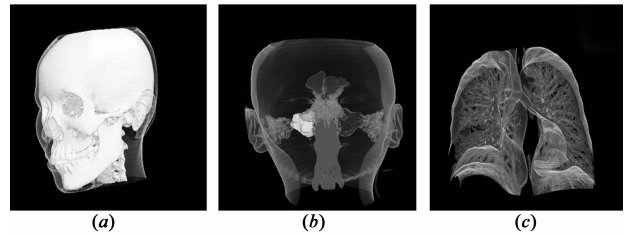
利用本文的算法对医学影像数据进行三维重建,实验结果如图 4 所示,其中图 4(a)是头部体绘制结果,(b)头部多器官标定的透明显示结果,(c)是肺部体绘制结果.用本文的算法对不同规模数据集进行测试,绘制分辨率为 512×512 时,不同数据集的绘制速度见表 1.

表 1　不同规模体数据的绘制速度

数据集	数据集大小	绘制分辨率	(FPS/s)
耳道	128×128×30	512×512	28
牙齿	256×256×161	512×512	23
肺部	512×512×88	512×512	18
头部	512×512×200	512×512	15
心脏	512×512×512	512×512	13

在上述算法研究基础上,本文实现了基于 GPU 的三维可视化系统,如图 5 所示,主要包括不同三维显示算法如:SSD、VR、MinIP、MIP;重建结果的三维操作如:放大平移、包围盒调节、动态阈值调节、光照模型动态调节等.

从本文的实验结果可以看出:算法重建速度达到实时要求.对于 512^3 规模的数据,基本法重建速度达到 12FPS/s,满足实时要求,基于二维表的纹理传输函数,解决了医学影像多组织标定问题,在任务协同中利用 CPU 进行代理几何体的生成,在降低 GPU 资源消耗的同时,绘制质量没有受到影响.



(a)　(b)　(c)

图4　三维重建结果与纹理传输函数

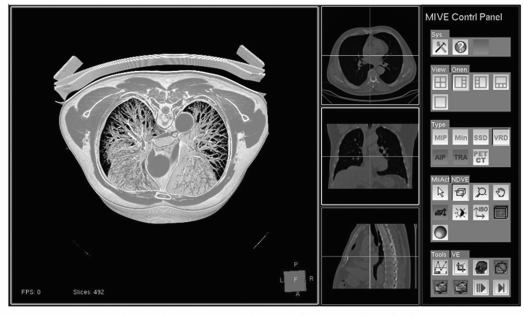


图5　基于GPU的三维可视化系统

参考文献:

[1] R A Drebin, L Carpenter, P Hanrahan. Volume rendering[J]. Computer Graphics, 1988, 22(4): 65 – 74.

[2] Cullip T J, Neumann U. Accelerating Volume Reconstruction with 3D Texture Mapping Hardware[R]. Unversity of North Carilina, Technical Report UNC, 1993, 7: 23 – 45.

[3] Krüger. Westermann. Linear algebra operators for GPU implementation of Numerical algorithms[J]. ACM Transactions on Graphics, 2003, 22(3): 908 – 916.

[4] Michael Guthe, Akos Balazs, Reinhard Klein. GPU-base trim-

ming and tessellation of NURBS and T-Spline surfaces[J]. ACM Trans Graph,2005,24(3):1016 – 1023.

[5] Moreland K,Angel A. The FFT on a GPU[A]. In:Proc. of the Graphics Hardwardware[C]. 2003,23(6):546 – 552

[6] Klaus Mueller,Roni Yagel. Rapid 3-D Cone-Beam Reconstruction with Simultaneous Algebraic Reconstruction Technique Using 2-D Texture Mapping Hardware[J]. IEEE Transaction on Medical Imaging,2000,12(9):221 – 230.

[7] Paul M. Novotny,Jeffrey A Stoll,Nikolay V Vasilyev,GPU-Based Real-Time Instrument Tracking with Three Dimensional Ultrasound[M]. Medical Image Computing and Computer-Assisted Intervention-MICCAI 2006,2006. 112 – 118.

[8] Kuo J,Bredthauer G R,Castellucci J B,von Ramm OT. Interactive volume rendering of real-time three-dimensional ultrasound images[J]. IEEE Transactions on Ultrasonics,Ferroelectrics and Frequency Control,2007,54(2):313 – 318.

[9] Kindlmann G,Durkin J W. Semi-automatic generation of transfer functions for direct volume rendering[J]. IEEE Symposium on Volume Visualization,1998,24(2):79 – 86.

[10] R. Westermann,T Ertl. Efficiently using graphics. hardware in volume rendering applications[A]. Proceedings of SIGGRAPH[C]. New York:ACM Press,1998,32(4):169 – 179.

作者简介:

杨金柱(通信作者) 男,1979 年生,博士,东北大学医学影像计算教育部重点实验室讲师,研究方向为医学影像处理、三维可视化,计算机图形学.

E-mail:yangjinzhu@neusoft.com

赵大哲 女,1960 年生,博士,东北大学医学影像计算教育部重点实验室主任,博士导师,柏林工业大学博士,主要研究方向医学影像计算,数据挖掘、工作流管理.