第 35 卷 第 3 期 计 算 机 工 程 2009 年 2 月
Vol.35 No.3 Computer Engineering February 2009
·博士论文· 文章编号：1000—3428(2009)03—0010—03 文献标识码：A 中图分类号：TP311.13

# 基于 Ray-casting 算法的并行存储系统

魏继增，孙济洲
(天津大学计算机科学与技术学院，天津 300072)

**摘 要：** Ray-casting 算法是一种高质量的直接体绘制算法，但绘制速度过慢，因此设计基于 Ray-casting 算法的硬件专用体系结构已成为研究的热点。而存储系统又是制约整个体系结构的瓶颈部件，其性能的优劣直接影响整个系统的运行速度。该文针对直接体绘制中的 Ray-casting 算法设计了无访存冲突的八体低位交叉并行存储系统 VOXMEM 提高吞吐率，并提出相应的体素存储分配策略和地址计算方法。该并行存储系统采用基于页模式的 SDRAM 实现，并通过仿真实验获得了令人满意的结果。

**关键词：** 直接体绘制；Ray-casting 算法；低位交叉并行存储系统；页模式

# Parallel Memory System Based on Ray-casting Algorithm

**WEI Ji-zeng, SUN Ji-zhou**
(Institute of Computer Science and Technology, Tianjin University, Tianjin 300072)

**【Abstract】** Ray-casting algorithm is a widely recognized method for high quality direct volume rendering, but the speed of rendering is too slow. Therefore, the design of specific hardware architecture based on Ray-casting is a hot research point. But memory system becomes the bottle-neck of the whole architecture and its performance directly affects the running speed of the whole system. This paper designs a conflict-free eight-bank low-order interleaving parallel memory system VOXMEM to increase the throughput, and voxel distribution function and address computation method are also proposed. The parallel memory system is achieved based on page mode SDRAM. Satisfied results are got through the emulation experiment.

**【Key words】** direct volume rendering; Ray-casting algorithm; low-order interleaving parallel memory system; page mode

## 1 概述

直接体绘制(direct volume rendering)随着体数据规模越来越大，导致绘制速度与用户要求实时进行显示处理之间的矛盾越来越严重，因此在保证绘制质量的前提下，速度成为体绘制技术进一步发展的瓶颈。

近 10 年直接体绘制加速技术的研究重点在针对某具体算法设计专用硬件体系结构，国外学者已取得较大成果，如德国 Tübingen 大学的 VIZARDⅡ[1-2]、Saarland 大学的 SaarCOR[3]和 RPU[4]、纽约州立大学石溪分校的 RACEⅡ[5]等。国内对此方面的研究主要停留在利用传统图形硬件进行加速，而对专用硬件体系结构的研究开展较少。

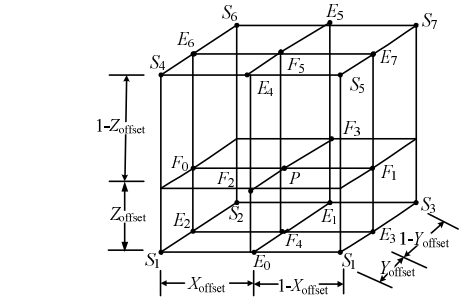
## 2 Ray-casting 算法

Ray-casting[6]是像序直接体绘制的典型代表。其基本原理为：从投影平面上每一个像素出发，沿视线方向发出一条射线进入体素空间。在这条射线上等间距选取若干重采样点，计算每个重采样点在体空间中的位置，然后用所选定的重构函数通过插值计算得到重采样点的值，再通过传递函数和明暗计算确定其不透明度及颜色值，最后对各个重采样点按照从后向前或者从前向后进行图像合成。该算法主要缺点是绘制速度较慢，最大的优点是能够产生高质量的图像。

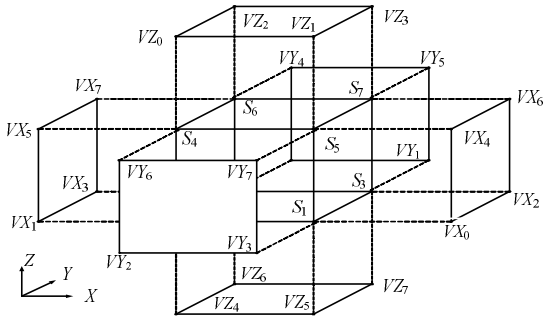
## 3 体素存储分配策略及 VOXMEM 设计

### 3.1 绘制模型分类

Ray-casting 对每一个重采样点进行明暗计算时(Phone 光照模型)，需要对其进行梯度估计。根据梯度估计的不同方法将 Ray-casting 分为 2 类绘制模型：快速绘制模型和高质绘制模型，如图 1 所示。




(a)快速绘制模型




(b)高质绘制模型

**图 1 Ray-casting 算法绘制模型**

**基金项目：** 国家自然科学基金资助项目“实时体绘制关键技术及支撑体系结构研究”(60373061)

**作者简介：** 魏继增(1981－)，男，博士研究生，主研方向：计算机体系结构，并行计算，计算机图形学；孙济洲，教授、博士生导师

**收稿日期：** 2008-10-31 **E-mail：** weijizeng_2001@163.com

图 1(a)为快速绘制模型，首先使用包围重采样点 $P$ 的 8 个体素($S_0$,$S_1$,…,$S_7$)的值和重采样点的坐标偏移($X_{\text{offset}}$, $Y_{\text{offset}}$, $Z_{\text{offset}}$)通过线性插值计算出 $E_0$,$E_1$,…,$E_7$ 的值，然后再次利用线性插值计算 $F_0$, $F_1$,…,$F_5$ 的值，最后计算重采样点 $P$ 的值 $D$ 和梯度估计在 3 个方向上的分量。图 1(b)为高质绘制模型，对梯度进行更准确的计算，除需要包围某个重采样点一个体元(体素由 $S$ 标记)，还需要与该体元在 3 个方向上共面的 6 个相邻体元(体素分别由 $VX$, $VY$, $VZ$ 标记)。重采样点值的计算方法与快速模型相同。梯度估计计算过程如下：首先利用标记 $S$ 的 8 个体素 $S_0$, $S_1$,…,$S_7$ 和标记 $VX$ 的 8 个体素 $VX_0$,$VX_1$,…,$VX_7$ 通过中心差分的方法得到 $S$ 体素梯度在 $X$ 方向上的分量。对这 8 个体素的梯度在 $X$ 方向分量进行 3 次线性插值可得到重采样点的梯度在 $X$ 方向上的分量，采用同样的方法可计算出重采样点的梯度在 $Y$, $Z$ 方向上的分量。

快速绘制模型针对每个重采样点需要存储系统 VOXMEM 能够在一个存储周期之内并行取出 8 个体素，而高质绘制模型针对每个重采样点需要分 4 个存储周期并行取出 32 个体素($S$ 类、$VX$ 类、$VY$ 类、$VZ$ 类)，而这些采样点存储位置不连续。因此无论是哪种绘制模型，为了能对体素进行并行存储，采用八体低位交叉并行存储的存储器组织方式。

### 3.2 体素存储分配策略

低位交叉并行存储系统的主要目的是提高存储器访问速度。假设体数据为三维网格结构，规模是 $512^3$×16 bit。每个体素位置坐标为($X$, $Y$, $Z$)，其中，$0 \leqslant X \leqslant 512$，$0 \leqslant Y \leqslant 512$，$0 \leqslant Z \leqslant 512$。按系统性能要求相对于某个重采样点的各个体元必须在一个存储周期内获得，因此整个存储系统需要 8 个独立的存储模块和 1 个无冲突体素分配策略。独立存储模块将组成八体低位交叉并行存储系统 VOXMEM。无冲突分配策略如下：

将所有体素按位置坐标的最低有效位分为 8 个子集 $M_i$，如下式所示：

$$M_i = \{Voxel_{(x,y,z)} \mid Z_0Y_0X_0 = i,\ i = 0,1,\cdots,7,(X,Y,Z)\text{为体素坐标}\} \tag{1}$$

每个子集对应一个存储模块。这样保证每个体元中的 8 个体素分布在不同的存储模块中，避免了访问冲突。对于某个体素，其所在存储模块块号 $\beta$ 可由下式得到：

$$\beta = M_i = \beta_2\beta_1\beta_0 = Z_0Y_0X_0 \tag{2}$$

体素在各个存储模块中的分配情况如图 2 所示。

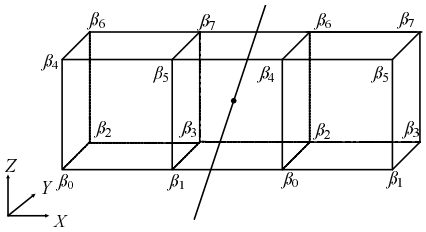



**图 2 体素分配示意图**

对于任意子集中的体素，假设彼此之间的相对位置不变，则该子集中的体素又构成三维网格结构。由图 2 可知，每个体素所属的存储模块号间隔出现，因此每个子集中的体素在新构成的三维网格结构中的新位置坐标由下式所示：

$$\overline{NC} = (NX, NY, NZ) = (X_{8\cdots1}, Y_{8\cdots1}, Z_{8\cdots1}) \tag{3}$$

此坐标就是体素在其存储模块中的块内地址。

考虑图 2，观察到在同一方向上每 2 个共面的体元有 4 个体素被共用，这样将导致某个体元的体素在各自存储模块中块内地址不同(共面 4 个体素的块内地址与另 4 个体素不同)。这样不满足并行存储系统设计要求，降低存储系统的吞吐率。为解决此问题，假设每个存储模块由 8 个存储单元(memory unit)构成，将每个存储模块中的体素按新坐标位置 $\overline{NC}$ 的最低有效位分配到 8 个存储单元中，假设每个存储模块内存储单元标号为 $\alpha$，则

$$\alpha = \alpha_2\alpha_1\alpha_0 = NZ_2NY_1NX_0 \tag{4}$$

每个体素在存储单元中的地址可由下式所得：

$$\overline{NC}' = (NX_{7\cdots1}, N_{7\cdots1}, NZ_{7\cdots1}) = (X_{8\cdots2}, Y_{8\cdots2}, Z_{8\cdots2}) \tag{5}$$

这样每个体元的 8 个体素属于不同存储模块的不同存储单元中，同时在各个存储单元中的相对位置相同，完全避免了访问冲突，保证了吞吐率。采用目前市场上主流的基于页模式(page mode)的 SDRAM 可实现 VOXMEM。

### 3.3 VOXMEM 八体低位交叉并行存储系统的设计

SDRAM 内部存储单元构成二维阵列，因此对存储单元寻址时分 2 次赋予地址。其内部每一行称为一页，每次选通行地址之后将该行中所有的数据缓存在 Cache 中，若下次读取仍在此行则 Cache 不需刷新；若所需数据不在此行，则需对 Cache 进行刷新缓存新的一行的数据。

基于 SDRAM 的特性，VOXMEM 的任务是将所有体素都分配到各个存储设备中去，使页能够频繁的使用。对于规模为 $512^3$×16 的体数据所需总存储容量为 256 MB。已知该存储系统由 8 个存储模块组成，每个存储模块又由 8 个存储单元构成，则该存储系统共需 64 个存储单元，每个存储单元容量为 2 M×16 bit。选取 2 片存储容量为 2 M×8 bit(2 048 row×1 024 column×8 bit)的 SDRAM 采用位扩展方式组成一个存储单元。将这样的 8 个存储单元集成在一起构成一个存储模块，同理 8 个存储模块再次按低位互连方式组成 VOXMEM 存储系统。每个存储单元设置一个 Cache，缓存从存储单元中读出的某一行(页)体素。处在同一存储单元中的体素可以按 2 种方式存放：

(1)按 $X$, $Y$, $Z$ 中任一方向存放体素。

(2)将整个体数据划分成若干大小相同数据块，存储单元中每一行(页)存储一个数据块。

第 2 种方式在性能上要优于第 1 种，光线可能来自任一方向，对于第 1 种方式若按 $X$ 方向存储，当投射光线与 $X$ 轴平行时，则 Cache 的命中率大约为 100%。但是当投射光线与 $Y$ 轴或 $Z$ 轴平行时，每处理一个重采样点都要对 Cache 进行一次刷新，命中率为 0。若采用第 2 种方式，假设以 8×8×8 为一个数据块，则无论光线从哪个方向射入，只有当重采样点落在边界体元时才需要对 Cache 进行刷新，此时 Cache 命中率为 87.5%。

按上述存储单元的组织结构，所有存储单元在同一行中的存储量为 1024×64×16 = 1Mb，共 64 K 个体素。选取规模为 32×64×32 单元为一个子体，共 2 K 个子体，恰好为选取 SDRAM 的行数(2 个 SDRAM 组成 1 个存储单元)。而每个存储单元每一行中需存放 1 K 体素，所以将每个子体再细分成 8×16×8=1 K 的数据块。

## 4 基于 VOXMEM 地址计算方法

为了对体素进行读取需要完成将三维坐标转换为 SDRAM 的行地址与列地址。此转换过程分为 2 个部分：体素新位置坐标的确定和行列地址的计算，由 2 个转移函数 $\zeta$ 和 $\eta$ 完成。

### 4.1 体素新位置坐标的确定

**定义** 参考点(Reference Point, RP)是包围某个重采样点 $X_R=(X_R,Y_R,Z_R)$ 的 8 个体素中最靠近坐标原点的体素，其坐标为 $\overline{RP}=(U,V,W)=(\lfloor X_R\rfloor,\lfloor Y_R\rfloor,\lfloor Z_R\rfloor)$。

体素新位置坐标可由式(3)得到。若计算体素子集中所有体素的坐标，就要事先存储原始坐标，这样将浪费存储空间。而根据对参考点的定义，每个重采样点都唯一对应一个参考点，因此对于高质绘制模型而言，每个参考点都确定一组 *S* 类、*VX* 类、*VY* 类和 *VZ* 类体素。最终，体素子集中体素新位置坐标可由相关的参考点决定。

根据上述论述转移函数 $\zeta$ 完成体素新位置坐标的计算，由于体素分 4 类，因此函数 $\alpha$ 分为 4 种情况考虑(对于快速绘制模型只需计算 *S* 类体素即可)：

(1)*S* 类体素：

$$\overline{RP}\rightarrow\overline{NC}=\begin{cases}NX=U_{8\ldots1}+U_0=\lfloor U/2\rfloor+U_0 & \text{for}\quad \beta_0=0\\ NX=U_{8\ldots1}=\lfloor U/2\rfloor & \text{for}\quad \beta_0=1\\ NY=V_{8\ldots1}+V_0 & \text{for}\quad \beta_1=0\\ NY=V_{8\ldots1} & \text{for}\quad \beta_1=1\\ NZ=W_{8\ldots1}+W_0 & \text{for}\quad \beta_2=0\\ NZ=W_{8\ldots1} & \text{for}\quad \beta_2=1\end{cases}$$

(2)*VX* 类体素：将 *S* 类体素公式的第 1 行用下式替代。

$NX=U_{8\ldots1}+\overline{U_0}$ for $\beta_0=0$; $NX=(-1)^{\overline{U_0}}+U_{8\ldots1}$ for $\beta_0=1$

(3)*VY* 类体素：将 *S* 类体素公式的第 2 行用下式替代。

$NY=V_{8\ldots1}+\overline{V_0}$ for $\beta_1=0$; $NY=(-1)^{\overline{V_0}}+V_{8\ldots1}$ for $\beta_1=1$

(4)*VZ* 类体素：将 *S* 类体素公式的第 3 行用下式替代。

$NZ=W_{8\ldots1}+\overline{W_0}$ for $\beta_3=0$; $NZ=(-1)^{\overline{W_0}}+W_{8\ldots1}$ for $\beta_3=1$

### 4.2 行列地址计算

体素新位置坐标仅表示体素之间的相对位置关系，但体素是存储在 SDRAM 存储器中，需要将新位置坐标转换成能对存储器进行读取的行地址与列地址。即构造一个转移函数 $\eta$ 将三维地址转换为行地址与列地址。

又由 3.3 节的讨论可知，VOXMEM 对体素的存储不是采用固定的某一方向依次存储的，而是将体数据划分成若干大小相同的数据块，每个存储单元的每一行存储一个数据块。对于规模为 $512^3\times16$ bit=256 MB 的体数据而言，选用 2 片 SDRAM(2 048 row×1 024 column×8 bit)组成一个存储单元，将每个规模为 2 M×16 bit 的体数据按 8×16×8 分成若数据块，数据块具体存储顺序如下：先沿 *X* 方向存储依次存储 8 个体素，然后沿 *Y* 方向存储 16 行，最后沿 *Z* 方向存储 8 层。因为每个体素新的位置坐标(*NX*,*NY*,*NZ*)的最低有效位($NX_0$,$NY_0$,$NZ_0$)确定其属于哪个存储单元，所以三维地址到行列地址的转换函数 $\eta$ 根据存储顺序定义如下：

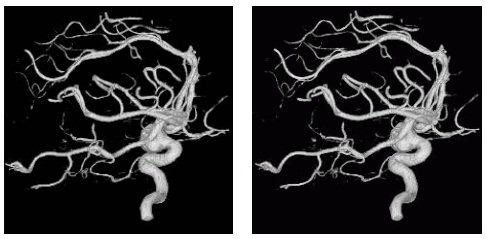
$$\eta:\overline{NC}\rightarrow ADC=NZ_{3\ldots1}\times2^7+NY_{4\ldots1}\times2^3+NX_{3\ldots1}$$
for column access

$$\overline{NC}\rightarrow ADR=NZ_{7\ldots4}\times2^7+NY_{7\ldots5}\times2^4+NX_{7\ldots4}$$
for row(page) access

## 5 实验结果与分析

对 VOXMEM 八体低位交叉并行存储系统的设计采用 NEC 生产的 2 M×8 bit SDRAM。利用 Altera 的 Cyclone II 设计 SDRAM 存储控制器。

将 VOXMEM 与其他硬件支持部件进行连接，对规模为 $512^3\times16$ bit 体数据(CT Vessel)在 2 种绘制模型上进行仿真，视点绕 *Z* 轴旋转 360°，每 1°绘制一次，产生 360 帧图像，图像分辨率为 512×512。绕 *Z* 轴旋转 90°后，效果如图 3(a)、图 3(b)所示，存储系统性能和绘制速度如表 1 所示。同时将此硬件体系结构与利用 GPU 加速的方法进行比较，采用快速绘制模型，帧数和分辨率与上述相同，GPU 加速算法在一台配置 Pentium 4 CPU 2.4 GHz, 2 GB RAM, Nvidia Geforce 8 400 MB GT GPU(显存 256 MB)PC 机上实现，效果如图 4(a)、图 4(b)所示，性能比较如表 2 所示。表 3 为此硬件体系结构与 VIZARDⅡ硬件加速器性能比较。

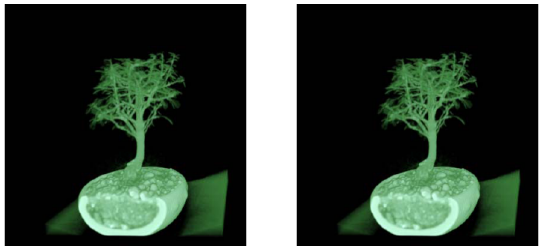


(a)CT Vessel 快速绘制模型　(b)CT Vessel 高质绘制模型

**图 3　使用 VOXMEM 进行绘制效果的比较**

**表 1　存储系统性能和绘制速度**

| 体数据 | 规模/bit | 数据块大小 | 绘制模型 | VOXMEM 存储系统存取时间 | Cache 命中率/(%) | 帧率/fps |
|---|---|---|---|---|---|---|
| CT Vessel | $512^3\times16$ | 8×16×8 | 快速<br>高质 | 27.4 ns/<br>8 voxel | 98.6 | 20.6<br>5.2 |



(a)CT 盆景 GPU 加速绘制结果　(b)盆景硬件快速绘制结果

**图 4　与 GPU 加速的绘制效果比较**

**表 2　与 GPU 绘制速度的比较(体数据规模 $256^3\times8$ bit)**

| GPU 与专用硬件 | 总时间/s | 帧率/fps |
|---|---|---|
| GPU 加速 | 12.85 | 28 |
| 专用硬件绘制(快速) | 8.67 | 41.5 |

**表 3　与 VizardⅡ绘制速度的比较(体数据规模 $256^3\times8$ bit)**

| VizardⅡ与专用硬件 | 总时间/s | 帧率/fps |
|---|---|---|
| VizardⅡ | 51.42 | 7.0 |
| 专用硬件绘制(快速) | 8.67 | 41.5 |

表 1 显示在一个存储周期之内(30 ns)可顺利读取计算一个重采样点所需 8 个体素，Cache 命中率较高，使此存储系统的性能得到了充分发挥。对于快速绘制模型，其绘制速度约为高质绘制模型的 4 倍。表 2、表 3 显示，本文 Ray-casting 专用硬件体系结构绘制速度约为采用 GPU 进行加速的 1.5 倍，为 RPU 的近 6 倍，获得更好的实时交互绘制效果。

## 6 结束语

本文针对 Ray-casting 专用硬件支撑体系结构中的瓶颈部件——存储系统进行分析，采用并行存储系统以每周期同时读取多个体素的方法解决了速度瓶颈，实现对大规模体数据的实时交互绘制。由此可见，针对具体体绘制算法设计相应的专用硬件体系结构是解决大规模体数据绘制的最优途径。后续工作将考虑采用 DDR, DDR Ⅱ等高速存储器代替 SDRAM 以获得更高的存取速度。

(下转第 17 页)