一种统一的硬件加速自适应 EWA Splatting 算法

陈 为[1)] 夏佳志[1)] 张 龙[2)] 于 洋[1)] 郑文庭[1)] 彭群生[1)]

[1)](浙江大学 CAD&CG 国家重点实验室 杭州 310058)

[2)](杭州电子科技大学图形图像研究所 杭州 310018)

摘 要 提出了一种新的硬件加速自适应 EWA(椭圆加权平均)Splatting 算法框架,可同时适用于三维体数据和点模型.算法将高斯重建核与低通图像滤波核结合,得到反走样、无模糊的高质量图像.提出一种高效的自适应滤波方法,减少了高质量 EWA Splatting 的计算量.提出了自适应体 EWA Splatting 的 3 种数据存储模式和一系列高级特性,其中包括交互式分类、体-面混合绘制策略和自适应浮点累加.展示了如何在可编程图形处理单元(GPU)中计算体数据和点模型数据的 EWA Splat 基元.实验表明,文中的方法在一台普通微机上每秒可绘制 1500 万~2000 万个基元,达到较高的图像质量与交互的绘制速度.

关键词 体绘制;点绘制;Splatting;EWA 滤波;反走样;硬件加速

中图法分类号 TP391 **DOI 号**: 10.3724/SP.J.1016.2009.01571

A Uniform Hardware-Accelerated Adaptive EWA Splatting Algorithm

CHEN Wei[1)] XIA Jia-Zhi[1)] ZHANG Long[2)] YU Yang[1)] ZHENG Wen-Ting[1)] PENG Qun-Sheng[1)]

[1)](*State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou* 310058)

[2)](*Institute of Graphics and Image, Hangzhou Dianzi University, Hangzhou* 310018)

Abstract This paper presents a novel framework for hardware-accelerated adaptive EWA (elliptical weighted average) Splatting. EWA Splatting combines a Gaussian reconstruction kernel with a low-pass image filter for high image quality without aliasing artifacts or excessive blurring. This paper introduces an efficient adaptive filtering scheme to reduce the computational cost of high quality EWA Splatting, and shows how to compute the EWA Splat primitives for volume data and for point-sampled surface data on modern graphics processing units (GPUs). To accelerate the rendering, the splat geometry and data attributes are assembled locally in video memory. For adaptive EWA volume Splatting, it proposes three data storage modes and several advanced features including interactive classification, hybrid surface-volume rendering and adaptive floating-point accumulation. The current implementation renders 15~20 millions primitives in a consumer PC. Several results for rectilinear volume data and point-sampled surfaces demonstrate the high image quality and interactive rendering speed of the proposed approach.

Keywords volume rendering; point-based rendering; Splatting; EWA filter; anti-aliasing; hardware acceleration

---

收稿日期:2008-08-14;最终修改稿收到日期:2009-01-04.本课题得到国家"九七三"重点基础研究发展规划项目基金(2009CB320803)、国家自然科学基金(60503056)和国家"八六三"高技术研究发展计划项目基金(2007AA01Z339)资助.**陈 为**,男,1976 年生,博士,副教授,主要研究方向为可视化.**夏佳志**,男,1984 年生,硕士,主要研究方向为可视化.**张 龙**,男,1981 年生,博士,主要研究方向为自然场景实时模拟、可编程图形硬件加速.**于 洋**,男,1984 年生,硕士研究生,主要研究方向为可视化.**郑文庭**(通信作者),男,1974 年生,博士,副研究员,主要研究方向为计算机图形学.E-mail: wtzheng@cad.zju.edu.cn.**彭群生**,男,1947 年生,博士,教授,博士生导师,主要研究领域为真实感图形、数字几何处理、虚拟现实、红外成像仿真和生物计算等.

1 引 言

Splatting 最初是一种物体空间的直接体绘制方法[1]. 它采用三维重建核,结合每个标量在标量场上采样重建连续信号,并预计算每个点采样的三维核函数沿视线方向的二维积分,并在图像平面上累积其贡献. Splatting 的最大优点是只处理相关的体素,可同时得到较高的绘制质量与可交互性. 由于使用了预积分重建核,在处理大量稀疏体数据[2]时,Splatting 在质量和效率上均可胜过光线追踪算法[3]和三维纹理切片方法[4].

最近几年基于图形硬件的直接体绘制有了很大的进展,包括光线追踪[5]、纹理切片[4]、shear-warp 和 shear-image rendering[6]、体 Splatting[2,7-8]. 高质量的 Splatting 必须同时考虑质量和效率. 特别地,Splatting 算法需要恰当的图像空间滤波以避免高频走样.

随着现代激光测距和光学扫描的普及,点模型逐渐成为基于网格的边界表示的一种强有力的补充. Splatting 是实时绘制大尺度点模型的主要方法. Zwicker 等提出了一种基于 EWA(椭圆加权滤波)的点模型 Splatting 算法. 原文方法未采用 GPU 加速计算,绘制速度很慢. 本文将在我们前面的工作[7]的基础上,提出一个同时适用于体模型和点模型的自适应 EWA Splatting 框架,在减少 EWA 计算量的同时,得到高质量的反走样绘制结果(图 1). 我们还将提出一个硬件加速 EWA 体 Splatting 框架,实现了交互速度的高质量体绘制、可交互的传输函数设计以及体-面混合 Splatting. 本文第 2 节讨论相关工作;第 3 节提出对点模型 Splatting 和体 Splatting 的自适应 EWA 重采样滤波;第 4 节介绍本文提出的硬件加速自适应 EWA Splatting 框架;第 5 节展示一些体数据和表面采样点数据的绘制结果;第 6 节总结本文的工作,并讨论未来的研究方向.

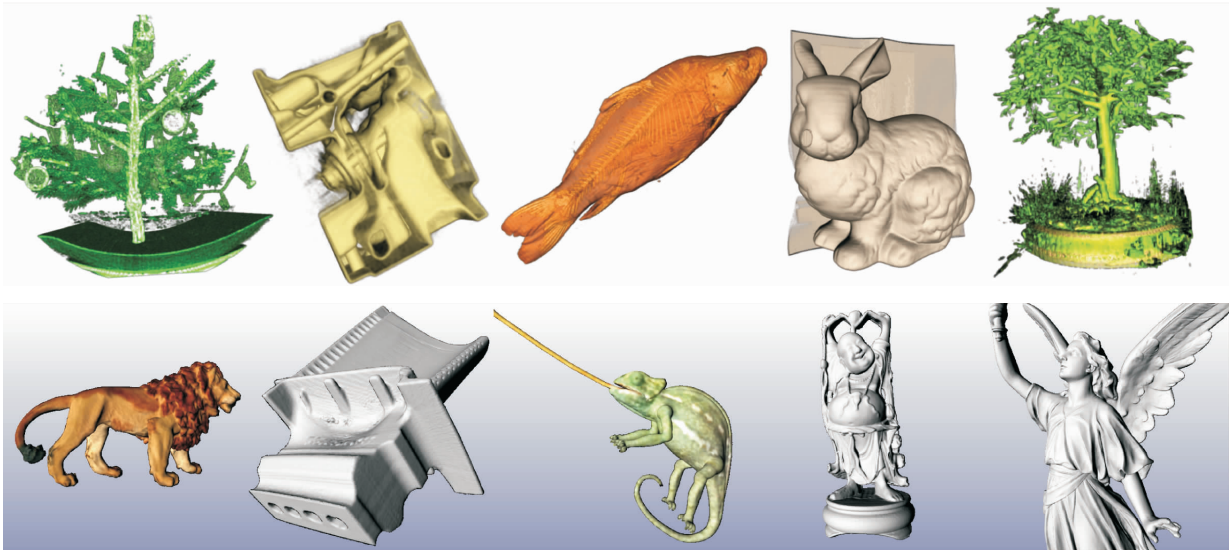


图 1(上列(从左到右):自适应 EWA 体 Splatting 绘制 C-tree(512^3),Engine($256^2\times110$),Carp($256^2\times512$),Bunny($512^2\times361$)和 Bonsai($512^2\times189$)的结果. 下列(从左到右):自适应 EWA 面 Splatting 对以下模型的绘制结果:Lion,183408 个点; Blade,1279519 个点; Chameleon,101685 个点; Buddha,1060220 个点; Lucy,10072906 个点)

2 相关工作

在本节中,我们总结规则体数据和点模型 Splatting 的工作以及它们的硬件实现.

2.1 体 Splatting

体 Splatting 灵活地支持体网格与点采样几何数据的混合. 从 Westover 的工作[1]起,大部分体 Splatting 算法都致力于改善图像质量,其中包括光线驱动的透视 Splatting[8]、边缘保持[9]、去除 popping 和模糊现象[10]. 体 Splatting 的走样问题由 Swan 等[11]和 Mueller 等[12]首先提出. 他们使用了距离相关的策略得到了类似低通滤波的结果. 文献[13]引入 EWA 滤波,避免了表面纹理走样. Zwicker 等[14]提出的 EWA Splatting 方法采用了与之相似的路线. Welsh 和 Mueller[15]使用了一种基于小波变换和预计算 Splat 基元的分层次算法.

标准的 GPU 加速 Splatting 方法[1]使用纹理映

射硬件来实现投影和扫描转换. Xue 和 Crawfis[16] 比较了一系列的硬件加速 Splatting 算法,但没有考虑反走样的问题. Chen 等[7] 提出了一种硬件加速自适应 EWA 体 Splatting 方法,减少了 EWA Splatting 的计算量,并在 GPU 上实现了整个流程. Neophytou 和 Mueller[17] 提出了一种用于图像对准的体 Splatting 的智能方法,这种方法在可编程图形硬件中实现,能够得到高质量图像和较高的帧率,并且还扩展到了对不规则体数据的处理. Vega-Higuera 等[2] 改进了文献[7]中提出的基于点 Sprite 基元的框架,并应用在神经血管的可视化上,但算法只采用了高斯重建滤波核,图像质量有所降低.

2.2　点模型 Splatting

自从点基元第一次提出[18],Splatting 就成为绘制点几何的普遍技术. QSplat 系统[19] 将多分辨率层次结构结合到点模型的处理中. 虽然它能达到交互级的帧率,但仅用一个正方形表示一个点使图像质量显著降低. 顺序点模型树(SPT)[20] 将文献[19]的层次结构转换成一个线性数组集,有利于高效图形硬件的实现.

Pfister 等[21] 提出以一个复杂的面元表示点. 为消除透视投影的走样,Zwicker 等提出 EWA 曲面重采样[22]. 但未优化的软件实现的 EWA 面 Splatting 算法速度极慢. Ren 等[23] 提出了一种世界空间描述的 EWA 面 Splats,并描述了图形硬件上的高效实现. 由于硬件限制,一个点需用一个四边形表示,使得每个点的 EWA 滤波计算需重复四次.

在使用硬件支持点 Sprite 基元的基础上,Botsch 和 Kobbelt[24] 在屏幕空间进行滤波. 为加快 Splat 计算,他们采用高斯滤波核,但也降低了图像质量. Guennebaud 和 Paulin[25] 实现了在屏幕空间的高质量 EWA 重采样滤波,并随后提出了多步 Splatting 方法[26]. Zwicker 等[27] 为了实现透视正确的 Splat 光栅化,提出了一种高效技术,使尖锐特征得以保留. Botsch 等[28] 扩展了文献[26]的思想,将之与 EWA 重采样滤波的简单近似相结合,每秒 2 千万以上个点,并得到较高质量. 最近,Weyrich 等[29] 提出了一种用于硬件体系结构获得了完全实时的点模型 Splatting,但没有采用 EWA 滤波.

2.3　基于 Splat 的硬件加速绘制的挑战

基于 Splat 的硬件加速绘制面临的主要挑战如图 2 所示.

在具体实现中,体 Splatting 与点模型 Splatting 有一些细微区别. 体 Splatting 在第 1 阶段判断出对

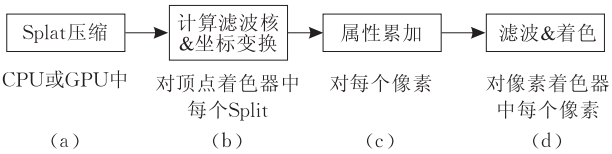


图 2　硬件加速 Splatting 的流水线模型

最终图像有贡献的体素,称作预分类过程. 所有的 Splat 都被组织为切片序列,并按轴平行[7] 或者图像平行[17] 逐片遍历. 在第 2 阶段,计算出每个 Splat 的位置和半径及其滤波核. 然后,记录下当前切片的贡献. 最后阶段进行滤波与逐像素的 shading 和 lighting. 而点模型 Splatting 在第 1 阶段以特定辅助数据结构组织 Splat,例如线性点模型树. 在第 2 阶段计算 Splat 的形状和滤波核之后,第 3 阶段中采用多步渲染技术来实现可见性计算,避免了不可见 Splat 的冗余计算. 最后,第 4 阶段采用了滤波计算和光照明计算.

相关文献表明,第 3 阶段和第 4 阶段中的硬件优化潜力已经被充分开发. 因此,本文将集中改进第 1 和第 2 阶段. 我们提出一种自适应的滤波模式以减少第 2 阶段中高质量重采样滤波核的计算量. 相应地,我们在第 1 阶段中提出一种基于聚类的自适应滤波核选择模式:将体素预排序成一个线性数组序列,并压缩存储在显存中,得到高效的交互式分类.

3　自适应 EWA 重采样滤波

我们的自适应 Splatting 方法基于 Zwichker 等[14] 提出的 EWA Splatting 方法. 因此,我们在 3.1 节中简单回顾这项技术. 在 3.2 节中将介绍我们的自适应 EWA 重采样滤波.

3.1　EWA 重采样滤波

Splatting 中,体素或点元用 3D 或 2D 重建核表示,通常选择椭圆高斯核 $Gv(t-p)$,其中心为 p,3×3 的矩阵 $\boldsymbol{V}$ 为其方差矩阵:

$$Gv(t-p)=\frac{1}{(2\pi)^{3/2}|\boldsymbol{V}|^{1/2}}\mathrm{e}^{-\frac{1}{2}(t-p)^{\mathrm{T}}\boldsymbol{V}^{-1}(t-p)} \quad (1)$$

在 Splatting 过程中,透视变换产生的采样率变化常导致走样问题. EWA Splatting 将投影后的核与 2D 低通滤波核进行卷积,生成 EWA 重采样滤波. 设物体空间映射到相机空间视点变换的旋转部分为 3×3 的矩阵 $\boldsymbol{W}$. 我们将相机空间坐标标记为 $\boldsymbol{u}=(u_0,u_1,u_2)$;初始的相机空间 $\boldsymbol{u}=\boldsymbol{0}$ 位于投影中心,图像平面为 $u_2=1$;体素 k 的相机空间坐标记为

$\boldsymbol{u}_k$；图像空间坐标标记为 x，体素 k 的图像空间位置为 x_k；进一步地，点 u_k 从相机空间到图像空间的透视投影的 Jacobian 矩阵为 3×3 的矩阵 $\boldsymbol{J}_{uk}$：

$$\boldsymbol{J}_{u_k}=\begin{pmatrix}\frac{1}{u_{k_2}} & 0 & -\frac{u_{k_0}}{u_{k_2}^2}\\ 0 & \frac{1}{u_{k_2}} & -\frac{u_{k_1}}{u_{k_2}^2}\\ \frac{u_{k_0}}{\|(u_{k_0},u_{k_0},u_{k_0})\|} & \frac{u_{k_1}}{\|(u_{k_0},u_{k_0},u_{k_0})\|} & \frac{u_{k_2}}{\|(u_{k_0},u_{k_0},u_{k_0})\|}\end{pmatrix} \tag{2}$$

给出物体空间中重建核的 3×3 方差矩阵，则其到图像空间的变换为 $\boldsymbol{V}_k=\boldsymbol{J}_{uk}\boldsymbol{W}\boldsymbol{V}_k''\boldsymbol{W}^{\mathrm{T}}\boldsymbol{J}_{uk}^{\mathrm{T}}$. EWA 体重采样滤波 $\rho_k(\boldsymbol{x})$ 为

$$\rho_k(\boldsymbol{x})=\frac{1}{2\pi|\boldsymbol{J}_{u_k}^{-1}|\ |\boldsymbol{W}^{-1}|\ |\boldsymbol{M}_k^{-1}|^{\frac{1}{2}}}\mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_k^{\mathrm{T}})\boldsymbol{M}_k(\boldsymbol{x}-\boldsymbol{x}_k)} \tag{3}$$

$$\boldsymbol{M}_k=(\hat{\boldsymbol{V}}_k+\boldsymbol{V}_k)^{-1} \tag{4}$$

其中，$\boldsymbol{V}_h$ 是高斯低通滤波的 2×2 方差矩阵，通常选择单位矩阵. 2×2 的方差矩阵 $\hat{\boldsymbol{V}}_k$ 由 $\boldsymbol{V}_k$ 删掉第 3 行和第 3 列得到①.

EWA 点重采样滤波可类似推出. 将点集标记为 $\{\boldsymbol{P}_k=(p_{k_0},p_{k_1},p_{k_2})\}_{k=1}^n$，局部参数化空间到世界空间的变换和从世界空间到相机空间的视点变换中的旋转部分分别记为 $\boldsymbol{S}_k$ 和 $\boldsymbol{W}$，将重建核与低通核的 2×2 方差矩阵分别标记为 $\boldsymbol{V}_k''$ 和 $\boldsymbol{V}_h$. $\boldsymbol{P}_k$ 的 EWA 点重采样滤波 $\rho_k(\boldsymbol{x})$ 为

$$\rho_k(\boldsymbol{x})=\frac{1}{2\pi|\boldsymbol{J}_{u_k}^{-1}|\ |\boldsymbol{W}^{-1}|\ |\boldsymbol{S}_k^{-1}|\ |\boldsymbol{M}_k^{-1}|^{\frac{1}{2}}}\mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_k^{\mathrm{T}})\boldsymbol{M}_k(\boldsymbol{x}-\boldsymbol{x}_k)} \tag{5}$$

$$\boldsymbol{M}_k=\boldsymbol{V}_k'+\boldsymbol{V}_h,\ \boldsymbol{V}_k'=\boldsymbol{J}_k\boldsymbol{W}\boldsymbol{S}_k\boldsymbol{V}_k''(\boldsymbol{J}_k\boldsymbol{W}\boldsymbol{S}_k)^{\mathrm{T}} \tag{6}$$

其中 $\boldsymbol{x}_k$ 是 $\boldsymbol{P}_k$ 的屏幕空间坐标，$\boldsymbol{J}_k$ 表示将 $\boldsymbol{P}_k$ 从相机空间映射到屏幕空间的变换 m_k 的 Jacobian 矩阵.

3.2 自适应 EWA 重采样滤波

EWA 重采样滤波减轻了走样问题，但其计算代价较高. 我们提出自适应 EWA Splatting 以简化计算，同时保持高质量结果. 分类标准如下[7]：

(1) 当体数据远离视点，仅采用低通滤波.

(2) 体数据靠近视点，仅采用重采样滤波.

(3) 体数据与视点距离在两者之间，采用 EWA 重采样滤波.

自适应 EWA 体重采样滤波

根据文献[7]的推导，我们有如下距离相关的自适应 EWA 体 Splatting 分类标准. 设重建核和低通滤波核的截断半径分别为 r_k 和 r_h，椭圆的长径和短径可分别由式(3)推出：

$$r_0=\sqrt{\frac{r_k^2}{u_{k_2}^2}+r_h^2},\ r_1=\sqrt{\frac{r_k^2(u_{k_0}^2+u_{k_1}^2+u_{k2}^2)}{u_{k_2}^4}+r_h^2} \tag{7}$$

距离相关的 EWA 滤波公式如下，其中 $c_{\min}$ 和 $c_{\max}$ 为控制参数

$$\begin{cases}\boldsymbol{H}_k(\boldsymbol{x})=\boldsymbol{x}^{\mathrm{T}}\cdot\hat{\boldsymbol{V}}_k^{-1}\cdot\boldsymbol{x}, & u_{k_2}<\frac{r_k}{r_h}\times c_{\min}\\ \boldsymbol{H}_k(\boldsymbol{x})=\boldsymbol{x}^{\mathrm{T}}\cdot\boldsymbol{V}_h^{-1}\cdot\boldsymbol{x}, & u_{k_2}>\frac{r_k}{r_h}\times c_{\max}\\ \boldsymbol{H}_k(\boldsymbol{x})=\boldsymbol{x}\cdot(\hat{\boldsymbol{V}}_k+\boldsymbol{V}_h)^{-1}\cdot\boldsymbol{x}, & \text{其它}\end{cases} \tag{8}$$

自适应 EWA 点重采样滤波

EWA 点重采样滤波的近似同样取决于重采样滤波与低通滤波之间的平衡. 同时，重采样滤波核低通滤波的影响可以由 $|\boldsymbol{V}_k'|^{\frac{1}{2}}$ 和 $|\boldsymbol{V}_h|^{\frac{1}{2}}$ 唯一确定(见式(6)). 当重建核与低通核的截断半径分别为 r_k 和 r_h 时，由式(6)有

$$\begin{aligned}|\boldsymbol{V}_k'|^{\frac{1}{2}}&=\frac{A_k}{\pi}\sqrt{\left(\frac{u_{k_0}}{u_{k_2}}\right)^2+\left(\frac{u_{k_1}}{u_{k_2}}\right)^2+1}\\&\le\frac{A_k}{\pi}\sqrt{2\cdot\tan^2\left(\frac{fov}{2}\right)+1}\end{aligned} \tag{9}$$

其中 $(u_{k_0},u_{k_1},u_{k_2})$ 为 $\boldsymbol{P}_k$ 的相机空间坐标，$A_k=\frac{\pi r_k^2\cos\theta_k}{u_{k_2}^2}$ 为重建滤波的影响区域面积，θ_k 为 $\boldsymbol{P}_k$ 的点法向与向量 $(u_{k_0},u_{k_1},u_{k_2})$ 的夹角. 低通滤波 h 的有效面积为 $A_h=\pi\cdot r_h^2$，A_k 与 A_h 之比决定了自适应 EWA 点 Splatting：

$$\begin{cases}\boldsymbol{H}_k(\boldsymbol{x})=\boldsymbol{x}^{\mathrm{T}}\cdot(\boldsymbol{V}_k')^{-1}\cdot\boldsymbol{x}, & u_{k_2}<\frac{r_k\sqrt{\theta_k}}{r_h}\times c_{\min}\\ \boldsymbol{H}_k(\boldsymbol{x})=\boldsymbol{x}^{\mathrm{T}}\cdot\boldsymbol{V}_h^{-1}\cdot\boldsymbol{x}, & u_{k_2}>\frac{r_k\sqrt{\theta_k}}{r_h}\times c_{\max}\\ \boldsymbol{H}_k=\boldsymbol{x}\cdot(\boldsymbol{M}_k)^{-1}\cdot\boldsymbol{x}, & \text{其它}\end{cases} \tag{10}$$

其中，$c_{\min}$ 与 $c_{\max}$ 参数用于调节质量与效率之间的平衡.

4 EWA Splatting 硬件加速框架

自适应 EWA Splatting 可以处理点采样曲面和规则或不规则体数据. 为降低计算代价，待处理的数据需要用特殊数据结构组织成面片或者块，然后计算各块的滤波策略并应用到相应面片或者块中的所

① 本文中，用头标标注的矩阵表示由原矩阵删掉第 3 行和第 3 列得到.

有体素上.在本节中,我们叙述如何将自适应 EWA Splatting 方法加入到硬件加速的 Splatting 框架中.

4.1　自适应 EWA 体 Splatting

我们用一个点 Sprite 基元(代理几何)表示一个 Splat,将 Splat 几何及相关属性压缩到顶点缓存集.根据视点方向选择最接近的主轴,逐层绘制与主轴正交的切片中的 Splat.每个 Splat 的数据解压缩及其滤波核、分类和照明的计算都在顶点处理步骤进行.其次,EWA 滤波在像素处理阶段中逐段完成.将同一个切片中 Splat 的贡献累加到一个中间缓存,然后从前到后地将各层切片合成到一个累加缓存,得到最终结果.我们的方法有 4 个创新点:(1)提出一个硬件实现的自适应 EWA Splatting 策略,在保持高图像质量的同时,得到比标准 EWA Splatting 更高的绘制性能;(2)采用一种将代理几何(例如表示 Splat 的点 Sprite)和体属性数据存储在本地图形硬件上的保留模式,避免了立即模式算法将每个 Splat 分别送到绘制流水所遇到的 CPU 与 GPU 之间的带宽瓶颈.我们将 Splat 几何与属性压缩到顶点缓存中的策略,在内存消耗与计算性能上,都要胜过其它的数据存储模型.(3)提出的基于列表的预分类技术,允许交互式分类.(4)提出一种新的自适应累加模式,避免了整点数帧缓存精度不足造成的颜色量化失真现象,极大地改善了整体性能.

4.1.1　自适应 Splat 计算

我们将每个切片的四边形按矩形面片分组.Splatting 时,动态地计算每个面片 4 个角上体素的相机空间坐标 u_{k_2} 并根据式(8)进行评估.如 4 个顶点都符合上限标准,则采用重建滤波,如都符合下限标准,则采用低通滤波,否则采用完全的 EWA 重采样滤波.在滤波核与可编程顶点渲染器中所有 Splat 的分类和照明计算完成后,颜色、透明度和滤波核相关系数都被送往像素处理阶段.EWA 滤波在可编程像素渲染器中执行.每个 Splat 的滤波核只需计算一次,其性能远高于纹理四边形表示[7](数据见表 4).

4.1.2　数据压缩和解压缩

无压缩数据需要很大的内存空间.以尺寸为 256^3 的体数据为例,所有体素的标量密度和梯度向量需要 64 兆字节.加上代理几何,用一个点 Sprite(显存中的一个点)表示一个体素,共需 16 兆顶点.每个点 Sprite 包括位置(3 个浮点数,共 12 字节)、体纹理坐标(3 个浮点数,共 12 字节),共 24 字节.此外,轴平行 Splatting 的 3 种遍历顺序需要将所有点 Sprite 的代理几何存储三次,所以每个体素的代理几何需要 72 字节.总内存需求是 1216 兆字节.我们的数据压缩和解压缩策略利用了线性或规则体数据的规则性,将代理几何与体属性存储在顶点缓存中,保存每个 Splat 的属性,即原始灰度值和梯度向量,需 4 字节.此外,需 4 字节保存体素的景物空间位置,将其 3 个坐标值变换到体纹理空间并标记为 t_x,t_y 和 t_z 3 个整数,以图 3(a)中描述的(i_x,i_y,i_z,$m_x\times32+m_y\times4+m_z$)形式保存为 4 个 8 位无符号整数.因此最大内存需求为 $256^3\times8\times3=384$ 兆字节.但通常只有 5%～20%的不透明体素需绘制.处理每个顶点的第 1 步是从压缩的数据中恢复 Splat 几何.我们用一个小查找表找出 m_x,m_y 和 m_z.然后用向量(t_x,t_y,t_z)来计算体素中心的景物空间坐标.整个解压缩过程可在可编程顶点渲染器中执行.

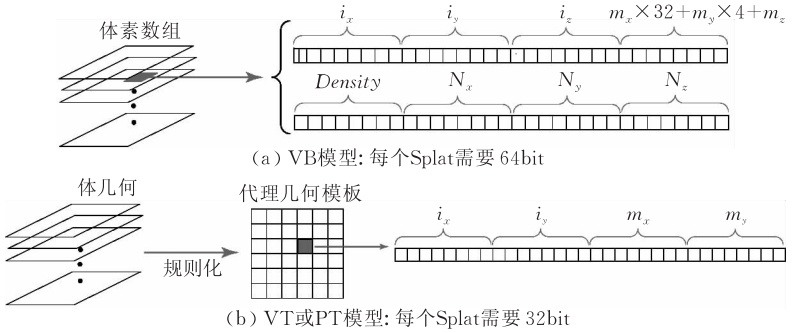


图 3　VB、VT 和 PT 模式在显存中的数据压缩

另一种方法是将代理几何存储在顶点缓存中,并将体属性压缩成体纹理.体属性可在顶点处理阶段或者在像素处理阶段读取.这里可进一步利用代理几何的规则性,因为体属性和代理几何是分开存储的.因为连续两层之间的体纹理坐标差值是一个常数,一层代理几何信息可为体中所有层共享,这层代理几何被称为代理几何模板(见图 3(b)).同一层中,沿遍历方向的体纹理坐标不会变化,这就只需两

个纹理坐标值.沿遍历方向,从一层到下一层只需将纹理坐标增加一个常量.共需保存 3 个遍历顺序的代理几何模板,因此 256^3 大小的体数据只需保存 19.2 兆顶点.Splat 位置的编码方法十分相似:将体纹理坐标 t_x 和 t_y 保存为 4 个 8bit 整数 i_x,i_y,m_x,m_y.其中,$m_x=t_x/256$,$i_x=t_x \bmod 256$,t_y 与之类似.这样,256^3 大小体数据集的代理几何内存总消耗仅为 76.8 兆字节($256\times256\times4\times3$).

为简单起见,将我们的方法和其他两种方法分别标记为 VB、VT 和 PT 模式.我们的方法并不像光线跟踪方法和纹理切片方法那样需要体纹理存取,这在如下 3 点上胜过其它两种模式:首先,没有硬件纹理映射功能限制,例如对体纹理或大规模体纹理的支持.其次,在图形硬件中体纹理映射的性能可能低于 2D 纹理映射或者顶点处理.在顶点缓存中存储体属性并在顶点处理阶段读取,比使用体纹理的方法要快得多.最后,我们的方法与 Splatting 的优点自然相符:只有相关的 Splat 及其属性才会保存在显存中.因为 Splatting 通常用于处理大规模稀疏体数据,虽然我们的方法存储了相关体素体属性的 3 份拷贝,但并未占用大量内存;相反,尽管 VT 和 PT 模式只用到了一份体纹理数据,但处理大规模体数据还是需要大量内存.

4.1.3 交互式分类

交互式分类是体传输函数应用的关键步骤.后分类策略是指先绘制所有相关体素,再进行分类,它比预分类策略要慢得多.预分类策略是指在绘制前挑选出透明体素,通常占总数的 5%～20%.在我们保留模式的硬件实现中,预分类需要收集不透明体素的顶点缓存,并在传输函数变化时将之载入图形硬件.顶点缓存的构造十分耗时,且将它们载入到显存中涉及到从 CPU 到 GPU 的大量数据传输,因此不能实时地改变传输函数.

我们建立了一个如图 4 所示的辅助数据结构来解决这个问题.基本思路基于 Mueller 等提出的基于列表的 Splatting 算法[10].首先基于体素的亮度值对体素进行桶排序.在文献[10]中,建立了整体的等值列表.不同的是,我们计算的是每个切片每个遍历方向的等值列表.所有的顶点缓存都跨越了等值体素的 256 个间隔.这些缓存被预载入到图形硬件中用于交互式绘制之前,被相应地排序并重新整理到等值顶点缓存中.在实际操作中,它们根据大小存储在显存或内存中.指向每个等值数组开头的指针被保存在主存中.当传输函数被交互改变时,对应的指针被迅速地收集和合并,以将可见的体素送往绘制流水线.

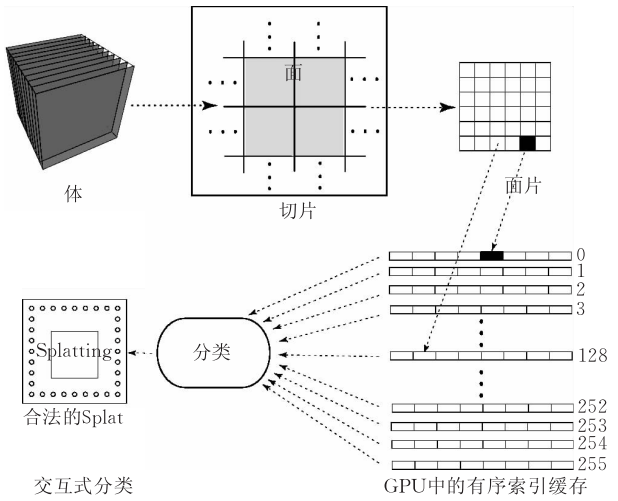


图 4 交互式分类的辅助数据结构(为了得到最佳性能,每个面片中的顶点通常不超过 65535 个)

4.1.4 混合体和点 Splatting

一般来说,体 Splatting 产生的图像比 Splatting 模糊.因为后者将重建核的形状与点法向结合,保留了更多尖锐特征.Zwicher 等引入了一种直接从体数据中提取和绘制点的方法[14],将体重建核扁平化为点重建核.类似地,自适应 EWA 体重采样滤波也可推导出一种自适应 EWA 扁平化点采样滤波.它采用与式(10)相同的公式.其硬件实现与自适应 EWA 体 Splatting 的硬件实现完全兼容.唯一的区别是式(5)、(6)和式(10)中的 Jacobian 矩阵、滤波核与自适应滤波的计算.

此外,面-体混合 Splatting 可通过在同一体数据中对不同部分分别采用体滤波核与扁平化面滤波核实现.例如,半透明体素可用自适应 EWA 体重采样滤波绘制,而不透明度等值面体素可以用扁平化面重采样滤波处理,从而得到更精确的图像.

4.1.5 自适应浮点精度

为累计出所有 Splat 的贡献,在自适应 EWA 体 Splatting 流水中采用了两种累加操作(见图 5).虽然顶点与像素处理中的算术计算都是浮点精度实现,但图形硬件中的累加缓存往往只有有限的颜色深度,例如每个颜色通道 8 位.当距离大到一定程度(见式(3)和式(7)),因为精度不足,一个单独的体素对一个像素的贡献被量子化为 0,导致严重的走样现象(见图 6(c)).最近图形硬件的 16 位浮点混合属性成功地消除了图 6(d)所示的绘制走样,但同时

也牺牲了性能,性能下降幅度与 Splat 数目成正比.在我们的实验条件下,以 512×512 分辨率绘制一个包含 7 万个体素,缓存中有 256 个厚片的场景,16 位浮点与 8 位整点数混合模型分别需要 10ms 和 5ms.以 256×256 分辨率绘制,则分别需要 6ms 和 4ms.图 6(a)~(d)的 fps 分别为 19.25、9.90、32.10 和 13.16.另一方面,如图 6(a)、(b)所示,当视点在适中范围内,使用 8 位整点数缓存并不会产生走样.因此在这种情况下浮点精度并非必要.可见从视点到体数据的距离决定了是否必需浮点精度.

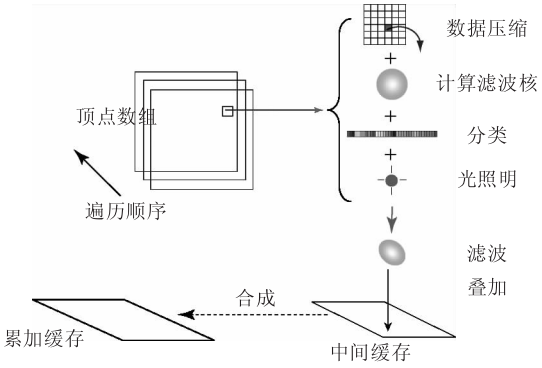
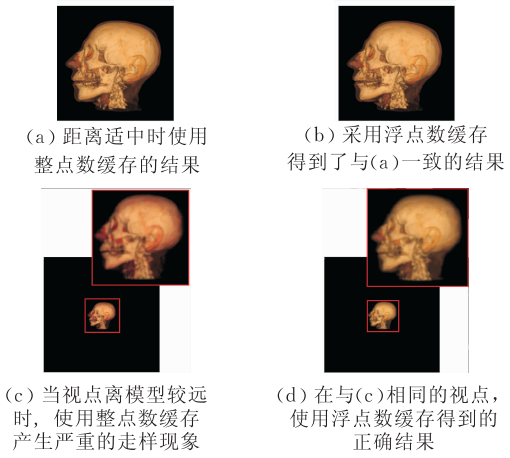


图 5　硬件加速 EWA 体 Splatting 流程



(a) 距离适中时使用整点数缓存的结果　(b) 采用浮点数缓存得到了与(a)一致的结果

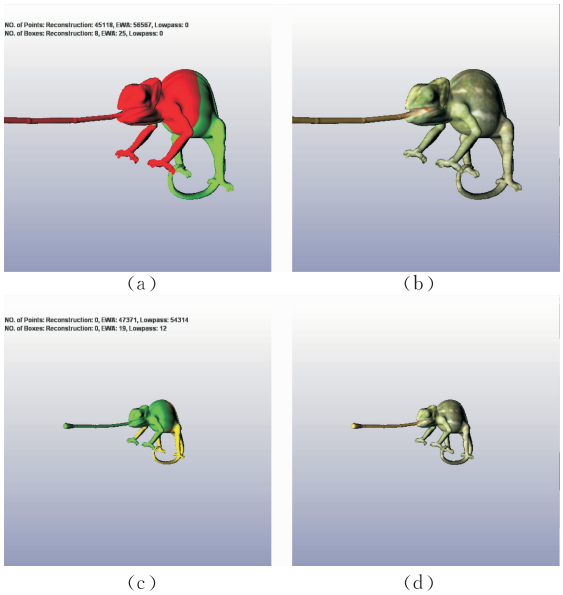
(c) 当视点离模型较远时,使用整点数缓存产生严重的走样现象　(d) 在与(c)相同的视点,使用浮点数缓存得到的正确结果

图　6(注:(c)和(d)的右上角分别是放大图)

基于此,我们提出一种简单有效的自适应浮点精度策略.在 Splatting 前,预构造待处理数据的一个稍大的包围盒和整数点与浮点精度.基于整体考虑选择合适的精度选择:若视点到包围盒的距离大于$\frac{r_h}{r_k}\times c_{acc}$,则选择浮点精度.反之则选择整数精度.通常阈值 $c_{acc}=c_{max}\times 0.3$.对每一帧,我们计算包围盒在图像平面上的投影几何,并将之重构为一系列三角形作为代理几何.精度模式选择和紧密代理几何计算都在 CPU 中完成,只需很低的代价.事实上,浮点精度只在视点离体数据很远时才在一个很小的区域内使用,因此绘制质量和性能可同时保证.

4.2　自适应 EWA 点 Splatting

硬件加速点 Splatting 框架包括 3 个步骤:可见性计算、属性累加和最后的 shading.高效的自适应滤波是本文方法的关键部分.注意到我们的自适应滤波分类策略可在任意的层次化数据结构上进行,例如八叉树.实际计算中,为每个点采样模型建立一棵有向包围盒树(OBB 树)[30],并逐层进行动态自适应滤波.在 OBB 树建立后,属于叶节点的采样点都被归类到顶点数组中.这个顶点数组的每个元素都以一个点 Sprite 及其点属性表示.以广度优先顺序将所有非根节点递归地合并到父节点,得到一个线性根节点顶点数组.然后将这个顶点数组以一块或多块顶点缓存的形式预装入显存.每个 OBB 树的包围盒以及指向顶点数组对应区间的指针被保存在主存中.在 Splatting 过程中,递归地计算当前节点包围盒顶点的相机空间坐标 u_{k_2}.与自适应 EWA 体 Splatting 类似,自适应阈值由式(10)求得.节点分类后,用对应的滤波核来绘制该节点所对应的顶点缓存区域内的所有点.式(10)中的控制参数 c_{min} 和 c_{max} 分别设为 0.3 和 $2.0\times(1.0+2.0\times\tan(fov/2)^2)$.图 7(a)和(c)描述了在近视点和远视点下,Chameleon 模型的滤波分类情况.相应的自适应 EWA 面 Splatting 绘制结果见图 7(b)和(d).



(a)　(b)　(c)　(d)

图 7　Chameleon 模型(共 101685 个点)的自适应 EWA 面滤波分类图示(图(a)靠前部分采用重建滤波,尾部采用 EWA 滤波;图(c)靠前部分采用 EWA 滤波,尾部采用低通滤波;图(b)和(d)分别显示了图(a)和(c)的绘制效果)

5 实验结果

我们在一台配备 Intel P4 2.4 GHz CPU、NVidia 6800 GT 显卡、2GB 内存的微机上进行了实验. 图形硬件加速是基于 DIRECTX 9.0c SDK 实现的. 报告中所有图片的分辨率都是 512×512. 我们硬件实现的体 Splatting 集成了一些常用的硬件优化技术，例如早期 z-测试、延迟绘制(deferred shading)和透明度裁剪(opacity-culling)等. 例如，我们对每 4～6 个切片做一次早期 z-culling，一般可以使算法性能提高 20%～50%.

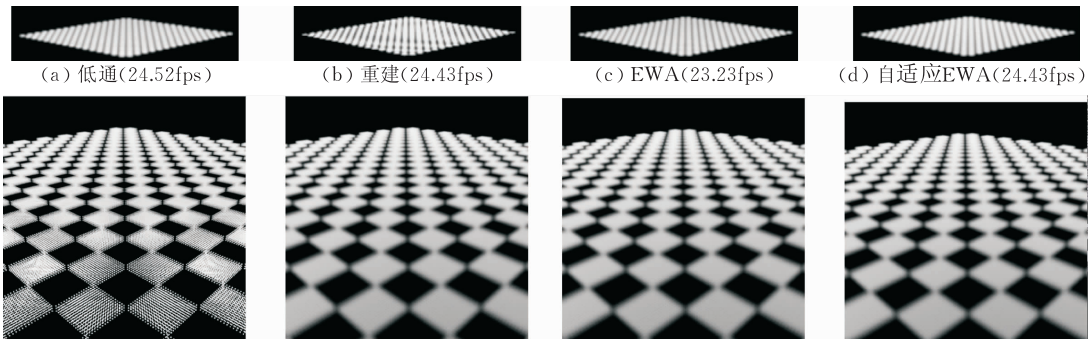
自适应 EWA 体采样滤波和面采样滤波都采用了 3 个不同的顶点着色器. 自适应策略的主要效率提升来自于缩小或放大时的滤波计算简化. 此外，因为重建滤波和低通滤波的 footprint 大小比 EWA 重采样滤波的 footprint 小，自适应 EWA 重采样滤波需要更少的光栅和像素处理，从而得到更高的帧率. 表 1 分别列出了体采样和面采样的顶点着色指令数.

表 1 顶点着色器的指令数目

数据类型	EWA	重建滤波	低通滤波
规则体	42	32	28
Rectilinear volume	46	32	28
点采样表面	49	40	32

我们比较了 VB、VT 和 PT 模式，IM(原始的立即模型)和 QD(四叉树表示)[7]的性能与显存耗费. EWA 体 Splatting 处理不同数据集的结果见表 4. 四叉树表示的瓶颈主要在于顶点处理阶段，因为每个体素都必须处理 4 个顶点. VB、VT 和 PT 模型采用了点 sprite 基元，克服了这一困难. 在三者中，VB 模型性能最佳并且需要的显存最少. 我们使用的 NVidia6800 GT 显卡的最大体纹理分辨率仅为 256^3，所以 QD、VT 和 PT 模型不能适用于 Bonsai、Carp、Bunny 和 C-tree① 数据集. 在 IM 模式下，每一个 Splat 都分别送到图形流水线，这明显意味着瓶颈在于 CPU 与 GPU 之间的带宽，而延迟模式绘制技术(QD、VB、VT 和 PT)具有显著的性能提升. 在接下来的实验中，所有的数据集都在 VB 数据存储模式下处理.

我们以 512×512×3 的棋盘格模型为例说明自适应 EWA 体 Splatting 的效率和质量，其中所有的体素都设置为不透明. 图 8(b)和(c)比较了采用重建滤波的 Splatting 和采用 EWA 滤波的图像质量. 同样地，图 8(e)和(g)分别表示采用低通滤波和 EWA 滤波的图像结果. 通过这些比较可以清楚地看到，采用不恰当的滤波核，即使是更高效的滤波核，也会导致结果的走样. 采用 EWA 体重采样滤波的 Splatting 纠正了这些错误，但计算代价很大. 自适应 EWA Splatting 在大幅降低计算代价的情况下，得到了可与全部采用 EWA 滤波的方法相比的图像质量，图片见图 8(d)和(h).



(a) 低通(24.52fps) (b) 重建(24.43fps) (c) EWA(23.23fps) (d) 自适应EWA(24.43fps)

(e) 低通(14.63fps) (f) 重建(13.80fps) (g) EWA(12.85fps) (h) 自适应EWA(13.30fps)

图 8 使用不同滤波核的自适应 EWA 体 Splatting 绘制 $512^2\times3$ 棋盘格模型图示

表 2 列出了自适应 EWA Splatting 策略(标记为 A-F 和 A-N)相对 EWA 体 Splatting 方法(标记为 E-F 和 E-N)的性能提升. 第 2 列和第 3 列给出了当视点较近时两种方法的帧率. 第 5 和第 6 列给出了当视点较远时的帧率. 第 4 列和最后一列给出了在两种情况下，自适应 EWA 滤波方法的性能提升. 绘制结果见图 1 上.

我们实现了一个简单的基于列表的传输函数预分类用户界面(图 9(g)). 用户可以交互地调整 R、G、B 和 α 颜色通道对应的 4 条曲线，曲线的变化被转换到一个位于显存的一维转换函数查找表，绘制结果也会即时更新. 当传输函数不变时，我们分类策

① Engine, Tree, Head, Carp, Bonsai, Bunny 和 C-tree 数据集的体分辨率分别为 $256^2\times110$, $256^2\times128$, $208\times256\times225$, $256^2\times512$, $512^2\times189$, $512^2\times361$ 和 512^3.

略的绘制速度几乎与预分类一样.图 9(a)～(f)给出了交互式分类时头模型的一系列绘制结果.(a)～(f)的帧率分别为8.4、20.06、20.10、20.06、20.06和19.30fps.未使用该策略时,更新一个 256^3 的体数据集的顶点缓存需要 3s.

表 2　自适应 EWA 体 Splatting 与 EWA 体 Splatting 的 fps 效率比较

数据集	fps 性能					
	E-N	A-N	提升/%	E-F	A-F	提升/%
Engine	18.20	19.10	5	43.15	49.05	14
Tree	32.05	34.10	6	61.82	68.65	11
Head	15.05	15.97	6	31.05	35.15	13
Bonsai	12.85	13.60	7	16.06	18.10	11
Carp	16.60	18.03	8	26.32	29.53	12
Bunny	2.60	2.86	10	2.80	3.20	14
C-tree	3.98	4.41	11	4.50	5.77	28

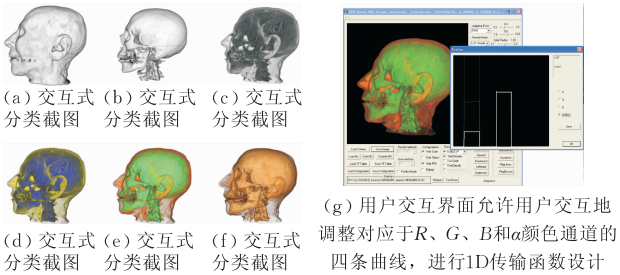


图 9　基于列表的交互式预分类图示

图 10(a)和(b)比较了 EWA 体 Splatting 和 EWA 点模型 Splatting.图 10(d)展示了用 EWA 体-面混合 Splatting 绘制一个 64^3 箱盒模型的结果,与之对照的是图 10(c)展示的 EWA 体 Splatting 的绘制结果.

我们在表 3 和图 1 下中总结了自适应面 Splatting 对于一系列点采样模型的绘制性能和图像质量.Chameleon、Lion、Dragon、Buddha、Blade 和 Lucy 模型的点数分别为 101685、183408、882954、1060220、1279519 和 10072906.自适应面 Splatting 得到了几乎相同的图像质量和 10%的性能提升.

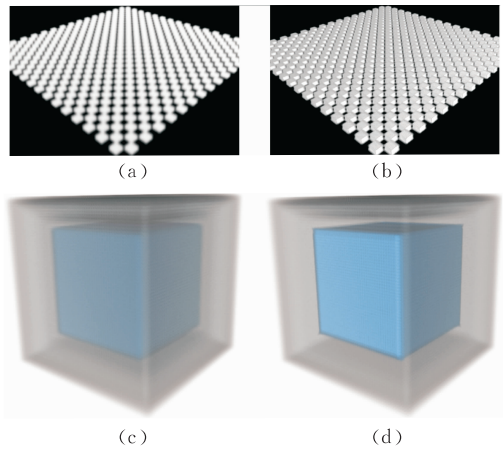


图 10　点 Splatting 与 EWA 体-面混合 Splatting 图示

表 3　适应 EWA 面 Splatting 与 EWA 面 Splatting 的 fps 性能比较

模型	fps 性能		
	EWA	Adaptive	提升/%
Chameleon	58.00～174.00	60.00～202.00	4～16
Lion	90.00～101.00	93.00～119.00	3～18
Dragon	16.20～16.48	17.13～19.10	6～16
Buddha	16.50～19.84	17.30～22.91	5～15
Blade	16.73～23.80	17.50～27.50	5～16
Lucy	1.66～1.67	1.78～1.89	7～14

表 4　EWA 体 Splatting 与 QD、VB、VT、PT 和 IM(立即模型)的性能(第 3～7 列)与显存消耗(第 8～12 列)的对比(因为显卡体纹理分辨率的限制,QD、VT 和 PT 模型不能处理 Bonsai、Carp、Bunny 和 C-tree 模型)

数据集	#splat	QD	VB	VT	PT	IM	QD	VB/M	VT/M	PT/M	IM/M
Engine	594430	11.21	27.43	21.64	15.66	2.85	78.8	14.3	36.0	36.0	14.3
Tree	274866	19.00	39.00	33.39	28.55	4.53	56.6	6.6	36.9	36.9	6.6
Head	693032	8.70	19.25	6.80	11.29	2.48	106.1	16.6	56.2	56.2	16.6
Bonsai	1445577	—	15.35	—	—	1.78	252.2	15.4	205.9	205.9	15.4
Carp	642659	—	21.94	—	—	2.30	256.6	34.7	151.6	151.6	34.7
Bunny	9449256	—	2.75	—	—	0.30	1172.2	226.8	491.9	491.9	226.8
C-tree	6893663	—	4.12	—	—	0.37	1115.9	165.4	619.6	619.6	165.4

6　结论和未来的工作

本文提出了一种适用于直接体绘制和点模型绘制的硬件加速自适应 EWA Splatting 算法.实验证明,我们的方法可得到高质量无走样图像,需要的计算量比完全的 EWA Splatting 更少.我们在硬件加速的体 Splatting 框架中采用自适应 EWA Splatting 策略,该框架的关键特性包括高效代理几何压缩与快速解压缩、对交互式分类的支持、体-面混合绘制以及自适应浮点精度缓存.我们还实现了一个自适应 EWA 面采样 Splatting 算法.

当切片的方向突变时,基于切片的结构会带来 popping 问题.Mueller 等[10]提出图像平面平行的核切片和遍历顺序.注意到我们的框架与遍历方式无关,因此可直接整合图像平面平行的方法.此外,我

们希望在硬件加速框架下,将我们的自适应 EWA Splatting 策略应用到不规则数据上.

参 考 文 献

[1] Westover L. Footprint evaluation for volume rendering//Proceedings of the ACM SIGGRAPH. Dallas, TX, USA, 1990: 367-376

[2] Vega-Higuera F, Hastreiter P, Fahlbusch R, Greiner G. High performance volume splatting for visualization of neurovascular data//Proceedings of the 2005 IEEE Visualization Conference. Minneapolis, MN, USA, 2005: 271-278

[3] Levoy M. Display of surfaces from volume data. IEEE Computer Graphics & Applications, 1988, 8(5): 29-37

[4] Engel K, Kraus M, Ertl T. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading//Proceedings of the 2001 ACM SIGGRAPH/Eurographics Workshop on Graphics Hardware. Los Angeles, CA, USA, 2001: 9-16

[5] Kruger J, Westermann R. Acceleration techniques for gpu-based volume rendering//Proceedings of the IEEE Visualization 2003. Seattle, WA, USA, 2003: 38-44

[6] Pfister H, Hardenbergh J, Knittel J, Lauer H, Seiler L. The VolumePro real-time ray-casting system//Proceedings of the ACM SIGGRAPH 1999. Los Angeles, CA, USA, 1999: 251-260

[7] Chen W, Ren L, Zwicker M, Pfister H.. Hardware-accelerated adaptive EWA volume splatting//Proceedings of the IEEE Visualization 2004. Austin, TX, USA, 2004: 67-74

[8] Mueller K, Yagel R. Fast perspective volume rendering with splatting by utilizing a ray driven approach//Proceedings of the 1996 IEEE Visualization Conference. Ottawa, Canada, 1996: 65-72

[9] Huang J, Crawfis R, Stredney D. Edge preservation in volume rendering using splatting//Proceedings of the 1998 IEEE symposium on Volume visualization. North Carolina, USA, 1998: 63-69

[10] Mueller K, Shareef N, Huang J, Crawfis R. High-quality splatting on rectilinear grids with efficient culling of occluded voxels. IEEE Transactions on Visualization and Computer Graphics, 1999, 5(2): 116-134

[11] Swan J E, Mueller K, Möller T, Shareef N, Crawfis R, Yagel R. An anti-aliasing technique for splatting//Proceedings of the 1997 IEEE Visualization Conference. Phoenix, AZ, USA, 1997: 197-204

[12] Mueller K, Moeller T, Swan J, Crawfis R, Shareef N, Yagel R. Splatting errors and antialiasing. IEEE Transactions on Visualization and Computer Graphics, 1998, 4(2): 178-191

[13] Heckbert P. Fundamentals of texture mapping and image warping[Ph. D. dissertation]. Department of Electrical Engineering and Computer Science, Berkeley, 1989

[14] Zwicker M, Pfister H, Baar J V, Gross M. EWA splatting. IEEE Transactions on Visualization and Computer Graphics, 2002, 8(3): 223-238

[15] Welsh T, Mueller K. A frequency-sensitive point hierarchy for images and volumes//Proceedings of the 2003 IEEE Visualization Conference. Seattle, USA, 2003: 56

[16] Xue D, Crawfis R. Efficient splatting using modern graphics hardware. Journal of Graphics Tools, 2004, 8(3): 1-21

[17] Neophytou N, Mueller K. GPU accelerated image aligned splatting//Proceedings of the International Workshop on Volume Graphics 2005. New York, USA, 2005: 197-205

[18] Levoy M, Whitted T. The use of points as display primitives. CS Department, University of North Carolina, Chapel Hill: Technical Report 85-022, 1985

[19] Rusinkiewicz S, Levoy M. QSplat: A multiresolution point rendering system for large meshes//Proceedings of the ACM SIGGRAPH 2000. Phoenix, AZ, USA, 2000: 343-352

[20] Dachsbacher C, Vogelgsang C, Stamminger M. Sequential point trees//Proceedings of the ACM SIGGRAPH 2003. Las Vegas, NV, USA, 2003: 285-288

[21] Pfister H, Zwicker M, van Baar J, Gross M. Surfels: Surface elements as rendering primitives//Proceedings of the ACM SIGGRAPH 2000. New Orleans, Louisiana, USA, 2000: 335-342

[22] Zwicker M, Pfister H, Baar J V, Gross M. Surface splatting//Proceedings of the ACMSIGGRAPH 2001. Los Angeles, CA, USA, 2001: 371-378

[23] Ren L, Pfister H, Zwicker M. Object-space EWA surface splatting: A hardware accelerated approach to high quality point rendering//Proceedings of Eurographics 2002. Saarbrücken, Germany, 2002: 461-470

[24] Botsch M, Kobbelt L. High-quality point-based rendering on modern GPUs///Proceedings of the Pacific Graphics 2003. Alberta, Canada, 2003: 335-343

[25] Guennebaud G, Paulin M. Efficient screen space approach for hardware accelerated surfel rendering//Proceedings of the Vision, Modeling and Visualization Conference. Germany, 2003: 1-10

[26] Guennebaud G, Barthe L, Paulin M. Deferred splatting. Computer Graphics Forum, 2004, 23(3): 653-660

[27] Zwicker M, Räadsädnen J, Botsch M, Dachsbacher C, Pauly M. Perspective accurate splatting//Proceedings of the Graphics Interface 2004. London, Ontario, Canada, 2004: 247-254

[28] Botsch M, Hornung A, Zwicker M, Kobbelt L. High-quality surface splatting on today's GPUs//Proceedings of the Eurographics Symposium on Point-Based Graphics. Long Island, New York, USA, 2005: 17-24.

[29] Weyrich T, Heinzle S, Aila T B, Fasnacht D, Oetiker S, Botsch M, Flaig C, Mall S, Rohrer K, Felber N, Kaeslin H, Gross M. A hardware architecture for surface splatting. ACM Transactions on Graphics, 2007, 26(3): 90

[30] Gottschalk S, Lin M C, Manocha D. OBBTree: A hierarchical structure for rapid interference detection//Proceedings of the 1996 ACM SIGGRAPH. New Orleans, LA, USA, 1996: 171-180

CHEN Wei, born in 1976, Ph. D., associate professor. His current research interests focus on visualization.

XIA Jia-Zhi, born in 1984, M. S.. His current research interest is visualization.

ZHANG Long, born in 1981, Ph. D.. His current research interests include photo-realistic large-scale forest rendering and volume rendering.

YU Yang, born in 1984, M. S. candidate. His current research interest is visualization.

ZHENG Wen-Ting, born in 1974, Ph. D., associate professor. His current research interests include computer graphics, programmable graphics hardware accelerate.

PENG Qun-Sheng, born in 1947, Ph. D., professor, Ph. D. supervisor. His current research interests include photo-realistic graphics, digital geometry processing, virtual reality, infrared imaging simulation and biological computing.

Background

To interactively process large-scale point clouds, a high efficient point rendering method has to be exploited. Most efforts have focused on direct display of point samples without connectivity, of which splatting is the dominant solution. As splatting is concerned, two aspects must be considered. On one hand, high performance is a major consideration for many time-critical applications. On the other hand, visually plausible image quality requires special cares because Splatting processes the rendering primitives in an individual manner. This paper is following the authors' former work. It presents a hardware-accelerated EWA volume splatting framework that allows interactive high quality volume rendering, interactive transfer function design, and hybrid surface-volume splatting. The work is supported by the National Natural Science Foundation of China under grant No. 60503056.