基于多分辨率裁剪纹理的体裁剪技术

罗月童[1]， 薛 晔[1]， 龙鹏程[2]， 伍国永[1]

（1. 合肥工业大学计算机与信息学院 VCC 研究室，安徽 合肥 230009;
2. 中国科学院等离子体物理研究所，安徽 合肥 230031）

摘 要：为解决单分辨率裁剪纹理的数据规模过大的问题，该文提出了一种具有 INDEX-DATA 二级结构的多分辨率裁剪纹理。文章主要介绍了多分辨率裁剪纹理的定义和结构，并讨论多分辨率纹理的构建方法和基于多分辨率纹理的体裁剪方法，最后对实验结果进行了分析讨论。该文方法已在自主知识产权可视化软件 SVIP 中得到应用，取得了令人满意的结果。

关 键 词：计算机应用；INDEX-DATA 纹理；多分辨率；体裁剪；直接体绘制
中图分类号：TP 391
文献标识码：A **文 章 编 号：**1003-0158(2011)03-0001-05

Volume Clipping Technology Based on Multi-resolution Clipping Texture

LUO Yue-tong[1], XUE Ye[1], LONG Peng-cheng[2], WU Guo-yong[1]

(1. VCC Division, School of Computer and Information, Hefei University of Technology, Hefei Anhui 230009, China;
2. Institute of Plasma Physics, Chinese Academy of Sciences, Hefei Anhui 230031, China)

Abstract: A multi-resolution clipping texture with INDEX-DATA structure is presented to solve the huge size problem of single-resolution clipping texture data. Firstly, definition and structure of multi-resolution clipping texture are introduced; then multi-resolution clipping texture's construction method and volume clipping technology based on multi-resolution clipping texture are discussed; finally, experiment results are discussed. The method presented in the paper has been applied in visualization software SVIP and the result is satisfactory.

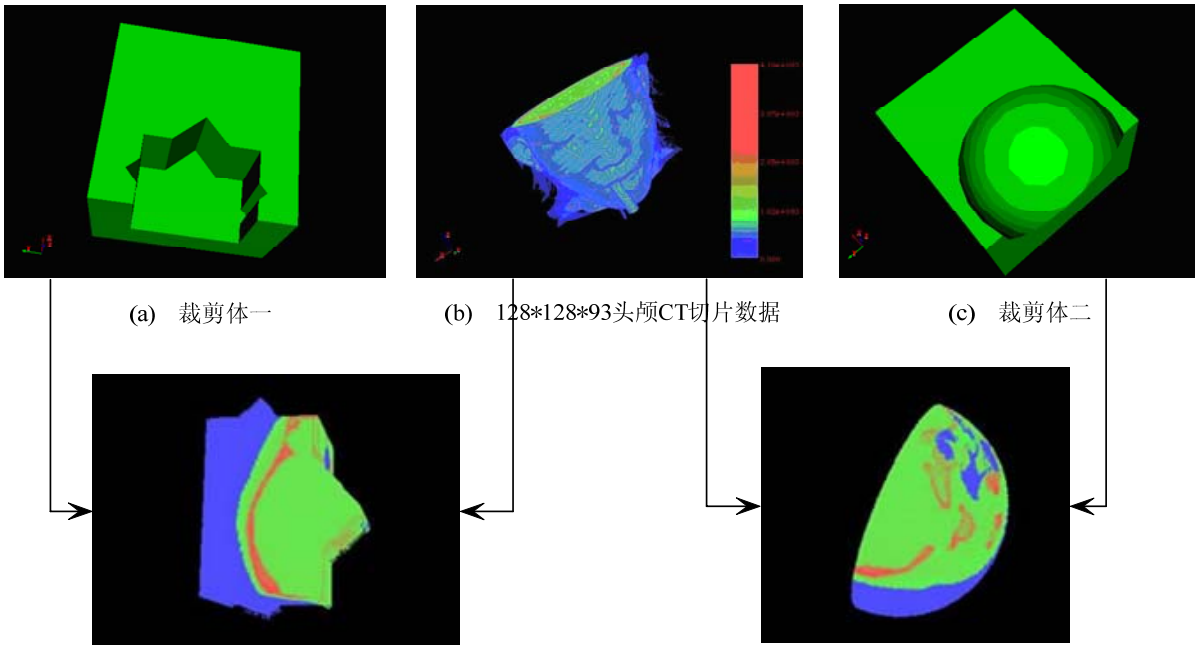
Key words: computer application; INDEX-DATA texture; multi-resolution; volume clipping; direct volume rendering

直接体绘制(DVR: Direct Volume Rendering)是可视化复杂数据集的重要工具，用户通过调节传递函数(TF: Transform Function)来显示/隐藏数据集中的特征。因为 TF 函数是全局函数，所以

收稿日期：2009-12-18
基金项目：安徽省自然科学基金资助项目（090412066）；合肥工业大学博士学位专项基金资助项目；国家自然科学基金资助项目（60573174；60673028）
作者简介：罗月童（1978-），男，安徽青阳人，副教授，博士，主要研究方向为计算机图形学，科学计算可视化，虚拟现实。

它难以凸显数据中的局部特征，虽然人们对 TF 函数进行了大量研究[1]，但这个问题至今依没能得到有效解决[2]。

虽然人们对 TF 函数进行了大量研究，但关于体裁剪技术的研究很少，深入的研究就更少。最简单的体裁剪方法是基于 OpenGL、DirectX 裁剪面的体裁剪方法[3]，但这类方法只能裁剪出简单的形状，难以实现如图 1(c)和图 1(d)所示的常用效果。Westerman 和 Ertl[4]提出基于蒙板测试的体裁剪方法，该方法能实现复杂的裁剪效果，但实现困难。随着可编程 GPU 技术的出现，研究基于 GPU 的体裁剪技术成为主流，Daniel Weiskop 等人[2]提出基于深度的体裁剪技术，该技术能实现精确裁剪且额外开销小，但它难以应用于非凸裁剪体。基于裁剪纹理的体裁剪技术一方面对裁剪体形状没有限制，另一方面基于可编程 GPU 很容易实现，所以它的应用非常广泛，但为了保持裁剪的准确性，通常需要很高分辨率的裁剪纹理，从而使得裁剪纹理过大。针对裁剪纹理太大的问题，本文提出多分辨率裁剪纹理，在保持体裁剪效果不变的情况下压缩裁剪纹理。



(a) 裁剪体一　(b) 128*128*93头颅CT切片数据　(c) 裁剪体二

(d) 裁剪效果一　(e) 裁剪效果二

图 1 体数据裁剪

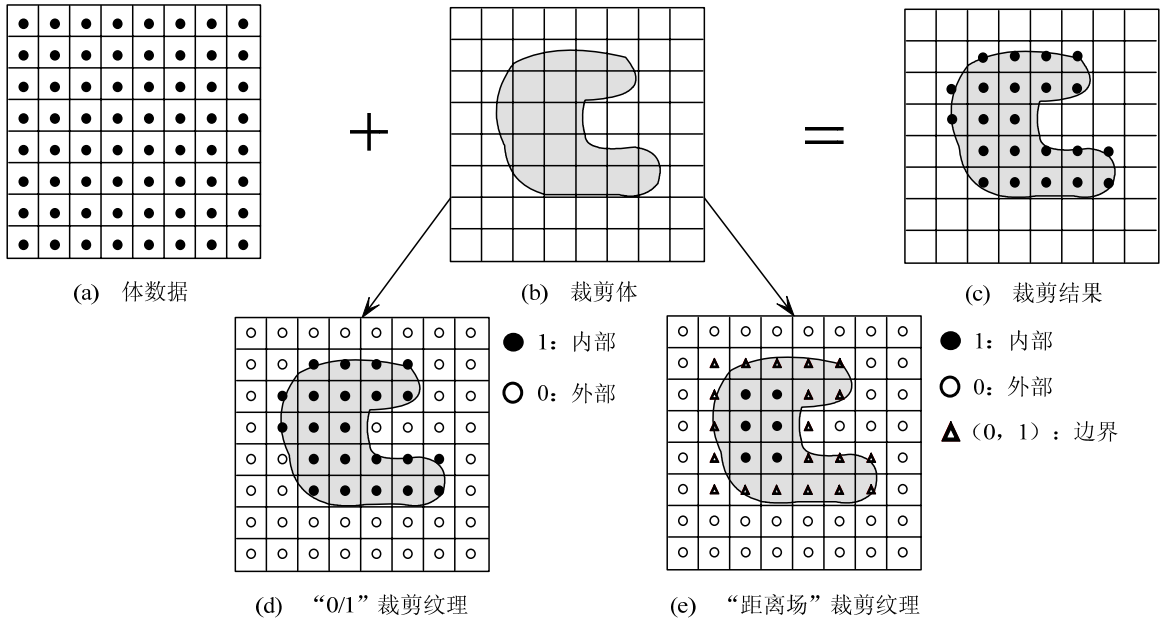
1 基于裁剪纹理的体裁剪技术

体裁剪技术是指在 DVR 方法中实现显示指定内容的方法和技术，通常使用 3D 实体指定裁剪区域，并将 3D 实体称作裁剪体，如图 2 所示（为便于表达采用 2D 图表示）。基于裁剪纹理的体裁剪技术首先将 3D 实体离散成裁剪纹理，然后基于裁剪纹理实现体裁剪。最简单的裁剪纹理是 0/1 裁剪纹理，裁剪体内部的体元用 1 表示，其它体元用 0 表示。基于 0/1 裁剪纹理的体裁剪容易在边界处引起锯齿走样或使边界变得模糊不清的问题，所以文献[2]对 0/1 裁剪进行改进，提出距离场裁剪纹理。在距离场裁剪纹理中，每个体元中存储了按下述方法计算得到的距离：首先计算该体元中心点到裁剪体边界的距离（内部点的距离为正，外部点的距离为负），然后按公式

$$f(d)=\begin{cases}0 & ,d\leqslant -D/2\\ d/D+1/2 & ,-D/2<d<D/2\\ 1 & ,d\geqslant D/2\end{cases}$$ 进行归一化。

其中 D 表示裁剪纹理中体元对角线距离，d 表示采样点到裁剪体边界的最近距离。$d\leqslant -D/2$ 表示体元完全处于裁剪体外部，$-D/2<d<D/2$ 表示体元有可能和裁剪体边界相交，而 $d\geqslant D/2$ 表示体元完全处于裁剪体内部。在基于距离裁剪

纹理的体裁剪算法中，在自定义的片段程序中查询距离裁剪纹理，如果该片段的裁剪纹理值不小于 0.5 则保留该片段，否则将该片段裁减掉。因为这种方法使用距离场裁剪纹理中 ISOVALUE=0.5 的等值面表示裁剪体边界，能有效克服 0/1 裁剪纹理的问题，所以本文在距离场裁剪纹理的基础上实现多分辨率裁剪纹理。



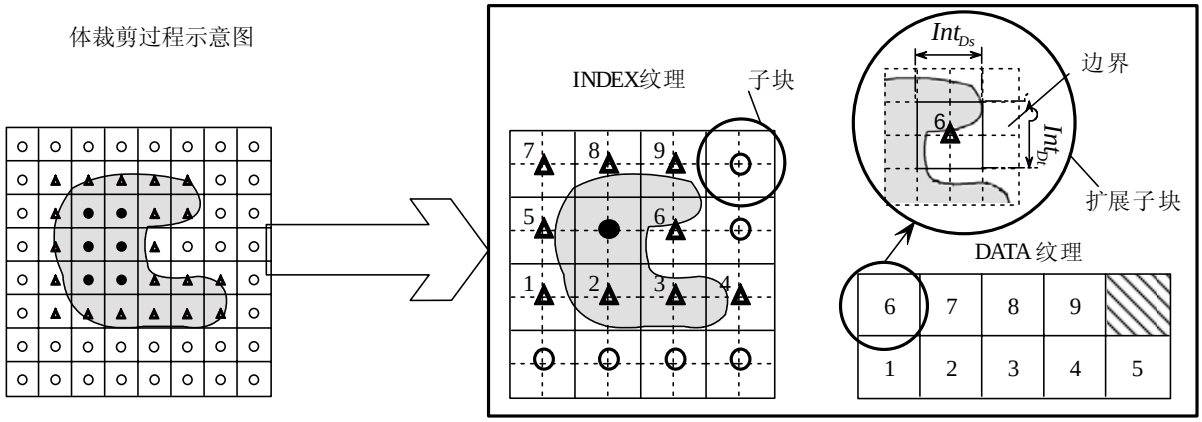
(a) 体数据　(b) 裁剪体　(c) 裁剪结果

(d) “0/1”裁剪纹理　(e) “距离场”裁剪纹理

图 2　体裁剪过程示意图

2　多分辨率裁剪纹理

虽然基于裁剪纹理的体裁剪技术具有灵活、方便实现等优点，但高分辨率裁剪纹理需要占用大量纹理内存空间。观察发现，裁剪纹理只在裁剪体边界附近需要高分辨率，其它部分可以使用较低分辨率并不影响效果，因此本文提出如图 3(b)所示多分辨率裁剪纹理——INDEX-DATA 裁剪纹理，其中 INDEX 纹理采用低分辨率，描述裁剪体的总体结构，DATA 纹理采用高分辨率，用于刻画裁剪体的边界细节。INDEX 纹理中体元的可能取值及它们的意义如表 1 所示，DATA 纹理保存 INDEX 纹理中(r_{Dc}, s_{Dc}, t_{Dc}, 0.5)体元所对应的高分辨率采样数据。



(a) 单分辨率裁剪纹理　(b) 多分辨率裁剪纹理

图 3　单分辨率裁剪纹理与多分辨率裁剪纹理的对比

表 1 INDEX 纹理体元的取值范围

编号	取值(r, g, b, α)	取 值 意 义
1	(0, 0, 0, 1)	$\alpha=1$：表示该体元完全处于裁剪体内部，(0, 0, 0)无意义。
2	$(r_{Dc}, s_{Dc}, t_{Dc}, 0.5)$	$\alpha=0.5$：表示该体元与裁剪体边界相交，(r_{Dc}, s_{Dc}, t_{Dc})表示该体元在 DATA 纹理中对应纹理块中心点的纹理坐标。
3	(0, 0, 0, 0)	$\alpha=0$：表示该体元完全处于裁剪体外部，(0, 0, 0)无意义。

2.1 多分辨率裁剪纹理的构建方法

对分辨率为 $N_w \cdot N_h \cdot N_d$ 的裁剪纹理，按下列步骤可生成相应的多分辨率裁剪纹理：

（1）将裁剪纹理分割成一组大小为 k^3 的子块，其中 N_w、N_h 和 N_d 均能被 k 整除，且 $2 \leqslant k \leqslant \min(N_w, N_h, N_d)$，则 INDEX 纹理大小为 $\frac{N_w}{k} * \frac{N_h}{k} * \frac{N_d}{k}$。

（2） 然后按下述方法处理每个方块：

1） 遍历子块中所有元素的值：如果值都为 1，则转向 2)；如果值均为 0，则转向 3)；否则转向 4)；

2） 将 INDEX 纹理中对应体元设为(0, 0, 0, 1)，然后转到 5)；

3） 将 INDEX 纹理中对应体元设为(0, 0, 0, 0)，然后转到 5)；

4） 按图 3(b)所示方式将“扩展子块”移动到 DATA 纹理中并依次排列，“扩展子块”是将当前子块在上、下、左、右、前、后六个方向各扩展一个体元所形成的$(k+2)^3$ 子块，扩张的目的是解决纹理查询中的边界问题；然后将 INDEX 纹理中对应体元设为$(r_{Dc}, s_{Dc}, t_{Dc}, 0.5)$，其中$(r_{Dc}, s_{Dc}, t_{Dc})$为子块中心点的 DATA 纹理坐标；

5） 结束对当前子块的处理。

（3） 保存 INDEX 纹理和 DATA 纹理，结束。

经过上述处理，得到大小为 $\frac{N_w}{k} * \frac{N_h}{k} * \frac{N_d}{k}$ 的 INDEX 纹理，DATA 纹理的大小取决于原始裁剪纹理，但通常会远远小于原始纹理，因为事实上只有少量子块和裁剪体边界相交。k 的取值对多分辨率裁剪纹理的性能有影响，实验表明 $k=4$ 时效果较好。

2.2 基于多分辨率裁剪纹理的体裁剪

基于多分辨率裁剪纹理的体裁剪技术也是通过在自定义片段程序中查询裁剪纹理，然后根据查询结果决定对应片段的去留。假设当前片段的 INDEX 纹理坐标为 (s_I, t_I, r_I)，那么在片段程序首先以纹理坐标 (s_I, t_I, r_I) 按 GL_NEAREST 方式查询 INDEX 纹理获取纹理值 $(s_{Dc}, t_{Dc}, r_{Dc}, \alpha_I)$；然后基于 α_I 的值进行以下处理：

（1） $\alpha_I = 1.0$，表明当前片段处于裁剪体内部，保留当前片段；

（2） $\alpha_I = 0.0$，表明当前片段处于裁剪体外部，抛弃当前片段；

（3） $\alpha_I = 0.5$，表明当前片段处于裁剪体边界周围，需通过进一步查询 DATA 纹理才能决定当前片段的去留。其中(s_{Dc}, t_{Dc}, r_{Dc})表示当前片段所对应 DATA 纹理块的中心坐标，查询步骤如下：

1） 计算 DATA 纹理坐标(s_D, t_D, r_D)；

2） 按 GL_LINEAR 方式查询(s_D, t_D, r_D)处的 DATA 纹理值 α_D；

3） 如果 $\alpha_D \geqslant 0.5$，则保留当前片段，否则抛弃当前片段。

本文基于文献[5]中的方法计算(s_D, t_D, r_D)，计算公式为

$$\begin{cases} s_D = fraction\left(s_I \cdot \left(N_{Iw} - 1\right) - 0.5\right) * Int_{Ds} + s_{DS} \\ t_D = fraction\left(t_I \cdot \left(N_{Ih} - 1\right) - 0.5\right) * Int_{Dt} + t_{DS} \\ r_D = fraction\left(r_I \cdot \left(N_{Id} - 1\right) - 0.5\right) * Int_{Dr} + r_{DS} \end{cases}$$

其中 N_{Iw}、N_{Ih} 和 N_{Id} 分别表示 INDEX 纹理的宽度(width)、高度(height)和深度(depth)；Int_{Ds}、Int_{Dt} 和 Int_{Dr} 分别表示 DATA 纹理中数据子块的宽度、高度和深度，如图 3(b)所示；函数 $fraction(x)$ 表示取 x 的小数部分。

3 实验结果与分析

本文算法已在自主开发可视化软件 SVIP (Scientific Visualization Integrated Platform)[7]中实现和应用，并且作者基于 SVIP 对本文算法进行

了测试。测试环境如表 2 所示，实验结果如表 3 所示。由实验结果本文得如下结论：

• 多分辨率纹理能有效压缩裁剪纹理，且压缩比和裁剪体形状有关。虽然对复杂结构裁剪体的压缩效果相对较差，但仍能获取很高的压缩比。

• 和单分辨率裁剪纹理的相比，使用多分辨率裁剪纹理会降低渲染速度，但依旧能够实现实时的渲染效果。渲染速度下降主要是因为在片段着色器程序中存在二级纹理查询。

表 2　测试环境

	显卡	CPU	内存
硬件环境	NVIDIA Geforce 7600	AMD Athlon 64 X2 3600+ 2.01 GHz	512 M
软件环境	测试平台	分辨率	相关参数
	SVIP	800*800	N=128、k=4
实验数据	体数据	裁剪体一	裁剪体二
	头颅 CT 切片，图 1(a)	简单的裁剪体，图 1(b)	复杂的裁剪体，图 1(c)

表 3　实验结果

裁剪体	压缩率	渲染速度 FPS		效果图
		单分辨率裁剪纹理	多分辨率裁剪纹理	
裁剪体 A	22.54%	32.3	21.3	图 1(d)
裁剪体 B	14.07%	32.3	22.1	图 1(e)

注：压缩率 =（INDEX 纹理+ DATA 纹理）/单分辨率纹理

4　结　论

体裁剪技术是对 TF 函数的重要补充，在解决 DVR 难以揭示数据集内部细节特征的问题上有重要价值。随着可编程 GPU 技术的发展，基于裁剪纹理的体裁剪技术得到广泛使用，但该方法的不足之处是其体纹理太大。本文基于“仅裁剪体边界处需高分辨率裁剪纹理”的事实，提出 INDEX-DATA 结构的多分辨率裁剪纹理，以减少裁剪纹理的内存需求。实验表明 IDEX-DATA 结构的多分辨率裁剪纹理能有效节约纹理内存，尽管同时它也影响了渲染速度，不过仍能获得实时渲染效果。

参 考 文 献

[1] Pfister H, Lorensen B, Bajaj C, et al. The transfer function bake-off [J]. IEEE Computer Graphics and Applications, 2001, 21(3): 16−22.

[2] Daniel Weiskopf, Klaus Engel, Thomas Ertl. Interactive clipping techniques for texture-based volume visualization and volume shading [J]. IEEE Transactions on Visualization and Computer Graphics, 2003, 9(3): 298−312.

[3] Volz W R. Gigabyte volume viewing using split software/hardware interpolation [C]//Proc. 2000 Symp. Volume Visualization, 2000: 15−22.

[4] Westermann R, Ertl T. Efficiently using graphics hardware in volume rendering applications [C]// SIGGRAPH 1998 Conf. Proc., 1998: 169−179.

[5] Kraus M, Ertl T. Adaptive texture maps [C]// Proceedings of Graphics Hardware 2002, 2002: 7−15.

[6] Binotto A P, Comba J L, Freitas C M. Real-time volume rendering of time-varying data using a fragment-shader compression approach [C]// Proceedings of the 6th IEEE Symposium on Parallel and Large-data Visualization and Graphics (PVG'03), 2003: 69−76.

[7] 罗月童, 龙鹏程, 薛　晔, 等. 面向中子学分析的集成可视化平台 SVIP 的发展研究[J]. 核科学与工程, 2007, 27(4): 374−378.