一种快速高效的反走样算法 *

刘 燕[a,b], 张建伟[a,b]
(四川大学 a. 计算机学院; b. 视觉合成图形图像技术国防重点学科实验室, 成都 610065)

摘 要: 快速近似反走样是一种后处理式反走样技术,其效果好、效率高,但由于是根据亮度值检测边缘,导致模型纹理贴图上的条纹和高光部分也会被检测出来,从而造成不必要的反走样。针对这一问题,该算法提出了一种改进的反走样方法:根据物理和阴影信息检测出模型与阴影边缘,再以边缘周围状况计算确定混合朝向,最后结合亚像素覆盖率进行反走样运算。实验表明,该算法在保留了原快速近似反走样效率的同时,有效地消除了不必要的反走样。

关键词: 图形图像处理; 反走样; 边缘检测; 亚像素覆盖率
中图分类号: TP391.9 **文献标志码**: A **文章编号**: 1001-3695(2012)03-1138-03
doi:10.3969/j.issn.1001-3695.2012.03.092

Fast and effective anti-aliasing algorithm

LIU Yan[a,b], ZHANG Jian-wei[a,b]
(*a. School of Computer Science, b. State Key Laboratory of Fundamental Science on Synthetic Vision, Sichuan University, Chengdu* 610065, *China*)

Abstract: Fast approximate anti-aliasing is a post process anti-aliasing technology, which performs perfectly and effectively. It detects the edges by luminance, so stripes of texture and highlight on models are detected, resulting in unnecessary anti-aliasing. To solve the problem, this paper provided a new method: detected the edges of models and shadows, based on physical information and shadows information, and then computed blending direction according to edges' ambient conditions. Finally, the blending direction was combined with sub-pixel coverage to achieve the goal of anti-aliasing. Experiments show that this algorithm can effectively eliminate the unnecessary anti-aliasing with retaining its excellent effect.

Key words: graphics and image processing; anti-aliasing; edge detection; sub-pixel coverage

走样(aliasing)指的是光栅图形显示器中用离散量表示连续量引起的失真,相应地,消除这种失真现象即为反走样。由于图形的走样对图形质量有很大的影响,不容忽视,所以反走样技术领域的研究也一直是计算机图形学的研究热点。

反走样的基本思想是将图像边缘与周围的像素颜色进行混合,然后用新生成的点来替换原位置上的点,以达到柔化物体外形、消除锯齿的效果。目前图形学中的反走样方法主要分为硬件式[1]、后处理式[2,3]、混合式三类。硬件方法即增加采样率,主要是靠提高采样点的数目以达到反走样的目的,采样率越高,锯齿越小,边缘看上去越平滑。该方法原理简单直接,可以很好地解决走样问题,但大大增加了对资源的消耗,并且受限于显示分辨率。目前硬件式方法如超采样及多重采样均运用这个原理,先进行边缘检测,然后在边缘处进行超采样,以较小的代价获得高反走样效果。后处理方法的主要思想是对整幅图像全部绘制完毕后,在显示之前作边缘检测,然后对边缘像素点作平滑处理。一般是以某像素及周围点的颜色来计算最终像素的颜色,如均匀区域采样、随机采样、加权随机采样等。该方法具有整体效果好的优点,但由于边缘检测对整幅画面进行搜索,且邻近像素之间都要作比较,再加上平滑所需要的时间,所以在计算量和存储量上的花费代价比较大。而混合式方法就是硬件式和后处理式方法结合后的产物,它既降低了硬件开销和计算量,又保持了较高的效率和很好的反走样效果,但缺点是需要修改原有的图形渲染流水线。

FXAA[4](fast approximate anti-aliasing)属于第二类方法,最早由 Timothy Lottes 在 2009 年提出。该算法快速、高效,但会将模型表面纹理贴图上的条纹和高光部分一同检测为边缘,并进行反走样。另外,该算法中子偏移量的计算复杂且阈值繁多,因此对于某些模型不能达到很好的反走样效果。针对这些问题,本文提出了不同的边缘检测及子偏移量计算的方法。

1 算法

1.1 原理

针对物体边缘和阴影边缘产生走样的问题,利用延迟光照算法,第一遍绘制过程在缓存中生成位置和法线信息,第二遍时生成阴影信息,结合这三类信息进行边缘检测。对于非边缘像素,立即返回并不再作后续处理;对于边缘像素,则根据其周围状况确定混合朝向,再结合亚像素覆盖率进行混合计算,达到平滑效果。

1.2 主要步骤

1)边缘提取

边缘检测[5,6]是基于物体和背景之间在灰度或纹理特性

收稿日期: 2011-08-16; **修回日期**: 2011-09-27 **基金项目**: 国家自然科学基金资助项目(60903118,60832011)

作者简介: 刘燕(1987-),女,四川简阳人,硕士研究生,主要研究方向为计算机图形学、虚拟现实;张建伟(1972-),男,四川成都人,博导,主要研究方向为计算机图形学、数字图像处理、实时软件工程(zhangjianwei@scu.edu.cn).

上存在某种不连续性进行的一种检测技术,大致可分为基于物理与基于属性的方法两类。基于物理方法的原理是利用法线、位置、深度值等信息进行判断;基于属性方法的原理是利用亮度、色彩等信息来进行判断。

原 FXAA 算法中是以像素为对象,利用亮度进行边缘检测,因此会将模型纹理贴图上的条纹和高分部分也提取出来,进行不必要的反走样。针对这个问题,本文利用延迟着色生成的屏幕上每个像素点的法线、位置和阴影信息来准确快速地提取物体及阴影边缘。

(1)利用法线信息提取边缘。场景中每一点的法线方向与其所在平面的法线方向相同。相邻的两点,如果位于一个平面,则其法线夹角为 0;如果位于一个平滑的曲面,则法线夹角较小;如果位于边缘,则法线夹角较大。

设相邻两点的法线向量分别为 nor_1、nor_2,法线夹角为 φ,由于法线向量是单位化的,因此绝对值都等于 1:

$$nor_1 \cdot nor_2 = |nor_1||nor_2|\cos\varphi = \cos\varphi \tag{1}$$

分别计算当前像素法线与其上、下、左、右四个方向相邻像素法线的夹角余弦值,获取夹角大小,与经验阈值进行比较,判断当前像素是否为边缘。

通过法线测试边缘可以提取绝大部分边缘信息,但对于场景中平行的平面,两平面上的点法线夹角为 0,但不会被检测为边缘,所以需要再利用位置信息继续进行判定。

(2)利用位置信息提取边缘。位置信息中保存的是当前像素的空间三维坐标,假设当前点与其相邻点的坐标分别为 (x,y,z),(x_1,y_1,z_1),可知两点的距离:

$$\text{distance} = \sqrt{(x-x_1)^2+(y-y_1)^2+(z-z_1)^2} \tag{2}$$

通过距离判断当前点是处于边缘,如式(3):

$$\frac{\max\{\text{distance}_i, i=1,2,3,4\}}{\min\{\text{distance}_i, i=1,2,3,4\}} > \text{DisThreshold} \tag{3}$$

其中,distance_i 分别表示当前点与其上、下、左、右四个方向相邻点的距离。如果最大与最小距离之比大于阈值,则认为当前点处于边缘。用黑线表示检测出的边缘,如图 1 所示,(a)为法线检测结果,(b)为法线结合位置信息检测结果。可以看出,利用位置信息检测很好地解决了法线检测遗留的问题。

(3)利用阴影信息提取边缘。上述操作将场景中模型的边缘完整地检测出来,但投影到地面上的阴影边缘却还需要缓存中的阴影信息进行检测。

延期着色[7]时,利用一张纹理的 α 通道保存阴影信息。根据当前像素是否在阴影中,将 α 的值分别赋为 0.0、1.0。如果某像素与周围像素同时处于阴影中或者同时处于光照下,则该像素不是阴影边缘;反之,则该像素是阴影边缘。

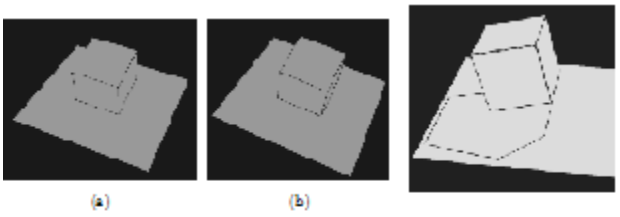
设两像素的阴影状态值分别为 InShadow_0、InShadow_1,表示状态一致的变量 IsSame:

$$\text{IsSame} = \text{abs}(\text{InShadow}_0 - \text{InShadow}_1) \tag{4}$$

通过阴影信息判断当前像素是否是边缘,如式(5):

$$\max\{\text{IsSame}_i, i=1,2,3,4\} > \text{ShadowThreshold} \tag{5}$$

其中,IsSame_i 分别表示当前像素与其上、下、左、右四个方向像素的阴影状态是否一致,其值为 0.0 或 1.0,所以 ShadowThreshold 取两者中间任意即可。这种方法很好地检测出了阴影的边缘,结合前面两个步骤,可以得到准确的边缘检测结果。用黑线表示检测出的边缘,效果如图 2 所示。



(a)　(b)

图 1　模型边缘　　图 2　模型边缘和阴影边缘

用黑线表示边缘,边缘检测效果如图 3 所示。可以看出,相较于左侧图亮度检测算法,右侧图中纹理贴图上的条纹边缘未被检测出来,从而避免了不必要的反走样,使实验效果更准确,同时减少了计算量。

图 3　边缘检测结果比较

2)确定混合朝向

在提取边缘后,对不同走向的边缘像素进行平滑处理时会采取不同的计算,所以将边缘的像素点分为不同的朝向,如下:

$$X=\begin{bmatrix} NW & N & NE \\ W & M & E \\ SW & S & SE \end{bmatrix}, A=\begin{bmatrix} 0.25 & -0.5 & 0.25 \\ 0.5 & -1.0 & 0.5 \\ 0.25 & -0.5 & 0.25 \end{bmatrix}, A^{\mathrm{T}}=\begin{bmatrix} 0.25 & 0.5 & 0.25 \\ -0.5 & -1.0 & -0.5 \\ 0.25 & 0.5 & 0.25 \end{bmatrix}$$

$$P = A \times X, Q = X \times A^{\mathrm{T}}$$

$$\begin{cases} \text{edge}H = |p[0][0]| + |p[1][1]| + |p[2][2]| \\ \text{edge}V = |Q[0][0]| + |Q[1][1]| + |Q[2][2]| \\ \text{Horizon} = \text{edge}H \geqslant \text{edge}V \end{cases} \tag{6}$$

其中:矩阵 X 的元素对应于当前像素与周围八个像素的颜色值;edge$H$ 表示当前像素与相邻上、下两像素的颜色值差;edge$V$ 表示当前像素与相邻左、右像素的颜色值差。两者进行比较,若 edge$H$ 较大,则表明该像素处于横向边缘,混合朝向为竖直方向。反之,则该像素处于纵向边缘,混合朝向为水平方向。

3)确定子偏移量

子偏移量对最后的反走样效果有至关重要的作用,因此获取混合朝向后,此处将对不同的朝向分别计算子偏移量。图像边缘有一个特性是梯度[8]方向,即函数值或灰度值的最大增长方向。混合朝向与梯度方向一致,该算法中将颜色差值近似作为梯度值,并将子偏移量初始位置在梯度值小的一方。

以竖直混合朝向的边缘像素进行说明,假设当前像素与相邻上、下两像素的颜色差值分别为梯度 $\nabla\varphi_1$ 和 $\nabla\varphi_2$,则子偏移量初始值 offset 为

$$\text{offset} = \begin{cases} 0.25(d_{\text{width}}, d_{\text{height}}) & \nabla\varphi_1 \leqslant \nabla\varphi_2 \\ -0.25(d_{\text{width}}, d_{\text{height}}) & \nabla\varphi_1 > \nabla\varphi_2 \end{cases} \tag{7}$$

其中,d_{width} 和 d_{height} 分别表示单个像素的宽度和高度。

为了获取更精确的颜色值,进一步计算最终子偏移量 subOffset,如式(8):

$$\text{subOffset} = \text{offset} \times \left(1 + \sqrt{\frac{\text{maxCol} - \text{minCol}}{\text{centerCol}}} \times 0.5\right) \tag{8}$$

其中:maxCol 和 minCol 分别表示当前像素的四个亚像素中最大颜色值和最小颜色值;centerCol 表示当前像素中心位置的颜色值;颜色值此处用分量 R、G、B 的和值进行表示。

水平混合朝向的边缘像素,其子偏移量的求解方法与上述方法相同,但求解初始值时应注意梯度值的不同。

4)混合平滑

(1)计算混合参数

平滑处理的本质即将两颜色值进行线性混合,那首先就得计算获取混合参数。具体操作如下:

计算出当前像素中四个亚像素的亮度值,如式(8):

$$luma = color.g \times (0.587/0.299) + color.r \tag{9}$$

由亮度值之比获取混合参数:

$$mix = \frac{lumaMin}{lumaMax} \tag{10}$$

其中,lumaMin 和 lumaMax 是当前像素的四个亚像素采样点中的最大亮度值和最小亮度值。

(2)颜色混合

$$color = rgbS \times mix + rgbO(1 - mix) \tag{11}$$

其中:rgb$S$ 表示根据子偏移量取出的颜色值;rgb$O$ 表示当前像素中四个亚像素和相邻四个像素的平均颜色值;mix 是之前获取的混合参数。混合后效果如图 4 所示。

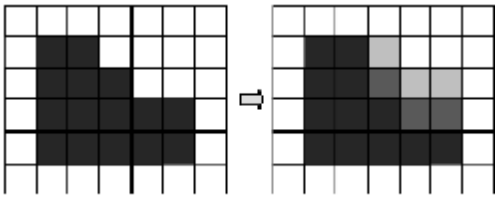


图 4　混合效果

本文算法通过借鉴快速近似反走样方法后处理式的思想,结合物理边缘检测和简单的子偏移量求解方法,在系统可以承受的范围内,得到真实性和实时性综合后的结果。

1.3　算法描述与流程图

通过上述算法步骤可见,要进行反走样处理,首先需要保存存储着最后绘制结果的纹理,并利用延期着色时缓存中的物理信息和阴影信息准确检测出边缘,然后将边缘划分为横向与纵向两类,分别获取其混合朝向,再根据混合朝向计算边缘像素的子偏移量,最后将像素子偏移量处取出的颜色值与亚像素和相邻像素的平均颜色值进行混合,实现平滑处理。流程如图 5 所示。

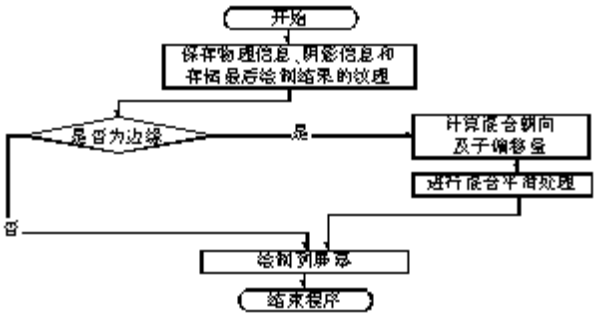


图 5　算法流程

2　实验结果与分析

本文使用的实验系统为 Intel® Core™ 2 Duo E7500(2.93 GHz) CPU、GTX260 GPU。实验采用较为简单的正方体模型和复杂的铁路模型。

无反走样与本文反走样算法的效果对比如图 6 所示。

从图 6(c)可以看到,无反走样算法时,铁路隧道口边缘处的走样十分严重,而运用本文算法(图 6(d)),反走样效果明显。快速近似反走样算法与本文算法的效果对比如图 7 所示。

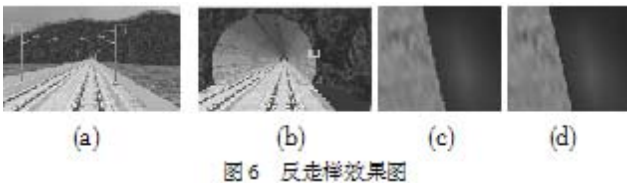


图 6　反走样效果图

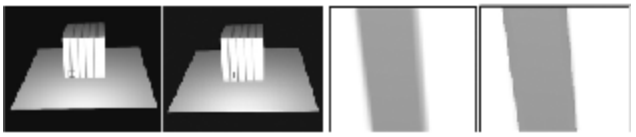


图 7　反走样效果对比

从图 7 可以看出,本文中的算法与快速近似反走样算法一样具有很好的反走样效果,并且很好地改善了它会将模型表面纹理贴图上的条纹进行不必要反走样的缺陷。另外,本文算法没有过多地使用经验阈值,不用针对模型对各阈值进行调整,扩大了应用范围。无反走样算法、快速近似反走样算法、本文算法反走样的性能对比如表 1 所示。

表 1　反走样方法帧率对比

反走样方法	平均帧率/fps
无反走样	760
快速近似反走样	703
本文算法	716

从表 1 可以看到,使用快速近似反走样算法后,帧率比无反走样有所下降,但下降幅度不到 1%,而本文算法使用了不同的子偏移量求解方法,在一定程度上进一步提高了效率,保证了实时性,同时满足了实时绘制对效果和效率的需求。

3　结束语

本文提出了一种快速有效的反走样算法,结合了物理边缘检测和快速近似反走样算法中后处理的优点,在保持了其优秀的反走样效果的同时,有效地改善了快速近似反走样算法会将模型表面纹理贴图上的条纹和高光部分进行多于反走样的情况,并且由于该算法中子偏移量的求解简明易解,在一定程度上提高了效率。该算法很大的一个优势是节省资源,但在处理一些细小的物体如电线、栏杆等时效果不明显。下一步,将在这个方向作进一步的改进,扩大算法应用范围。

参考文献:

[1] 陈为. 硬件加速反走样体 Splatting 算法[J]. 计算机辅助设计与图形学学报, 2005,17(4):677-682.

[2] 沈强,张波,陈淑珍. 计算机图形学反走样技术及实现[J]. 武汉大学学报:自然科学版, 1997, 43(1):113-118.

[3] 骆朝亮. 一种支持多线宽直线反走样算法[J]. 计算机技术与发展,2010,20(9):102-105.

[4] LOTTES T. FXAA[EB/OL]. (2011-01-25)[2011-06-23]. http://developer. nvidia. com/sites/default/files/akamai/gamedev/files/sdk/11/FXAA_WhitePaper. pdf.

[5] 张太发,高朝阳. 数字图像边缘检测方法的分析与研究[J]. 计算机技术与发展,2010,20(10):141-145.

[6] 宋莹,陈科,林江莉. 基于图像分块的边缘检测算法[J]. 计算机工程,2010,36(14):196-197.

[7] 王豪,段茗,李西猴,等. 一种延期着色的快速反走样算法[J]. 计算机工程,2010, 36(24):251-255.

[8] 周虹,罗晓曙,何富运. 一种改进的亚像素偏移算法研究[J]. 广西师范大学学报:自然科学版,2010,28(1):170-173.