各向异性滤波器的硬件实现

徐起超, 杜慧敏*, 曾泽沧, 王鹏超

(西安邮电大学电子工程学院 西安 710121)
(fv@xupt.edu.cn)

摘 要: 在计算机图形学中, 通常采用各项同性滤波器逼近异性滤波器以减少纹理映射中发生纹理走样, 逼近算法中实现MIP-MAP层级包含计算覆盖区域边长以及求对数等操作, 用二次逼近或者Cordic算法等实现时电路较大. 为了易于算法的硬件实现, 提出用线性逼近计算覆盖区域边长和对数的算法. 该算法用一次移位和一次加法实现覆盖区域边长计算, 用一次加法实现对数计算, 降低了硬件实现成本. 在Xilinx的ZC706开发板上实现了文中算法, 实验结果表明, 该算法所计算 MIP-MAP 的层级数与原算法的计算误差绝对值为 1 的概率为 7%.

关键词: 各向异性滤波器; 纹理映射; 纹理反走样; 覆盖区域边长
中图法分类号: TP391.41 **DOI:** 10.3724/SP.J.1089.2019.16953

Hardware Implementation of an Anisotropic Texture Filter

Xu Qichao, Du Huimin*, Zeng Zecang, and Wang Pengchao

(*School of Electronic Engineering, Xi'an University of Posts & Telecommunications, Xi'an* 710121)

Abstract: In order to decrease the texture aliasing during texture mapping, one of methods is to approximate aniso-tropic filtering by isotropic filtering in computer graphics field. In such an approximation, MIP-MAP layer could be obtained using footprint and logarithm, whose implementations cause larger size of the circuits in Cordic or quadratic approximation. In this paper, a linear approximation to calculate footprint and logarithm is presented to simplify their circuits and footprint calculation with one adder and one shifter and logarithm with one adder were implemented. The absolute value of error of MIP-MAP layers between the approximation and origin is less than 7%. The approximate anisotropic filtering proposed in this paper was validated on Xilinx ZC706 board.

Key words: anisotropy; texture mapping; texture anti-aliasing; footprint

纹理映射可以极大地提高计算机图形绘制的真实感. 但是在纹理映射过程中, 纹理空间与像素空间大部分时候不存在一一映射, 有时候需要对纹理进行缩小, 将指纹理空间中的多个纹素映射到一个像素上, 在显示设备上会出现"锯齿(jaggies)"或"阶梯状(staircase)"现象, 称之为走样(aliasing)[1-2], 造成视觉上的不舒服感. 研究人员提出一些算法减少纹理映射产生的走样(即纹理反走样技术)[3-4], 以获得高质量的图形渲染画面.

根据纹理滤波(texture filtering)方式, 纹理反走样可分为各向同性滤波[5]和各向异性滤波[6-7]. 各向同性滤波算法包括最近邻点采样, 双线性

收稿日期: 2017-10-19; 修回日期: 2018-10-15. 基金项目: 陕西省重点研发项目(2017ZDXM-GY-005). 徐起超(1990—), 男, 硕士研究生, 主要研究方向为集成电路系统设计; 杜慧敏(1966—), 女, 博士, 教授, 硕士生导师, 论文通讯作者, 主要研究方向为计算机体系结构; 曾泽沧(1967—), 男, 学士, 讲师, 主要研究方向为电子与系统; 王鹏超(1990—), 男, 硕士研究生, 主要研究方向为电路与系统.

滤波[8], MIP-MAP 滤波和三线性采样[5]. 各向异性滤波器在纹理映射过程中能更好地减少走样, 因此目前高性能 GPU 中都包含该功能, 主要有逼近椭圆加权平均, EWA(elliptical weighted average)算法[9-10], 椭圆覆盖区域近似算法[11], 四边形覆盖区域近似算法[12]和利用各向同性滤波方式实现各向异性滤波算法[1,13]. 其中, EWA 算法的纹理映射效果最好, 但计算复杂度高, 仅作为一个纹理映射的评价标准; 椭圆覆盖区域近似算法和四边形覆盖区域近似算法均是对 EWA 算法进行改进, 先用 MIP-MAP 算法计算出纹理映射的层级[5], 再根据该层级上的覆盖区域进行加权平均, 从而减少对纹理存储的访问次数, 提高纹理映射速度. 但是, 改进算法的复杂度依然很高, 用软件实现速度依然较慢. 利用各向同性滤波方式实现各向异性滤波算法的核心是将双线性插值, 三线性插值与 MIP-MAP 结合, 利用一个像素逆映射到纹理空间一个平行四边形区域逼近 EWA 算法, 计算相对简单, 便于硬件实现.

虽然利用各向同性滤波方式实现各向异性滤波算法较其他反走样方法的实现复杂度较低, 但是用软件实现依旧实时性较差. 本文根据利用各向同性滤波方式实现各向异性滤波的原理, 提出适于硬件实现的覆盖区域边长的逼近计算算法和对数逼近计算算法, 并对逼近算法的误差进行分析. 设计并实现了相关的硬件电路, 在ZC706开发板 SOPC 系统上进行了验证, 结果表明, 采用本文算法的硬件电路实现纹理反走样效果良好.

1　改进各向同性滤波实现各向异性滤波算法

1.1　各向同性滤波实现各向异性滤波原理

纹理映射有 2 种实现方式: 一种是正向映射(纹理扫描方式), 即将纹理图案映射到物理空间的物体表面上, 再经过投影变换映射到图像空间上, 最终显示在屏幕上; 另外一种是逆向映射(屏幕扫描方式), 即将图像空间中的像素逆向映射到纹理空间. 利用各向同性滤波方式实现各向异性滤波[6]属于逆向映射.

像素空间像素 $P(x,y)$ 及逆向映射到纹理空间的覆盖区域(简称覆盖区域)如图 1 所示. 其中, s_x, t_x, s_y, t_y 分别是 P 的纹理坐标 (s,t) 在 x 方向和 y 方向上的偏导数[5]. 如果 s_x, s_y 不相等, 称该映射为各向异性.

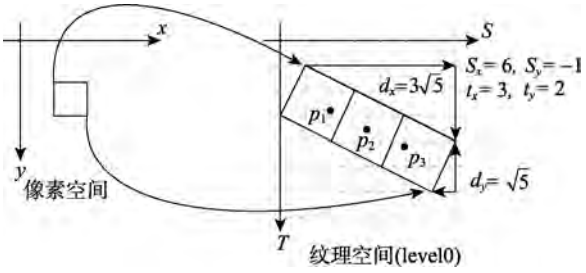


图 1　各向同性滤波方式实现各项异性滤波示意图

覆盖区域最长边为 $d_x = 3\sqrt{5}$, 短边为 $d_y = \sqrt{5}$; 长边与短边的比值为 3, 即可以将各向异性的覆盖区域划分为 3 个各向同性覆盖区域. 图 1 中 p_1, p_2, p_3 分别为 3 个各向同性滤波覆盖区域对应的 3 个采样点.

(1) 计算 P 在覆盖区域的纹素采样点

根据纹理空间与像素空间的映射关系, 求出像素空间映射到纹理空间覆盖区域的最长边 $d_{\max}$ 和最短边 $d_{\min}$, 即

$$\begin{cases} d_x = \sqrt{s_x^2 + t_x^2} \\ d_y = \sqrt{s_y^2 + t_y^2} \\ d_{\max} = \max(d_x, d_y) \\ d_{\min} = \min(d_x, d_y) \end{cases} \tag{1}$$

根据长边和短边之比 $d_{\max}/d_{\min}$, 求出纹理空间所需采样纹素的个数

$$N = \min\left(d_{\max}/d_{\min}, N_{\max}\right) \tag{2}$$

其中, $N_{\max}$ 为用户设置的最大采样点数.

从式(1)可以看出, 无论是求 $d_{\max}$ 还是 $d_{\min}$ 都需要作 4 次平方操作, 2 次加法操作, 1 次比较操作和 1 次开平方操作; 求 N 时需要作除法操作. 计算复杂, 不易于硬件实现.

(2) 计算 P 对应的 MIP-MAP 层级数

P 应用 MIP-MAP 的层级

$$l = \text{lb}\,(d_{\max}/N) = \text{lb}\,d_{\max} - \text{lb}\,N \tag{3}$$

(3) 计算 P 的颜色值

通过加权平均纹理空间的采样纹素, 计算像素 P 的颜色

$$c = \frac{1}{N}\sum_{1}^{N} c(s,t) \tag{4}$$

其中, $c(s,t)$ 表示纹理坐标 (s,t) 对应的纹理值. 该运算需要对采样点的个数求加法和除法运算.

1.2　覆盖区域边长计算改进

为了降低硬件实现的复杂度, 本节将讨论式(1)

的易于硬件实现的逼近算法.

首先讨论覆盖区域边长(即三角形斜边)计算逼近方法. 如图 2 所示, d_x 为直角三角形的斜边, s_x 与 t_x 为直角三角形的 2 条直角边, 其值是按文献[5]方法计算的偏导数. 由于偏导数具有方向性, 因此图2中直角边的大小用绝对值表示; 计算斜边

$$d_1=\sqrt{s_x^2+t_x^2}\ .$$

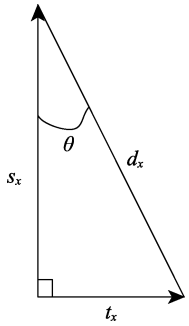


图 2 直角三角形示意图

本文用线性近似公式逼近 d_1, 其中, $\alpha\in[0,1]$, 且 $s_x>t_x$, 斜边逼近为

$$d_2=s_x+\alpha t_x \tag{5}$$

令 $k=\dfrac{t_x}{s_x}$, $k\in[0,1]$, 把式(4)(5)代入绝对误差公式中得到

$$e_\alpha=|\,d_2-d_1\,| \tag{6}$$

并且化简得到

$$e_\alpha=\left|s_x(1+\alpha k-\sqrt{1+k^2})\right|.$$

为求出 e_α 的最小值, 需要确定 α 值. 最优化的 α 应该使得在 $k\in[0,1]$ 内 $\int_0^1 e^2\mathrm{d}k$ 最小. 积分函数为

$$f(\alpha)=\int_0^1 e^2\mathrm{d}k \tag{7}$$

计算式(7)得

$$f(\alpha)=\int_0^1\left(1+\alpha k-\sqrt{1+k^2}\right)^2\mathrm{d}k\approx$$

$$\frac{1}{3}\alpha^2-0.219\alpha+0.0377.$$

$f(\alpha)$ 在 $\dfrac{\partial f(\alpha)}{\partial\alpha}=0$ 得到最小值, 当 $\alpha\approx0.3284$ 时误差值最小.

为了易于硬件设计, 本文取 α 为 0.5 和 0.25, 此时 αt_x 可以用右移位 1 位或者 2 位 t_x 实现. 表 1 所示为这 2 种情况下的直角三角形斜边逼近均方误差值.

下面分析当 α 为 0.5 和 0.25 时, 各向同性滤波

表 1 不同系数下的直角三角形斜边逼近均方误差值

α	均方差/%
0.5	1.16
0.25	0.38
0.3284	0.18

MIP-MAP 层级 l 层级计算的误差. 根据式(3)得到误差

$$e_1=\mathrm{lb}(d_2)-\mathrm{lb}(d_1)=\mathrm{lb}\left(\frac{d_2}{d_1}\right)=\mathrm{lb}\left(\frac{1+\alpha k}{\sqrt{1+k^2}}\right).$$

注意到 $k=\dfrac{t_x}{s_x}=\dfrac{\sin(\theta)}{\cos(\theta)}$, 可以将 e_1 写成是 θ 的函数并简化得到

$$e_1(\theta)=\mathrm{lb}(\cos(\theta)+\alpha\sin(\theta)) \tag{8}$$

由 k 值的范围, 不难得出 $\theta\in\left[0,\dfrac{\pi}{4}\right]$, 且 $e_1(\theta)$ 均匀变化. 容易证明式(8)具有单调性与连续性, 其层级均方误差公式为

$$e_2(\theta)=\frac{1}{\frac{\pi}{4}-0}\int_0^{\frac{\pi}{4}}\left(e_1(\theta)\right)^2\mathrm{d}\theta \tag{9}$$

分别把系数 0.5, 0.25 和 0.3284 代入式(9), 其层级均方误差值计算如表 2 所示.

表 2 层级计算误差对比分析

α	均方误差/%
0.5	+1.60
0.25	+0.38
0.3284	+0.28

从表 1 和表 2 中可以明显看出, 当 α 为 0.5 时, 计算斜边均方误差与计算层级均方误差都非常小; 当选择 α 为 0.5, 对最小直角边进行右移一位操作, 加上最长边就可以获得直角三角形的斜边长度; 而选择 α 为 0.25 时, 需要最小直角边右移 2 位. 从硬件实现的角度, 选择 α 为 0.5 资源占用的资源更小. 因此当 $s_x>t_x$ 时, d_x 斜边计算公式可变为

$$d_x=s_x+0.5t_x \tag{10}$$

同理, d_y 斜边计算公式可变为

$$d_y=s_y+0.5t_y \tag{11}$$

从式(1)(10)(11)可以明显看出, $d_{\min}$ 或者 $d_{\max}$ 只需要 2 次加法运算、2 次右移、2 次比较运算, 在保证误差的情况下, 可极大地减少运算量并易于硬件实现.

1.3 采样点计算

用式(2)计算纹理空间的纹素采样点的个数 N, 但是 N 中包含了除法计算, 计算复杂.

本文考虑覆盖区域的长边和短边比的对数

$$L=\mathrm{lb}\left(d_{\max}/d_{\min}\right) \tag{12}$$

式(12)确定了三线性插值中 MIP-MAP 2 级层级之间的插值系数.

为了硬件实现方便, 采样点的数目一般应该为 2^n ($n\geqslant 0$), 可以通过对 L 进行下取整[6] ($\lfloor L \rfloor$) 获得 $N=2^{\lfloor L \rfloor}$.

在透视投影中, 当像素远离视点时其映射后的纹理覆盖范围为长边远大于短边的不规则多边形. 这种情况下长边和短边比较大, 导致采样点个数增多, 增加了硬件实现的复杂度. 一般硬件上限制采用最大采样点数目 $N_{\max}$ [13], 则式(4)像素的纹理值为 $c=\dfrac{1}{\min(N,N_{\max})}\sum_{i=1}^{N}c(s,t)$.

1.4 对数计算改进

对数属于超越函数一种, 可以采用文献[14]算法实现. 本文采用线性逼近对数计算, 对于任意的正数 x, 均可变为 $(1+f_{\mathrm{r}})\times 2^n$ 的形式. 其中, f_{r} 表示纯小数. 对 $\mathrm{lb}(x)$ 变换形式, 得到

$$\mathrm{lb}(x)=\mathrm{lb}\left(2^n(1+y)\right)\approx n+y \tag{13}$$

其中, $y\in[0,1]$, 由式(13)得出 $\mathrm{lb}(x)$ 绝对误差为

$$e_3=|(n+y)-\left(n+\mathrm{lb}(1+y)\right)|=|y-\mathrm{lb}(1+y)|$$

$\mathrm{lb}(x)$ 均方误差值

$$e_4=\frac{1}{1-0}\int_0^1(e_3)^2\mathrm{d}y .$$

采用线性计算方式逼近对数, 计算 y 的均方误差, 其值为+3.12%.

1.5 整体误差分析

由于直角三角形的斜边计算改进和对数计算改进均会影响层级误差, 因此需要整体上计算改进后算法的层级与原算法的层级误差为 1 的概率以及对应 s_1 的范围.

在如图 2 所示的直角三角形中, 设 $s_x>t_x$, 记

(1) $t_x=ks_x$, $k\in(0,1]$,

(2) $s_x=2^M(1+2s_1)$, $s_1\in[0,0.5)$; 其中, M 为正整数. 采用原算法, 直角三角形斜边边长

$$d_x=\sqrt{s_x^2+t_x^2}=2^M(1+2s_1)\sqrt{1+k^2} \tag{14}$$

用对数计算层级, 把式(14)代入 Ewins 等[5]提出的基于各向同性滤波 MIP-MAP 层级的计算公式 $l=\mathrm{lb}(d_x)$, 得到

$$l(k,s_1,M)=\mathrm{lb}\sqrt{s_x^2+t_x^2}=M+\mathrm{lb}((1+2s_1)\sqrt{1+k^2}) \tag{15}$$

直角三角形斜边边长近似值

$$d_{x1}=s_x+0.5t_x=2^M(1+0.5k)(1+2s_1) \tag{16}$$

采用改进后的对数计算方法, 把式(16)代入式(13), 得到

$$l_1(k,s_1,M)=\mathrm{lb}(d_{x1})=M+\mathrm{lb}(1+2s_1+0.5k+ks_1)\approx M+2s_1+0.5k+ks_1 \tag{17}$$

显然, 式(15)与式(17)计算的层级相差最大为 1. 当层级差为 1 时, 可推导出

$$\begin{cases}\mathrm{lb}\left((1+2s_1)\sqrt{1+k^2}\right)\geqslant 1\\ 2s_1+0.5k+ks_1<1\end{cases} \tag{18}$$

$$\begin{cases}\mathrm{lb}\left((1+2s_1)\sqrt{1+k^2}\right)<1\\ 2s_1+0.5k+ks_1\geqslant 1\end{cases} \tag{19}$$

由于 $s_1\in[0,0.5)$, 结合式(18), 可推导出

$$\frac{1}{\sqrt{1+k^2}}-0.5\leqslant s_1<\frac{1}{1+0.5k}-0.5 \tag{20}$$

同理, $s_1\in[0,0.5)$ 结合式(19), 可推导出

$$\frac{1}{1+0.5k}-0.5\leqslant s_1<\frac{1}{\sqrt{1+k^2}}-0.5 \tag{21}$$

由于 $k\in(0,1]$, 容易证明

$$1+0.5k>\sqrt{1+k^2} \tag{22}$$

由式(22)可知式(20)不成立, 式(21)成立. 所以在 $k\in(0,1]$ 时, 可得当

$$s_1\in\left[\frac{1}{1+0.5k}-0.5,\frac{1}{\sqrt{1+k^2}}-0.5\right]$$

时, 计算层级相差为 1.

因此, $k\in(0,1]$, $s_1\in[0,0.5)$ 时, 计算层差为 1 的概率

$$p=\int_0^1\int_{\frac{1}{1+0.5k}-0.5}^{\frac{1}{\sqrt{1+k^2}}}\mathrm{d}s_1\mathrm{d}k=0.0704 \tag{23}$$

由式(22)计算出的概率可知, 采用改进后算法计算的层级与采用原算法计算的层级差值绝对值为 1 有 7%概率.

1.6 各向同性滤波实现各向异性滤波算法改进

改进算法步骤如下:

Step1. 计算纹理坐标相对应像素的偏导数 s_x, t_x,

s_y 和 t_y.

Step2. 按照式(1)(10)(11)计算单个像素映射到纹理空间覆盖区域的最长边和最短边.

Step3. 计算像素映射到纹理空间覆盖区域的长边和短边比的对数, 以及纹理空间的纹素采样点 N.

Step4. 确定 MIP-MAP 层级 l.根据最长边 $d_{\max}$ 来确定 MIP-MAP 基本层级 $B=\mathrm{lb}(d_{\max})$; 根据系统设置和长短边的对数比调整 B 值, 形成最后 MIP-MAP 层级 l.

Step5. 按照式(4)加权平均, 计算像素最终的纹理值.

其中, 当 MIP-MAP 层级 l 为整数时, 对每个采样点进行邻近点采样或双线性滤波; 当 l 为非整数时, 分别对 $\lfloor l \rfloor$ 和 $\lceil l \rceil$ 对应的层级采用邻近点采样或双线性插值, 并在 2 个层级之间进行线性插值, 即三线性插值. 根据采样点个数 N 把每个采样点计算出的纹理值进行加权平均, 求出像素的纹理值.

2 硬件架构设计

2.1 顶层设计

基于改进算法, 本文提出一种实现结构, 如图 3 所示. 图 3 中, 主要包括地址计算模块和滤波计算模块, 且纹理映射范围为 [0,1] [13]. 地址计算模块实现算法中 MIP-MAP 层级地址的计算, 采样点个数的计算以及采样点偏移量的计算; 滤波模块根据控制信息实现采样点的三线性插值、双线性插值或最近邻点采样, 并对采样值进行加权平均, 从而计算最终像素的纹理值. 图 3 中的参数预存模块用于存储初始化参数.

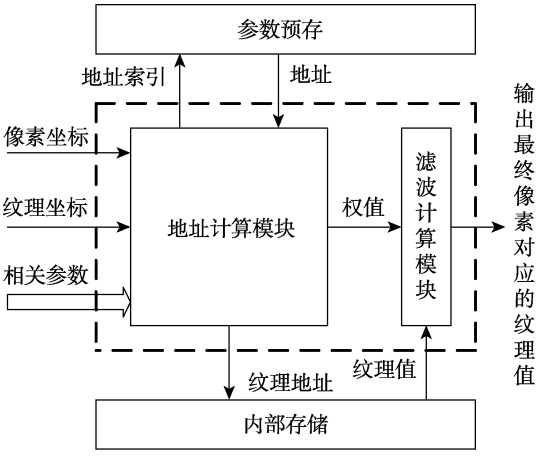


图 3 本文整体硬件设计架构

2.2 地址计算模块

地址计算模块的框架如图 4 所示. 本文架构针对双线性滤波, 当进行邻近点采样和三线性插值时仅需要对权值进行调整, 当进行纹理映射时依次处理一个像素. 该模块完成以下计算:

(1) 偏导数计算. 计算给定的纹理坐标 (s,t) 沿像素 x 方向和 y 方向的偏导数 s_x, t_x, s_y 和 t_y.

(2) 层级与纹素采样点个数计算. 根据给定的像素坐标和 s_x, t_x, s_y 和 t_y, 计算对应的 MIP-MAP 层级 l 以及采样点的个数 N.

(3) 权值计算. 根据屏幕分辨率设置和纹理采样点坐标、MIP-MAP 层级等信息, 计算邻近点采样、双线性插值和三线性插值所需要的邻近纹理采样点的权值.

(4) 采样点地址计算. 利用纹理采样点坐标的整数部分计算出 4 个相邻纹素的地址, 应用这些地址访问 4 个相邻纹理值.

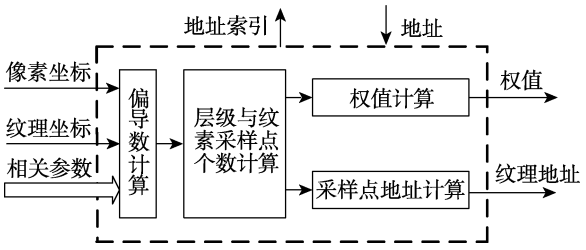


图 4 地址计算模块框架

2.3 滤波计算模块

滤波计算模块采用 7 级流水线, 根据从地址计算模块输出的权值和从存储单元取出的纹理颜色值计算每个采样点的纹理值; 再根据从地址计算模块输出采样点的个数 N 对所有采样点对应的纹理值进行求和取平均, 从而得到像素最终对应的纹理值.

3 实验结果及分析

本文设计的纹理反走样架构在 FPGA 开发板上进行了验证. 开发 OpenGL ES 2.0 应用程序, 在 Xilinx 的 ZC706 开发板 SOPC 系统上对一幅图形进行贴图处理, 并观察纹理走样效果. 原始纹理图如图 5 所示, 其分辨率为 256×256.

在 ZC706 开发板 SOPC 系统验证平台上实现的效果如图 6 所示. 在图 6 中, 纹理贴图的模式如下.

(1) GL_LINEAR. 在 MIP-MAP 基本层级上使用双线性滤波;

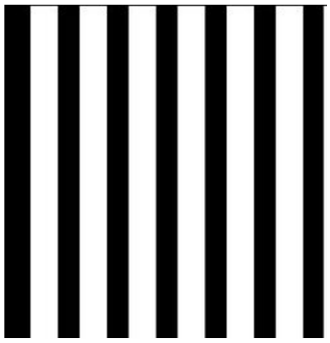


图 5　原始路型纹理

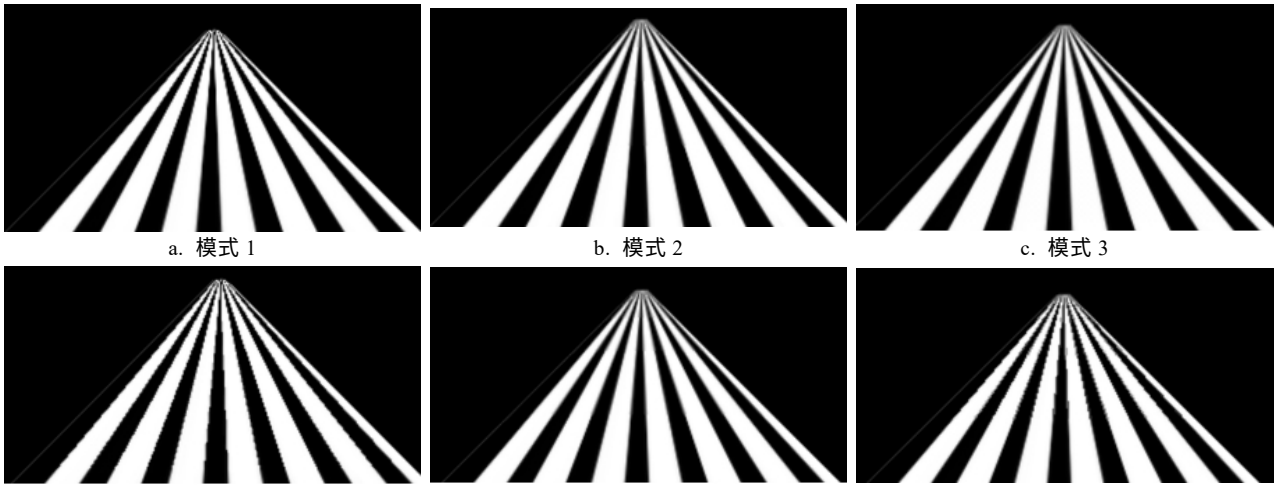
(2) GL_LINEAR_MIPMAP_LINEAR. 在选择的采样点上，MIP-MAP 层各自作双线性滤波之后，在 MIP-MAP 层之间使用线性插值，又称三线性滤波；

(3) GL_LINEAR_MIPMAP_NEAREST. 选择最邻近的 MIP-MAP 层之后，在该 MIP-MAP 层上进行双线性滤波；

(4) GL_NEAREST. MIP-MAP 层上使用最近邻点采样；

(5) GL_NEAREST_MIPMAP_LINEAR. 在选择的采样点上，MIP-MAP 层各自作最近邻点采样之后，在 MIP-MAP 层之间使用线性插值；

(6) GL_NEAREST_MIPMAP_NEAREST. 选择最邻近的 MIP-MAP 层之后进行最近邻点采样.



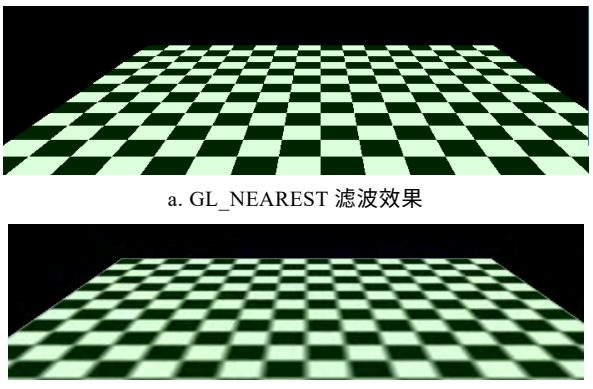
a. 模式 1　b. 模式 2　c. 模式 3

d. 模式 4　e. 模式 5　f. 模式 6

图 6　ZC706 开发板实现效果

从图 6 可以看出，纹理缩小时，GL_LINEAR_MIPMAP_LINEAR 滤波实现的效果最好.

棋盘格采用 GL_NEAREST 滤波和 GL_ LINEAR_MIPMAP_LINEAR 滤波效果进行比较. 如图 7 所示.

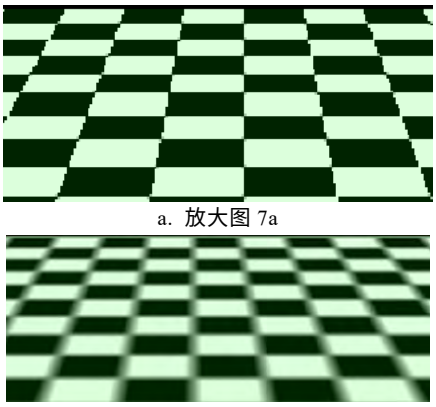


a. GL_NEAREST 滤波效果

b. GL_LINEAR_MIPMAP_LINEAR 滤波效果

图 7　棋盘纹理反走样实现效果

为了进一步对比图 7a 与图 7b 的滤波效果，分别对其进行放大观察，如图 8 所示. 从图 8a 可以明显看出，其边缘部分有明显的“锯齿”走样现象，边缘有明显的平滑效果. 本文算法可以很好地完成 3D 图形滤波, 3D 贴图的纹理反走样效果如图 9 所示.



a. 放大图 7a

b. 放大图 7b

图 8　棋盘纹理反走样实现局部放大效果

图 9 3D 纹理贴图三线性滤波反走样实现效果

本文设计的纹理反走样架构在 Xilinx ZC706 开发板 SOPC 平台上工作频率为 50 MHz, 资源消耗情况如表 3 所示. 为验证本文架构的面积与功耗, 在 Synopsys 公司的 Design Compiler 工具 0.13 μm SMIC 工艺进行综合, 如表 4 所示.

表 3 资源消耗情况

资源	估计	可利用	利用率/%
FF	672	437200	0.15
LUT	3357	218600	1.54
I/O	147	362	40.61
BUFG	1	32	3.12

表 4 DC 综合结果分析

频率/MHz	功耗/mW	面积/μm^2
166.67	54	1846160.13

4 结 语

本文对各向同性滤波逼近实现各向异性滤波算法中的覆盖区域边长计算及对数运算提出了易于硬件实现的算法, 用 1 次移位和 1 次加法实现了覆盖区域边长计算, 用 1 次加法实现对数计算, 减少了实现面积. 采用流水线设计并实现了一种适应于嵌入式设备的纹理反走样硬件架构, 在 Xilinx ZC706 FPGA 开发板上进行了验证.

参考文献(References):

[1] Han J H. 3D graphics for game programming[OL]. [2017-10-19]. https://www.crcpress.com/3D-Graphics-for-Game-Programming/Han/p/book/9781439827376.

[2] Du Huimin, Du Qinqin, Ji Kaibo, *et al*. Survey on the post-processing anti-aliasing techniques[J]. Journal of Xi'an Institute of Posts and Telecommunications, 2016, 21(1): 7-15(in Chinese)
(杜慧敏, 杜琴琴, 季凯柏, 等. 后处理反走样技术综述[J]. 西安邮电大学学报, 2016, 21(1): 7-15)

[3] Lv Y B, Xu J F. Research and application on the means of transparent texture anti-aliasing[C] //Proceedings of the 3rd International Symposium on Information Science and Engineering. Los Alamitos: IEEE Computer Society Press, 2011: 382-385

[4] Cant R, Langensiepen C, Rhodes D. Fourier texture filtering[C] //Proceedings of the 15th International Conference on Computer Modelling and Simulation. Los Alamitos: IEEE Computer Society Press, 2013: 123-128

[5] Ewins J P, Waller M D, White M, *et al*. MIP-map level selection for texture mapping[J]. IEEE Transactions on Visualization and Computer Graphics, 1998, 4(4): 317-329

[6] Bóo M, Amor M. High-performance architecture for anisotropic filtering[J]. Journal of Systems Architecture, 2005, 51(5): 297-314

[7] Olano M, Mukherjee S, Dorbie A. Vertex-based anisotropic texturing[C] //Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware. New York: ACM Press, 2001: 95-98

[8] Blinn J. Hyperbolic interpolation[J]. IEEE Computer Graphics and Applications, 1992, 12(4): 89-94

[9] Miller G S, Hoffman C R. Illumination and reflection maps: simulated objects in simulated and real environments[OL]. [2017-10-19]. http://www.pauldebevec.com/ReflectionMapping/illumap.pdf.

[10] Mavridis P, Papaioannou G. High quality elliptical texture filtering on GPU[C] //Proceedings of the Symposium on Interactive 3D Graphics and Games. New York: ACM Press, 2011: 23-30

[11] Crow F C. Summed-area tables for texture mapping[J]. ACM SIGGRAPH Computer Graphics, 1984, 18(3): 207-212

[12] Shin H C, Lee J A, Kim L S. SPAF: sub-texel precision anisotropic filtering[C] //Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware. New York: ACM Press, 2001: 99-108

[13] Ewins J P, Waller M D, White M, *et al*. Implementing an anisotropic texture filter[J]. Computers & Graphics, 2000, 24(2): 253-267

[14] Cao Guangjie, Du Huimin, Wang Pengchao, *et al*. Hardware implementation of elementary function approximation by using a piecewise cubic polynomial interpolator[J]. Journal of Computer-Aided Design & Computer Graphics, 2016, 28(1): 180-187(in Chinese)
(曹广界, 杜慧敏, 王鹏超, 等. 分段三次多项式逼近初等函数的硬件实现[J]. 计算机辅助设计与图形学学报, 2016, 28(1): 180-187)