

基于最优移位双线性插值的图像缩放旋转硬件加速研究

丁家隆, 杜慧敏, 曹广界, 王鹏超, 杜琴琴

(西安邮电大学 电子与信息工程学院, 陕西 西安 710121)

摘 要: 针对最优双线性插值算法进行精度优化, 采用 ARM A9 处理器软件定点实现比浮点提高 13.5 倍处理速度. 并提出一种高效的硬件实现结构, 在 Xilinx Zedboard 开发板上验证了该结构, 实验结果表明: 在典型情况下可以比 ARM A9 软件定点实现提高 9 倍速度.

关键词: 图像缩放; 图像旋转; 最优移位双线性插值; 硬件加速

中图分类号: TP391.41

文献标识码: A

文章编号: 1000-7180(2016)07-0115-06

The Hardware Acceleration Research of Image Zooming and Rotation Based on The Opt-shifting Bilinear Interpolation

DING Jia-long, DU Hui-min, CAO Guang-jie, WANG Peng-chao, DU Qin-qin

(College of Electronics and Information Engineering, Xi'an University of Posts & Telecommunications, Xi'an 710121 China)

Abstract: The paper optimize the accuracy of the opt-shifting bilinear interpolation. Implementation uses the ARM A9 processor software fixed-point implementation than 13.5 times floating point processing speed. Then the paper puts forward a highly efficient hardware implementation structure and implemented on Xilinx Zedboard. The experimental results show that in typical cases: hardware can 9 times faster than implementation by the ARM A9 software fixed-point.

Key words: image scaling; image rotation; opt-shifting bilinear interpolation; hardware acceleration

1 引言

图像缩放旋转处理在手机、便携式数字娱乐设备和工业控制等众多嵌入式应用领域中有着广泛的应用^[1]. 图像缩放中常见的方法分为非线性和线性两种.

非线性方法由于包含积分、二重积分等复杂运算, 运算量非常大, 故不利于硬件电路实现^[1-3].

线性方法中邻近取样法计算简单, 但效果很差, 造成了明显的马赛克效应^[4]; 双三次插值效果好, 但是计算量大, 每计算一个插值点的像素值, 需要至少 20 次乘法, 15 次加法^[5]; 标准双线性插值计算量适中, 但是缩放后的图像边缘模糊. 为了克服标准双线性插值算法的缺点, 保留其优点, Thierry Blu^[6]等人提出最优移位双线性插值算法, 该算法并不是调节

插值信号的移动, 而是使用一个统一的且与信号绝对独立的平移量, 在此平移量基础上进行采样, 并且该平移量存在一个最优值为 1/4. 最优移位双线性插值与标准的插值方法相比大大提高了图像质量, 并且图像质量与双三次插值的效果相当^[6].

本文在研究最优移位双线性插值算法基础上, 根据精度要求, 优化算法, 用软件定点实现该算法, 比浮点实现提高 13.5 倍的速度. 采用并行、流水、资源复用等思想设计了一种高效的硬件结构. 在 Xilinx Zedboard 开发板上分别采用 ARMA9 和本设计进行一系列软硬件实验, 实验结果说明: 该硬件结构效果理想, 加速明显, 资源占用少; 影响硬件加速比的主要因素为旋转角度和缩放倍数, 与图片大小无关.

收稿日期: 2015-09-09; 修回日期: 2015-10-27

基金项目: 国家自然科学基金项目(61136002); 陕西省科技发展计划项目(2011k06-47); 西安市科技发展计划项目(CXY1440(10)); 西安邮电大学创新基金项目(CXL2014-25)

2 最优移位双线性插值算法

本节介绍文献[6]提出来的最优移位双线性插值算法.

2.1 预处理

假设一张图片大小为 $v \times h$, 则每一行有 v 个点, 每个点的灰度值分别为 $f_0, f_1, f_2, \dots, f_{v-2}, f_{v-1}$, 通过预处理, 图像中的每个点得到新的计算系数. 每行第 n 个点的系数 c_n 计算如式(1)所示, 其中 τ 为平移系数, 并且文献[6]中指出图片缩放效果最好时, τ 值位于 0.21 左右, 可以取 $1/4$. f_n 为图像每行第 n 个点的灰度值. 每行的计算从第 0 个点开始计算, 在预处理中行处理后进行列处理. 在对一张图片进行过一次行处理后, 图片每个点都会得到一个 c 值. 此时再次利用式(1), 按列对图片进行一次列处理. 与行处理不同的是: 式(1)中的 f_n 变为了纵列第 n 个点在行处理中所计算出来的 c 值.

$$c_n = \begin{cases} f_0, n = 0 \\ -\tau \times c_{n-1} + (1 + \tau) \times f_n, 0 < n < v \end{cases} \quad (1)$$

2.2 坐标转换

预处理之后采用式(2)计算 x 轴和 y 轴的偏移系数, 其中 $x_{\text{deviation}}$ 和 $y_{\text{deviation}}$ 分别为 x 轴和 y 轴的偏移系数; x_{origin} 和 y_{origin} 为旋转中心点的 x 和 y 轴坐标; θ 为旋转角度; x_{shift} 和 y_{shift} 分别表示图片在 x 轴和 y 轴正方向上平移的距离; z_{oomx} 和 z_{oomy} 分别表示图片在 x 轴和 y 轴的缩放倍数.

$$\begin{aligned} x_{\text{deviation}} &= x_{\text{origin}} - \\ &\cos\theta \times (x_{\text{shift}} + x_{\text{origin}}) / z_{\text{oomx}} - \\ &\sin\theta \times (y_{\text{shift}} + y_{\text{origin}}) / z_{\text{oomy}} \end{aligned} \quad (2)$$

$$\begin{aligned} y_{\text{deviation}} &= y_{\text{origin}} - \\ &\sin\theta \times (x_{\text{shift}} + x_{\text{origin}}) / z_{\text{oomx}} - \\ &\cos\theta \times (y_{\text{shift}} + y_{\text{origin}}) / z_{\text{oomy}} \end{aligned}$$

利用偏移系数和式(3)可以计算出旋转缩放之后位于坐标 $(x_{\text{coord}}, x_{\text{coord}})$ 的点在未旋转缩放之前所在的采样位置 $(x_{\text{ss}}, y_{\text{ss}})$.

$$\begin{aligned} x_{\text{ss}} &= x_{\text{samp}} - \tau \\ y_{\text{ss}} &= y_{\text{samp}} - \tau \\ x_{\text{samp}} &= x_{\text{deviation}} - \\ &\sin\theta \times y_{\text{coord}} / z_{\text{oomy}} + \\ &\cos\theta \times x_{\text{coord}} / z_{\text{oomx}} \\ y_{\text{samp}} &= y_{\text{deviation}} + \\ &\cos\theta \times y_{\text{coord}} / z_{\text{oomy}} + \\ &\sin\theta \times x_{\text{coord}} / z_{\text{oomx}} \end{aligned} \quad (3)$$

图 1 为图片旋转示意图, 其中黑线框为原始图片, 虚线框为旋转后的图片. 需要注意的是, 如果 $x_{\text{samp}} \in (-\infty, -0.5] \cup [v - 0.5, +\infty)$ $\cup y_{\text{samp}} \in (-\infty, -0.5] \cup [h - 0.5, +\infty)$, 代表 $(x_{\text{coord}}, x_{\text{coord}})$ 这个点在旋转缩放后位于图 1 中右斜线部分, 即代表 $(x_{\text{coord}}, x_{\text{coord}})$ 点在原始图像中并不存在. 此时对于该点的像素值直接输出 0.

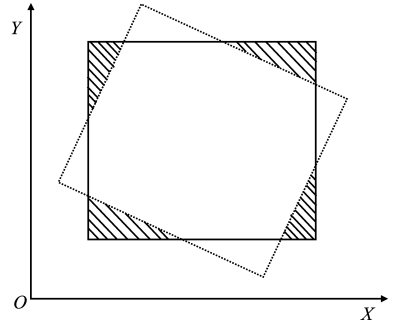


图 1 旋转示意图

2.3 插值系数计算

利用转换后的坐标 $(x_{\text{ss}}, y_{\text{ss}})$, 再利用式(4)求得图 2 中 $(x_{\text{ss}}, y_{\text{ss}})$ 临近左上角的坐标 (i, j) .

$$\begin{aligned} i &= \begin{cases} 0, [x_{\text{ss}}] < 0 \\ v - 1, [x_{\text{ss}}] \geq v \\ [x_{\text{ss}}], \text{other} \end{cases} \\ j &= \begin{cases} 0, [y_{\text{ss}}] < 0 \\ h - 1, [y_{\text{ss}}] \geq h \\ [y_{\text{ss}}], \text{other} \end{cases} \end{aligned} \quad (4)$$

由于 $(x_{\text{ss}}, y_{\text{ss}})$ 周围的 4 个点有可能超出图片外, 如果 4 个点有超出图片外的就取边界值, 所以 i 和 j 的取值有 3 种情况.

得到 i, j 以后, 通过式(5)即可求得双线性插值所需的插值系数 $w_{11}, w_{12}, w_{21}, w_{22}$, 这 4 个系数分别对应图 2 中的 $w_{11}, w_{12}, w_{21}, w_{22}$.

$$\begin{aligned} w_{11} &= x_{\text{ss}} - i \\ w_{12} &= 1 - w_{11} \\ w_{21} &= y_{\text{ss}} - j \\ w_{22} &= 1 - w_{21} \end{aligned} \quad (5)$$

2.4 双线性插值

$c_{11}, c_{12}, c_{21}, c_{22}$ 分别代表图 2 中 $(i, j), (i+1, j), (i, j+1), (i+1, j+1)$ 4 个点在经过预处理后得到的 c 值, 则目标图像 $(x_{\text{coord}}, x_{\text{coord}})$ 点处的像素值如式(6)所示.

$$\begin{aligned} f &= w_{22} \times (c_{11} \times w_{12} + c_{12} \times w_{11}) + \\ &w_{21} \times (c_{21} \times w_{12} + c_{22} \times w_{11}) \end{aligned} \quad (6)$$

对于采用 RGB 模式的彩色图片,只需对 RGB3 个分量单独运算,最后合成一幅完整的 RGB 模式图像.

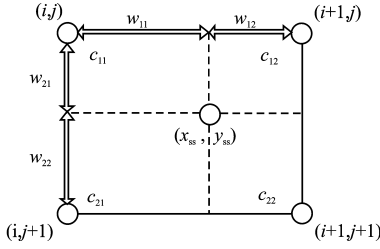


图 2 双线性插值原理图

3 算法优化

3.1 图像预处理优化

将平移系数 $\tau = 1/4$,代入式(1),得到式(7).

$$c_n = \begin{cases} f_0, n = 0 \\ f_n + 2^{-2} \times f_n - 2^{-2} \times c_{n-1}, 0 < n < v \end{cases} \quad (7)$$

式(7)是迭代计算,展开后得到 c_n :

$$\begin{aligned} c_n &= (1 + 2^{-2}) \times f_n - 2^{-2} \times (1 + 2^{-2}) \times f_{n-1} \\ &\quad \dots \dots + 2^{-2(n-i)} (1 + 2^{-2}) \times f_i \\ &\quad - 2^{-2(n-i+1)} \times (1 + 2^{-2}) \times f_{i-1} \\ &\quad \dots \dots + 2^{-2(n-1)} (1 + 2^{-2}) \times \\ &\quad f_1 - 2^{-2n} \times f_0 \\ &\quad (0 \leq i \leq n) \end{aligned} \quad (8)$$

从式(8)看出距离第 n 个点 $n-i$ 位置的点对 c_n 影响因子 IF 为式(9):

$$\text{IF} = 2^{-2(n-1)} (1 + 2^{-2}) \times f_i \quad (9)$$

我们可以看出: i 越大,IF 值越小,即离点 n 距离越远的点,对预处理计算影响越小.

采用软件分析式(10),得到 $|\text{sum}(\text{IF})_i|$ 最大值与 $n-i$ 的关系如表 1 所示.从表 1 可以看出距离预处理所计算点 10 个点以外的点的像素值,对计算影响已经小于 0.332.由于预处理后的 cn 值为图片像素,需要对小数部分进行 4 舍 5 入截断,得到 0 ~ 255 之间的整数.当 $|\text{sum}(\text{IF})_i|$ 最大值小于 0.5 时,对计算结果不造成影响所以,可以不考虑 10 点(包含第 10 个点)距离以外的点.

$$\begin{aligned} C_n &= (1 + 2^{-2}) \times f_n - 2^{-2} \times (1 + 2^{-2}) \times f_{n-1} \\ \dots \dots &2^{-2(n-i-1)} (1 + 2^{-2}) \times f_{i+1} + \text{sum}(\text{IF})_i \\ \text{sum}(\text{IF})_i &= 2^{-2(n-i)} (1 + 2^{-2}) \times f_i - \\ &2^{-2(n-i+1)} (1 + 2^{-2}) \times f_{i-1} \dots \dots + \\ &2^{-2(n-1)} (1 + 2^{-2}) \times f_i - \\ &2^{-2n} \times f_0 \end{aligned} \quad (10)$$

表 1 $|\text{sum}(\text{IF})_i|$ 最大值与 $n-i$ 的关系

$n-i$	$ \text{sum}(\text{IF})_i $ 最大值
5	10.625 0
6	5.312 5
7	2.656 3
8	1.328 1
9	0.664 1
10	0.332 0
11	0.166 0

这样在不影响计算结果的情况下,将一个递归算法变成一个非递归算法,将算法转化成了定点运算,硬件上用移位加法实现 c_n 的计算.

3.2 图像双线性算法精度选择

由双线性插值的原理图(图 2)可以看出 w_{11} 、 w_{21} 都是 0 ~ 1 之间的小数,式(6)中 c_{11} 、 c_{12} 、 c_{21} 、 c_{22} 是预处理后的图片像素,值为 0 到 255 之间的整数.再利用式(5)将式(6)改写成式(11).由于是 4 个正数相加,所以计算中不需要符号位.并且可以求得式(11)中 f 的极大值为 $256 = 2^8$,所以整数数位为 8 位.

$$\begin{aligned} f &= (1 - w_{21}) \times (1 - w_{11}) \times c_{11} + (1 - w_{21}) \times \\ &w_{11} \times c_{12} + w_{21} \times (1 - w_{11}) \times c_{21} + w_{21} \times \\ &w_{11} \times c_{22} \end{aligned}$$

$$0 \leq w_{11} \leq 1$$

$$0 \leq w_{21} \leq 1$$

$$0 \leq c_{11} \leq 255, \quad c_{11} \in z \quad (11)$$

$$0 \leq c_{12} \leq 255, \quad c_{12} \in z$$

$$0 \leq c_{21} \leq 255, \quad c_{21} \in z$$

$$0 \leq c_{22} \leq 255, \quad c_{22} \in z$$

采用浮点和定点分别计算式(11),然后求绝对误差,得到式(12).

$$E = f_{\text{float}} - f_{\text{fix}}$$

$$\begin{aligned} f_{\text{float}} &= (1 - w'_{21}) \times (1 - w'_{11}) \times c_{11} + \\ &(1 - w'_{21}) \times w'_{11} \times c_{12} + \\ &w'_{21} \times (1 - w'_{11}) \times c_{21} + \\ &w'_{21} \times w'_{11} \times c_{22} \end{aligned}$$

$$\begin{aligned} f_{\text{fix}} &= (1 - w_{21}) \times (1 - w_{11}) \times c_{11} + \\ &(1 - w_{21}) \times w_{11} \times c_{12} + \\ &w_{21} \times (1 - w_{11}) \times c_{21} + \\ &w_{21} \times w_{11} \times c_{22} \end{aligned} \quad (12)$$

$$w_{21} = [w'_{21} \times 0.5^{-i} + 0.5] \times 0.5^i$$

$$w_{11} = [w'_{11} \times 0.5^{-i} + 0.5] \times 0.5^i$$

$$0 \leq w_{11} \leq 1$$

$$0 \leq w_{21} \leq 1$$

$$0 \leq c_{11} \leq 255, \quad c_{11} \in z$$

$$0 \leq c_{12} \leq 255, \quad c_{12} \in z$$
$$0 \leq c_{21} \leq 255, \quad c_{21} \in z$$
$$0 \leq c_{22} \leq 255, \quad c_{22} \in z$$

式中, w_{11}, w_{21} 为浮点表示的 w 值, w_{11}, w_{21} 为定点表示的 w 值, i 为定点数的小数位宽. 采用软件建立数学模型后, 求得绝对误差最大值 E 与 i 的关系如图 3 所示.

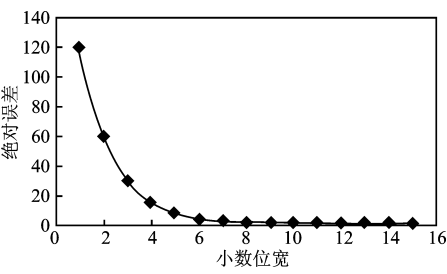


图 3 定点数小数位宽与绝对误差趋势图

图 3 可以看出, 当小数位宽达到一定范围时, 绝对误差就小于 0.5. 另一方面, 式(11)计算的是图片像素值, 最后输出值需要进行四舍五入的取整操作, 所以绝对误差小于 0.5 时, 增加小数位宽不影响计算结果. 从表 2 得出小数位宽取 9 即可保证计算的准确性.

表 2 定点数小数位宽与绝对误差

小数位宽(i)	绝对误差(定点数计算结果与浮点数计算结果)
5	7.486 43
6	3.741 22
7	1.801 72
8	0.943 51
9	0.447 94
10	0.234 10
11	0.113 40

综上所述, 双线性插值部分采用定点运算而不用浮点运算. 在设计中我们采用 17 位(8 位整数, 9 位小数)运算单元实现双线性插值. 该结论也适用于标准双线性插值.

4 硬件架构

本文设计可通过配置参数实现 2048×2048 以下图片的处理. 图 4 为硬件电路的顶层框图, 包括预处理模块、查找表模块、坐标转换模块、插值系数模块、地址产生模块和双线性插值模块. 对于彩色图片的处理, 我们实际将图片分为 R、G、B, 3 路进行并行处理.

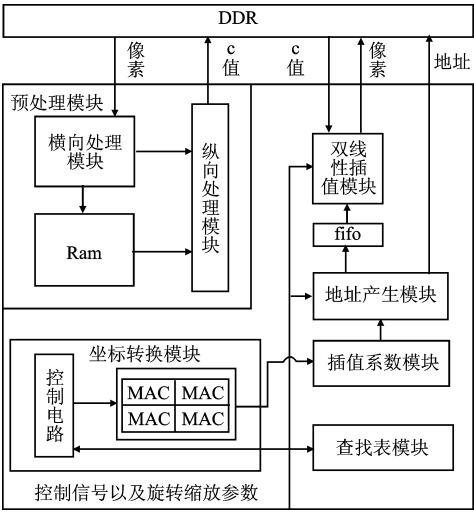


图 4 硬件电路顶层框图

4.1 预处理模块

预处理模块主要包含行处理模块、列处理模块以及 $1 \times v$ (v 为图片横向点数) 大小的 ram. 算法原理中列处理必须位于行处理之后. 为了提高电路速度, 本模块采用了一块 RAM 来实现并行预计算.

采用一块 $1 \times v$ (v 为图片横向点数) 大小的 ram 来缓存行处理后的输出值, 解决了列预处理必须在整个图片行预处理完成之后的问题, 实现了行、列预处理同时进行. 图 5 为预处理的一个示意图. 在 (i, j) 点行处理计算的同时, 将 $(i, j-1)$ 点预处理值从 ram 中读出来. 在下一个时钟周期, 利用 (i, j) 点行处理和 $(i, j-1)$ 点预处理值进行 (i, j) 列处理. 在列处理完成后, 将 (i, j) 预处理值写回 ram 覆盖 $(i, j-1)$ 点在 ram 中的预处理值, 同时也将 (i, j) 点预处理写回 DDR. 采用此策略后, 预处理可以比串行实现预处理提升一倍速度.

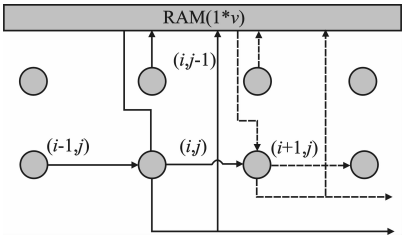


图 5 预处理示意图

4.2 双线性插值模块

双线性插值模块可以用定点运算单元完成, 在设计中采用了如图 6 所示的流水线结构. 其中加法器为 1 个节拍. 由于乘法器运算的时间限制了整个电路的工作频率, 所以, 采用 4 级流水线实现(17 位定点加法器一个节拍在 $0.13 \mu\text{m}$ 工艺下速度为 350

MHz,4 级流水的 17 位定点乘法器才能达到 350 MHz)以提高电路速度.图 6 中 MUL 代表乘法器,ADD 代表加法器.和乘法器同级的 DFF 为 4 个 D 触发器串联,和加法器同级为 1 个 D 触发器.

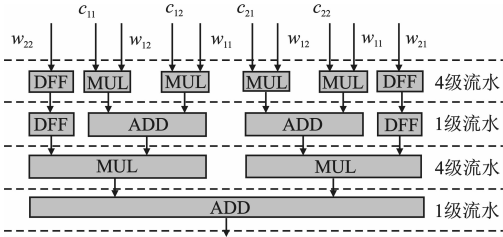


图 6 双线性插值流水示意图

4.3 查找表模块

在设计中采用查找表计算正余弦和倒数三个超越函数,原因在于:

- (1)缩放系数具有局限性,一般缩放系数都位于 0.5~4,线性插值在放大倍数过大以后效果不理想.
- (2)人的视觉感受对旋转角度精度要求不高.

倒数表和三角函数表的参数如表 3 所示,设计中利用正弦函数和三角函数曲线特点,可以用 0°~90°的正弦表实现 0°~360°的三角正余弦查找表,这样缩小了查找表的大小,提高了查找速度.

表 3 查找表参数		
表名称	深度	范围
倒数	56	0.125~7
正弦	723	0~90(度)

4.4 坐标转换模块

坐标转换模块实现了式(2)、(3).如图 3 所示,该模块由控制电路和 4 个定点 MAC 单元组成.每个 MAC 单元实现定点 $a * b + c$ 功能,MAC 单元位宽 22 位,其中 1 位符号位,整数位 12 位,小数位 9 位.小数位取 9 位是由于 2.2 节推论指出最终双线性插值的 w_{11}, w_{21} 精度只需要达到 9 位.整数位选取 12 位由于图片大小要处理 $2\,048 \times 2\,048$ 以下的图片,所以整数部分小于 $2\,048 \times 2 = 2^{12}$ (12 位整数就可以表示).

对于相同的旋转缩放控制系数,式(2)只需计算一次,而图片每个点都需要计算一次式(3),式(3)中的 x_{ss} 和 y_{ss} 必须在偏移系数计算结果出来之后再计算.另一方面本模块运算较多,如果要实现 x_{ss} 和 y_{ss} 的流水计算(假设偏移系数已经计算出来)必须要有 4 个定点 $a \times b + c$ 功能的 MAC 单元.所以,坐标转换模块复用了 4 个 MAC 单元先计算偏移系数再流

水计算(x_{ss}, y_{ss}),减少电路面积.

4.5 插值系数模块

插值系数模块采用两级流水,实现式(4)、(5).采用两级流水的原因是由于,该部分一级流水会成为电路延迟最大的一级,为了提高电路工作频率,插入一级寄存器.

4.6 地址产生模块

地址产生模块是利用插值系数模块得到的坐标 (i, j) 和控制信号中给出的 Base_addr(图片在 DDR 中的首地址),计算出 (i, j) 点在 DDR 中的地址.地址计算公式如下:

$$\text{addr}_{11} = v \times i + j + \text{Base_addr}$$

(13)

$$\text{addr}_{21} = v \times (i + 1) + j + \text{Base_addr}$$

式中, v 为图片每行的点数. addr_{11} 和 addr_{21} 分别为图 2 中的 c_{11} 和 c_{21} 在 DDR 内的地址.

4.7 fifo 的深度

图 3 中 fifo 的作用是解决 $w_{11}, w_{12}, w_{11}, w_{22}$ 和 DDR 读出的图片预处理 c 值之间的同步问题.由于读取 c 值的第一个地址是和 $w_{11}, w_{12}, w_{11}, w_{22}$ 一起计算出来的,但是 DDR 会在一定周期数后才能开始回写 c 值,然后第一组 $w_{11}, w_{12}, w_{11}, w_{22}$ 才能开始用于计算,所以我们需要一个 fifo 来缓存 $w_{11}, w_{12}, w_{11}, w_{22}$ 的值, fifo 的深度取决于图 7 中的 I 值(图 7 中 I 值为 4, fifo 深度设计为 4,实际的 I 值取决于接口).

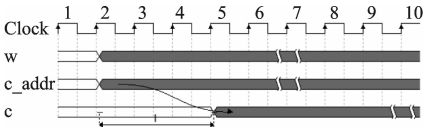


图 7 fifo 深度的影响因素 I

5 结果及分析

5.1 验证平台

验证平台框图如图 8 所示.在 ZedBoard 上面移植 Linux 系统,在 Linux 下读取图片像素信息到 DDR,之后硬件作为外设开始工作,并与 DDR 通信完成图片处理,最后 Linux 再通过 HDMI 将处理后的图片显示到显示器.

5.2 硬件性能分析

本设计在 SynopsysDC 下采用 0.13 μm 工艺库综合后工作频率为 350 MHz.如果针对一张 512×512 的图片,处理一张图片时间为

$$2.857(\text{ns}) \times 512 \times 512 = 0.000\,748(\text{s}).$$

由于本设计是作为操作系统的外设工作,因此

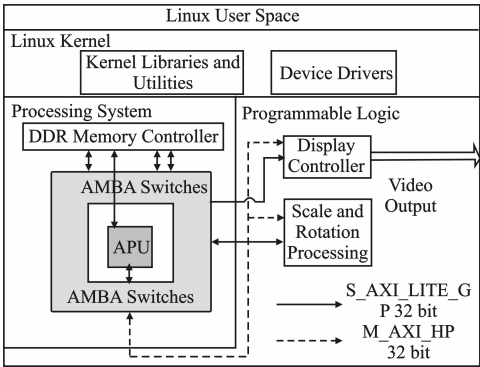


图 8 验证平台

构建的测试平台中,电路实际处理一张图片的时间与操作系统调度有关系,不能达到理想的计算时间。

本设计采用 32 位 AXI 总线,在总线接口速率为 100 MHz 环境下进行测试.定义两个指标来说明本设计的效率:

面积比=目标图像显示面积 / 原始图面积

硬件加速比=硬件执行时间 / 软件执行时间

其中原始图像显示面积指图 1 中的实线框,目标图像显示面积指图 1 中实线框减去右斜线部分。

图 9 所示为针对不同的旋转缩放参数加速比测试结果.对于不同图片大小,测试结果基本一致,其中横轴表示的面积比,纵轴表示的硬件加速比.从图 9 可以看出,目标图像显示区域较小时(面积比较小),硬件加速比较低.当目标图像显示面积与原始图像一样时(面积比为 1),硬件加速比达到最大值 120(倍)。

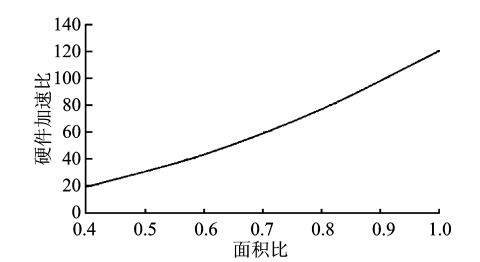


图 9 硬件加速比曲线图

对图 9 的结论的解释是:当出现图 1 情况时,即因为旋转或者缩放,图像中的部分像素超出了屏幕的显示区,因此不显示在屏幕上.这样,软件上无需再计算这些像素,即图 1 右斜线部分,减少软件代码的指令数量,从而减少软件执行时间,所以软件执行时间与旋转缩放参数和图片大小有关(还是与显示的区域有关系).硬件电路为流水处理,每个时钟周期计算一个点,所以执行时间只与图片大小有关。

如果设计中考虑接口的问题,对不在区域的显示做专门的处理,可以进一步减少在面积比较低时,硬件处理图像的时间。

表 4 所示为在面积比为 1 的条件下,采用 Zed-Board 开发板 ArmA9 处理器进行软件浮点执行和软件定点执行,以及采用本设计硬件执行时间的比较.执行时间根据 5 组典型的缩小放大参数下,4 次重复实验测试结果求平均得到。

表 4 软硬件执行时间比较(面积比为 1)

图片大小	软件浮点实现/s	软件定点实现/s	FPGA/s	加速比	软件定点实现与软件浮点实现比较	FPGA与软件定点实现比较	FPGA与软件浮点实现比较
128×128	0.92	0.069	0.0076	13.33	9.07	121.05	
256×256	3.75	0.27	0.031	13.88	8.70	120.96	
512×512	14.85	1.1	0.127	13.50	8.66	120.75	
1 024×1 024	59.99	4.5	0.502	13.33	8.96	119.50	
2 048×2 048	241.38	17.5	2.001	13.70	8.74	120.62	

5.3 与三线性插值的比较

针对和本文算法效果相似的双三次插值算法,如果硬件实现双三次插值算法,主要的资源占用来源于乘法器,双三次插值算法每个周期要输出一个像素,需要至少 20 个乘法器,而本文算法只需要 14 个乘法器.本文算法与双三次线性插值算法资源比较见表 5。

表 5 本文算法与双三次线性插值算法资源比较

资源	最优移位双线性插值	双三次线性插值
SliceLUTs *	3 775	5 880
SliceRegisters	3 222	6 660

本设计对彩色图片进行旋转缩放结果表明,图片在放大后没有马赛克效应,并且边缘清晰,没有明显的模糊现象。

6 结束语

针对高质量图像缩放处理时运算量大且执行时间长的问题,选用一种效果与双三次插值算法类似的最优移位双线性插值算法进行硬件加速研究.最后得出硬件加速比曲线以及影响因素.实验结果表明该方案加速明显、缩放效果好、资源占用低、适用性高.可以应用于实时性要求较高的图像处理场合。

(下转第 124 页)

调度过程中,利用改进算法进行实验,耗费的时间比传统算法短,控制精度较高.

5 结束语

通过软件调度优化控制设计可以减小内存开销,避免系统空间资源的浪费,提高资源分配能力. 以往的软件调度方法采用提前分配机制下静态规划控制方法,无法为所有软件调度用户提供分配空间,且效率较低. 本文对此提出一种基于关联博弈的软件调度线性规划控制算法. 利用仿真实验进行了性能测试和验证. 结果表明,采用本文算法能有效实现软件调度的关联博弈线性规划控制,耗费的时间短,控制精度较高,用户对对象对软件也的利用率也有所提高.

参考文献:

[1] 包晓安,谢晓鸣,张娜,等. 基于缺陷关联度的 Markov 模型软件优化测试策略[J]. 软件学报, 2015, 26(1): 14-25.

[2] 陈旖,许力,张美平. 适用于大规模无线传感器网的流量优化系统设计[J]. 计算机应用, 2015, 35(4): 905-909.

[3] 庄福振,罗平,何清,等. 迁移学习研究进展[J]. 软件学报, 2015, 26(1): 26-39.

[4] 程美英,倪志伟,朱旭辉. 萤火虫优化算法理论研究综述[J]. 计算机科学, 2015, 42(4): 19-24.

[5] 邓传华,范通让,高峰. Hadoop 下基于统计最优的资源调度算法[J]. 计算机应用研究, 2013, 30(2): 417-419.

[6] 薛又岷,严玉萍,古嘉玲,等. 两种基于 K 近邻特征选择算法的对比分析[J]. 电子设计工程, 2016, 22(1): 19-22.

[7] Chong S K, Gaber M M, Krishnaswamy S, et al. Energy conservation in wireless sensor networks: a rule-based approach[J]. Knowledge and Information Systems, 2011, 28(3): 579-614.

[8] Oliveira L M, Rodrigues J J. Wireless sensor networks: a survey on environmental monitoring[J]. Journal of Communications, 2011, 6(2): 143-151.

[9] Hackmann G, Sun F, Castaneda N, et al. A holistic approach to decentralized structural damage localization using wireless sensor networks[J]. Computer Communications, 2012, 36(1): 29-41.

[10] Zhang Guo-peng, Cong Li, Ding En-jie, et al. Fair and efficient resource sharing for selfish cooperative communication networks using cooperative game theory [C]// IEEE International Conference on Communications (ICC). kyoto: IEEE, 2011: 1-5.

作者简介:

刘海姣 女, (1982-), 讲师. 研究方向为计算机图形图像处理. E-mail: rtian1@163. com

赵书田 男, (1979-), 硕士, 讲师. 研究方向为计算机应用和软件编程.

(上接第 120 页)

参考文献:

[1] Haji-Saeed B, Woods C L, Kierstead J, et al. A differential interpolation image correction approach for bi-directional resonant scanners[C]// SPIE Defense, Security, and Sensing. International Society for Optics and Photonics. [s. l.]: IEEE, 2010.

[2] Drakopoulos V, Bouboulis P, Theodoridis S. Image compression using affing fractal interpolation on rectangular lattices[J]. Fractals-complex Geometry Patterns & Scaling in Nature & Society, 2011, 14(4): 259-269.

[3] Salehi Shirin, Mahdavi-Nasab, Homayoun. New image interpolation algorithms based on dual-tree complex wavelet transform and multilayer feedforward neural networks[J]. IJICIC Editorial Office, 2012, 8(10A): 6885-6902.

[4] Zhang J, Fan X, Huang C, et al. Multispectral image-superresolution reconstruction based on B-spline interpolation[C]// Sixth International Symposium on Mul-

tispectral Image Processing and Pattern Recognition. International Society for Optics and Photonics. [s. l.]: IEEE, 2009.

[5] 张阿珍, 刘政林, 邹雪城, 等. 基于双三次插值算法的图像缩放引擎的设计[J]. 微电子学与计算机, 2007, 24(1): 49-51.

[6] Blu T, Thévenaz P, Unser M. Linear interpolation re-vitalized[J]. Image Processing, IEEE Transactions on, 2004, 13(5): 710-719.

作者简介:

丁家隆 男, (1991-), 硕士研究生. 研究方向为电路与系统. E-mail: 904812006@qq. com.

杜慧敏 女, (1966-), 博士, 教授, 硕士生导师. 研究方向为计算机体系结构.

曹广界 男, (1990-), 硕士研究生. 研究方向为集成电路系统设计、计算机图形学.

王鹏超 男, (1990-), 硕士研究生. 研究方向为电路与系统.

杜琴琴 女, (1989-), 硕士研究生. 研究方向为计算机体系结构;