

基于叠加的图像缩放优化及 FPGA 实现

武文斌,高俊雄,王耘波

(华中科技大学 微电子学与固体电子学系,湖北 武汉 430074)

摘 要: 针对数字图像缩放处理消耗较多硬件资源的问题,利用坐标转换和叠加处理的方法,进行图像缩放的硬件设计,采用流水线设计,转换为两种整数倍缩放图像叠加的形式,使其在 FPGA 上更加易于实现.最后在 Xilinx 的 XC7A200T 芯片上对算法进行验证,得到数据后用 MATLAB 显示缩放后图像,效果良好.

关键词: 叠加;图像缩放;FPGA

中图分类号: TP391

文献标识码: A

文章编号: 1000—7180(2017)11—0128—03

An Optimized Image Scaling Based on Superposition and Its FPGA Implementation

WU Wen-bin, GAO Jun-xiong, WANG Yun-bo

(Dept. of Microelectronics and Solid State Electronics, Huazhong University of Sci. & Tech., Wuhan 430074, China)

Abstract: Aiming at solving the problem of hardware resource consumption in the feild of image scaling, the hardware design of the image scaling is carried out by using the method of coordinate transformation and superposition. Besides we can make it easy to implement on FPGA in the form of two interger times scaled images overlaying, with the pipeline architecture design. Finally the algorithm is verified on the Xilinx XC7A200T chip and the data is transfromed to images with MATLAB which confirms at can work well.

Key words: superposition;image scaling;FPGA

1 引言

随着互联网的广泛应用以及信息数据化的普及,数字图像处理被应用到各种科学研究和社会生活领域,而图像缩放技术是其重要的基础内容之一.常用的灰度数字图像处理方法分为基于插值和基于边缘两类,前者包括最近邻域法、双线性插值、双三次插值等,后者包括边缘缩放法等^[1].由于如今在许多应用领域中对数字图像处理的效果和实时性要求越来越高,因此常用硬件来实现图像的缩放.在算法的硬件实现方面,主要考虑的是缩放效果和硬件资源使用两方面,其中基于插值算法的硬件实现相对简单,但最近邻域法存在马赛克和锯齿^[2],双线性插值法高频信息丢失严重,双三次插值计算量大^[3],基于边缘算法计算较为复杂,硬件上难以实现.

本文借助刘强提出的任意比例图像缩放算

法^[4],设计了一种通过处理目标图像(即缩放后的图像 ImageObj)坐标转换到源图(ImageSou)坐标,再进行叠加等一系列处理的形式,从而实现了将缩放图像转化为两个整数倍缩放图像叠加,相对于双线性插值算法,避免了对浮点数的处理,硬件实现上资源消耗更少,并且处理后的图像更清晰.

2 算法原理与优化

刘强等人在算法理论方面,证明了对于任意自然数 $M、N(M > N)$,必存在非负整数 $K、L、a$ 使得式(1)成立,其中, $a = [M/N]$ (其中 $[]$ 为取整), $L = M \% N, K = N - L$:

$$\frac{N}{M} = \frac{(K+L)}{K * a + L * (a+1)} \tag{1}$$

由上式可以得到任意比例系数的缩放,均可以由两种整数倍的缩放图像(ImageSca1、ImageSca2)

的叠加来完成^[4].

根据这一理论,实现算法的关键是得到 ImageObj、ImageSca1、ImageSca2 三个图像数据之间的关联规律.我们的做法是,首先寻找 ImageObj 的坐标地址(X_Obj,Y_Obj)与 ImageSca1 的坐标地址(X_Sca1,Y_Sca1)和 ImageSca2 的坐标地址(X_Sca2,Y_Sca2)之间的规律,从而将 ImageObj 中的每个像素点映射到 ImageSca1 和 ImageSca2 中的相对应像素点,完成坐标转换的第一步.然后,寻找(X_Sca1,Y_Sca1)和(X_Sca2,Y_Sca2)与 ImageSou 的坐标地址(X_Sou,Y_Sou)之间的规律,根据双线性插值算法,很容易将 ImageSca1 和 ImageSca2 中的像素点映射到 ImageSou 中的相对应像素点,完成坐标转换的第二步,继而将 ImageObj 与 ImageSou 联系在一起.

具体转换关系图(以 4/5 缩小为例)如图 1 所示,第一次坐标转换表达式见式(2)(其中[]为取整),第二次坐标转换表达式见式(3).

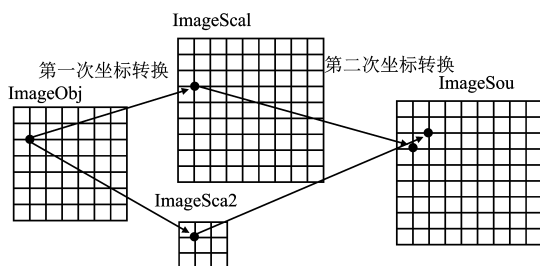


图 1 4/5 缩放转换图

$$\begin{cases} X_{Sca1} = \left[X_{Obj} * \left(\frac{M}{a} \right) + \frac{1}{2} \right] \\ Y_{Sca1} = \left[Y_{Obj} * \left(\frac{M}{a} \right) + \frac{1}{2} \right] \end{cases} \quad (2)$$

$$\begin{cases} X_{Sca2} = \left[X_{Obj} * \left(\frac{M}{a+1} \right) + \frac{1}{2} \right] \\ Y_{Sca2} = \left[Y_{Obj} * \left(\frac{M}{a+1} \right) + \frac{1}{2} \right] \end{cases}$$

$$\begin{cases} X_{Sou1} = X_{Sca1} * a \\ Y_{Sou1} = Y_{Sca1} * a \\ X_{Sou2} = X_{Sca2} * (a+1) \\ Y_{Sou2} = Y_{Sca2} * (a+1) \end{cases} \quad (3)$$

对于 4/5 缩小,由式(1)可以得到 $N=4, M=5, a=1$,即 $4/5 = (3+1)/(3*1+1*2)$,故 ImageObj 为 3 个 1/1 缩放图像 ImageSca1 和 1 个 1/2 缩放图像

ImageSca2 的叠加.将 M, N, a 值代入式(2)和式(3)完成坐标转换,为了在硬件更加容易实现,进行一定的转化,见式(4)和式(5),在得到像素数据后,叠加再做除以 4 处理.

$$\begin{cases} X_{Sca1} = \left[X_{Obj} + \left\lfloor \frac{X_{Obj} + 1}{2} \right\rfloor \right] \\ Y_{Sca1} = \left[Y_{Obj} + \left\lfloor \frac{Y_{Obj} + 1}{2} \right\rfloor \right] \end{cases} \quad (4)$$

$$\begin{cases} X_{Sca2} = \left[\frac{X_{Obj} * 4 + X_{Obj} + 4}{8} \right] \\ Y_{Sca2} = \left[\frac{Y_{Obj} * 4 + Y_{Obj} + 4}{8} \right] \end{cases}$$

$$\begin{cases} X_{Sou1} = X_{Sca1} \\ Y_{Sou1} = Y_{Sca1} \\ X_{Sou2} = X_{Sca2} * 2 \\ Y_{Sou2} = Y_{Sca2} * 2 \end{cases} \quad (5)$$

3 算法的 FPGA 实现

图像缩放系统由双口 BRAM(DBRAM)、坐标转换一模块、坐标转换二模块、地址信号产生模块、数据处理模块等构成,见图 2.工作过程:首先,源图像素数据信号 Data_Sou 与一个 Start 信号输入,Start 在 Data_Sou 之前,检测到 Start 信号后,产生写使能与地址信号将 Data_Sou 存入 DBRAM 中. ImageObj 坐标地址(X_Obj,Y_Obj)不断自加,同时将 X_Obj、Y_Obj 输入坐标转换一模块,得到两种缩放图像 ImageSca1、ImageSca2 对应的坐标地址(X_Sca1,Y_Sca1)和(X_Sca2,Y_Sca2),再通过坐标转换二模块,得到坐标地址(X_Sou1,Y_Sou1)和(X_Sou2,Y_Sou2),由地址信号产生模块生成对应的地址信号 addr1 和 addr2,输入到 DBRAM 中,读出 2 个像素数据 Data_Sca1 和 Data_Sca2,经过叠加和除法处理得到 ImageObj 中坐标地址为(X_Obj,Y_Obj)的像素数据 Data_Obj,存入 BRAM 中,等待 VGA 模块调用,此设计为流水线形式,每个时钟周期完成 ImageObj 中一个像素数据的处理.

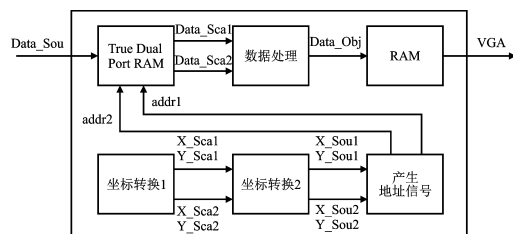


图 2 缩放图像系统结构

更具体的缩放模块实现见图 3,根据变换后的表达式(4)和表达式(5),设计硬件实现,“/2”为“>>1”,“/4”为“>>2”,“* 2”为“{,1'b0}”,“* 4”为“{,2'b0}”,设计中使用了一个乘法器,Y_Sou1 和 Y_Sou2 轮流使用,使用信号 YC_Val1 和 YC_Val2 从乘法器输出中取出各自的结果。

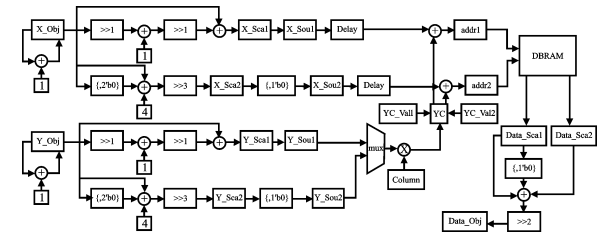


图 3 缩放模块实现

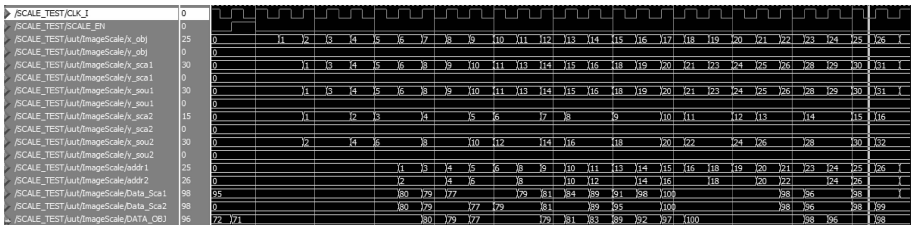


图 4 仿真结果图

5 结束语

本文将一种图像缩放算法成功的进行了 FPGA 实现,通过坐标转换和像素数据叠加的方式,将任意比例的图像缩放转化为两个整数倍图像缩放的叠加,从而避免了双线性插值中的复杂浮点数运算,减小了硬件开销,且尽量保留了源图中的高频分量.验证结果表明,图像缩放达到了较好的效果。

参考文献:

[1] 张梁,王景存,梅镖. 视频缩放在 FPGA 中的应用和实现[J],电子技术应用,2016,42(6):34-37.
[2] 王平,全吉成,赵柏宇. 基于双线性插值的图像缩放在 GPU 上的实现[J]. 微电子学与计算机,2016,33(11):

4 实验结果

本缩放图像设计在 Xilinx 的 Artix7 系列的 XC7A200T 芯片上进行验证. 首先在 MATLAB 上将 640×480 分辨率灰度图像的数据生成,形式为 16 进制的一维数组,在 Modelsim 上仿真时将数据读入,仿真过程中将 Data_Obj 写入文件“data_out_file.txt”,仿真结束后在 MATLAB 上读取“data_out_file.txt”,生成 512×384 分辨率灰度图. 仿真结果图见图 4,在 ISE14. 7 环境下综合,频率达到 250. 8 MHz.

129-132.
[3] 丁家隆,杜慧敏,曹广界,等. 基于最优移位双线性插值的图像缩放旋转硬件加速研究[J]. 微电子学与计算机,2016,33(7):115-124.
[4] 刘强,晋兆虎. 基于 FPGA 的任意比例图像缩放算法[J]. 数字技术与应用,2011,29(12):126-127.

作者简介:

武文斌 男,(1991-),硕士研究生. 研究方向为数字图像处理. E-mail:283078893@qq. com.
高俊雄 男,(1974-),讲师,硕士生导师. 研究方向为数字图像处理.
王耘波 男,(1965-),副教授,硕士生导师. 研究方向为数字图像处理.