

文章编号:1007-2780(2009)06-0882-04

基于 FPGA 的高分辨实时监控图像缩放设计

胡小龙,冯 彬

(中南大学 信息科学与工程学院,湖南 长沙 410075, E-mail: xinke013@163.com)

摘 要:介绍了一种基于图像的双三次线性插值缩放算法的设计方法,并通过 FPGA 验证了设计的可行性。重点讨论了视频缩放的插值算法,对两种实现方法在硬件资源利用率及实施效率方面进行了比较并论证了块状插值实现方法的优越性。最终设计实现了高分辨率实时视频图像的缩放。

关 键 词:图像缩放;现场可编程门阵列;双三次插值
中图分类号: TP331.2; TP752.1 **文献标识码:** A

1 引 言

实时视频图像的监控现在已在很多领域得到应用^[1]。近年来,现场可编程门阵列(Field Programmable Gate Array, FPGA)在实时图像处理中的应用越来越多。一些 FPGA 器件不仅提供了丰富的逻辑资源,还提供了硬件乘法器和大量的片上存储器,为实现高性能的高效率数字信号处理功能提供了方便,而在 FPGA 设计过程中需要对 FPGA 的优化设计加以考虑^[2]。

图像插值是图像处理的一个基本问题^[3]。计算机图像处理、图像渲染、医用图像的重构都需要使用到图像插值^[4,5]。常用的图像插值方法包括最近邻插值、双线性插值、双三次插值^[6-8]以及 B 样条插值等^[9]。

本文介绍了一种基于图像的双三次线性插值缩放算法的设计方法,并通过 FPGA 验证了设计的可行性,实现了高分辨率实时视频图像的缩放。

2 双三次线性图像缩放算法原理及 FPGA 设计

2.1 算法原理

三次插值的名称来自于其计算时最高次幂为三次。在数值分析中,插值算法可以表示为:

$$f(x) = \sum_{k=0}^{n-1} h(x) \times S_k \quad (1)$$

其中, $h(x)$ 为插值系数, S_k 为第 k 个原函数的值。

不同的插值算法根据所取像素点个数不同有不同的算法思想。插值图像的每一个像素点 $g(i, j)$ 都可以在原始图像中找到一个 $n \times n$ 大小的对应邻域。输出像素 $g(i, j)$ 可以通过对其原始图像中对应邻域进行二维卷积运算得到。

双三次线性插值的核心在于求得插值系数 h , 所采用的数学模型如下:

$$h(x) = \{a|x|^3 + b|x|^2 + c|x| + d\}, \\ x \in (m, n) \quad (2)$$

算法采用 4×4 大小领域进行计算,每得到一个输出像素点灰度值都需要取输入图像 $4 \times 4 = 16$ 个像素点的灰度值。常用的差值系数 h 的数学模型为:

$$h(x) = \begin{cases} 1 - 2|x|^2 + |x|^3, & 0 \leq |x| < 1 \\ 4 - 8|x| + 5|x|^2 + |x|^3, & 1 \leq |x| < 2 \\ 0, & \text{其他} \end{cases} \quad (3)$$

图 1 为算法实现示意图,已知 f 代表原始图像像素点灰度值, F 代表插值像素灰度点。

事实上,由于二维卷积模板在 x 和 y 方向可以分离,因此二维卷积运算也可以进一步分解为 2 次一维卷积来实现,即:对 4×4 邻域中的图像数据首先沿行的方向进行逐行的一维卷积,卷积的结果再沿列的方向进行一次一维卷积以得到最终的输出像素值,如图 2 所示。

双三次插值的最大计算量在于对于每个输出像素点都需要计算其水平和垂直方向的 h 值,

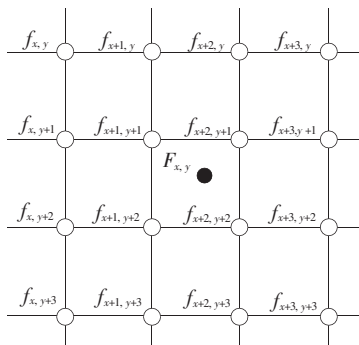


图 1 算法实现示意图
Fig. 1 Algorithm diagram

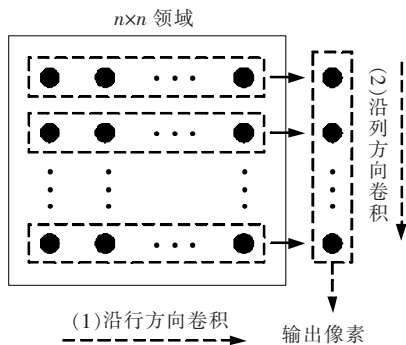


图 2 二维卷积实现示意图
Fig. 2 Two-dimensional convolution diagram

h 值的计算涉及大量的乘方和乘法运算,会占用大量的系统资源,而点逼近算法则在一定程度上简化了计算 h 的过程。点逼近算法将 $[0,1]$ 空间分成若干等份,这里是 8 份,即 8 个区间: $[0,1/8)$, $[1/8,2/8)$, $[2/8,3/8)$, $[3/8,4/8)$, $[4/8,5/8)$, $[5/8,6/8)$, $[6/8,7/8)$, $[7/8,1)$ 。落在某一区间的所有点采用同一 h 值。 h 值的计算对每幅输出图像计算一次,只计算 $0,1/8,2/8,3/8,4/8,5/8,6/8,7/8$ 这 8 个位置的值。将计算得到的值存

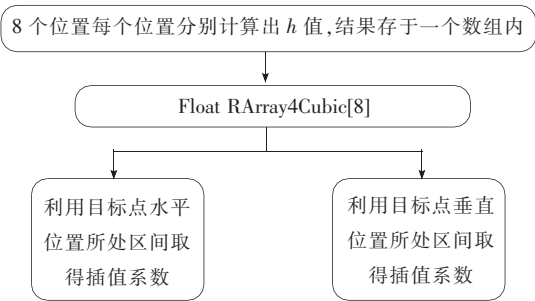


图 3 双三次插值系数计算过程
Fig. 3 Calculation of bicubic interpolation coefficient

储于数组 RArray4Cubic 内。图 3 为 h 值计算过程的流程图。

2.2 FPGA 设计

实时监控每秒传输 50 帧 $1\,280 \times 1\,024$ 分辨率的图像,为了能够满足系统要求选用 Xilinx 公司的 Virtex4 系列 XC4VLX100 的 FPGA 芯片实现实时缩放。图 4 是系统硬件总体结构图。控制模块用来实现输入输出数据控制,并且根据接收到的缩放倍数数据调整插值计算模块工作。

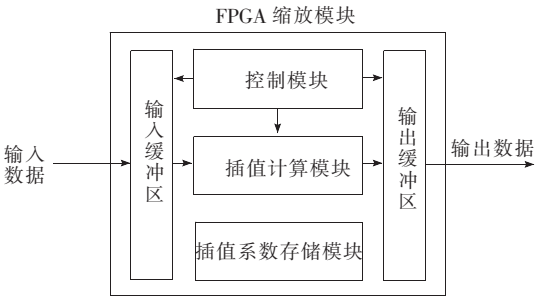


图 4 系统硬件设计图
Fig. 4 Diagram of system hardware design

要实现每秒传输 50 帧 $1\,280 \times 1\,024$ 分辨率的图像,FPGA 内部存储空间尚不能达到要求,需要外部存储器,并且需要设计存储控制器。

3 两种实现方法的比较研究

在 FPGA 设计中,更少地消耗 FPGA 内部逻辑资源数量和在 FPGA 上稳定运行时所能达到的更高频率是一对矛盾,也就是面积和速度的矛盾。为了在充分利用系统资源的同时尽量提高系统的综合频率,要求在设计中能够对二者进行综合评价。

与面积和速度相对应,本文采用的双三次图像插值算法在 FPGA 上采用了如下的两种实现方法:

(1)带状插值

将目标图像像素一次一行进行计算(图 5)。所要计算的像素点需要的源像素不进行存储,而通过与存储器的通信完成。这种与存储器的频繁通信需要大量的握手信息,要实现图像实时显示需要设计一个比较长的流水线。这种实现方法每次只需要存储计算目标像素点的 16 个原始像素点,节约了片上的 Block RAM 资源,是一种以速度换取面积的算法思想。

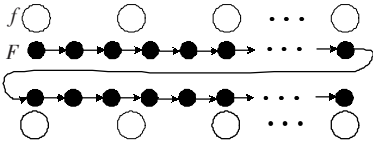


图 5 带状插值实现示意图
Fig. 5 Band interpolation diagram

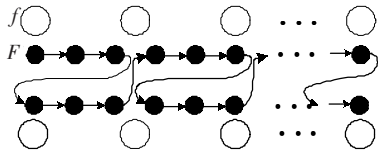


图 6 块状插值实现示意图
Fig. 6 Block interpolation diagram

(2)块状插值

将目标图像像素分块进行计算(图 6)。一次性尽可能多地读入原始像素点存在 FPGA 片上 RAM 资源中,可以避免与存储器的频繁通信。由于算法是以 4 行为一组进行插值点的计算,因而在 FPGA 内部开辟一部分空间来存储 4 行原始图像的像素点。根据算法可知,算出一个插值点需要 16 个原始像素点,而若以 16 个原始像素点为一个单位块,计算出所有在块内的插值点则可以最大限度地减少与外部存储器的通信,从而有利于系统的实时监控。这是一种以面积换取速度的算法思想。

一个设计消耗的 FPGA 内部逻辑资源的数量,可以用消耗的触发器和查找表的个数或者是等效逻辑门数来衡量。表 1 指出了在一路实时视频图像情况下两种实现方法系统资源占用情况。由表 1 可以看出,块状插值比带状插值多占用了 3~4 倍的系统资源。经过综合测试,两种方法均可以很好地完成 1 280×1 024 分辨率下 50 帧/s 的图像进行实时的放大缩小工作。对于多路的实时视频图像缩放来说,带状插值会由于频繁与外部存储器通信而影响系统的实时效果,而块状插值相对于带状插值进行多路视频图像的缩放受到系统资源限制则更加明显。

表 1 两种实现方法系统资源占用比较表

Table 1 Comparison of the system resources occupied realized by tow methods

系统资源	可使用(个)	带状插值使用(个)	带状插值利用率(%)	块状插值使用(个)	块状插值利用率(%)
Slice	49 152	829	1.7	3 617	7.3
触发器	98 304	738	0.8	2 792	2.8
4 输入 LUT	98 304	1 665	1.7	6 736	6.8
DSP48	96	24	25	30	62

4 实验结果

实验中使用一片 Xilinx 公司的 Virtex4 系列 XC4VLX100 的 FPGA 芯片。4 片 2M×4bank×32 bit 的 hy5duDDR,用来进行 4 路视频信号的帧缓存;一片 AD9888 进行视频信号采集;一片 ADV7401

将处理后数字视频信号转化为 VGA 信号在显示器上显示。实验流畅地实现了视频信号的缩放处理。其实际效果如图 7,8,9 所示。



图 7 原始图像
Fig. 7 Original image



图 8 放大 2.25 倍后的图像
Fig. 8 Enlarged 2.25 times image



图 9 缩小 0.25 倍后的图像
Fig.8 Narrowed 0.25 times image

5 结 论

介绍了基于 FPGA 实现双三次插值的模块,

参 考 文 献:

[1] 张广拓. 嵌入式网络视频监控系统的研究与设计[D]. 哈尔滨:哈尔滨工程大学,2007.

[2] Ian Kuon, Jonathan Rose. Area and delay trade-offs in the circuit and architecture design of FPGAs [C]// *Proceedings of the 16th International ACM/SIGDA Symposium on Field Programmable Gate Arrays*, New York:ACM, 2008:149-158.

[3] Castleman Kenneth R. 数字图像处理 [M]. 朱志刚,林学闾,林定机,译. 北京:电子工业出版社,1998.

[4] Acharya Tinku, Tsai Ping-Sing. *Computational Foundations of Image Interpolation Algorithms* [M]. New York: ACM Ubiquity, 2007:1-17.

[5] Raanan Fattal. Image up sampling via imposed edge statistics [J]. *ACM Transactions on Graphics*, 2007, 26(3):1-7.

[6] 张辉,胡广书. 基于二维卷积的图像插值实时硬件实现 [J]. 清华大学学报(自然科学版), 2007, 47(6):33-36.

[7] 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.

[8] Keys Robert G. Cubic convolution interpolation for digital image process [J]. *IEEE Trans. on ASSP*, 1981, 29 (6):1153-1160.

[9] Park Stephen K, Schowengerdt Robert A. Image reconstruction by parametric cubic convolution [C]//*Computer Vision, Graphics, and Image Processing*, Amsterdam:Elsevier, 1983:258-272.

Design on FPGA Based High-Resolution and Real-time Monitoring Image Zooming

HU Xiao-long, FENG Bin

(School of Information Science and Engineering, Central South University, Changsha 410075, China,E-mail :xinke013@163.com)

Abstract

An interpolation algorithm used in image zooming was introduced and a real-time monitoring image zooming based on FPGA was presented. The algorithm was discussed. Both of the implementations were compared to verify in hardware resource utilization and efficiency, and demonstrated the superiority of the block interpolation. The system is proved to be stable when it is applied to the monitoring image zooming with real-time and high-resolution.

Key words:image zooming; FPGA; bicubic interpolation

作者简介:胡小龙(1969—),男,湖南常德人,博士,副教授,研究方向为嵌入式系统、计算机网络。