边缘检测算法的 FPGA 实现①

周光宇[1], 刘慧忠[2]

[1](宁波大红鹰学院 机械与电气工程学院, 宁波 315175)
[2](陕西科技大学 电气与信息工程学院, 西安 710021)

摘 要: 边缘检测算法是最常见的图像处理算法. 分析了 Prewitt 算子和 Sobel 算子两种边缘检测算法的原理, 利用 DSP Builder 工具在 MATLAB/Simulink 下完成了这两种算子的图像边缘检测算法的设计, 并将该设计直接在 FPGA 中实现. 利用 FPGA 的并行处理能力, 大大缩短了图像处理所需的时间.

关键词: 边缘检测; 图像处理; 现场可编程门阵列; 算法; DSP Builder

Implementation of Edge Detection Algorithm in FPGA

ZHOU Guang-Yu[1], LIU Hui-Zhong[2]

[1](Ningbo Dahongying University, Ningbo 315175, China)
[2](Shaanxi University of Technology &Science, Xi'an 710021, China)

Abstract: Edge detection algorithm is the most common image processing algorithms. The principles of two kinds of edge detection algorithm—Prewitt operator and Sobel operator are analyzed. The DSP design of the image edge detection algorithm is completed by using DSP Builder tool in a friendly environment MATLAB/Simulink. And then the design is implemented directly in FPGA. It can shorten the image processing time with the capability of parallel process by using FPGA to process image.

Key words: edge detection; image process; FPGA; algorithm; DSP builder

0 引言

数字图像处理在现代社会各个领域扮演着极其重要的角色, 尤其是在汽车、制造、包装、造纸、医疗成像等领域. 随着计算机计算能力的增加以及成本的降低, 数字图像处理的应用范围呈爆炸式增长, 从工业检测到医疗器械, 都成为了数字图像处理的应用领域[1]. 边缘检测是数字图像处理的一个最基本的处理方法, 作用在于识别出图像中亮点变化明显的点, 描绘出图像中物体的轮廓[2]. 传统的图像处理平台基于串行计算机结构, 它的工作方式为将操作级算法分解为一连串的有 ALU(Arithmetic logic Unit)执行的算术或逻辑电路操作来串行执行所有的计算, 而 CPU 的其他部分则为 ALU 提供必要的数据. 无论图像的尺寸和数量是多少, 早期的串行计算机系统由于速度太慢而不能用于处理图像. 图像的规则机构使人们考虑利用硬件系统进行设计以及开发其并行性, 这是因为硬件系统本质上是并行的. 基于硬件实现的图像处理系统速度非常快, 但它们最大的问题是存在相对的不灵活性. 一旦配置完成, 这些系统能很好的执行任务, 但是如果任务的本质发生改变, 再想重新配置这些系统, 即使有可能实现也是非常困难的[3][4]. 20 世纪 80 年代, FPGA(Field Programmable Gate Array)技术的引入为数字逻辑设计开创了新的可行性. FPGA 同时具备了硬件固有的并行性及软件的灵活性, 其功能函数可被重新编程或者重新配置. FPGA 即现场可编程门阵列, 由于其可编程逻辑的大容量、灵活性、并行处理的能力, 内嵌 DSP 等模块, 可以解决 PC 机不能并行运行处理, 鲁棒性不强等问题, 大大缩短了图像处理所需要的时间[3]. 为了方便对 FPGA 开发, Altera 芯片公司开发了基于 Simulink 的 FPGA 设计工具 DSP Builder, 实

① 基金项目:国家国际科技合作项目(2010DFB43660);陕西省科技统筹创新工程计划项目(2012KTCQ01-19)
收稿时间:2015-01-19;收到修改稿时间:2015-03-23

现了 Simulink 系统仿真模型到 FPGA 实现代码之间的无缝链接，将高级语言转化为硬件描述语言，能够将以图像处理为代表的复杂算法直接在 FPGA 中，以并行结构实现，大大缩短了产品研制与开发周期，具有重要的现实意义.

本文利用 Altera 公司的 DSP Builder，设计了两种图像边缘检测算法的模型，在 MATLAB/Simulink 环境下进行了算法级的仿真，最后利用 DSP Builder 的硬件在环测试功能，将算法编译下载到 FPGA 中进行硬件级的仿真验证.

1　DSP Builder工具介绍

Altera 公司在 2002 年推出了 DSP 设计工具 DSP Builder. DSP Builder 依赖于 Mathworks 公司的数学工具 MATLAB/Simulink，以 Simulink 的 Blockset 形式出现. DSP Builder 通过帮助用户建立在算法友好环境下的 DSP 设计硬件描述来缩短 DSP 开发周期. DSP Builder 集成了 MATLAB/Simulink 系统设计工具和 Altera Quartus II 和第三方综合与仿真工具的算法开发、仿真、验证功能. DSP Builder 信号编译器模块读取由 DSP Builder 和 MegaCore®模块构建的 Simulink 模型文件，生成相对应的硬件描述语言 VHDL 文件，同时生成用于控制与编译的 Tcl 脚本，而后进行综合，并完成仿真[5].

针对 FPGA 上算法模型的不同，DSP Builder 提供了手动流程和自动流程两套设计流程. 图 1 是基于 MATLAB、DSP Builder、QuartusII 等工具完成设计的流程图.

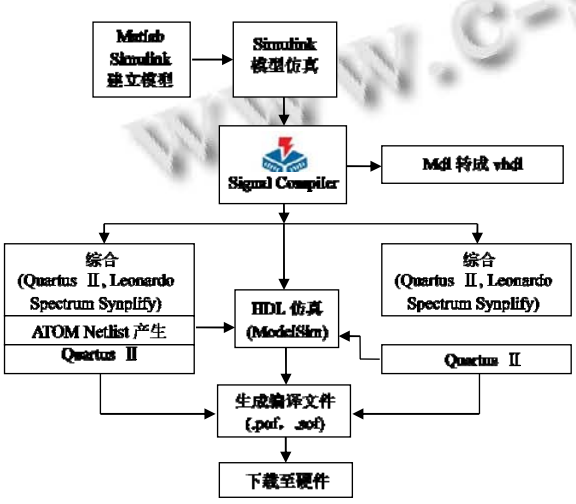


图 1　DSP Builder 的设计流程图

2　图像边缘检测的实现

2.1 边缘检测原理

图像的边缘是图像的最基本特征，是指图像灰度值局部变化显著的地方. 一幅图像可以看作是图像灰度值的连续函数的取样点矩阵，而图像的边缘就是图像灰度值函数变化较大的地方. 梯度是描述函数变化的一种度量，函数在某点的变化最大，就是在该点的梯度值最大[6]. 在一维情况下，阶跃边缘同图像的一阶导数局部峰值有关. 梯度是一阶导数的二维等效式，定义为:

$$G(x,y)=\begin{bmatrix}G_x\\G_y\end{bmatrix}=\begin{bmatrix}\partial f/\partial x\\\partial f/\partial y\end{bmatrix} \tag{1}$$

而梯度的幅值可由式(2)给出:

$$|G(x,y)|=\sqrt{{G_x}^2+{G_y}^2} \tag{2}$$

在实际应用中，通常用绝对值来近似梯度幅值，如式(3)

$$G(x,y)=|G_x+G_y| \tag{3}$$

2.2 边缘检测算法实现原理及方法

基于一阶导数的边缘检测算子有 Roberts 算子、Sobel 算子、Prewitt 算子，它们都是梯度算子. 本文采用的是 Prewitt 算子和 Sobel 算子，它们除了能对边缘点进行检测，还对噪声具有一定的抑制能力. 其中，Sobel 算子检测效果较好，但是经典的 Sobel 算子也存在不足，对垂直和水平方向敏感，对其他方向不敏感，使得有些方向的边缘检测不到.

如图 2(a)和(b)所示，其中S_x和S_y分别是 Prewitt 边缘算子和 Sobel 算子在水平和垂直方向上的卷积核，他们分别与P_5邻域每个像素做卷积，虽然是对P_5这个像素做卷积，却必须同时知道P_1到P_9这几个点的像素值.

Sx=

-1	0	1
-1	0	1
-1	0	1

Sy=

-1	-1	-1
0	0	0
1	1	1

P=

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

(a)Prewitt 边缘检测算子

Sx=

-1	0	1
-2	0	2
-1	0	1

Sy=

-1	-2	-1
0	0	0
1	2	1

(b)Sobel 边缘检测算子

图 2　边缘检测算子

从而可以知道P_5点在水平方向的梯度公式为:

$$G_x = x_1 \times P_1 + x_2 \times P_2 + x_3 \times P_3 + x_4 \times P_4 + x_5 \times P_5 + x_6 \times P_6 + x_7 \times P_7 + x_8 \times P_8 + x_9 \times P_9 \quad (4)$$

这里P_5点在垂直方向的梯度公式也和上式基本一样, 只是P_5邻域上的每个像素与 S_y 上的每个点相乘.

Prewitt 算子的水平和垂直方向的梯度公式可以用式(5)表示:

$$G_x = (P_3 + P_6 + P_9) - (P_1 + P_4 + P_7)$$
$$G_y = (P_7 + P_8 + P_9) - (P_1 + P_2 + P_3) \quad (5)$$

Sobel 算子的水平和垂直方向的梯度可以用公式(6)表示:

$$G_x = (P_3 + 2P_6 + P_9) - (P_1 + 2P_4 + P_7)$$
$$G_y = (P_7 + 2P_8 + P_9) - (P_1 + 2P_2 + P_3) \quad (6)$$

这样算得图像中每个邻域中P_5点梯度, 再将该点的水平和垂直方向的梯度取绝对值后相加得$|G_x| + |G_y|$. 将这个值与阈值比较, 如果$|G_x| + |G_y|$大于阈值, 则该像素被声明为边界像素, 当前像素的值赋值 255(白色); 否则为一般像素, 当前像素的值赋值 0(黑色), 从而实现了边缘检测. 阈值可通过对图像灰度作直方图分析, 再借助一定的经验反复调整就可以得出[7].

2.3 边缘检测算法的各个模块 DSP Builder 建模

2.3.1 边缘检测算法实现流程

如图 3 所示, 先将待处理像素排成 3×3 邻域窗口, 该窗口数据进入梯度计算模块, 并行计算出水平、垂直方向上的梯度值, 然后进行取绝对值后相加, 并与区分度阈值比较, 输出结果. FPGA 实现方式就是通过边缘检测实现函数的模块对输入的每个像素进行依次处理, 也就是流水化处理. 每个像素都是单独操作, 每个模块都是在同时运行, 实现了并行处理.

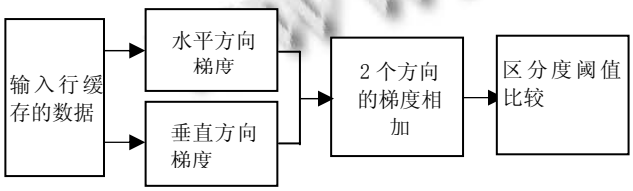


图 3 边缘检测算法流程图

2.3.2 3×3 邻域窗口模块

本文采用串行流水的 3×3 滑动窗口采集图像每个像素点的灰度值, 这样可以利用 Prewitt 梯度算子和 Sobel 梯度算子很方便的计算出水平和垂直方向的梯度. 3×3 滑动窗口设计如图 4 所示, 其中 Delay 模块是延迟一个时间单位的寄存器, Memory Delay 模块是 640 个时间单位的延迟(图片像素 640480). rl 到 r3_d2 就代表滑动窗口中的 8 个像素点的灰度值, 因为在求梯度中不需要知道 P5 点的灰度值, 故只有 8 个像素点的灰度值.

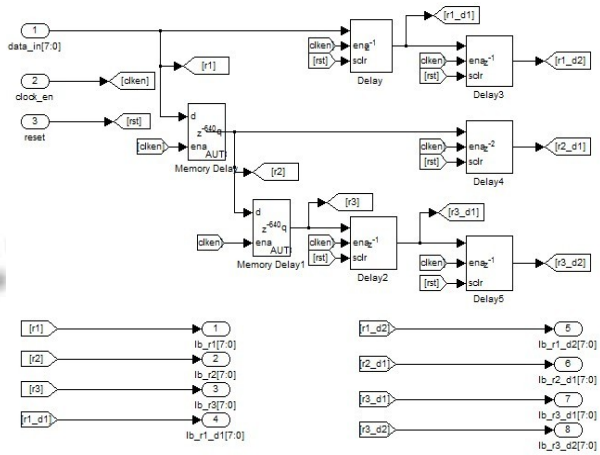


图 4 3×3 滑动窗口实现 8 抽头行缓存

2.3.3 梯度计算模块

图 5 和图 6 可以清楚得看到 Prewitt 边缘算法和 Sobel 边缘算法在梯度计算上的区别, Sobel 边缘算法还需要两个乘法器分别将各点对应的权值计算进去. 图 6 为 Prewitt 边缘算法水平方向梯度计算模块, 可以看出, 该模块使用了 4 个加法器和 1 个减法器, 实现了 6 个相关像数的水平梯度计算.

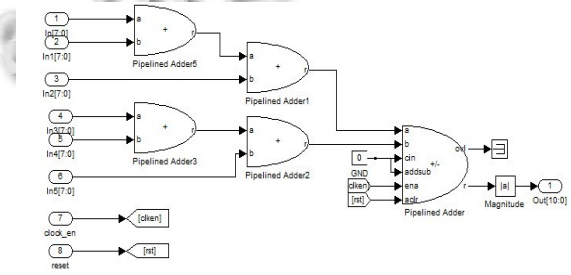


图 5 利用 Prewitt 算子计算水平方向梯度

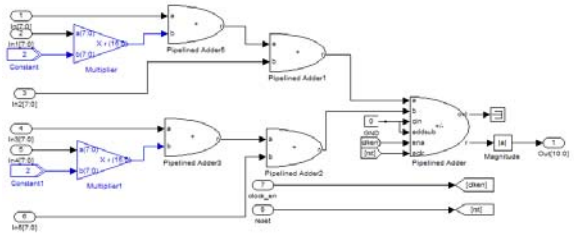


图 6 利用 Sobel 算子计算水平方向梯度

2.3.4 梯度与阈值比较模块

本文中梯度采用$|G_x|+|G_y|$来表示，并与预置的阈值进行比较，判断是否为边界像素. 其内部结果如图 7 所示，其中输入为 P_5 点在水平和垂直方向的梯度，通过一个流水加法器实现加法运算，BusConversion 实现了将 11 位的数据截取低 8 位的功能，然后与比较器 Comparator 进行比较，如果梯度小于区分度阈值，选择器 Multiplexer 中输出为一般像素，灰度值为 0(黑色). 如果梯度大于区分度阈值，则在选择器 Multiplexer 中输出为边界像素，灰度值为 255(白色).

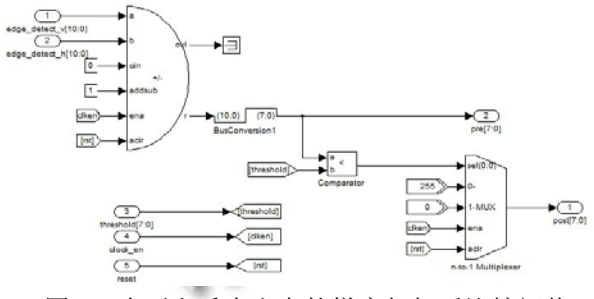


图 7　水平和垂直方向的梯度相加后比较阈值

2.3.5 数据有效信号模块

数据有效信号模块是由计数器、比较器、锁存器组成的，如图 8 所示. 当数据有效时，该模块输出 1，无效时输出 0.

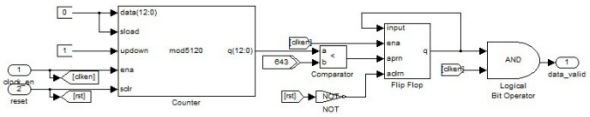
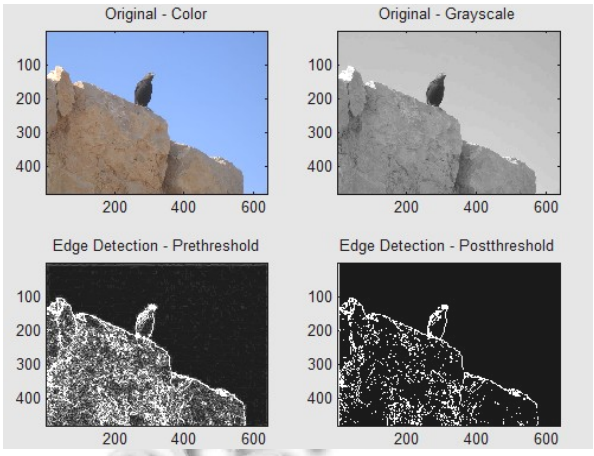


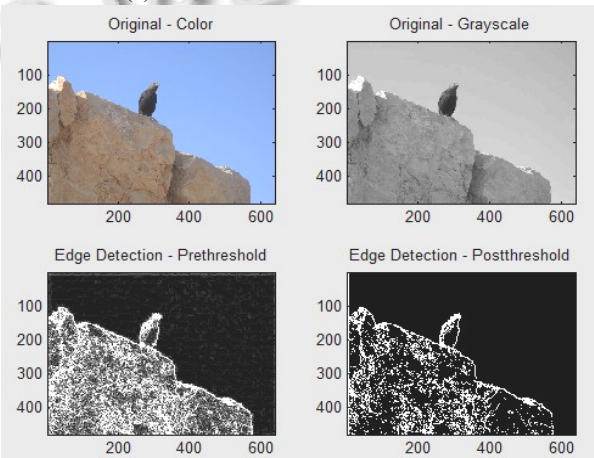
图 8　数据有效信号模块

3　边缘检测模型仿真

HIL(Hardware–in–the–Loop)硬件在环仿真测试系统以实际控制器加虚拟对象的半实物仿真系统. 它可以加快模型所生成代码的仿真速度与验证 Verilog 代码的实时性，以及所生成的 Verilog 代码在实际硬件中运行的正确性. 通过 USB Blaster 并将代码以及图像数据下载到 Alter FPGA 芯片 EP3C40F484 中，数据处理完成后返回 DSP Builder 中，其结果如图 9 所示. 从中我们可以看到算法实现了图像的边缘检测，达到了实际效果. Prewitt 算子可以很好地检测边缘点，也可以抑制噪声，Sobel 算子能提供比较精确地边缘的方向信息，对噪声的信息有平滑效果，但是边缘定位精度不高.



(a)Prewitt 算子边缘检测 HIL 结果



(b)Sobel 算子边缘检测 HIL 结果

图 9　HIL 边缘检测仿真结果

在 DSP Builder 中分别导出 Prewitt 算子和 Sobel 算子边缘检测算法的 Verilog 代码，在 Quartus II 编译代码，结果显示如表 1. 从表中可以看出整个算法所占用的逻辑资源和内存非常少，完全可将算法移植到图像采集系统中而不会增加对芯片的要求. 在 100MHz 时钟频率下，一副像素为 640×480 的图像，采用 FPGA 这种并行的结构算法，只需 3.072ms(640×480×0.01×10–3ms)，远远小于计算机处理图像的时间.

表 1　代码 Quartus II 编译结果

EP3C40F 484	逻辑单元 (LE)	寄存器 (registers)	引脚 (pins)	内存 (memory bits)
Prewitt 算子代码	324 (<1%)	218 (<1%)	34 (10%)	16384 (1%)
Sobel 算子代码	420 (1%)	257 (<1%)	34 (10%)	16384 (1%)

5 结论

本文分析了边缘检测算法的原理, 提出用 FPGA 这种硬件并行的方法来实现图像边缘检测算法, 利用 DSP Builder 工具对图像边缘检测算法在 MATLAB/Simulink 环境下设计了两种算子的边缘检测算法, 并在 FPGA 中进行了硬件在环测试, 而测试结果也表明 FPGA 对图像进行边缘检测的实时性是满足要求的. 将 FPGA 应用到图像处理中, 可以大大缩短图像处理的时间, 与传统的图像处理技术相比, 有很大的优势, 可以促进图像处理应用到更加广泛的行业.

参考文献

1 刘中合,王玉亮,李邦明等.数字图像处理及特征测量技术研究.山东农业大学学报(自然科学版),2007,(38):121–124.

2 袁春兰,熊宗龙,周雪花等.基于 Sobel 算子的图像边缘检测研究.激光与红外,2009,(39):86–87.

3 朱明,鲁剑锋.基于DSP+FPGA结构图像处理系统设计与实现.计算机测量与控制,2004,(9):866–869.

4 Bailey DG. Design for Embedded Image Processing on FPGAs. John Wiley & Sons (Asia) Pte Ltd, 2011: 11–18.

5 Altera Corporation. DSP Builder Handbook, 2012.

6 宁赛男,朱明,孙宏海等.一种改进的 Sobel 自适应边缘检测的 FPGA 实现.液晶与显示,2014(29):395–400.

7 杨新华,寇为刚.基于 FPGA 的 Sobel 算子图像边缘检测算法.仪表技术与传感器,2013(1):102–104.