用 CPLD 实现嵌入式平台上的实时图像增强

武汉大学电子信息学院(430079) 俞诗鲲 郑建生 曾 欣

摘 要：提出了在嵌入式平台上用 CPLD 实现实时图像增强算法的解决方案，并加以实现。重点讨论了经过改进的图像增强算法以及使用 CPLD 实现的具体方法，介绍了所采用的嵌入式平台的总体结构。

关键词：嵌入式 CPLD 实时处理 图像增强

通常，在拥有 DSP 或 FPGA 的嵌入式平台上，有关图像信号处理的算法部分都由 DSP 和 FPGA 完成。但是相对于标准的 PC 平台来说，嵌入式平台的资源有限得多，而且由于成本的原因，中央处理器的速度也通常无法与 PC 相比。因此，在 PC 机上用软件可以轻易实现的图像处理算法，完全移植到嵌入式平台上就要颇费一番周折了。

为了达到实时图像处理的目的，除了最大限度地发挥中央处理器的图像处理能力外，还需要合理地分配任务。DSP 芯片的优势在于乘除运算的能力，由于其特殊的流水线结构和处理单元，大部分 DSP 都能在单周期内完成在 PC 上需若干个周期才能完成的乘法运算，所以在进行诸如 FFT、DCT 等运算时优势明显；相反在进行简单的加减运算时，由于时钟频率和总线宽度都无法与 PC 机相比，效率不高。因此，如果能用硬件实时实现这些相对简单却又繁琐的运算，就可以大大提高系统的总体性能。

1 改进的图像增强算法

图像增强是图像处理中用于改善图像质量以及图像视觉效果的一种方法。在 DSP 平台上采用直方图均衡实现实时图像增强是一种常用的方法。对一幅连续图像，其具有灰度 G 的阈值面积(所有轮廓线所包围的面积)为 $A(G)$，则其直方图 $H(G)$ 定义为：

$$H(G)=\lim_{\Delta G\to 0}\frac{A(G+\Delta G)-A(G)}{\Delta G}=\frac{d}{dG}A(G),$$

对于数字图像，G 为整数，$A(G)$ 表示灰度值大于等于 G 的象素个数，当 $\Delta G=1$，$H(G)=A(G+1)-A(G)$。

如果对 $A(G)$ 做一次系数为 G_m/A_0 的比例变换，G_m 表示灰度级的最大值，A_0 表示图像的面积(在数字图像中为象素总数)。这就是一种线性直方图均衡。这种直方图均衡的具体实现如下：

(1)对于图像 $\{P_{i,j}|i=1,2,\ldots,n;\ j=1,2,\ldots,m\}$，就灰度 G，$G=0,1,\ldots,255$，求出直方图 $H(G)$；

(2)由 $A(G+1)=A(G)+H(G)$ 求出阈值面积 $A(G)$，$G=1,2,\ldots,255$；

(3)求出变换后的灰度分度值 $h_{new}(G)=255\ A(G)/A_0$，$A_0=nm$；

(4)$P_{ij}=h_{new}(P_{ij})$。

借助 LUT，可使运算以最快速度实现。

考察直方图均衡的实现过程可以发现，这是一种有限区间内的单调变换。从其频域特性看，直方图均衡改变了已有频率成分的分布，使它们分布得更加均匀，但并不增加新的频率成分。直方图均衡对于彩色(灰度)值集中在低端的图像，可起到较明显的视觉改善作用。但对于那些色彩分布很不均匀、频带较窄，特别是整体偏亮的图像，效果就不明显了。

本文采用一种新的图像增强方法，将对图像的边缘增强处理与均衡结合起来，并且这些运算最终可由硬件实现。

对于连续图像 P，其局部边缘可由对应空间梯度的幅值 $|\nabla P|=[(\frac{\partial P}{\partial x})^2+(\frac{\partial P}{\partial y})^2]^{\frac{1}{2}}$ 表示，取其一阶近似 $\Delta P_{i,j}=2P_{i,j}-P_{i,j-1}-P_{i-1,j}$，可得图像 $\{P_{i,j}|i=1,2,\ldots,n;\ j=1,2,\ldots,m\}$ 在 (i,j) 的边缘信息。

如果不计 $P_{i,j}$ 的取值范围，可直接对图像 $\{P_{i,j}|i=1,2,\ldots,n;\ j=1,2,\ldots,m\}$ 进行修正：

$$P'_{i,j}=P_{i,j}+\Delta P_{i,j},$$

其中，$P'_{i,j}$ 表示 $P_{i,j}$ 修正后的值。显然，图像 $\{P_{i,j}|i=1,2,\ldots,n;j=1,2,\ldots,m\}$ 按此规则修正后边缘值的变化更为强烈，边缘更为突出，可达到边缘增强的效果。同时，由于在原图像上叠加了梯度值，使得修正后的图像的频谱有一定的扩展。但由于没有对 $P_{i,j}$ 的取值作约束，这样处理后的象素值可能会溢出，例如对于每个色彩通道为 8 位的图像，处理后的数值可能会大于 255 或小于 0。因此，通常要对其进行归一化处理，即：

$$P_{new}=255\times(P'-P'_{min})/(P'_{max}-P'_{min})。$$

但用硬件实现乘除运算可能会占用很多资源，上述公式即便以运算实现都是很不经济的。本文采用预拉伸加饱和/截止的方法，在不牺牲频率特性的基础上达到减少计算量的目的。

考察 $\Delta P_{i,j}$ 与 $P_{i,j}$ 的直方图，分别取得它们的右峰值所对应的横座标，记为 G_Δ 和 G，并找到 k，使得 $kG_\Delta+G=255$，

则修正公式变为 $P'_{i,j}=P_{i,j}+k\Delta P_{i,j}$。其中 $k\Delta P_{i,j}$ 可以 LUT 实现。修正后的 $P'_{i,j}$ 可在[0,255]上进行饱和/截止运算。

2 用 CPLD 实现实时的图像增强

本文所采用的改进图像增强算法的主要成份是差分、累加以及饱和/截止。这些运算都是加减法及逻辑运算，都属于 ALU 的简单操作，适合硬件实现。本文采用 CPLD 实现所提出的算法。以对具有 30fps 的 1280×1024 RGB 图像计算 $\Delta P_{i,j}$ 为例，每计算一点 $\Delta P_{i,j}$ 需要 4 次加（减）运算，即总的需要 1280×1024×3×30×4=471,895,200 次加（减）运算。如果采用的 DSP 的速度是 100MHz，且假定所有运算都是单周期的，则仅仅该运算就需要 4.7s！所以采用 CPLD 实现某些运算是必需的。

采用 CPLD 实现运算（例如边缘处理中涉及的求梯度运算），还需解决数据的暂存问题。本文以一片高速 SRAM 作为数据缓冲区。由于图像数据的采样输入的频率也很高，需要充分合理地安排好每一次操作的时序，充分利用已参与运算的数据及中间结果，减少数据进出 SRAM 的次数。

2.1 基于 E1-DSP 的网络图像采集平台

在分析具体实现方法前，先简要介绍所采用的硬件平台。该平台主要用于远程图像采集和以太网传输，其图像通道结构如图 1 所示。

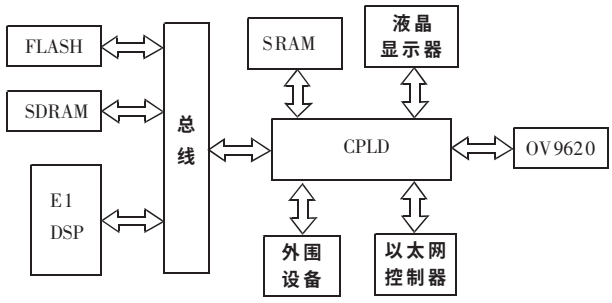


图 1 嵌入式平台数据总线结构框图

OV9620 是 CMOS 的数字图像传感器，负责采集连续的数字图像；中央处理器使用德国 HYPERSTONE 公司的 E1 系列 RISC DSP，它集 DSP 和 RISC 于一身，可以加载 OS，方便地实现任务调度、内存管理等功能，大大提高系统的总体性能；CPLD 的基本功能是作为 E1 总线接口控制模块，本文还将用它实现图像增强运算。

2.2 算法的总流程

为了实现实时的读写和运算，需要由外部电路产生 24MHz×4 的时钟 EXCLK 作为读写时钟，所有时序都由 CMOS 时钟和 EXCLK 控制，可以做到完全同步。具体流程如图 2 所示。

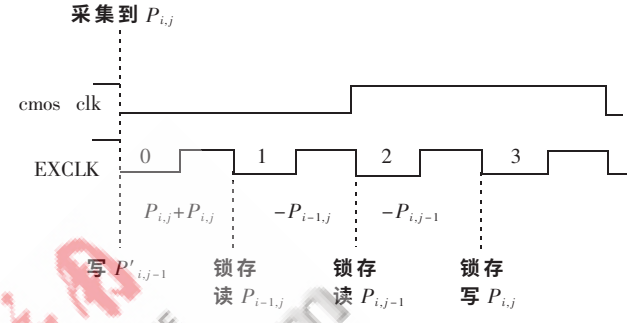


图 2 总体运算流程

(1)在 CMOS 时钟到来时，从 CMOS 传感器的数据输出口采集 $P_{i,j}$，并实现加法运算 RESULT=$P_{i,j}$+$P_{i,j}$，同时用 EXCLK 的第 0 个时钟向 SRAM 写入 $P'_{i,j-1}$ 或 $P'_{i,m-1}$（本行最后一个数据，下一次操作应换行）；

(2)在 EXCLK 的第 1 个时钟锁存 RUSELT，由 SRAM 读入 $P_{i-1,j}$，并做减法运算 RESULT=RESULT-$P_{i-1,j}$；

(3) 在 EXCLK 的第 2 个时钟锁存 RUSELT，由 SRAM 读入 $P_{i,j-1}$，并做减法运算 RESULT=RESULT-$P_{i,j-1}$；

(4) 在 EXCLK 的第 3 个时钟锁存 RUSELT，同时写入 $P_{i,j}$。

然后开始下一个点的运算。

2.3 硬件实现的逻辑结构

用 CPLD 实现该算法所采用的逻辑结构如图 3 所示。

其中，加模块实现 2×$P_{i,j}$ 运算，生成 9 位的运算结果交给减模块；减模块在 EXCLK 的第二和第三个时钟分别读入 $P_{i-1,j}$ 和 $P_{i,j-1}$ 进行减法运算，并把结果存回 result 寄存器。由于两次减法在时间上是错开的，因此只需要一个减法器就够了，节约了内部资源。

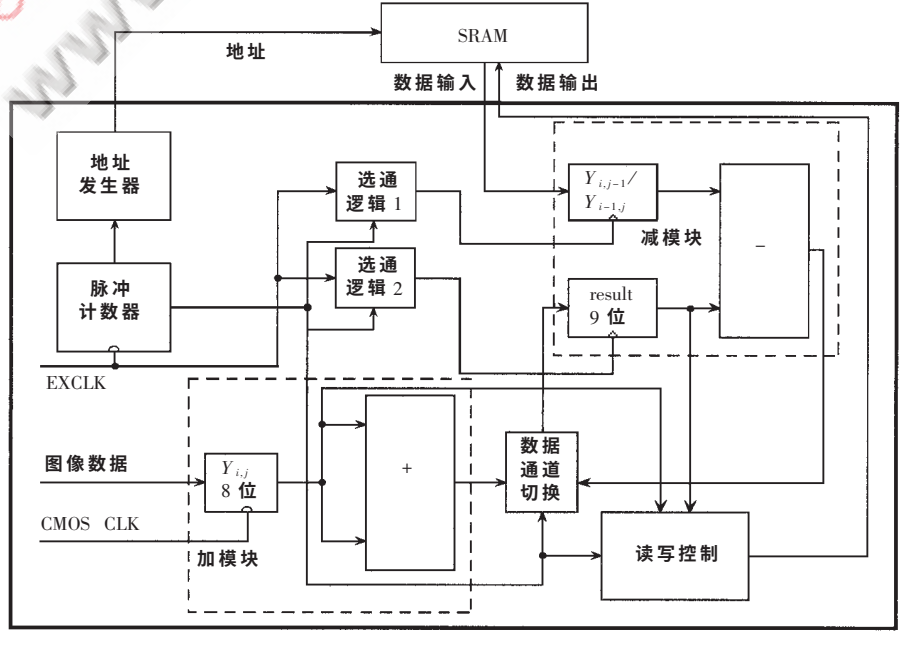


图 3 图像增强算法的硬件实现结构

图 3 中的脉冲计数器是一个模 4 计数器，所有的读写时序和运算时序都由它控制。数据通道切换模块控制流入 result 寄存器的数据流，在第一个 EXCLK 时钟让加法器的结果进入 result，其余的时间都让减法器的结果进入 result。两个选通逻辑模块对 EXCLK 起门控作用，选通逻辑 1 允许第 1 个和第 2 个时钟通过，用来锁存从 SRAM 读入的数据；选通逻辑 2 允许第 1、2、3 个时钟通过，用来锁存三次运算的结果。

SRAM 的读写操作由地址发生器和读写控制模块共同实现。由于四次读写操作的地址都不同，且不连续，无法用普通的地址计数器实现。这里采用地址计数器加偏移的相对寻址法，具体结构如图 4 所示。

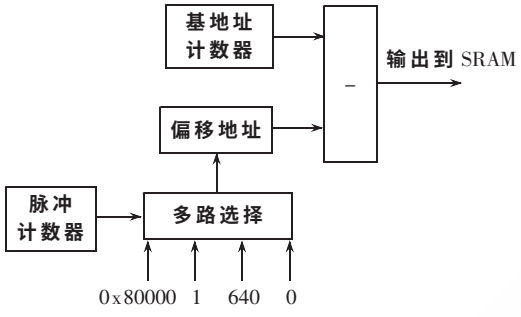


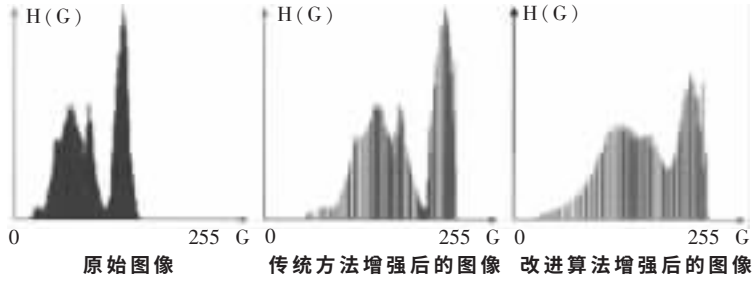
图 4 地址发生模块的结构

地址计数器中保存 $P_{i,j}$ 的地址，它由 cmos clk 作为时钟实现累加；偏移地址则由脉冲计数器模块控制，分别选择 $P'_{i,j-1}$、$P_{i-1,j}$、$P_{i,j-1}$ 和 $P_{i,j}$ 的偏移地址；最后做减法运算得到绝对地址送到 SRAM。

通过上述设计和优化，完全可以在结构和功能都比较简单的 CPLD 上实现实时的图像增强处理。

由于采用了改进的图像增强算法，在处理窄频带的图像时收到了非常好的效果，部分测试结果如图 5 所示。

与传统的处理方法相比，改进后的算法对图像的均衡效果更为明显一些，而且由于展宽了频带，图像的细节更加丰富，图像更加明艳和清晰。



原始图像　传统方法增强后的图像　改进算法增强后的图像

图 5 直方图比较

以上算法都在 CPLD 上实现，并没有占用 DSP 的处理时间，因而节省了大量的运算时间。笔者做过一个实际测试，在 100MHz 主频的 E1 DSP 上用 C 编程实现一帧 640×480 RGB 图像的增强算法大约需要 100ms（如果用汇编语言编程或对程序作优化可使性能提高一些），而且要占用大量存储资源。这样的运算速度只适合静止图像的处理。所以，如果不做简化处理或采用更高性能的 DSP，根本无法做到实时处理。由此可见，采用硬件处理的方法可以极大地提高系统的总体性能。

综上所述，在拥有 DSP 的嵌入式平台上使用 CPLD 实现改进的图像增强算法是可行的，对于实时的图像处理是一种高效的解决方法。

参考文献

1 Shoup,R.G.Real-time image manipulation using soft hardware Systems. Man and Cybernetics, 1993. Systems Engineering in the Service of Humans. Conference Proceedings.International Conference,17~20 Oct. 1993;3: 343~348
2 Verkest,D., Desmet, D., Avasare, P., etc. Design of a Secure,Intelligent, and Reconfigurable Web Cam Using a C Based System Desing Flow. Signals, Systems and Computers, 2001.Conference Record of the Thirty-Fifth Asilomar Conference on, 2001;1(11): 463~467
3 Hyperstone DSP. http://www.hyperstone.com
4 Lattice CPLD. http://www.latticesemi.com
5 刘 涛. Visual C++实现数字图像增强处理. 2002, http://www.chinabyte.com

（收稿日期：2003-08-29）