文章编号:1001-5078(2013)12-1411-05

·图像与信号处理·

光学图像高斯平滑滤波的DSP优化

汤 达[1],牛照东[1],王丁禾[1],陈曾平[1],李晓鹏[2]
(1. 国防科学技术大学ATR重点实验室,湖南 长沙 410073;2. 中国人民解放军77108部队,四川 成都 611233)

摘 要:高斯平滑滤波方法具有优良的噪声平滑性能和边缘保留能力,但运算量相对较大,限制了其在实时图像处理系统中的应用。本文基于高斯平滑掩膜的可分解性,提出了一种单次遍历实现两次卷积的高斯平滑滤波DSP优化方法。首先,依次打包读取4×8图像区域作为基本运算单元,有效降低对数据的重复访问;其次,在基本运算单元内,利用内联函数并行计算横向模板的一次卷积;然后,重组并复用横向模板卷积单元直接进行纵向模板的二次卷积。最后,以基本运算单元为单位遍历处理图像,计算平滑滤波结果。实验表明,利用TMS320C6455定点DSP对一幅320×240×8bit图像进行5×5高斯模板滤波耗时0.187ms,是优化前滤波耗时的1/35,具有较大的工程应用价值。

关键词:光学图像;高斯平滑滤波;算法优化;实时;数字信号处理器
中图分类号:TP391 **文献标识码**:A **DOI**:10.3969/j.issn.1001-5078.2013.12.21

Optimization of optical Image Gaussian smoothing filtering based on DSP

TANG Da[1], NIU Zhao-dong[1], WANG Ding-he[1], CHEN Zeng-ping[1], LI Xiao-peng[2]
(1. ATR Key Lab, National University of Defence Technology, Changsha 410073, China;
2. Unit No. 77108, Chengdu 611233, China)

Abstract: Gaussian smoothing filtering has a good performance in averaging the noise and preserving the edge. However, the large amount of calculation limits its application in real-time image processing system. Consequently, in view of the decomposability of the Gaussian smoothing mask, an optimized Gaussian smoothing filtering method which achieves successive convolution within single traversal is proposed in this paper. Firstly, reading 4×8 image area is as a basic operation unit to reduce data access. Secondly, lateral convolution is parallelly computed by using the inline function, to longitudinal convolution is calculated by recomposing and reusing lateral convolution results. Finally, the image with basic operation unit is traversed, smoothing filtering results are calculated. The experimental results show that the optimized code execution time is improved more than 35 times compared to the original C code when a 320×240×8bit image is smoothed based on TMS320C6455. The proposed method has great value in engineering application.

Key words: optical image; Gaussian smoothing filtering; algorithm optimization; real-time; DSP

1 引 言

光学传感器在传输、接收和处理图像数据的过程中,往往会存在一定程度的噪声干扰,恶化了图像的质量。高斯平滑滤波具有优良的噪声平滑性能和边缘保留能力,适合平滑图像、去除噪声,但运算量较大,限制了其在实时图像处理系统中的应用。因此,有必要对高斯平滑滤波算法进行优化,使其满足图像处理系统的实时性需求。

目前,已有专家学者针对算法的硬件优化进行研究,并取得了一定的成果。文献[1]、[2]介绍了

基金项目:部委级预研项目(No. 51301030404-1)资助。
作者简介:汤 达(1988-),男,硕士研究生,主要研究方向为图像处理与自动目标识别。E-mail:albert_tang@126.com
收稿日期:2013-05-07

DSP 代码开发流程以及代码优化的思路与方法，但并未结合实例进行具体说明；陈松[3]等人对三种图像预处理算法进行了优化，但同样缺乏对优化方法的详细陈述；雷涛[4-5]等人提出了一种空域低通滤波优化方法，但该方法对滤波掩膜的系数取值有严格的限制，具有较大的局限性；黄德天[6]等针对中值滤波进行了不同程度的优化，取得了较好的效果，但其优化方法仍有较大的改进空间。

本文基于 TMS320C6x 系列[7-8]定点处理器，利用高斯滤波掩膜的可分解性，提出了一种单次遍历实现两次卷积的高斯平滑滤波 DSP 优化方法。实验结果表明，该方法能够显著提升滤波效率，达到了算法优化的目的。

2 基本的空域滤波优化方法

空域滤波在图像空间中的实现是通过在待处理图像中逐点移动掩膜进行卷积来完成的。常用的掩膜尺寸为 3×3 或 5×5，掩膜系数随功能变化而变化。图 1 给出了两种尺度的高斯平滑掩膜，用于平滑图像、减小噪声。本文就以图 1(a)所示5×5 高斯平滑掩膜为例，说明对空域滤波的优化，对其他尺度滤波掩膜，同样可借鉴本文优化思想。

$$F_{5\times5}=\frac{1}{84}\begin{bmatrix}1&2&3&2&1\\2&5&6&5&2\\3&6&8&6&3\\2&5&6&5&2\\1&2&3&2&1\end{bmatrix}\qquad F_{3\times3}=\frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$$

(a)5×5 Gaussian smoothing mask (b)3×3 Gaussian smoothing mask

图 1 两种高斯平滑掩膜

Fig. 1 Two Gaussian smoothing masks

图 2 描述了传统的空域滤波过程，*A*0 等点代表待滤波图像数据，虚线方框表示滤波掩膜遍历至 *C*2 点时，滤波掩膜覆盖到的图像区域。将该区域内每个点的像素值与掩膜对应位置滤波系数相乘，累加后除以滤波系数总和，即可得到滤波结果 R_{11}。如此，对单个像素点滤波，需要 25 次乘法运算、24 次加法运算和 1 次除法运算。

分析传统空域滤波耗时原因，不难发现，其中存在大量的数据重复访问。如对 *C*2、*C*3 两点的滤波均用到了 *C*1 至 *C*4 点所在列图像数据。如果两次滤波过程均读取掩膜覆盖区域内全部图像数据，将造成 *C*1 至 *C*4 点所在列图像数据重复读取。*C*6*x* 编译器访问存储器是很费时的，为了提高数据处理率，应使一条 *Load/Store* 指令能够访问多个数据[9]。针对上述问题，利用内联函数[10]_memd8()，一次读取图 2 中第一行 *A*0 到 *A*7 的全部 8 个原始图像数据。如此，只需五条数据读取指令，就可以读取图 2 所示全部图像数据，同时对多个点进行滤波。

作为一种快速求点积和指令，_dotpsu4()函数将 32 位输入 *src*1 与 *src*2 的每 8 位对应相乘后的累加值作为结果输出，可实现同一行中 4 个相邻图像数据与滤波系数的相乘后累加工作。鉴于此，对图 2 所示图像区域内 *C*2 至 *C*5 点，调用_dotpsu4()函数分别计算相应覆盖区域与滤波掩膜的卷积和，除以滤波系数总和，即可得到滤波结果 R_{11}、R_{12}、R_{13} 和 R_{14}。

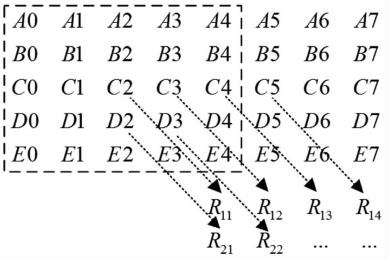

图 2 5×5 空域滤波示意

Fig. 2 5×5 Gaussian smoothing

利用上述方法，每次读取 40 个图像数据，可计算同行相邻 4 个像素点对应的滤波结果。其中，对单个像素点的滤波需要调用 10 次_dotpsu4()函数。横向遍历原始图像，对一个宽、高各为 *M* 和 *N* 原始图像，遍历 *MN*/4 次，即可得到空域滤波后结果图像。

3 高斯平滑滤波优化

3.1 高斯平滑掩膜的可分解性

在空域滤波的实际应用中，为了减少运算量，通常会在满足滤波性能的前提下，有针对性地设计滤波掩膜。常用的方法是利用两个一维滤波掩膜的逐次卷积来代替二维滤波掩膜，而高斯平滑滤波就是其典型代表。

二维高斯滤波掩膜(i,j)处滤波系数取值如下：

$$g(\sigma;i,j)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(i-i_0)^2+(j-j_0)^2}{2\sigma^2}}=\frac{1}{c^2}e^{-\frac{(i-i_0)^2+(j-j_0)^2}{2\sigma^2}} \tag{1}$$

式中，(i_0,j_0)为掩膜中心位置坐标；σ 为高斯分布参数，决定高斯函数的平滑程度；c 为归一化系数。令一维高斯掩膜滤波系数取值如下：

$$g(\sigma;i)=\frac{1}{c}e^{-\frac{(i-i_0)^2}{2\sigma^2}} \tag{2}$$

由式(1)、式(2)，以及高斯掩膜的中心对称性，有：

$$g(\sigma;i,j)=g(\sigma;i)\times g(\sigma;j)=g(\sigma;i)\times g(\sigma;W-j) \tag{3}$$

式中，W 为掩膜宽度。由式(3)，进一步得到二维高斯掩膜 $g(\sigma;i,j)$ 与两个相互垂直的一维高斯掩膜 $g(\sigma;i)$、$g(\sigma;j)$ 之间满足：

$$g(\sigma;i,j)=g(\sigma;i)*g(\sigma;j) \tag{4}$$

结合式(4)，将 $g(\sigma;i,j)$ 与待滤波图像 $f(i,j)$ 卷积，可得：

$$g(\sigma;i,j)*f(i,j)=g(\sigma;i)*g(\sigma;j)*f(i,j)=g(\sigma;i)*(g(\sigma;j)*f(i,j)) \tag{5}$$

式(5)说明了高斯平滑掩膜的可分解性[11-12]，二维高斯平滑掩膜与图像卷积等效于将图像与一维高斯平滑掩膜卷积，然后再将卷积结果与方向垂直的一维高斯平滑掩膜进行卷积。这样，对单个像素点滤波，只需要 10 次乘法运算、8 次加法运算和 2 次除法运算，大大减少了计算量。

3.2 高斯平滑滤波优化

高斯平滑掩膜的可分解性为高斯平滑滤波的进一步优化提供了理论基础。鉴于此，将 5×5 滤波掩膜分解为 5×1 横向掩膜 N 与 1×5 纵向掩膜 M 的相乘，高斯平滑滤波等效于原始图像先与 M 卷积(以下称横向卷积)后再与 N 卷积(以下称纵向卷积)。接下来，本文从以下几个方面考虑，对逐次卷积过程进行优化。

3.2.1 基本运算单元

为了最大限度减少算法耗时，应避免对滤波中产生的中间结果进行存取操作，这就要求上述逐次卷积过程应在一次遍历中完成。在明确这一需求后，需进一步确定每次遍历读取多大的图像区域作为基本运算单元进行逐次卷积，能使优化效果达到最优。

为了借助_mem4()函数实现对滤波结果的快速存储，应使每次遍历算得的滤波结果在行方向上相邻且为 4 的整数倍。为了借助_dotpsu4()函数实现图像数据与掩膜系数的相乘累加，每次读取同一行中相邻 8 个图像数据参与横向卷积，同理，每次使同一列中相邻 8 个横向卷积结果参与到纵向卷积。基于以上两点考虑，选择 8×8 区域作为卷积的基本运算单元，经横向卷积，得到卷积结果大小为 8×4，再经纵向卷积得到滤波结果为 4×4 大小区域。

3.2.2 重组卷积结果

重组卷积结果的目的就是整合横向卷积结果，以便利用内联函数快速完成纵向卷积。下面借助图 3(a)与图 3(b)，结合重组横向卷积结果，说明逐次卷积过程。

数据打包读取原始图像数据 $A0$ 至 $A7$ 及其正下方 7 行图像数据，横向卷积得到 $F1,G1,H1,I1,\cdots,Fi,Gi,Hi,Ii(i=1,\cdots,8)$。利用内联函数_packl4()与_pack2()对 $F1$ 等进行重组，如对 $F1$ 至 $F8$，利用式(6)、式(7)，将其组合为 2 个 32 位整数 $data\_h$ 与 $data\_l$。将 $data\_h$、$data\_l$ 以及滤波系数组合 $coeff\_h$、$coeff\_l$ 一同作为_dotpsu4()函数的输入，利用式(8)计算纵向卷积结果。式中，利用逻辑位移代替除法运算，X 为逻辑右移位数。

$$data\_h=\_packl4(\_pack2(F8,F7),\_pack2(F6,F5)) \tag{6}$$

$$data\_l=\_packl4(\_pack2(F4,F3),\_pack2(F2,F1)) \tag{7}$$

$$R_{ij}=(\_dotpsu4(coeff\_h,data\_h)+\_dotpsu4(coeff\_l,data\_l))>>X \tag{8}$$

通过横向卷积结果在两次卷积中的合理过渡，不仅使逐次卷积能够在对图像的单次遍历中完成，且由于逐次卷积过程中大量使用内联函数，使得算法执行效率得到了大幅提升。

3.2.3 复用卷积结果

分别读取原始图像中以像素点 (m,n) 和 $(m+4,n)$ 为左上角端点的 2 组 8×8 图像数据，各自与 M 卷积，得到 Fi、Gi、Hi、Ii。两次横向卷积结果前者 i 取值从 m 至 $m+7$，后者 i 取值从 $m+4$ 至 $m+11$。可见，后者前一半的卷积结果已由前者算得，无须重复计算，只需再计算后一半卷积结果即可。

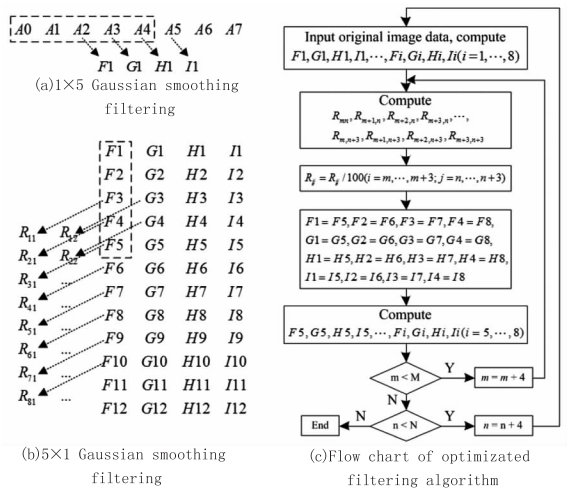

图 3 优化后滤波计算与流程图

Fig. 3 Calculation and flow chart of filtering algorithm after optimization

鉴于此，纵向遍历时，除了在图像顶端的初始遍历读取大小 8×8 图像区域，算得 8×4 横向卷积结果外，其余遍历时复用前一次遍历产生的 4×4 横向卷积结果，另外读取 4×8 基本运算单元，算得可复用于下次遍历的 4×4 横向卷积结果即可。在获得同等数量滤波结果情况下，复用过程结果，数据读取

量与卷积运算量均减半,且滤波过程无多余的数据存取,既节省了时间,又节约了空间。

3.2.4 滤波流程

综上所述,优化后高斯平滑滤波算法步骤如下,m、n 初始值均为0。

Step1:数据打包读取图像数据 $O'=O_{0n8\times8}$,其中 $O_{mn8\times8}$ 代表以原始图像中(m,n)点为矩阵左上角的 8×8 图像数据,读取图像数据后将 m 赋值为8;

Step2:O'的每一行与 M 卷积,得到 $F1,G1,H1,I1,\cdots,Fi,Gi,Hi,Ii(i=1,\cdots,8)$,记为 T;

Step3:T 的每一列与 N 卷积,得到 R_{mn},$R_{m+1,n}$,$R_{m+2,n}$,$R_{m+3,n}$,$\cdots$,$R_{m,n+3}$,$R_{m+1,n+3}$,$R_{m+2,n+3}$,$R_{m+3,n+3}$,除以系数累加和作为滤波结果输出;

Step4:利用式(9)-(12),复用卷积结果,更新 Fi、Gi、Hi、$Ii(i=1,\cdots,4)$;

$$F1=F5\quad F2=F6\quad F3=F7\quad F4=F8 \tag{9}$$

$$G1=G5\quad G2=G6\quad G3=G7\quad G4=G8 \tag{10}$$

$$H1=H5\quad H2=H6\quad H3=H7\quad H4=H8 \tag{11}$$

$$I1=I5\quad I2=I6\quad I3=I7\quad I4=I8 \tag{12}$$

Step5:数据打包读取原始图像数据 $O'=O_{mn4\times8}$,其中 $O_{mn4\times8}$ 代表以原始图像中(m,n)点为矩阵左上角的 4×8 图像数据;

Step6:O'的每一行与 M 卷积,得到 $F5,G5,H5,I5,\cdots,Fi,Gi,Hi,Ii(i=5,\cdots,8)$;

Step7:如果 m 小于原始图像高度 M,$m=m+4$,重复 Step3 至 Step6,否则跳转至 Step8;

Step8:如果 n 小于原始图像宽度 N,$n=n+4$,跳转至 Step1,否则结束滤波。

4 实验结果与分析

为了测试优化性能,利用图1(a)所示 5×5 高斯平滑滤波掩膜对不同尺寸的原始图像进行滤波,得到优化前后滤波耗时如表1所示。实验仿真环境为 DSP 集成开发环境 CCS v3.2,处理器为 TMS320C6455/1GHz,处理时间为 CCS 软件内部剖析时钟显示的程序耗时。

表1 优化前后滤波耗时对比

Tab.1 Time cost before and after optimization

Image parameter	-op3 compile/ms	Spatial filtering method[3]/ms	Gradual filtering/ms	Proposed method/ms	Increase of efficiency
160×120×8bit	1.625	0.093	0.083	0.050	32.50
240×180×8bit	3.733	0.219	0.197	0.110	33.94
320×240×8bit	6.667	0.393	0.418	0.187	35.65
480×320×8bit	15.177	0.888	0.904	0.422	35.96

由表1,在-o3模式下编译未经优化的高斯平滑滤波代码,对不同尺寸图像的滤波均耗时较长。利用文献[3]所述传统的空域滤波方法进行优化,滤波效率平均提升约16倍。借助高斯平滑掩膜的可分解性,对原始图像进行逐次卷积,该过程首先遍历原始图像进行横向卷积,再遍历横向卷积结果进行纵向卷积。逐次卷积耗时如表1第四列所示,对小尺寸图像的逐次卷积耗时相比传统空域滤波有小幅减小,对大尺寸图像的逐次卷积耗时反而超过了传统空域滤波耗时,这主要是因为两次遍历中横向卷积结果的存取消耗了大量的时间,且图像尺寸越大,存取操作耗时也就越多。利用本文方法,重组并复用横向卷积结果,使逐次卷积能够在对图像的单次遍历中完成,滤波效率较传统优化方法和逐次卷积方法有了进一步的提升。实验结果表明,对不同尺寸的图像,滤波效率提升32倍以上,且图像尺寸越大,滤波效率的提升就显著。

对一幅 $320\times240\times8$bit 图像,利用不同尺度的高斯平滑掩膜进行滤波,得到滤波耗时如表2所示。随着掩膜尺度的增大,掩膜分解前后算法运算量差异就越大。因此,越是大尺度的滤波掩膜,利用本文方法滤波,优化效果也就越显著。

表2 不同尺度掩膜优化性能对比

Tab.2 Results of optimization experiments with different scale masks

	-op3 compile/ms	Spatial filtering method[3]/ms	Proposed method/ms	Increase of efficiency
3×3 mask	0.375	0.196	0.163	2.30
5×5 mask	0.667	0.393	0.187	3.57
7×7 mask	8.923	3.837	1.693	5.27

5 结 论

本文以 TMS320C6x 系列 DSP 为应用平台,基于高斯平滑掩膜的可分解性,提出了一种单次遍历实现两次卷积的高斯平滑滤波 DSP 优化方法,首先,依次打包读取 4×8 图像区域作为基本运算单元,有效降低对数据的重复访问;其次,在基本运算单元内,利用内联函数并行计算横向模板的一次卷积;然后,重组并复用横向模板卷积单元直接进行纵向模板的二次卷积。最后,以基本运算单元为单位遍历处理图像,计算平滑滤波结果。实验表明,本文方法能够显著提升高斯平滑滤波效率,具有较大的工程应用价值。

参考文献:

[1] Diao Yiping, Zhao Xiaoqun. C code optimization for TI C6000 DSPs[J]. Microcomputer Applications, 2007, 28(5):544 – 548. (in Chinese)
刁一平,赵晓群. 基于 TI C6000DSP 的 C/C + +语言代码效率优化[J]. 微计算机应用, 2007, 28(5):544 – 548.

[2] YangGuangyu, Gao Xiaorong, Wang Li, et al. Technology of C/C + + Program Optimization Based on TI C6000 DSP[J]. Modern Electronics Technique, 2009, 33(8):544 – 548. (in Chinese)
杨光宇,高晓蓉,王黎,等. 基于 TI C6000 系列 DSP 的 C/C + +程序优化技术[J]. 现代电子技术,2009,33(8):544 – 548.

[3] Chen Song, Zheng Hong, Wu Xinghua, et al. Optimization method of digital image preprocessing algorithm based on DSP[J]. Electronic Instrumentation Customer, 2010, 17(3):58 – 60. (in Chinese)
陈松,郑红,吴兴华,等. 基于 DSP 的数字图像预处理算法优化方法[J]. 仪器仪表用户, 2010, 17(3):58 – 60.

[4] Lei Tao, Cao Xiaowei, Wu Qinzhang. The optimization of the space low-pass filtering module in the real-time image processing based on DSP[J]. Opto-Electronic Engineering, 2012, 39(5):116 – 120. (in Chinese)
雷涛,曹晓伟,吴钦章. 实时 DSP 图像处理空间低通滤波模块优化[J]. 光电工程,2012,39(5):116 – 120.

[5] Lei Tao, Zhou Jin, Wu Qinzhang. Research on optimization method of DSP real-time image processing software [J]. Computer Engineering, 2012, 38(14):177 – 180. (in Chinese)
雷涛,周进,吴钦章. DSP 实时图像处理软件优化方法研究[J]. 计算机工程,2012,38(14):177 – 180.

[6] Huang Detian, Chen Jianhua. Code optimization of DSP image processing[J]. Chinese Journal of Optics and Applied Optics, 2009, 2(5):452 – 459. (in Chinese)
黄德天,陈建华. DSP 图像处理的程序优化[J]. 中国光学与应用光学,2009,2(5):452 – 459.

[7] Texas Instruments. TMS320C6455 Technical Reference [EB/OL]. 2005. (SPRU965)

[8] Li Kun, Dong Wenjuan, Wang Weihua, et al. Implementation of a hardware platform for infrared image information processing and optimization of software based on double DSPs[J]. Infrared Technology, 2008, 30(9):533 – 536. (in Chinese)
李坤,董文娟,王卫华,等. 基于双 DSP 的红外图像信息处理硬件平台的实现及软件优化[J]. 红外技术, 2008, 30(9):533 – 536.

[9] Li Fanghui, Wang Fei, He Peikun. The principle and application of TMS320C6000 DSPs[M]. 2nd ed. Beijing: Publishing House of Electronics Industry, 2003. (in Chinese)
李方慧,王飞,何佩琨. TMS320C6000 系列 DSPs 原理与应用[M]. 2 版. 北京:电子工业出版社,2003.

[10] Texas Instruments. TMS320C64x/C64x + DSP CPU and Instruction Set Reference Guide [EB/OL]. 2008. (SPRU732)

[11] Byung Soo Moon. A gaussian smoothing algorithm to generate trend curves[J]. Korean Journal of Computational and Applied Mathematics, 2001, 8(3):507 – 518.

[12] Frank Y Shih, Yi-Ta Wu. Decomposition of arbitrary gray-scale morphological structuring elements[J]. Pattern Recognition, 2005, 38(12):2323 – 2332.