基于积分图像的快速选择掩模平滑算法

李金红,张有会,滑斌杰,程 浩
LI Jin-hong,ZHANG You-hui,HUA Bin-jie,CHENG Hao

河北师范大学 数学与信息科学学院,石家庄 050016
College of Mathematics and Information Science,Hebei Normal University,Shijiazhuang 050016,China
E-mail:jing831111@126.com

LI Jin-hong,ZHANG You-hui,HUA Bin-jie,et al.Rapid algorithm of selective masking smoothing based on integral image.Computer Engineering and Applications,2009,45(11):206-208.

Abstract: In selective masking smoothing algorithm,it is needed to calculate grayscale average and variance of a large number of image windows.Integral image and square integral image can be used to realize a rapid algorithm for calculating grayscale average and variance of image windows,and the calculating speed is independent of the windows' size.Using this rapid algorithm to calculate grayscale average and variance of each masking can improve the computing speed of selective masking smoothing algorithm,hence the rapid calculating of selective masking smoothing algorithm.

Key words: selective masking smoothing;integral image;average;variance

摘 要:在选择掩模平滑算法中,需要计算一幅图像的大量局部窗口的灰度均值和方差。而运用积分图像和平方积分图像,可以实现一种计算速度与图像窗口大小无关的灰度均值和方差的快速计算方法。将这种灰度均值和方差的快速计算方法运用于选择掩模平滑算法中,来计算各个掩模的灰度均值和方差,能够大大地提高算法的计算速度,从而实现了选择掩模平滑算法的快速计算。

关键词:选择掩模平滑;积分图像;均值;方差
DOI:10.3778/j.issn.1002-8331.2009.11.062 **文章编号**:1002-8331(2009)11-0206-03 **文献标识码**:A **中图分类号**:TP391

1 引言

图像在成像、数字化和传输的过程中,难免会受到各种噪声源的干扰和影响而使图像质量变差。为了抑制噪声,改善图像质量,必须对图像进行平滑处理。图像平滑是图像处理中的一个关键性环节,在实际情况中,它往往作为图像处理与识别的预处理,是图像后续处理(边缘检测、图像分割、特征提取、模式识别等)的基础。由于对大多数图像而言,人们不知道或不可能用简单的随机过程精确地描述信号和噪声的统计模型,因此在一段时间以来人们常采用不需要知道任何有关图像先验知识的固定窗口空间域局部平滑法。这种算法简单,计算速度快,但它在抑制噪声的同时会使图像产生模糊。为此,人们又提出了许多改进的局部平滑算法,以求在抑制噪声的同时,尽可能多地保留图像的细节。选择掩模平滑算法是其中一种比较成功的算法。它提供多个掩模,并通过比较各个掩模的方差大小来选择方差最小的掩模的均值作为最后的平滑输出。因此它在滤除噪声的同时能较好地保留图像的细节。但在算法实现中,因为要计算所有掩模的均值和方差,所以增加了算法的时间复杂度,限制了它在实时系统中的应用。

为了解决上述问题,本文对选择掩模平滑算法进行了分析和改进,提出了一种基于积分图像的快速选择掩模平滑算法。它在积分图像和平方积分图像的基础上,能够快速地计算所有掩模的均值和方差。

积分图像的思想最早是由 Paul Viola 等在文献[1]中提出的,并被应用到实时的对象检测框架中,来计算图像窗口的灰度值总和,显示出了其在规则区域(矩形或旋转矩形)内像素灰度值加和运算上的优势。此外,也有人利用积分图像来计算图像窗口的灰度均值[2-3]。

2 一般的选择掩模平滑算法

常用的图像平滑法,如邻域平均法、频域平均法等,都不可避免地带来平均化的缺憾,致使尖锐变化的边缘或细节变得模糊[4]。而选择式掩模平滑算法既可完成滤波操作,又不破坏区域边界的细节。该算法认为图像中的目标物体和背景统计特性不同,即具有不同的均值与方差。相对于图像中的平缓区域,含有尖锐边缘的区域一般具有较大的方差[5]。其基本思想如下:

首先对于图像中待处理的像素(x,y),选它的 5×5 邻域,在此邻域中以此像素(x,y)为基准点制作 9 个掩模,包括 1 个正方形掩模,4 个五边形掩模和 4 个六边形掩模,如图 1 所示。图

基金项目:河北省教育厅基金(the Foundation of the Educational Department of Hebei Province under Grant No.2006436)。

作者简介:李金红(1983-),女,硕士研究生,主要研究领域为图像处理、视频编码技术;张有会(1955-),男,教授,主要研究领域为图像处理、视频编码技术;滑斌杰(1974-),男,硕士研究生,主要研究领域为图像处理、视频编码技术;程浩(1983-),男,硕士研究生,主要研究领域为图像处理、视频编码技术。

收稿日期:2008-03-03 **修回日期**:2008-05-08

中用圆圈标出的像素点为待处理的像素(x,y),所标的数字表示掩模的编号。然后分别计算各个掩模的均值和方差,按方差排序,最小方差对应掩模的灰度均值就是待处理像素(x,y)的输出值。因此,选择掩模平滑算法又被称为最小均方差滤波[6]。

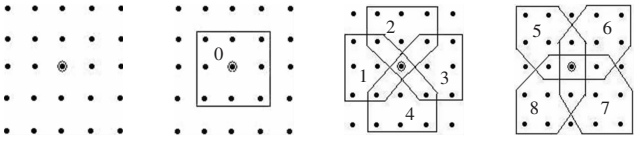

图 1　5×5 邻域和 9 个掩模

该算法以方差作为各个邻域灰度均匀性的测度。含有尖锐边缘的邻域,其方差必定很大,而不含边缘或灰度均匀的邻域,方差则很小。因此方差越小,其对应掩模的灰度就越均匀,待处理像素(x,y)选择该掩模的可能性就越大。若(x,y)本来就是边界上的点,那么用最小方差对应掩模的灰度均值代替(x,y)的灰度值,将不会破坏边界清晰度;若(x,y)为一噪声点,采用这种方法则起到了平滑作用。当然,仅通过一次处理,不一定完全除去噪声,可通过多次执行,直到灰度不再改变为止,以求最佳效果。可见通过选择掩模平滑算法处理后既可以消除噪声,又不破坏区域边界的细节。

算法具体实现步骤如下:

(1)对图像中的每一个待处理的像素,按图 1 做出 9 个掩模,分别计算每一掩模的灰度均值和方差。

(2)对 9 个掩模的方差排序,选出方差最小的掩模。

(3)用该掩模的灰度均值代替待处理像素的灰度值。

其中各掩模的灰度均值μ和方差σ^2的计算公式如下:

$$\mu=\frac{1}{N}\sum_{i\in\Omega}f_i \quad \sigma^2=\frac{1}{N}\sum_{i\in\Omega}(f_i^2-\mu^2)$$

式中,Ω是对应的掩模,f_i是掩模内各像素的灰度值,N是掩模内像素的个数。

选择掩模平滑算法具有完成滤波操作的同时,又不破坏区域边界细节的优点。但是对要处理的每一个像素,都需要计算 9 个掩模的均值和方差,其中 9 个掩模的均值需要 8+6×8=56次加减法和 9 次除法,9 个掩模的方差需要 13×8+17=121 次加减法和 14×8+18+9=139 次乘除法。那么对于一幅 $N\times N$ 大小的图像,要计算所有像素点的滤波输出,仅均值和方差的计算量就是:$195N^2$ 次加减法和 $148N^2$ 次乘除法,当 N 很大时,计算量将会剧烈增加。因此该算法的缺点是算法的时间复杂度大。

3　均值及方差的快速计算

在积分图像和平方积分图像基础上,可实现一种计算速度与图像窗口大小无关的灰度均值和方差的快速计算方法[7]。本文将这种灰度均值和方差快速算法应用于选择掩模平滑算法中,大大减少了算法中均值和方差的计算量,从而能提高了整个算法的计算速度。

3.1　积分图像

积分图像的任意一点(x,y)表示图 2 所示的原图像斜线区域的灰度值总和,即:

$$ii(x,y)=\sum_{x'\leqslant x,y'\leqslant y}i(x',y') \tag{1}$$

其中,$ii(x,y)$表示积分图像,$i(x',y')$表示原图像。$ii(x,y)$可用式(2)、式(3)迭代计算得到:

$$s(x,y)=s(x,y-1)+i(x,y) \tag{2}$$

$$ii(x,y)=ii(x-1,y)+s(x,y) \tag{3}$$

其中,$s(x,y)$表示一列的积分,且$s(x,-1)=0$,$ii(-1,y)=0$。求积分图像,只需遍历一次原图像即可得到,计算开销很小。

在没有积分图像的情况下,要计算一幅图像中大量矩形区域的灰度值总和,需要对每一个矩形区域进行遍历,计算量很大。而在积分图像的基础上计算则很方便,它可以避免对区域的遍历。例如,图 3 所示窗口 D(阴影)的灰度值总和,可以用积分图像相应点 1、2、3、4 的取值快速计算出来,即窗口 D 的灰度值总和=4+1−(2+3),且其计算速度与窗口的大小无关。

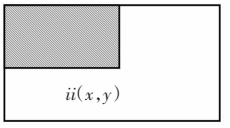

图 2　积分图像示意图

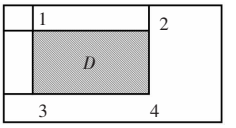

图 3　积分图像用法示意图

3.2　平方积分图像

为快速计算图像窗口的方差,在积分图像的基础上提出了平方积分图像的概念[7]。平方积分图像的任意一点(x,y)表示图 2 所示的原图像阴影区域的灰度值平方的总和,即:

$$sii(x,y)=\sum_{x'\leqslant x,y'\leqslant y}[i(x',y')\times i(x',y')] \tag{4}$$

其中 $sii(x,y)$为平方积分图像,$i(x',y')$为原图像,$sii(x,y)$可用式(5)、式(6)迭代计算得到:

$$ss(x,y)=ss(x,y-1)+i(x,y)\times i(x,y) \tag{5}$$

$$sii(x,y)=sii(x-1,y)+ss(x,y) \tag{6}$$

其中,$ss(x,y)$表示一列的平方积分,且 $ss(x,-1)=0$,$sii(-1,y)=0$。平方积分图像可与积分图像在对原图像的同一次遍历中算出,计算开销很小。运用平方积分图像可快速算出图 3 所示窗口 D 的灰度值平方总和。即 D 的灰度值平方总和=4+1−(2+3),其中 1、2、3、4 表示平方积分图像中相应点的取值,且计算速度与窗口 D 的大小无关。

3.3　灰度均值及方差的快速计算方法

假设某一矩形窗口的左上角顶点坐标为(x,y),窗口宽为 N 像素,高为 M 像素,窗口灰度矩阵为 $R[i][j]$,则窗口的灰度均值μ和方差σ^2的计算公式如下:

$$\mu=\frac{\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}R[i][j]}{M\times N} \tag{7}$$

$$\sigma^2=\frac{\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}(R^2[i][j]-\mu^2)}{M\times N} \tag{8}$$

公式(7)表明,直接计算窗口的灰度均值将需对窗口遍历一次。因为,项$\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}R[i][j]$表示窗口灰度值的总和,所以可以用积分图像来计算,这样能避免对窗口遍历,从而提高窗口灰度均值的计算速度。公式(8)则表明,直接求窗口的灰度方差,需要先计算窗口的灰度均值,然后再对窗口遍历一次才能完成。为避免对窗口的遍历,对公式(8)作如下变换:

$$\sigma^2=\frac{\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}(R^2[i][j]-\mu^2)}{M\times N}=\frac{\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}R^2[i][j]}{M\times N}-\frac{\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}\mu^2)}{M\times N}$$

即：

$$\sigma^2=\frac{\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}R^2[i][j]}{M\times N}-\mu^2 \tag{9}$$

其中，$\sum_{j=y}^{y+M-1}\sum_{i=x}^{x+N-1}R^2[i][j]$表示窗口的灰度值平方总和，可以用平方积分图像来计算。这样，根据公式(7)和式(9)，并利用预先得到的积分图像和平方积分图像，就可完全避免对每一个窗口的遍历，快速地计算出窗口的灰度均值及方差。

4 快速选择掩模平滑算法

将以上介绍的灰度均值及方差的快速计算方法运用于一般的选择掩模平滑算法中，可大大提高算法的计算速度，从而得到了快速选择掩模平滑算法。

4.1 掩模灰度均值和方差的计算

灰度均值及方差的快速计算方法利用预先得到的积分图像和平方积分图像，可完全避免对每一个窗口的遍历，从而提高了计算的速度。所以快速选择掩模平滑算法将运用此算法来计算图 1 的 9 个掩模的均值和方差。

假设原图像为 f，积分图像为 ii，平方积分图像为 sii，平滑后的图像为 g，当前要处理的像素为(x,y)，编号为 i 的掩模的均值和方差分别用 $mean[i]$和 $fangcha[i]$表示和存储，则当前的 9 个掩模的均值和方差计算如下：

对于图 1 中编号为 0 的 1 个正方形掩模，其均值和方差可由图 3 对应的公式直接计算，所以：

$mean[0]=[ii(x+1,y+1)+ii(x-1,y-1)-ii(x-1,y+1)-ii(x+1,y-1)]/9.0$

$fangcha[0]=[sii(x+1,y+1)+sii(x-1,y-1)-sii(x-1,y+1)-sii(x+1,y-1)]/9.0-(mean[0])^2$

而对于图 1 中编号为 1 的五边形掩模因其不是矩形，所以计算时，除了要用到积分图像和平方积分图像之外，还要用到原图像的部分灰度值。所以：

$mean[1]=[ii(x-1,y+1)+ii(x-2,y-1)-ii(x-2,y+1)-ii(x-1,y-1)+f(x,y)]/7.0$

$fangcha[1]=[sii(x-1,y+1)+sii(x-2,y-1)-sii(x-2,y+1)-sii(x-1,y-1)+f(x,y)^2]/7.0-(mean[1])^2$

同理可得 $mean[2]$,$fangcha[2]$,$mean[3]$,$fangcha[3]$,$mean[4]$,$fangcha[4]$。

4 个六边形掩模的计算与五边形掩模的计算方法类似，例如：

$mean[5]=(ii(x,y)+ii(x-2,y-2)-ii(x-2,y)-ii(x,y-2)-f(x-2,y)-f(x,y-2))/7.0$

$fangcha[5]=[sii(x,y)+sii(x-2,y-2)-sii(x-2,y)-sii(x,y-2)-f(x-2,y)^2-f(x,y-2)^2)]/7.0-(mean[5])^2$

同理可得 $mean[6]$,$fangcha[6]$,$mean[7]$,$fangcha[7]$,$mean[8]$,$fangcha[8]$。

4.2 算法实现步骤

假设原图像为 f，其积分图像为 ii，平方积分图像为 sii，平滑后的图像为 g。图像宽为 $wide$ 像素，高为 $height$ 像素。则快速选择掩模平滑算法的具体实现步骤如下：

(1)获取原图像 f 大小和图像数据；

(2)对原图像进行遍历，得到其积分图像 ii 和平方积分图像 sii，伪代码如下：

```
for:j=0,……,height-1
  for:i=0,……,wide-1
{
  //========积分图像的计算
    if(j==0)s(i,j)=f(i,j);
  else  s(i,j)=s(i,j-1)+f(i,j);
    if(i==0)ii(i,j)=s(i,j);
  else  ii(i,j)=ii(i-1,j)+s(i,j);
  //======平方积分图像的计算
  if(j==0)
      ss(i,j)=f(i,j)×f(i,j);
  else
      ss(i,j)=ss(i,j-1)+f(i,j)×f(i,j);
  if(i==0)
      sii(i)=ss(i,j);
  else
      sii(i,j)=sii(i-1,j)+ss(i,j);
}
```

(3)为输出图像 g 分配相同大小数据缓冲区，并以原数据初始化；

(4)逐行、逐列移动，循环确定输出图像各位置处的像素灰度值；

```
for:j=2,……height-3
for:i=2,……wide-3
{
```

计算 9 个掩模均值 $mean[n]$与方差 $fangcha[n]$，$n=0,1,2,\cdots,8$

确定最小方差=min{$fangcha[n]$,$n=0,1,2,\cdots,8$}，并求得最小方差对应的标号 n

```
g(i,j)=mean[n]
}
```

(5)把缓冲区中改动的数据复制到原数据区中。

4.3 算法时间复杂度比较

对于一幅大小为 $N\times N$ 的图像，用一般的选择掩模平滑算法和改进后的快速选择掩模平滑算法的时间复杂度对比见表 1。

表 1 算法时间复杂度对比表

	均值		方差		合计	
	加减法	乘除法	加减法	乘除法	加减法	乘除法
改进前	$56N^2$	$9N^2$	$121N^2$	$139N^2$	$177N^2$	$148N^2$
改进后	$41N^2$	$9N^2$	$50N^2$	$31N^2$	$91N^2$	$40\ N^2$

注：改进后的算法时间包括积分图像与平方积分图像的计算时间。

从表 1 中可以看到改进后的快速掩模平滑算法比改进前的一般选择掩模平滑算法所用的计算时间大大减少，尤其是乘除法的计算时间更是有很大的减少。

5 结束语

提出了一种基于积分图像的快速选择掩模平滑算法。该算法将基于积分图像和平方积分图像的灰度均值和方差的快速计算方法运用到选择掩模平滑算法中，来计算各个掩模的均值和方差，从而实现了选择掩模平滑算法的快速计算。通过比较，快速选择掩模平滑算法能够大大降低原算法的时间复杂度。

(下转 219 页)