基于多级空间压缩测度积分的中值计算方法

唐权华[1)] 雷金娥[2)] 周 艳[3)] 金炜东[3)]

[1)](西南交通大学信息科学与技术学院,成都 610031) [2)](南昌工程学院计算机科学系,南昌 330099)

[3)](西南交通大学电气工程学院,成都 610031)

摘 要 中值滤波由于可有效去除脉冲噪声,且能保护细节信息,因而成为应用最广泛的滤波算法之一,然而中值计算的速度问题在一定程度上制约了中值滤波的应用。为提高中值计算的速度,提出了一种利用测度积分快速计算中值的方法。该方法首先使用阶梯函数对数组进行拓展,并通过给出中值与函数测度积分的关系,实现了基于函数测度积分的中值计算算法;然后针对数组值分布范围较大时,中值计算时间增长的情况,给出了对测度空间进行压缩的解决方案;最后将空间压缩的方案推广到了多级压缩。实验表明,新方法计算中值的速度优于已往各种中值算法,并能与多数改进的中值滤波算法相结合。

关键词 中值计算 测度 测度积分 阶梯函数 空间压缩

中图法分类号: TP391.41 **文献标识码**: A **文章编号**: 1006-8961(2009)04-0642-05

Multi-scale Space Compressed Measure-integral Based Median Computation

TANG Quan-hua[1)], LEI Jin-e[2)], ZHOU Yan[3)], JIN Wei-dong[3)]

[1)](*School of Information Science & Technology, South. west Jiaotong University, Chengdu* 610031)

[2)](*School of Computer Science, Nanchang Institute of Technology, Nanchang* 330099)

[3)](*School of Electrical Engineering, South. west Jiaotong University, Chengdu* 610031)

Abstract Due to its effectiveness for removing impulse noise and preserving detail features, median filtering has long been a popular tool of filtering algorithm. But in practice, an important issue of applying median filtering is the filtering speed. In this paper, a fast median algorithm based on measure-integral is proposed. A step function is employed to expand the array for median, then the relationship between median and measure-integral is deduced and an algorithm is gained by it. To the question that the compute time of the algorithm increases rapidly when the values of the array or the function range become large, a method of compress the measure space is put forward, which is extended to multi-scale compress method at last. Experiments show that multi-scale space compressed measure-integral based median computation(MCMIM) has higher processing speed and can be combined with most of the earlier improved median filters.

Keywords median computation, measure, measure-integral, step function, space compression

1 引 言

中值体现了一组数据的集中趋势,其比均值更能体现实验数据的真实值,已被广泛应用于各学科的研究中。中值滤波因其能去除脉冲噪声,且能保护边缘信息而成为目前应用最广泛的图像滤波算法之一。然而,中值计算的速度问题在一定程度上制约了中值滤波的应用,即使在计算机硬件高速发展的今天,传统的中值滤波算法也较难应用于实时的

基金项目:国家自然科学基金项目(60572143)

收稿日期:2007-05-25;**改回日期**:2007-10-23

第一作者简介:唐权华(1976 ~),男。现为西南交通大学信息科学与技术学院博士研究生。研究领域为视频信号处理等。E-mail: quanhuatang@163.com

视频信号处理中。为提高中值滤波的速度,国内外学者做了大量的研究,已获得许多成果。张丽等人提出了使用均值加速中值滤波计算(MFM)的方法[1],使中值计算的速度得到一定提高,在滤波窗口较大时,均值加速后的中值运算时间约为传统算法的四分之一。李刚与范瑞霞提出了一种快速中值滤波算法——LFM(median filter based on the last frame)[2],该算法利用矩形窗口按行(列)移动时不改变元素的排序和中值信息的特点,使中值滤波算法的复杂度降低到 $O(N)$。曹治华、宋斌恒将这一结果推广到多种形状的滤波窗口[3]。此外,为提高中值滤波的性能,还出现了如自适应中值滤波[4](改变窗口大小)、矢量中值滤波[5]、加权中值滤波[6]、基于拓扑学的中值滤波[7]、特殊形状滤波窗口[3]的中值滤波等算法。

然而,上述针对滤波速度和性能的不同改进的滤波方法往往很难结合。尽管提高中值滤波速度的算法通常都利用了窗口大小固定、位置定向平移、形状不变等条件,可是在进行性能改进时却一般需要破坏这些条件。均值加速虽然没有利用特殊条件,但仅在一定程度上提高了运算速度,仍然很难应用于实时视频信号处理中。本文从中值计算的角度来解决中值滤波的速度问题,即先利用阶梯函数对中值问题进行建模,再使用基于测度积分的中值计算方法,并借助对测度空间进行压缩、多级压缩的手段来减弱算法速度对数据值分布范围大小的依赖性。

2　基于多级压缩测度积分的中值计算

2.1　数组中值与测度

为利用函数的测度积分进行中值计算,首先将中值的定义推广到函数。传统的中值定义如下:

定义 1　令 $\boldsymbol{A}=\{x_i\}(i=1,2,\cdots,N,N$ 为奇数$)$ 为待求中值的 1 维数组,$\tilde{\boldsymbol{A}}=\{\tilde{x}_j \mid \tilde{x}_j\in\boldsymbol{A}$,且 $\tilde{x}_j\leqslant\tilde{x}_{j+1},\forall 1\leqslant j<N\}$,则中值定义为 $\tilde{x}_{\frac{N+1}{2}}$。

事实上,求解中值并不一定需要排序,中值的本质是这样一个值,即数组中数值比它小的元素刚好过半。基于此,本文给出了中值的以下定义:

定义 2　令 $\boldsymbol{A}=\{x_i\}(i=1,2,\cdots,N,N$ 为奇数$)$ 为待求中值的 1 维数组,令

$$\boldsymbol{L}_k=\{x_{i_k}\mid x_{i_k}<x_k,x_{i_k}\in\boldsymbol{A}\}$$

$$\boldsymbol{G}_k=\{x_{i_k}\mid x_{i_k}\leqslant x_k,x_{i_k}\in\boldsymbol{A}\}$$

则称满足条件 $|\boldsymbol{L}_k|<\frac{N-1}{2}$,且 $|\boldsymbol{G}_k|\geqslant\frac{N-1}{2}$(其中 $|\bullet|$ 表示待求中值数组元素的个数)的 x_k 为数组 $\boldsymbol{A}$ 的中值。

显然,定义 2 与定义 1 等价,并且更能体现中值的本质。但从定义 2 出发仍然不便于计算中值,因为计算 $\boldsymbol{L}_k$ 的代价为 N,要找到满足条件的 k 值,代价还是会接近 N^2。为进一步分析中值,本文利用一个函数来模拟数组,函数定义如下:

$$f(t)=x_{[t]}(x_i\in\boldsymbol{A},[\bullet]\text{表示取整运算})\qquad(1)$$

则有 $f(i)=x_i(i=1,2,\cdots,N)$。进一步,函数中值定义如下:

定义 3　令 $D_f(y)$ 为 $f(t)$ 的测度积分,即 $D_f(y)=\int_{f(t)\leqslant y}\mathrm{d}t$,则称满足条件 $D_f(\xi)\leqslant\frac{N+1}{2}$,且 $D_f(y)>\frac{N+1}{2}(\forall y>\xi)$ 的 ξ 为函数 $f(t)$ 的中值。

容易证明,式(1)定义的函数中值与数组 $\boldsymbol{A}$ 的中值相等。计算函数 $f(t)$ 的中值 ξ 可以分两步进行,即首先计算 $f(t)$ 取不同值的测度并累加,直到累加和大于或等于 $(N+1)/2$,根据此思路即可得到基于测度积分计算中值的基本算法,算法步骤如下。(为方便计算,设 x_i 为整数,实际不为整数时,则可以先进行整型化。)

算法 1　步骤如下:

(1)令 $M(j)=0,0\leqslant j\leqslant j_{\max}$(其中 $[0,j_{\max}]$ 为函数的值域),并从 $i=0$ 到 N 统计 $f(t)$ 的测度,依次计算 $M(x_i)=M(x_i)+1$;

(2)令 $D(0)=M(0),j=1$;

(3)计算测度 M 的积分 $D(j)=\sum_{i=1}^{j}M(i)$,$D(j)=D(j-1)+M(j)$;

(4)如果 $D(j)>\frac{N}{2}$,则 j 即为所求的中值,结束,否则将 j 加 1,返回步骤(3)。

算法 1 中步骤(1)为求函数在各值上的测度,运算量为 N,步骤(2)~步骤(4)为对测度 M 进行积分,其最大的计算量为 $j_{\max}$,因此整个算法的最大计算量为 $N+j_{\max}$。当 $\frac{j_{\max}}{N}$ 较小时,此算法优势比较明显。但当 $\frac{j_{\max}}{N}$ 较大时,此算法与传统的中值算法相比优势较小,在某些情况下甚至可能处于劣势。

2.2 测度空间的压缩

当数组的值分布范围较大时,其对应的函数值域[0,$j_{\max}$]也较大,则算法 1 中求解测度积分的过程会耗费大量的时间。通过仔细分析可以发现,在这种情况下,积分过程出现了许多加零的无用运算。如果能除去这些加零运算,则积分过程的最大运算量可缩小到 N。本文提出用压缩测度空间的方法来去除或减少函数测度积分过程中的加零运算,当测度空间变小时,显然积分过程也就变短了。但空间压缩必须保证以下两个条件:(1)能从压缩后的空间恢复到未压缩空间的位置;(2)压缩后的积分值不变。为满足这两个条件,将压缩后的测度积分函数定义如下:

$$\tilde{D}_f(y)=\int_{f(t)\leqslant Ky}\mathrm{d}t \quad (\text{其中 } K \text{ 为压缩比例}) \quad (2)$$

由式(2)的定义及测度积分的基本性质容易得出 $\tilde{D}_f(y)=D_f(Ky)$,于是可以通过计算 $\tilde{D}_f(y)$ 来获得 $D_f(Ky)$ 的值,这样就可使积分运算的速度提高 K 倍。然而现在并不能直接得到中值,令 $\tilde{\xi}$ 为 $\tilde{D}_f(y)$ 的中值,即 $\tilde{D}_f(y)\leqslant\frac{N+1}{2}$ 且 $\tilde{D}_f(y)>\frac{N+1}{2}(\forall y>\xi)$,则得到的结论是 $D_f(K\xi)\leqslant\frac{N+1}{2}$ 且 $D_f(K(\xi+1))>\frac{N+1}{2}$,也就是说,只能由此得到所求中值的范围在 $K\xi\sim K(\xi+1)$ 之间。为进一步确定中值的位置,必须再求实际测度空间上的积分。不过,由于此时已经获得 $D_f(K\xi)$,故现在只需要再累加比 $K\xi$ 大的部分测度就可以了。由此就得到了以下新的中值求解算法。

算法 2 步骤如下:

(1)令 $M(j)=0,\tilde{M}(j)=0,0\leqslant j\leqslant j_{\max}$,从 $i=0$ 到 N 统计 $f(t)$ 的测度,依次计算

$$M(x_i)=M(x_i)+1,\tilde{M}\left(\left[\frac{x_i}{K}\right]\right)=\tilde{M}\left(\left[\frac{x_i}{K}\right]\right)+1$$

(2)令 $S(0)=\tilde{M}(0),j=1$;

(3)$S(j)=S(j-1)+\tilde{M}(j)$;

(计算压缩空间上的测度积分 $S(j)=\sum_{i=1}^{j}\tilde{M}(i)$)

(4)如果 $S(j)>\frac{N}{2}$,则令

$S(j)=S(j)-\tilde{M}(j),j=(j-1)\times K$,转入步骤(5),否则将 j 加 1,返回步骤(3);

(5)$S(j)=S(j-1)+M(j)$(计算原空间上的测度积分 $S(j)=\sum_{i=1}^{j}M(i)$);

(6)如果 $S(j)>\frac{N}{2}$,则 j 即为所求中值,结束,否则将 j 加 1,返回步骤(3)。

算法 2 最多需要 $3N+K+j_{\max}/K$ 次运算。要使运算量达到最小,应让 $3N+K+j_{\max}/K$ 关于 K 的导数为零,由此可得到 $K=\sqrt{j_{\max}}$。于是算法 2 的总体最大运算量应为 $3N+2\sqrt{j_{\max}}$。与算法 1 相比,两种算法的最大运算量的差值为 $j_{\max}-2\sqrt{j_{\max}}-2N$,如果 $j_{\max}-2\sqrt{j_{\max}}>2N$,则算法速度还可以得到提升。令传统的中值算法的运算量为 $\sigma N\log N,(\sigma>1)$,则与算法 2 的运算量的差为 $\sigma N\log N-3N-2\sqrt{j_{\max}}$,由此看出,当 $\frac{\sigma N\log N}{3N+2\sqrt{j_{\max}}}>1$ 时,算法 2 优于传统的中值算法。

2.3 测度空间的多级压缩

算法 2 在算法 1 的基础上加入了测度空间压缩的过程,虽然降低了算法复杂度对 $j_{\max}$ 的依赖性,但在 $j_{\max}$ 特别大,而 N 较小时,算法 2 仍然存在一定性能上的缺陷。为进一步提高测度积分的计算速度,可将算法 2 压缩测度空间的方法进行推广。算法 2 与算法 1 相比,改进的实质是利用压缩空间去计算非压缩空间的积分,同理也可以用压缩程度高的测度空间去计算压缩程度低的测度积分,也就是说,可以先将测度空间进行多级压缩,然后分级计算测度积分。这样无论函数值的分布范围有多大,总可以通过增大压缩程度和压缩层次来适应测度积分的计算。

在进行多级测度空间压缩以后,就可以迅速得到函数的测度积分,但由上一节的分析可知,压缩测度空间也会带来 $2N$ 数量的计算新空间测度的负担。由此可见,并非测度空间压缩的级别越多,算法性能就越好,还需要一个决定压缩级数的优化策略。为获得多级压缩后算法的运算量,先假设进行 L 级空间压缩,各级压缩比为 K_l,则计算各级空间上的测度需要的运算量为 $(2L+1)N$,计算测度积分的过程需要的运算量为 $\sum_{l=0}^{L}\frac{K_{l-1}}{K_l}$,从而总的运算量为 $(2L+1)N+\sum_{l=0}^{L}\frac{K_{l-1}}{K_l}$。由于 $\sum_{l=0}^{L}\frac{K_{l-1}}{K_l}=K_0=j_{\max}$,所以为使 $\sum_{l=0}^{L}\frac{K_{l-1}}{K_l}$ 最小,应使 $\frac{K_{l-1}}{K_l}=\sqrt[L]{j_{\max}}$,($\forall 0\leqslant l\leqslant$

L),于是总的运算量变为

$$(2L+1)N+(L+1)\sqrt[L]{j_{\max}} \tag{3}$$

其中前一部分为 L 的增函数,后一部分在(0,$+\infty$)上关于 L 先降后升,所以式(3)也应该是 L 的一个先降后升的函数,并且极小值只可能在后一部分的单减区间上。而后一部分的单减区间为(0,$\frac{\ln j_{\max}+\sqrt{\ln^2 j_{\max}+4\ln j_{\max}}}{2}$),是一个较小的范围,因而可以在这个区间上用试探的方法来获取式(3)的极小值。

一旦确定了需要分级压缩的次数,就可以像算法 2 一样,先计算各级压缩空间上的测度,再按空间压缩比从大到小计算函数的测度积分。进行多级测度空间压缩后的算法总结如下:

算法 3　步骤如下:

(1)从 $i=0$ 到$\frac{\ln j_{\max}+\sqrt{\ln^2 j_{\max}+4\ln j_{\max}}}{2}$计算 $C_i=N(i+1)+(i+1)\sqrt[i]{j_{\max}}$,如果 $C_i<C_0$ 则令 $C_0=C_i, L=i$。令 $K_0=j_{\max}, K_l=\sqrt[L-l+1]{j_{\max}}$, $l=1,2,\cdots,L$。

(2)对所有的 $l\in\{0,1,2,\cdots,L\}$,令 $M_l(j)=0$, $0\leqslant j\leqslant j_{\max}$,从 $i=0$ 到 N 统计 $f(t)$ 的各级测度,并依次计算 $M_l(x_i)=M_l(x_i)+1$;

(3)令 $S(0)=M_L(0), j=1, l=L$;

(4)$S(j)=S(j-1)+M_l(j)$;

(5)如果 $S(j)\leqslant\frac{N}{2}$则令 $j=j+1$,返回步骤(4),否则转步骤(6);

(6)令 $S(j)=S(j)-M_l(j), j=(j-1)\times K_l$,若 $l>0$,则令 $l=l-1$,转入步骤(4),否则 j 为所求的中值,结束。

由前面的分析知道,算法 3 的运算量为式(3),由于 L 待定,所以直接求式(3)的极值比较困难,但实验表明,在图像位深度大于 8 时,算法 3 的计算速度的确比算法 1 和算法 2 快。

3　实　验

3.1　实验说明

在编程实现时,为了提高程序的运行速度,还应该注意以下几方面问题:(1) 计算整数问题,计算的结果均保留整数;(2) 除法优化问题,计算机在做除法时会耗费较多的时间,因此总是将 $j_{\max}$ 扩展到 2^z ($z\in\mathbf{Z}$)进行计算;(3) 乘方问题,算法 3 要求 $\tilde{K}=\sqrt[L]{j_{\max}}$,而实际上并非对所有 $j_{\max}$、L 都能保证 $\tilde{K}$ 为整数,在 $\tilde{K}$ 不为整数时,处理方法是:若 $2^{z-1}<j_{\max}\leqslant 2^z$ ($z\in\mathbf{Z}$),则取

$$K_l=\begin{cases}2^{[\frac{z}{L}]l} & l<L\\ 2^{L\bmod z} & l=L\end{cases}$$

3.2　几种基于测度积分中值计算方法的比较

为对比几种基于测度积分的中值算法的性能,选择一般情况下通用的算法,即用算法 1、算法 2、算法 3 与传统中值算法进行了 2 维中值滤波实验。实验时,使用 VC + +6.0 进行编程,测试对象为 512 × 512pixel 大小的 Lena 图像,运行环境为 CPU 主频 1.6G、内存 512M 的 PC 兼容机,Windows XP 操作系统,实验结果如表 1 所示。

表 1　基于测度积分的中值算法(MM)、基于压缩空间测度积分的中值算法(CM)与基于多级压缩空间测度积分的中值算法(MCM)在不同窗口与颜色深度下的滤波时间对比

Tab. 1　Measure-integral based(MM), compressed measure-integral based (CM), and multi-scale compressed measure-integral based(MCM) median filters

图像	算法	各种滤波算法的滤波时间(ms)					
		3 × 3	5 × 5	7 × 7	9 × 9	11 × 11	13 × 13
4-bits 灰度图	MM	*47*	78	94	125	140	172
	CM	78	110	140	171	203	250
	MCM	141	156	218	266	312	375
8-bits 灰度图	MM	297	328	344	360	391	406
	CM	109	140	172	188	234	250
	MCM	157	187	234	281	344	391
16-bits 彩色图	MM	44 266	44 891	43 438	43 157	43 047	42 625
	CM	531	547	594	625	656	688
	MCM	297	328	422	516	531	594
24-bits 彩色图	MM	inf	inf	inf	inf	inf	inf
	CM	19 032	18 953	18 922	18 875	18 843	18 812
	MCM	657	813	937	1 078	1 219	1 328

从表 1 可见,当滤波图像的颜色深度较小时,无论窗口大小,直接使用测度积分计算中值都较快,且运算时间相差不大。但随着颜色深度增大,分级压缩和多级压缩的算法逐渐占优。对于 24bits 的彩色图,直接使用测度积分变得不可用,分级压缩的算法也变得很慢,而多级压缩算法则没受太大影响。综合考虑,一般情况下,使用多级压缩的测度积分是较好的选择。

3.3 基于测度积分算法与其他中值滤波方法对比

为了对比各种改进的中值计算方法的速度,使用 256 级灰度、512 × 512pixels 大小的 Lena 图像,对传统的中值滤波算法(SM)、均值加速中值滤波算法(MFM)、利用已排序信息的中值滤波算法(LFM)和基于多级压缩空间测度积分的中值算法(MCM),在 3.2 节所述的环境下进行了实验。为了更好地体现新算法可扩展的优势,实验过程中还在 MCM 算法中引入了部分 LFM 机制。实验结果如表 2 所示。

表 2 传统中值滤波算法与几种改进算法的运行时间对比

Tab. 2 Computation time of conventional median filter and improved filters

算法	各种滤波算法的运算时间(ms)					
	3 × 3	5 × 5	7 × 7	9 × 9	11 × 11	13 × 13
SM	266	1 619	5 859	15 765	34 703	67 015
MFM	190	805	2 163	3 941	7 711	13 403
LFM	131	269	576	878	1 304	1 812
MCM	105	125	157	183	212	243

4 结 论

当数组的值分布范围较小时,将测度积分的概念应用于中值计算会极大地提高运算速度,且受数组元素个数的影响较小;但当值分布范围增大时,由于测度积分会耗费较多的时间,所以必须对测度空间进行压缩。实验表明,对测度空间进行多级压缩后再进行中值计算,可以解决中值计算的速度问题。

由于基于多级空间压缩测度积分的中值计算方法不依赖于中值滤波的特定条件,它可以应用于任何使用中值的环境,并可以轻松地与以往的各种中值滤波改进算法进行结合,比如与加权中值滤波算法相结合,只需要将对应的权值作为测度计算即可。作者下一步将开展如何将该算法与中值滤波的各种变形,如自适应中值滤波、加权中值滤波、矢量中值滤波等算法进行结合的研究,以期获得速度更快、效果更好的图像滤波算法。

参考文献(References)

1 Zhang Li, Chen Zhi-qiang, Gao Wen-huan, *et al*. Mean-based fast median filter[J]. Journal of Tsinghua University(Science and Technology), 2004, **44**(9): 1157-1159. [张丽, 陈志强, 高文焕等. 均值加速的快速中值滤波算法[J]. 清华大学学报(自然科学版), 2004, **44**(9): 1157-1159.]

2 Li Gang, Fan Rui-xia. A new median filter algorithm in image tracking systems [J]. Journal of Beijing Institute of Technology, 2002, **22**(3):76-378. [李刚,范瑞霞. 一种改进的图像中值滤波算法[J]. 北京理工大学学报,2002, **22**(3): 76-378.]

3 Cao Zhi-hua, Song Bin-heng. A fast algorithm for median filtering in multiform window[J]. Application Research of Computers, 2006, **3**(1):85-88. [曹治华,宋斌恒. 多种形状窗口下的快速中值滤波算法[J]. 计算机应用研究,2006, **3**(1):85-88.]

4 Hu Wang, Li Zhi-shu, Huang Qi. An adaptive median filter based on the double windows and extremum-compressing[J]. Journal of Image and Graphics, 2007, **12**(1):43-50. [胡旺, 李志蜀, 黄奇. 基于双窗口和极值压缩的自适应中值滤波[J]. 中国图象图形学报, 2007,**12**(1): 43-50.]

5 Regazzoni C S, Teschioni A. A new approach to vector median filtering based on space filling curves[J]. IEEE Transactions on Image Processing,1997,**6**(7): 1025-1037.

6 Lin Tzu-chao. A new adaptive center weighted median filter for suppressing impulsive noise in images[J]. Information Sciences, 2007,**177**(4): 1073-1087.

7 Hakan Güray Senel, Richard Alan Peters, Benoit Dawant. Topological median filters [J]. IEEE Transactions on Image Processing, 2002,**11**(2): 89-104.