CH-011

FPGA のための小型 5×5、7×7 メディアンフィルタ
Compact 5×5, 7×7 Median Filters For FPGA

依田 晴夫†
Haruo Yoda

1. はじめに

メディアンフィルタはエッジ情報を保存しながらペッパーノイズを除去できる有効な画像処理フィルタである。しかし、その優れた特徴にもかかわらず、n×n ウィンドウ中の画素値を大きい順にソートしてその中央値（メディアン）を出力するという原理面での計算量の多さによって、広く工業用途に適用するまでには至っていない。これまで、ソートアルゴリズムの改良の他、ウィンドウ内画素値の濃淡ヒストグラムに着目した高速計算方法[1][2]も提案されているが、広い用途への適用に対しては計算速度、装置規模的に不充分である。

計算速度の課題に関しては、FPGA を用いた並列回路化が最も有効である。その回路規模の小型化についても、通常の3×3のメディアンフィルタに限定すれば、既に縦、横、斜め方向の部分ソートを用いた有効なアルゴリズムが提案され[3][4]、実装上の問題も種々検討されて[5][6][7][8]、ほぼ問題は解決されている。

しかし、5×5、7×7 メディアンフィルタについては、その有用さに関わらず、未だ有効な小型化アルゴリズムは見当たらない。ソート処理の工夫による5×5フィルタのリアルタイム化の検討も報告されているが[9]、回路規模的に広い用途への適用は難しい。本論文の目的は、5×5、7×7 フィルタに関しても、小型化を可能にする新たなアルゴリズムを提案することである。

本論文では、まず従来から知られている並列ソート回路の構成と、[3]で提案された小型 3×3 メディアンフィルタについて説明する。[3]では、その小型化アルゴリズムの正当性を数値計算結果のみから保証していので、ここでは改めて新たな方法でその正当性を証明する。次に、その方法を拡張し、5×5、7×7 フィルタに対する新たな小型化アルゴリズムを提案する。

また、新機能としてウィンドウ内の有効画素のみを対象としてそのメディアンを計算する機能を追加する。この機能は、例えば立体視処理で計算された距離画像の様に計算不能な無効画素が多く含まれる画像に対しても適用可能であり、上記の小型化とともにメディアンフィルタの適用拡大に有効である。

†高速信号処理研究家
Email: hyoda@snow.ocn.ne.jp

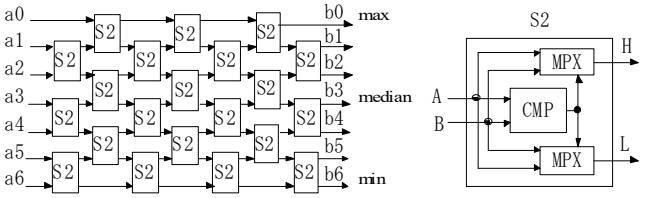
最後に、FPGA に実装した場合のリソース規模を推定し、提案方式のリソース量が従来方式に比較して大幅に削減されることを示す。

なお、メディアンフィルタに関しては、"Separable"フィルタ[10]、"Adaptive"フィルタ[11]などの研究もなされているが、出力結果が異なるので対象外とする。

2. 従来ソート回路と 3×3 メディアンフィルタ

図 1(a)に、"odd-even sort" 方式として知られている並列ソート回路の構成を示す。回路中のブロック S2 は２入力ソート回路であり、図(b)に示すように比較回路 CMP と二つの選択回路 MPX により、信号 A,B を入力して max[A,B]を H 端子に、min[A,B]を L 端子にそれぞれ出力する。

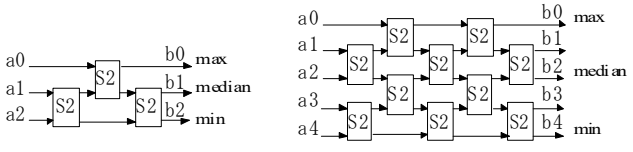
図(a)の構成において、各処理段ごとにタイミング調整用レジスタ（パイプラインレジスタ）を追加して実際に回路化すれば、入力 7 信号をソートしてその結果を 7 クロック後に並列出力する回路が実現できる。クロックごとに異なるデータを入力できるので、7 クロックのレイテンシーを持つリアルタイム回路となる。



(a) 7 入力ソート回路の構成　　(b) 2 入力ソート回路S2

図 1. "odd-even" 型並列ソート回路の例

図１は 7 入力ソート回路の例であるが、任意の入力数のソート回路も同様に構成できる。図 2 に後述の構成例に用いた３信号、５信号入力の各ソート回路の例を示す。一般に、k 個の入力要素のソート回路は k×(k-1)／2 個の２入力ソート回路で実現できる。



(a) ３入力ソート回路　　(b) ５入力ソート回路

図 2. ３入力、５入力ソート回路の構成例

次に小型 3×3 メディアンフィルタについて説明する。3×3 メディアンフィルタは図 3 を一つの 3×3 入力ウィンドウとした時、ウィンドウ内の 9 個の数値を大きい順に並べ、その中央値（メディアン）を出力するフィルタである。

a00	a01	a02
a10	a11	a12
a20	a21	a22

図 3. 入力ウィンドウ要素配列 A

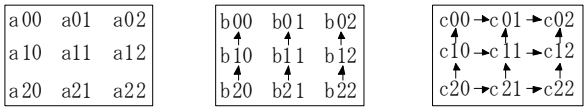
文献[3]によれば、3×3 ウィンドウのメディアンは次のステップで計算できる。

(1) 入力配列 A を各縦列でソートし配列 B とする。
{b00,b10,b20}=SORT[{a00,a10,a20}] (2.1)
{b01,b11,b21}=SORT[{a01,a11,a21}] (2.2)
{b02,b12,b22}=SORT[{a02,a12,a22}] (2.3)
(2) 配列 B を各横行でソートして配列 C とする。
{c00,c01,c02}=SORT[{b00,b01,b02}] (2.4)
{c10,c11,c12}=SORT[{b10,b11,b12}] (2.5)
{c20,c21,c22}=SORT[{b20,b21,b22}] (2.6)
(3) 配列 C の対角要素{c00,c11,c22}をソートする。
{d00,d11,d22}=SORT[{c00,c11,c22}] (2.7)
(4) d11 をウィンドウ内メディアンとして出力する。

これにより、従来必要だった 9 要素のソートが 7 個の 3 入力ソートで代替えでき、結果として 36 個必要だった 2 入力ソート回路が 21 個となり FPGA 上の回路規模が削減される。更に、通常の様に画像中の入力ウィンドウが 1 画素ずつ横に移動して設定される場合には、列ごとのソート結果はそれぞれ連続 3 ウィンドウで使用できるので、式(2.1)～(2.3)の計算は実質 1 回で済み、2 入力ソート回路は 15 個とさらに削減される。

文献[3]ではこの計算処理の正当性を、起こりうる全ての入力配列に対して数値的に確認することで保証している。ここでは次のステップにつなげるため新たな方法で以下上記の計算が正しいことを証明する。

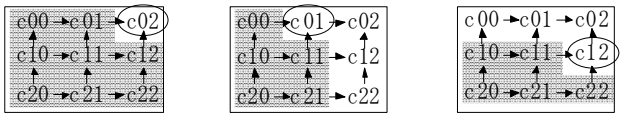
図 4.(a)は入力ウィンドウとなる前記の 3×3 画素配列 A である。この配列 A に対して列ごとに縦方向ソートして図(b)の配列 B を得る。図の矢印は数値の大小関係を示す不等号"≦"を示し、上側の数値は下側の数値より大であることを意味する。次に配列 B に対して行ごとに横方向ソートして図(c)の配列 C を得る。すなわち配列 C においては右側の数値は左側の数値より大である。この時、横方向ソートによって左右方向に数値移動が発生するが、付録の A1.【性質 1】に示したように、配列 B における上下方向の大小関係はソート後の配列 C においても保存される。



(a) 入力配列**A** (b) 列ソート後配列**B** (c) 行ソート後配列**C**

図 4. 列、行ソート後要素の大小関係

配列 C において要素間の大小関係に注目すると、右上隅の 3 要素はそれぞれ図 5 に示す網掛け部の要素を自分以下の数値としているので、右上隅 3 要素はそれぞれ自分以下の数値を 5 個以上持つこととなりメディアンとはなりえない。同様に対称位置となる左下隅の 3 要素は自分以上の数値を 5 個以上持つのでやはりメディアンとはなりえない。よって、残りの対角方向 3 要素{c00,c11,c22}をソートしてその中央値を出力すれば、求めるメディアンとなる。これにより、前記の計算ステップの正しいことが証明される。



(a)c02以下の要素 (b)c01以下の要素 (c) c12以下の要素

図 5. 右上 3 要素以下の要素 (網掛け部)

図 6 は、この方式に従った 3×3 メディアンフィルタの回路構成例である。図に示すように、ラスタ走査方式で順次入力される画像データに対して 2 個の 1 ライン長シフトレジスタ(1_LINE_SHIFTER)を配置すれば、ウィンドウ内の縦 3 画素を並列に出力できる。その 3 画素データを SORT3 によって縦ソートして 2 段のパイプラインレジスタを通せば、縦ソート済の配列 B データをそのまま並列に出力できる。

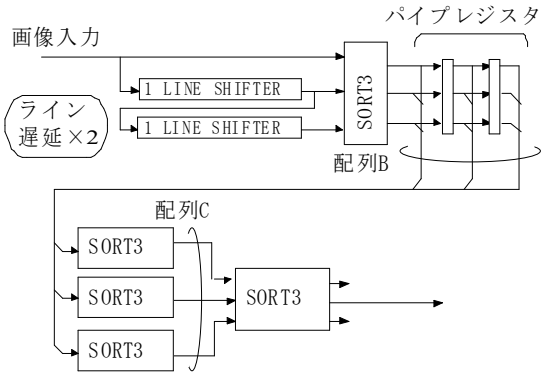


図 6. 3×3 メディアンフィルタの回路構成

この配列 B データに対して 3 個の 3 入力ソート回路を用いて行方向ソート処理し、最後に出力される配列 C の対角方向 3 画素を 3 入力ソートすれば、その中央値が求めるメ

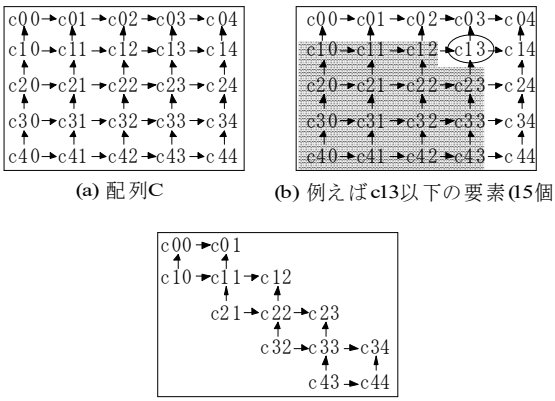
ディアンとなる。回路内の各処理は全てクロックごとに一方向に流れるパイプライン回路として構成できるので、1 画素入力と同時に 1 画素出力するリアルタイム回路とすることが出来る。この回路構成は、文献[3][5][6][7][8]記載のものと一致する。

3. 小型 5×5、7×7 メディアンフィルタ

小型 3×3 メディアンフィルタの考え方を 5×5、7×7 の場合へと拡張し、その小型化回路方式を提案する。

3.1. 5×5 メディアンフィルタ

5×5 入力配列 A に対してまず列ごとの縦ソートを行い、次に行ごとの横ソートを行い配列 C を得る。配列 C は前述の付録 A1.【性質 1】により、図 7(a)に示すような要素間大小関係を持つ。この大小関係を分析すると、右上隅の 6 要素は、図(b)に例示する丸印要素 c13 のように、それぞれ自分以下の数値要素を 14 個以上持つので、メディアンにはなりえない。同様に左下隅 6 要素も自分以上の要素を 14 個以上持つので、やはりメディアンになりえない。よって図(c)に示す残り 13 要素のメディアンが求めるメディアンとなる。



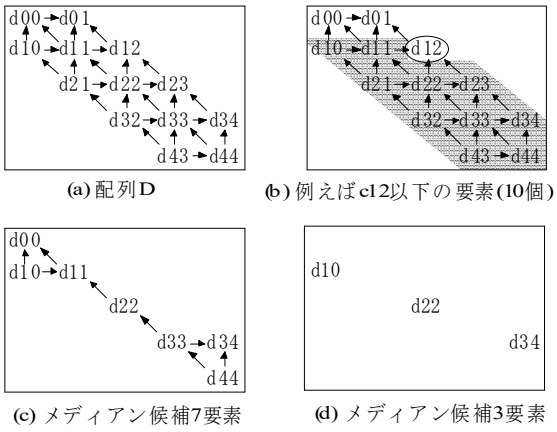
(a) 配列C　(b) 例えばc13以下の要素(15個)

(c) 明確なメディアン候補外を除去した配列Cの残り13要素

図 7. 配列 C の要素大小関係とメディアン候補要素

次に、配列 C の残り 13 要素を左上がり 45° 方向の斜め列ごとにソートし、図 8(a)に示す配列 D を得る。この時、斜め方向ソートによって各要素間の数値移動が発生するが、付録 A2.【性質 2】に示したように、配列 C の大小関係は配列 D においても保存される。

配列 D の数値間の大小関係に注目すると、上斜め列の 3 要素{d01, d12, d13}は、図(b)に例示する丸印要素 d12 のように、自分以下の要素を 7 個以上持ち、同様に下斜め列要素{d21, d32, d43}も自分以上の要素を 7 個以上持つからそれぞれメディアンになりえない。従って図(c)に示す残りの 7 数値のメディアンが求めるメディアンである。更に残り 7 要素の大小関係も詳細に分析すると、d00 と d11 は自分以下の要素を 4 個以上持ち、d44 と d33 も自分以上の要素を 4 個以上持つのでメディアンになりえない。



(a) 配列D　(b) 例えばc12以下の要素(10個)

(c) メディアン候補7要素　(d) メディアン候補3要素

図 8. 配列 D 要素の大小関係に基づく要素絞込み

結局、5×5 入力配列 A のメディアンは、図 8(d)に示す 3 要素(d10, d22, d34)のメディアンに一致する。ここで 3 要素のうち、d10 は斜めソートの最大値、d22 は斜めソートのメディアン、d34 は斜めソートの最小値にそれぞれ置き換えられるので、新たな小型化 5×5 メディアンフィルタの計算手順は次の通りとなる。

(1) 入力配列 A を各列でソートして配列 B とする。

$$\{b0i, b1i, b2i, b3i, b4i\} = SORT[\{a0i, a1i, a2i, a3i, a4i\}] \quad \text{但し、 } i=0,1,2,3,4 \tag{3.1}$$

(2) 配列 B を各行でソートして配列 C とする。

$$\{cj0, cj1, cj2, cj3, cj4\} = SORT[\{bj0, bj1, bj2, bj3, bj4\}] \quad \text{但し、 } j=0,1,2,3,4 \tag{3.2}$$

(3) 配列 D の必要 3 要素 d34, d22, d10 を計算する。

$$d34 = MIN[\{c01, c12, c13, c14\}] \tag{3.3}$$

$$d22 = MEDIAN[\{c00, c11, c22, c33, c44\}] \tag{3.4}$$

$$d10 = MAX[\{c10, c21, c32, c43\}] \tag{3.5}$$

(4) 3 要素のメディアンを計算し出力する。

$$出力 = MEDIAN[\{d10, d22, d34\}] \tag{3.6}$$

図 9 に 5×5 メディアンフィルタの回路構成例を示す。図では、3×3 の場合と同様に配列 C を生成し、4 入力最小値回路(MIN4)、5 入力ソート回路(SORT5)、4 入力最大値回路(MAX4)を配置し、その 3 個の出力からメディアンを計算出力する。図には部分計算ごとの遅延時間調整用パイプラインレジスタが省略されているが、計算単位ごとに適切にパイプラインレジスタを配置して回路化すれば、1 クロックで 1 画素を出力するリアルタイム処理回路が実現できる。

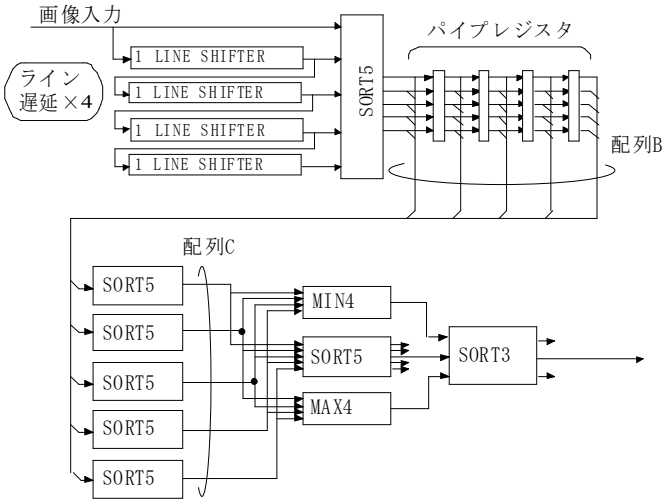
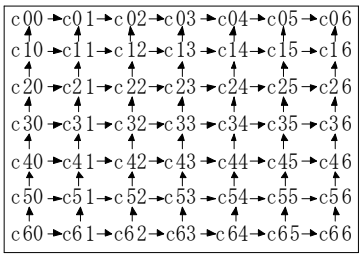


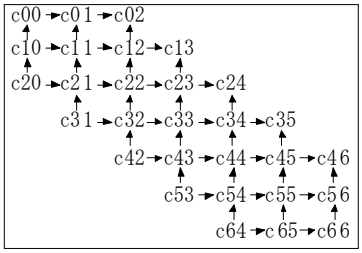
図 9. 5×5 メディアンフィルタの回路構成

3.2. 7×7 メディアンフィルタ

7×7 メディアンフィルタも 5×5 の場合と同様に、まず 7×7 入力配列 A に対して縦と横のソートを実行して図 10(a)の配列 C を得る。この配列 C において、右上隅の 10 要素は自分以下の要素を 25 個以上持ち、左下隅の 10 要素も自分以上の要素を 25 個以上持つのでメディアンとはなりえない。よって求めるメディアンは図(b)に示す残り 29 要素のメディアンに一致する。



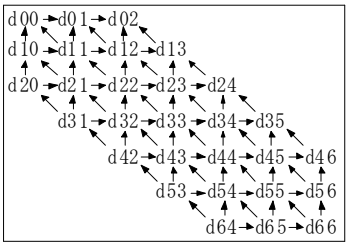
(a) 配列C



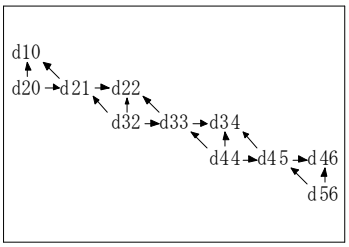
(b) 配列Cの残りメディアン候補29要素

図 10. 配列 C 要素の大小関係に基づく要素絞込み

次に、配列 C の残り 29 要素に対して左上がり 45° 斜め方向にソートして図 11(a)の配列 D を得る。この時、付録の A2. 【性質 2】により、配列 C における要素間の大小関係は配列 D においても保存される。



(a) 配列D

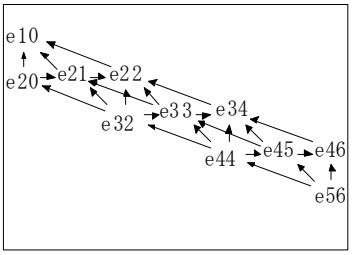


(b) 配列Dの残りメディアン候補11要素

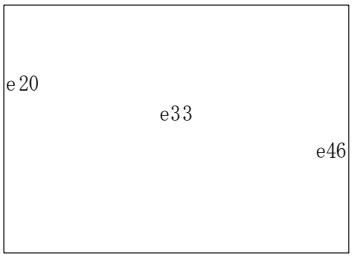
図 11. 配列 D 要素の大小関係に基づく要素絞込み

図 11(a)の配列 D において数値間の大小関係を分析すると、右上側 9 要素は自分以下の要素を 15 個以上持ち、左下側 9 数値も自分以上の要素を 15 個以上持つのでメディアンになりえない。この結果、図(b)に示す残り 11 数値のメディアンが求めるメディアンとなる。

残りの 11 数値は、配列 D 上で 3 本の 26° (=ArcTan[1/2])斜め列上に並んでいるので、更にこの斜め列ごとにソートして図 12(a)の配列 E を求める。この時、付録 A3. 【性質 3】に示したように配列 D 上の大小関係はやはり配列 E 上でも保存されている。



(a) 配列E



(b) 配列Eの残りメディアン候補3要素

図 12. 配列 E 要素の大小関係に基づく要素絞込み

図(a)の配列E上の要素の大小関係を詳細に分析すると、左上側 4 個の数値は自分以下の要素を 6 個以上持ち、右下側 4 個の数値も自分以上の要素を 6 個以上持つのでメディアンになりえない。結局、7×7 入力配列 A のメディアンは、配列 E における残り 3 画素{e20,e33,e46}のメディアンに一致することがわかる。

前述の変形の中で、配列 D の d46、d20、配列 E の e46、e20 は自分を含む斜め列のそれぞれ最小値と最大値に一致するので計算は更に簡易化できる。以上の結果から、新たな小型化7×7メディアンフィルタの計算手順は次のとおりである。

(1) 入力配列 A を各列でソートして配列 B とする。

{b0i,b1i,b2i,b3i,b4i,b5i,b6i}
=SORT[{a0i,a1i,a2i,a3i,a4i,a5i,a6i}]
但し、 i=0,1,2,3,4,5,6 (3.7)

(2) 配列 B を各行でソートして配列 C とする。

{cj0,cj1,cj2,cj3,cj4,cj5,cj6}
=SORT[{bj0,bj1,bj2,bj3,bj4,bj5,bj6}]
但し、 j=0,1,2,3,4,5,6 (3.8)

(3) 配列 C を斜め 45° 方向にソートして配列 D とする。このうち必要な 2 要素 d46,d20 はそれぞれ MIN,MAX 計算に置き換えて簡易化する。

d46=MIN[{c02,c13,c24,c35,c46}] (3.9)
{d01,d12,d23,d34,d45,d56}
=SORT[{c01,c12,c23,c34,c45,c56}] (3.10)
{d00,d11,d22,d33,d44,d55,d66}
=SORT[{c00,c11,c22,c33,c44,c55,c66}] (3.11)
{d10,d21,d32,d43,d54,d65}
=SORT[{c10,c21,c32,c43,c54,c65}] (3.12)
d20=MAX[{c20,c31,c42,c53,c64}] (3.13)

(4) 配列 D を斜め 26° 方向にソートしたときの必要 3 要素を計算する。

e46=MIN[{d10,d22,d34,d46}] (3.14)
e33=MEDIAN[{d21,d33,d45}] (3.15)
e20=MAX[{d20,d32,d44,d56}] (3.16)

(5) 3 要素のメディアンを計算し出力する。

出力=MEDIAN[{e46,e33,e20}] (3.17)

図 13 に 7×7 メディアンフィルタの回路構成例を示す。図では、5×5 の場合と同様に、配列 C を生成した後に、5 入力最小値回路(MIN5)、6 入力ソート回路(SORT6)2 個、7 入力ソート回路(SORT7)、5 入力最大値回路(MAX5)をそれぞれ配置して配列 D の 11 個の必要要素を計算し、更にその出力を入力とする 4 入力最小値回路(MIN4)、3 入力ソート回路(SORT3)、4 入力最大値回路(MAX4)を配置し、その出力のメディアン計算を行うことで、求める7×7メディアンフィルタの出力を得るようにしている。

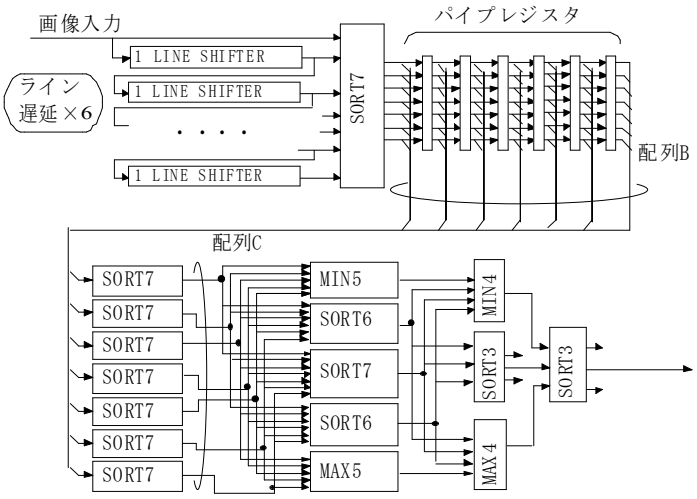


図 13. 7×7 メディアンフィルタの回路構成

4. 有効画素のみからのメディアン計算

入力画像中に画素値の計算できない無効画素が含まれている画像の場合には、ウィンドウ内の有効画素のみからそのメディアンを計算することも必要である。この機能は 5×5 や 7×7 などの大きなウィンドウサイズのメディアンフィルタに対しては特に重要となる。

有効画素のみからのメディアンの計算は、原理的には予め無効画素の数値を最小値（通常"0"）にしてからウィンドウ内の全要素をソートし、その時の有効画素数を m 個とすれば、ソーティング後の数値の大きい方から Floor[(m+1)/2]番目の数値を求めればよい。

この計算は、ウィンドウ中の無効画素の半分を最大値要素に変換してからそのメディアン値を求めることでも実現できる。前述の小型化フィルタ回路にはこの方式の方が適用容易である。

本稿で提案する図 9.の 5×5 メディアンフィルタ回路にこの機能を付加するためには、配列 B を計算する図 9.の上半分を図 14 の様に変更すればよい。

図 14 では、予め最小値に設定された無効画素の半分を二つの回路 SFT0、SFT1 で分担して最大値要素へと変換している。すなわち、並行入力される一つの列内の有効画素数 k を回路Σによって計算し、SFT0 回路において、Floor[k/2]の数値が 0 なら 2 個、1 なら 1 個、2 以上なら 0 個、列ソート後の数値の大きい方に最大値を挿入する。

更に列ごとの変換残に相当する k の最下位 bit はパイプラインレジスタで連続する 5 列分(5bit)を監視し、"最下位 bit=1"の列のうちの半数の列に一つずつ最大値を挿入する。この論理がやや複雑なので、回路図では 5bit 入力 ROM を参照し、その結果で SFT1 回路を駆動し、予め定められた適切な列データに最大値を挿入するようにしている。

これにより、配列 B の無効画素数の半数が最大値に変換されるので、図 9.の小型化回路を用いても有効画素のみからのメディアン計算が可能になる。この方法は7×7メディアンフィルタにおいても同様である。

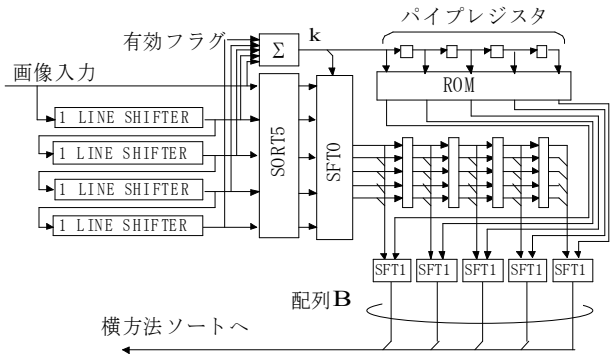


図 14. 有効画素のみメディアン計算への変更

5. FPGA における回路規模

本稿で提案した小型化メディアンフィルタ回路を実際にハードウェア記述言語 VerlogHDL で記述し、ALTERA 社の FPGA である Cyclone IV をターゲットとして QuartusII Version13.1 でコンパイルした。そのときの回路規模算定結果を表 5.1、表 5.2 に示す。ただし、この中にラインシフタを含む列画像切り出し回路部は含めていない。

表において、「従来方式」は前述の"odd-even"型ソート回路を使用した方式、「全画素」は通常のメディアンフィルタ、「有効画素」は有効画素のみからのメディアンフィルタをそれぞれ意味する。

表 5.1.　5×5 メディアンフィルタ回路規模比較

回路方式	従来方式		提案方式	
	全画素	有効画素	全画素	有効画素
LE 数	4,057	4,836	1,467	1,730
メモリ数	0 bit	84 bit	40 bit	194 bit

表 5.2.　7×7 メディアンフィルタ回路規模比較

回路方式	従来方式		提案方式	
	全画素	有効画素	全画素	有効画素
LE 数	15,457	18,213	3,973	4,766
メモリ数	0 bit	225 bit	560 bit	584 bit

この結果、LE (Logic Elements) 数のみの比較では、5×5 の場合で凡そ 1/2.8、7×7 の場合で凡そ 1/3.8 へと回路規模が削減されていることがわかる。削減は、要素数の大きなソーティング処理が少ない要素数のソート処理へと分割実行されることによるものであり、その効果はウィンドウサイズが大きいほど大きい。

なお、全ソーティングによる従来方式はメディアン値以外にも不必要に多くの要素データを出力しているが、QuartusII は不要出力に関わる回路要素を自動的に削除するので、回路規模の結果が大きく変わることはない。

6. おわりに

5×5、7×7 の比較的大きなウィンドウサイズを持つメディアンフィルタについて、コンパクトでリアルタイム処理可能な計算方式を新たに提案した。従来のウィドウ内要素の全ソーティングを行う方式に比べて、小さな要素数のソーティングを組み合わせて実行するため、1 クロック 1 画素出力のリアルタイム性を保持しながらもコンパクトな回路規模に収まった。

FPGA 上の回路規模は 5×5、7×7 の場合で従来比それぞれ 1/2.8 倍、1/3.8 倍の規模となった。メディアンフィルタはもともと回路規模の大きなフィルタなので、実用上その効果は大きい。

また、入力画像中に無効画素を含む特殊な画像であっても、有効画素のみからのメディアンフィルタ処理が実現できることを示した。これにより、本稿における提案手法は、例えば前述の立体視の視差画像の様に有効な視差値が計算できない場合であっても、適用できるようになる。

なお、提案以外のウィンドウサイズへの拡張も可能と思えるが、現時点では未検討である。また、リアルタイム性の実現を目的としたためFPGAによる並列回路化を中心としたが、提案アルゴリズムは比較演算を大幅に削減する効果があるため、ソフト処理であってもその高速化に有効と思われる。

付　録

A1.【性質 1】

図A1.1.に示すように、図(a)のN×N数値配列Aに対し、縦列ごとにソートして図(b)の配列Bを得たとする。この結果、配列 B の同一列上にある上側の数値は下側の数値より大となる。図(b)において矢印は数値間の大小関係を示す不等号"≦"を表す。次に配列 B の数値を横行ごとにソートし、図(c)の配列Cを得たとする。この結果、配列Cの同一行上にある右側の数値は左側の数値より大となる。

この時、行ごとのソート処理によって左右方向の数値移動が発生するにもかかわらず、図(c)に示すように配列Bにおける上下要素間の大小関係は配列 C においても継承される。

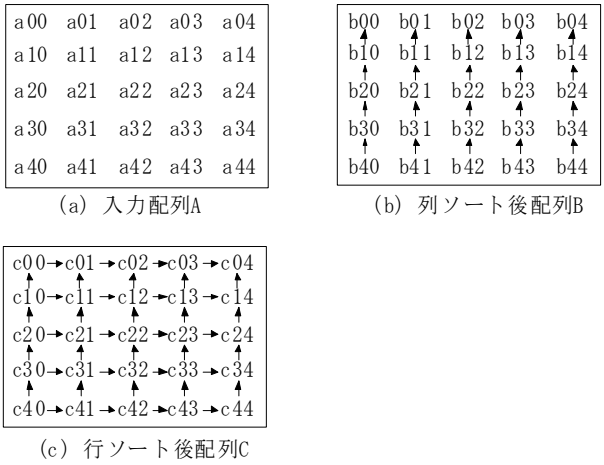


図 A1.1. 列行ごとソーティング後の要素間大小関係

【性質 1 の証明】

簡単のため図 A1.2. に示すように、行ごとソート前後の第 0 行と第 1 行に注目する。第 1 行の最大値 c14 がソート前に例えば b13 だったとすれば、第 0 行内に少なくとも一つ c14 より大きな数値 b03 が存在する。よって第 0 行のソート後最大値 c04 は b03 以上、すなわち c14 より大きい。

次に次大値 c13 がソート前に例えば b11 だったとすれば、第 0 行内に c13 より大きな数値が少なくとも 2 個 b01,b03 が存在する。従って、第 0 行次大値 c03 は c13 より必ず大きい。

第 1 行の 3,4,5 番目の要素 c12、c11、c10 についても第 0 行内に自身より大きい要素をそれぞれ 3､4､5 個以上持つから同様であり、結果として第0、1行間の大小関係は保存される。配列 C の任意の 2 行間で同様な関係が成立するので、配列 C の上で性質 1 は成立する。

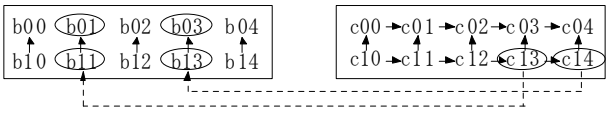


図 A1.2. 任意の 2 行に注目した行ごとソート

A2. 【性質 2】

横と縦のソーティングが終了した配列 C において、図 A2.1.(a)の破線で示す左上 45° 方向の列ごとにソート処理を行い、配列 D を得る。この結果、配列 D の同一 45° 列にある左上側の数値は右下側の数値よりも大きい。

この時、配列 C の異なる行間、列間の要素の大小関係は、図(b)に示すごとく斜めソーティングによる数値移動にも関わらず配列 D においても継承される。

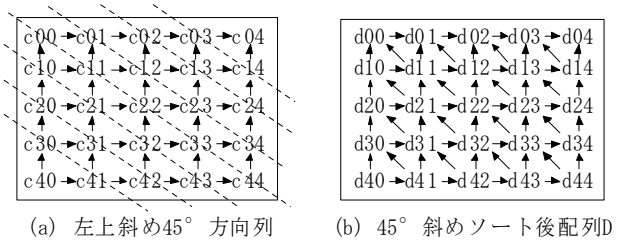


図 A2.1. 斜め 45° ソート後の要素間大小関係

【性質 2 の証明】

配列 C 上で隣接する任意の斜列に注目し、図 A2.2.(a)に示すように斜列 0、1 とする。この二本の斜列が配列 C の右上三角領域に含まれるなら、斜列 1 の要素数は斜列 0 の要素数より一つ少ない。そこで斜列 1 の最上位と最下位に

cmax=MAX[{c01,c12,c23,c34}]

cmin=MIN[{c00,c11,c22,c33,c44}]

なる cmax,cmin を図のように新たに追加し、縦方向大小関係、横方向大小関係にそれぞれ注目して並べ替えると、図(b1)、(b2)なる二つの配列を得ることが出来る。

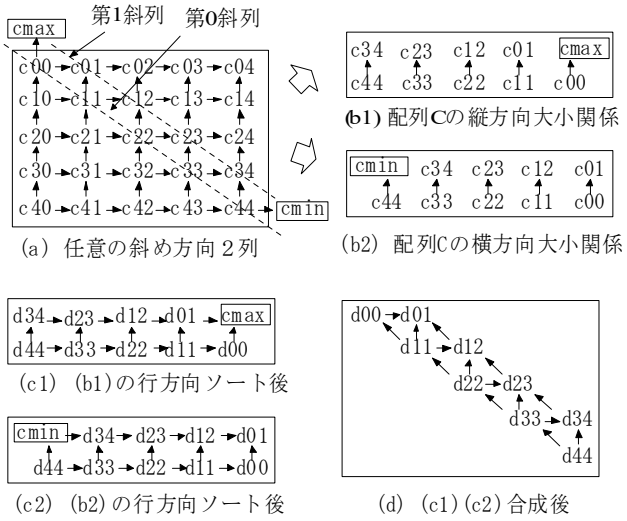


図 A2.2. 斜め方向ソーティングによる大小関係

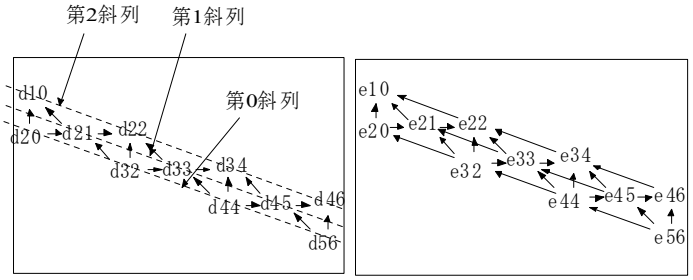
図(b1)、(b2)の配列を横行方向にそれぞれソートすると、【性質 1】により図(c1)(c2)の大小関係の配列が得られる。cmax は第 1 列の最大値、また cmin は cmin ≦ [{c01,c12,c23,c34}] でもあるから、ソート処理によって位置は変わらない。この関係を位置関係に注意して配列 D として復元すると、図(d)が得られる。すなわち、注目二斜列において図(a)の大小関係は保存される。

また、二本の斜列が配列 C の左下三角領域に含まれるなら、斜列 0 の要素数が斜列 1 の要素数より一つ少ない。この時は、斜列 0 の最上位に斜列 0 の最大値 cmax を追加し、斜列 0 の最下位に斜列 1 の最小値 cmin を追加するようにす

れば、前述の右上三角領域の場合と同等の結果が得られる。任意の隣接二斜列において図(a)に示す配列Cの要素間大小関係が継承されるので、【性質 2】は成立する。

A3.【性質 3】

縦、横、45°斜め方向ソート後の配列 D において、図 A3.1.(a)のように左上 26°斜め方向の 3 列を考える。この 3 列の各要素間の大小関係は【性質 1】【性質 2】により図の矢印の通りである。この配列に対して 26°斜め方向の列ごとに左上方向が最大になるようにソート処理し、図(b)の配列 E が得られたとする。



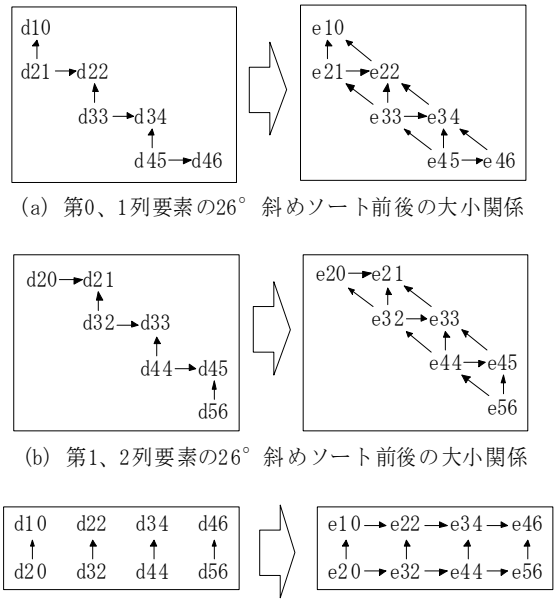
(a) 配列Dにおける26°斜め列　　(b) 26°斜めソート後の配列E

図 A3.1 26°斜めソート前後の要素間大小関係

この時、配列 E において、各要素の数値は 26°斜め方向ソーティングに関わらず配列 D で示された各要素間の大小関係を継承する。

【性質 3 の証明】

図 A3.1.(a)の 3 本の 26°斜列について 2 本ずつ選択して配列しなおすと、図 A3.2.(a)(b)(c)の左側の図がそれぞれ得られる。



(a) 第0、1列要素の26°斜めソート前後の大小関係

(b) 第1、2列要素の26°斜めソート前後の大小関係

(c) 第0、2列要素の26°斜めソート前後の大小関係

図 A3.2.　大小関係保存の説明図

この図において 26°斜めソートは、それぞれ 45°斜めソート、横行ソートに相当するから、それを実行すると【性質 2】【性質 1】により、図(a)(b)(c)のそれぞれ右側の大小関係が得られる。この関係を位置関係に注意して配列 E として復元すると、図 A3.1.(b)が得られる。この結果から、【性質 3】は成立する。

文　献

[1] T.S.Huang, G.J.Yang, and G.Y.Tang, "A Fast Two- Dimensional Median Filtering Algorithm," IEEE Trans. Acoust., Speech, Signal Processing, vol.27, no.1, pp.13–18, 1979.

[2] S. Perreault and P. Hebert, "Median Filtering in Constant Time," IEEE Transactions on Image Processing, vol.16, no.9, pp. 2389-2394, Sept.2007.

[3] G.L.Bates, S.Nooshabadi, " FPGA Implementation of a Median Filter," Proceedings of IEEE Region 10 Annual Conference on Speech and Image Technologies for Computing and Telecommunications, Vol.2, pp.437-440, Dec. 1997

[4] 浜村倫行,入江文平,"3×3 メディアンフィルタの高速アルゴリズム,"FIT 情報技術レターズ, vol.1, no.1, pp.141-142, 2002.

[5] M.A.Vega-Rodriguez,J.M.Sanchez-Perez,J.A.Gomez-Pulido, "An FPGA-Based Implementation for Median Filter Meeting the Real-Time Requirements of Automated Visual Inspection Systems," proc. 10th Mediterranean Conference on Control and Automation, Lisbon, Portugal, July. 2002.

[6] A.Sanny, V.K.Prasanna, "Energy-Efficient Median Filter on FPGA," 2013 International Conference on Reconfigurable Computing and FPGAs (ReConFig), Cancus, Mexico, Dec. 2013.

[7] S.S.Tavse, P.M.Jadhav, M.R.Ingle, "Optimized Median Filter Implementation on FPGA Including Soft Processor," International Journal of Emerging Technology and Advanced Engineering, vol.2, no.8, pp236-239, Aug. 2012.

[8] P.Walivadekar, D.S.Bormane, "Hardware and Software Implementation of Median Filter in Image Processing Application," International Journal of Engineering Sciences & Research Technology, vol.2, no.10, pp.2904-2909, Oct.2013.

[9] J.Scott, M.Pusateri and M.U.Mushtaq,"Comparison of 2D Median Filter Hardware Implementations for Real-Time Stereo Video," proc. 37th IEEE Applied Imagery Pattern Recognition Workshop, pp.1-6, Washington,USA, Oct.2008.

[10] P.M.Narendra, "A Separable Median Filter For Image Noise Smoothing," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol.3, pp.20-29, January 1981.

[11] H.Hwang, R.A.Haddad, "Adaptive Median Filters: New Algorithms and Results," IEEE Transaction on Image Processing ,vol.4, no.4, pp.499-502, April 1995.