Two-Dimensional Block of Spatial Convolution Algorithm and Simulation

Mussa Mohamed Ahmed mussa_m7@yahoo.com
Department of Electronics and Communication
Faculty of Engineering, University of Aden-Yemen.

Abstract

This paper proposes an algorithm based on sub image-partitioning strategy. The proposed scheme divides a grayscale (or color) image into overlapped 6×6 blocks each of which is partitioned into four small 3x3 non-overlapped sub-images. A new spatial approach for efficiently computing 2-dimensional linear convolution or cross-correlation between suitable flipped and fixed filter coefficients (sub image for cross-correlation) and corresponding input sub image is presented. Computation of convolution is iterated vertically and horizontally for each of the four input sub-images. The convolution outputs of these four sub-images are processed to be converted from 6×6 arrays to 4×4 arrays so that the core of the original image is reproduced. The present algorithm proposes a simplified processing technique based on a particular arrangement of the input samples, spatial filtering and small sub-images. This results in reducing the computational complexity as compared with other well-known FFT-based techniques. This algorithm lends itself for partitioned small sub-images, local image spatial filtering and noise reduction. The effectiveness of the algorithm is demonstrated through some simulation examples.

Keywords: Spatial Convolution, Algorithm, Partitioning, Flipping, Simulation.

1. INTRODUCTION

Convolution is a very useful tool to determine the response of a given system. For example, in a wavelet bank filter, bands are computed using convolution [1]. In image processing, convolution operation is useful for detecting the edges [2]. So far, several methods have been developed to compute 2-D system responses such as the circulant or Toeplitz matrices [3], sliding window method as a spatial filtering [4], which requires moving the center of a mask through an image and does not require folding a mask as in case of tabular method. However, the computation of responses using the above methods is a redundant one, so it has a global response computation, which causes beside noise amplification, spatially undesirable noise distribution on output image also. Thus, they are only suitable for computation of global signals and not applied for specific area of interest (inside the image). In other hand, convolution algorithms in principle are complex due to the number of operations. Block spatial convolutions help in implementing local convolution or short convolutions without using FFT, which only estimate the direct convolution.

In some literature [3]-[5] it is noticed that the authors consider the filter size as a measure of the number of operations per output pixel (sample), for example an impulse response of M×N coefficients requires an M×N multiplications per sample. In contrary to this the authors in [6] conclude that the number of operations involved in a direct convolution is less for small arrays size and may take the form of a trapezoidal function. We have adopted this conclusion as a basis in our present work and have tested another function, namely a falling ramp. The advantage of the proposed algorithm in this paper resides in that it can be used also for measuring the degree of similarity between sub-images, which is not found in the previously mentioned literature. The algorithm integrates the convolution and correlation in one processing system. Partitioning(segmentation) of image to sub-images are applicable in techniques for hiding information and water marking where the spatial domain technique is the most common one [7], [8]. Available literature reveals a lack of details of partitioning and sequence of blocks. For this reason, the present work gains an importance. Exploiting image processing locality in cache pre-fetching [9] is one of the benefits of partitioning.

The authors in [10] propose a scheme of partitioning based on an anticlockwise division of a rectangular block image of size (64x512) for the iris ring into eight sub-images of size (64x64). A vector consisting of an ordered sequence of sub-image features are then extracted from the local information contained in the eight sub-images.

In [11] a partition fusion technique for multi-focus images is developed for improving the image quality. It compares conventional partition fusion technique for image sub- blocks taking even sizes only, which might result in asymmetry and a modified partition fusion method where sub-blocks are selected of the fused image based on their clarity measures. The clarity measure of an image sub block was determined by second order derivative of the sub image.

Algorithms [10],[11] are based on gray image. A gray image can provide enough information to identify different individuals, but the common in nature that colors are different.
Moreover, block sizes used are large and need large processing filters, which result in increased distortion at borders of images in addition to processing complexity [6].

It is not necessary to use length of filters radix-2 as in [5],[11], some time we need symmetric filter with odd length. Length of the filters directly affects computation time of analysis and re synthesis; shorter filters are favored in more cases. Linear phase response can be achieved by using symmetric filters. In algorithm [5] step number seven computes the inverse 2-D FFT, which needs to keep quarter the output array only, this means loss of time used to compute other three quarters.

Our scheme is expected to be clear in comparing it with a conventional convolution such as the sliding-window or double-circulant matrix [3],[12],[13]. It suggests a new block convolution where blocks are convolved independently with local filters for more reliable and fast processing. In our algorithm we, in advance compute desired parts (small sub-images), what we need only and without redundant operations. The rest of the paper is organized as follows: In Section 2, we provide the problem statement of this work, section 3 proposed design our algorithm, which consist of two steps partitioning and short convolution. The partitioning approach is described in subsection 3.1, short convolution is presented in sub section 3.2, while mathematical processing section is performed in section 3.3, experiment and performance evaluation of our approach are presented also. Section 4 deals with the evaluation results of simulation while section 5 represents discussion and conclusion. Appendix includes two-Matlab functions.

2. PROBLEM STATEMENTS

To realize short convolutions we need overlapping between sub-images to prevent discontinuity and to make real time processing. Extracting features such as point and edge detection and exploiting corner point in image alignment (registration) etc, needs local partitioning the entire image to sub-images. In other hand, short convolution helps to implement spatial and direct convolution with less operations, this gives exact solution rather than the FFT estimated solutions.

Now the problem is how to design the three distinct steps of the algorithm: step 1 is the partitioning of the input image to overlapped sub-images followed by a second partitioning of the previous one into smaller sub-images. Step 2 is the block convolution after a suitable flipping. The double partitioning is also useful for the extraction of local features or any other desired operations. Finally, the increase of pixels resulting after the convolution of the overlapped sub-images is compensated by converting the 6x6 sub-images at the output of the four processors into 4x4 sub-images. The algorithm is tested experimentally using the Matlab software package to evaluate its performance.

3. PROPOSED DESIGN ALGORITHM

This paper explores a new method for computing the 2-D convolution that is based on the algorithm represented by Figs. 1 and 2 using two main steps: partitioning (partitioning) and filtering as well as similarity measuring. Discrete input image is applied to the splitter input. Splitter shows itself as primary filter or discretizer. It determines the sub image size of a specified periodic deterministic function. The four sampled output periodic blocks with specific size are applied to four convolution processors.

3.1 Partitioning

Assume that an X×Y grayscale (or color image layers) image (indices x and y as I(x, y)) (Fig. 1a) has to be partitioned into overlapped sub-images. Each has the dimension 6x6 as shown in Fig. 1b, X1×Y1= 6×6 and is a function of the indices x1 and y1; S1(x1, y1). As mentioned before, overlapping here reduces latency and prevents discontinuity resulting from partitioning, thus giving high efficiency. Each sub-image has to be partitioned to four small non-overlapped sub-images with indices x2 and y2; S2(x2, y2) the size of each of these sub-images must be equal to that of the spatial filter sizes; M×M=X2×Y2=3×3.

The first partitioning with overlap and non-overlap band (delta=Δ) is implemented using relations below:

$$\boldsymbol{\Delta = X1 - L = Y1 - L} \tag{1}$$

$$\boldsymbol{x = x1 + (i - 1)\Delta, \qquad i = 1,2,\ldots(X - L)/\Delta)} \tag{2}$$

$$\boldsymbol{y = y1 + (j - 1)\Delta, \qquad j = 1,2,\ldots(Y - L)/\Delta)} \tag{3}$$

Where the indices i and j stand for the horizontal and vertical overlapped blocks, respectively. For example, Fig. 1a shows a partitioning with L = 2, Δ = 4.

The second partitioning is repeated in the same manner as before in order to give non-overlapped small sub-images (indices x2 and y2) with parameters:

$$\boldsymbol{x1 = x2 + (i1 - 1)\,M, \qquad i1 = 1,2,\ldots..X1/M} \tag{4}$$

$$\boldsymbol{y1 = x2 + (j1 - 1)\,M, \qquad j1 = 1,2,\ldots..Y1/M} \tag{5}$$

Where the indices i1 and j1 stand for the horizontal and vertical non-overlapped blocks, respectively, and (X-L)/(X1-L) = m and (Y-L) / (Y1-L) = n; m×n are the maximum number of overlapped sub-images, but X1/M=m1, Y1/M = n1; m1×n1 are the maximum number of non-overlapped sub-images. It should be to note that in Eqs. 4 and 5 the non-overlapping enlarges the band Δ to number M.

The partitioning and convolution processes can be performed in the one of the following manners:

1. Each time, a 6×6 overlapping block is subdivided into 4 small non-overlapping blocks each of 3×3 elements and the resulting four sub-blocks are then subjected to the convolution with the corresponding processors as shown in Figs. 1b, c and 2. This method is presented in this paper. The Matlab code for this method is given in Fig. 3.

2. The entire image of size X×Y is first divided into overlapped 6x6 blocks. Then the resulting image is subdivided into non-overlapped blocks of the size 3×3. Then each 4 blocks of the size 3×3 are subjected to convolution. The advantage of this method resides in its simplicity however it requires more memory space.

In many cases, partitioning helps to make processing implemented independently "in parallel "[5], [11]. In this paper, as we will see later on, the two points that need to be highlighted are: first, the four-processor banks in Fig. 2 are operating in parallel, and second, number of operations is minimized by the factor of two. Taking into consideration these points, and in case of still image (single still picture), a fast image processing is expected. The issue of real time processing and video signal processing ' movie' requires further measures to ensure acceleration.

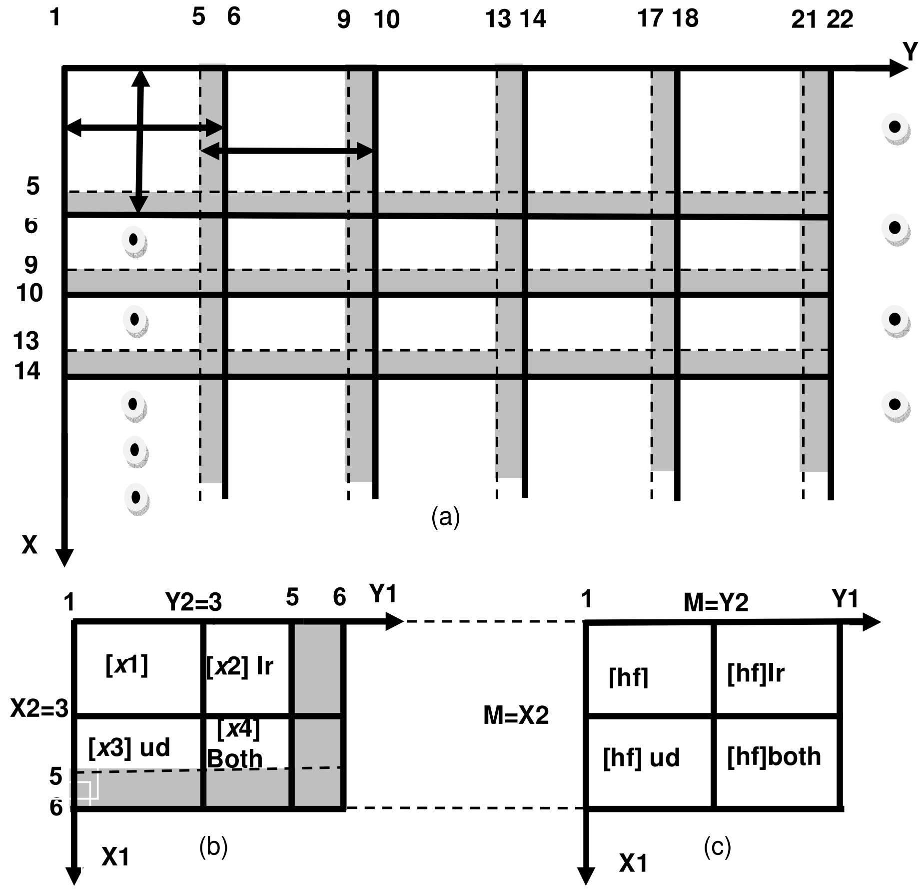

FIGURE 1: Layout of the Partitioning grid for the input Image Filtering pass.

(a) Formed input Image I(x, y) (b) First sub-Image with size X1×Y1 (Eq. 8)

(c) Impulse Responses of the System [hf] Partitioned to four M×M=3×3 Coefficients

3.2 Short Convolution Method

3.2.1 Description of Algorithm

The short convolution algorithm (length 3) as in Fig. 2 is used in this paper to develop an efficient implementation method for 2-D block processing. In our case each block filter inside the processor takes a set of input samples, (e.g. [x1]) and produces one pixel after convolution, (e.g. P_{11}). The size of the input set should coincide with the filter coefficients. It should be noted that an input samples set must decrease by one column from its preceding one and the same is applied to the filter coefficients. Referring to Fig. 2, the set [x1] is a 3×3-array, the set [x1c] is a 3×2-array (one column less than [x1]), and the set [x2c] is a 3×1-array (one column less than [x1c]. Similarly, the filter coefficients are denoted by [h3c], [h2c], and [h1c]. The convolution results in an array that is a function of the size; e.g., P11 [3×3], P12 [3×2], and P13 [3×1] (rows remain constant while columns change from 3 to1). These samples are returned as the first output row. The second output row is then obtained after omitting the first row from the original set [x1], and the third row from the original filter coefficient set [hf]. The new sets [x1r] and [h3r] are subjected to the convolution by repeating the previous processing method. The second row will be P21 [2×3], P22 [2×2], and P23 [2×1]. Finally, the third row is obtained after omitting the second row from the preceding input samples set and coefficients set. Detailed description of the signal designation is illustrated in Table1 and Matlab function [y1] = newconv2DO(x1, hf). From the analysis of the processing, we observe that a change between output rows is obtained by decreasing the applied signal indices downwards and the system coefficients upwards. However, to get samples of the output row, the input signal decreases by a column from left to right and the system coefficients from right to left. In addition, it is seen that the number of operations per output pixel is varying with the output row. At the end of the process, the overall processor outputs are four sets each of 3x3 samples. Intermediate level samples, are concatenated to form convolved intermediate block of 6x6 samples. Finally the result is processed using Eq. 10 to obtain a 4x4 matrix sub-images (see Appendix for the Matlab function [yo] =calloutO (y1, y2, y3, y4)).

Signal	Description
[I]	Input Image (size(X×Y), overlapped gray level sub-images (S_{x1y1} size X1×Y1=6×6 Figs. 1 and 2)
[x4], [x3], [x2], [x1] = S_{x2y2}	Four 3×3 arrays resulted from X1xY1 sub image (see Eq. 6).
[hf]	Flipped filter impulse response
[x2c], [x1c], …	Matrices after omitting the second column, first column, …, respectively, from the preceding matrix, e.g. [x1].
[x2r], [x1r], …	Matrices after omitting the second row, first row, …, respectively, from the preceding matrix, e.g. [x1]
[h3c], [h2c], [h1c]	Matrices after omitting the third column, second column, or first column, respectively, from hf.

TABLE 1: Signals and Description

3.2.2 Mathematical Processing

The processor is ideally suited for real-time image processing applications, such as edge enhancement and edge detection.

For the convolution-transform for each pixel shown in Fig. 2, the coefficients $\boldsymbol{P_{r,c}}$ is obtained by computing the two-dimensional dot product of [hf] and each 3×3 sub image.
Convolution results as obtained from the first processor (y1) are as follows:

$$\boldsymbol{P_{r,c}} = \sum_{x2}^{M} \sum_{y2}^{M} \boldsymbol{hf_{x2,y2} S_{x2,y2}} \tag{6}$$

$$\boldsymbol{y1} = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \tag{7}$$

In Eq. 6 the size of the filter $(M \times M)$ and the input signal must change from a maximum M (maximum degree of overlapping) to a minimum one (minimum degree of overlapping). In Fig. 2 the convolution blocks indicate that the size of arrays used for convolution decreases gradually from 3×3 to 1×1. This reflects that the number of operations is variable and follows a falling ramp function with the lag index, instead of a trapezoidal form as in [11]; i.e. processors work in one mode (decreasing) instead of three modes (increasing-constant-decreasing). Eq. 7 shows that, in our case the output convolution samples are equal to the input samples (Fig. 2); this is not the case in [5] where only a quarter of the output samples, estimated by inverse FFT, are used.
The remaining input sub-images are applied to the corresponding processors after appropriate flipping as given by Eq. 8 and require exactly the same operation steps as the first one. Flipping is applied differently in order to keep the algorithm applicable equally for all processors. Therefore, all processors perform same operation.

$$\boldsymbol{S_1(x1,y1)} = \begin{bmatrix} [x1] & [x2 \times J] \\ [J \times x3] & [J \times (x4 \times J] \end{bmatrix} \tag{8}$$

Here matrix J is the exchange matrix.

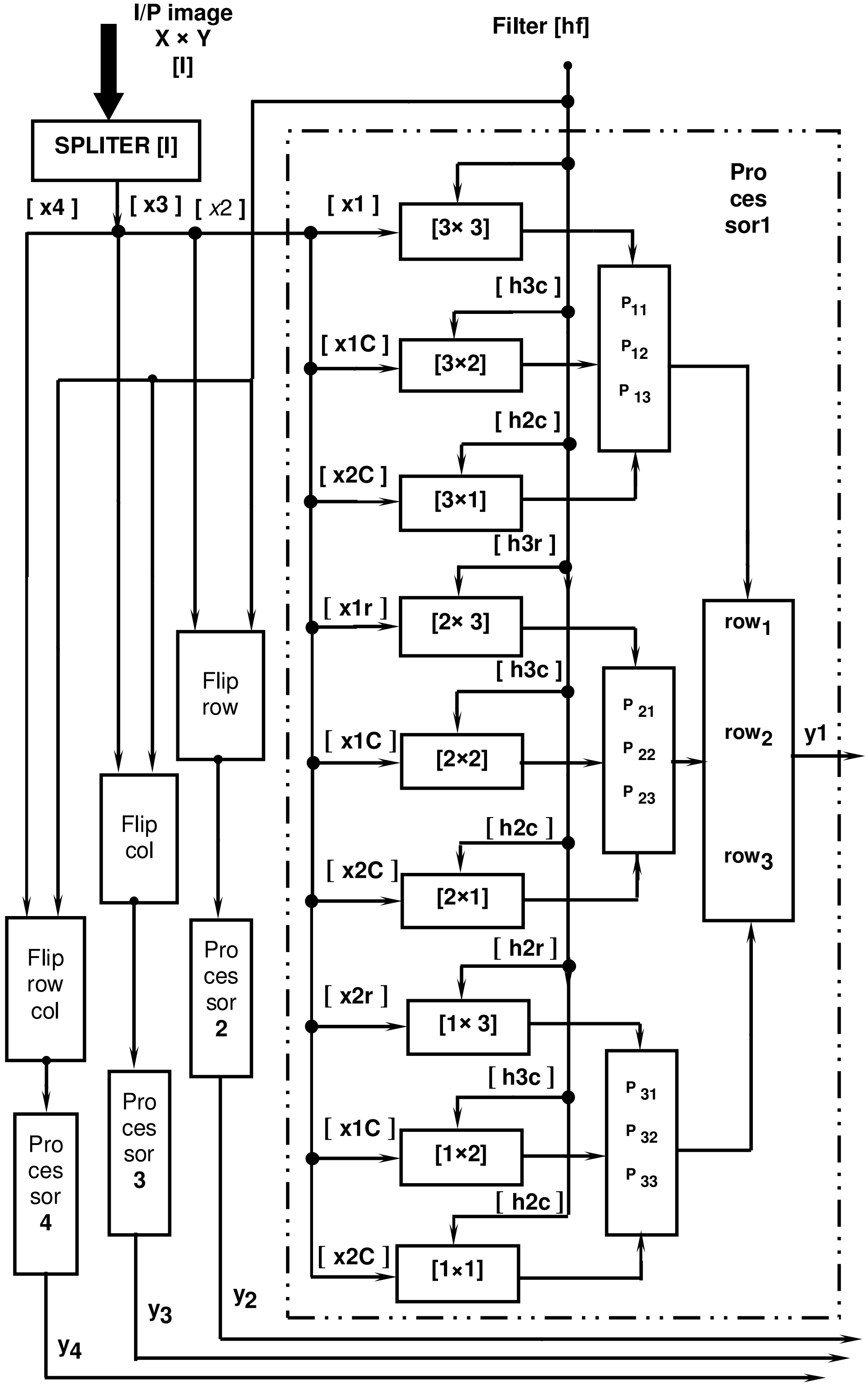

FIGURE2: Algorithm for small Convolution
Matlab function [y1] newconv2DO(x1, hf)

Similar operations as dictated by Eq. 8 are also performed with filter coefficients (Fig 1b).
An intermediate stage is introduced according to Eq. 9. The external function y_{IT} successively converts the results of the four processors (6x6 pixels) to the final processing level, which is saved as output of 4x4 blocks (Eq. 10).

$$y_{IT} = \begin{bmatrix} y_1 & y_2 \\ y_3 & y_4 \end{bmatrix} \tag{9}$$

From y_{IT} we get first final sub image results as y_{1f} size 4×4.

$$y_{1f} = \begin{bmatrix} C_1 & P_{1,2} + P_{1,4} & P_{1,3} + P_{1,5} & C_2 \\ P_{2,1} + P_{4,1} & P_{2,2} + P_{2,4} + P_{4,2} + P_{4,4} & P_{2,3} + P_{2,5} + P_{4,3} + P_{4,5} & P_{2,6} + P_{3,6} \\ P_{31} + P_{5,1} & P_{3,2} + P_{3,4} + P_{5,2} + P_{5,4} & P_{3,3} + P_{3,5} + P_{5,3} + P_{5,5} & P_{3,6} + P_{5,6} \\ C_3 & P_{6,2} + P_{6,4} & P_{6,3} + P_{6,5} & C_4 \end{bmatrix} \tag{10}$$

Where C1, C2, C3 and C4 are the local corner convolved points of the four sub-images. Eq. 10 points out that the increase in the number of pixels due to overlapping (Fig. 1) is compensated here horizontally then vertically or parallel after convolution takes place using function[yo]=calloutO(y1,y2,y3,y4) (see Appendix). This is similar to the methods, which apply operations after convolution methods such as overlap-add, or overlap-save methods [12]. It should be noted that corner points have an important particularity that they do not add to neighboring pixels of sub-images and maintain properties of their own convolved sub-images.
For sub image of size X2×Y2, to be convolved with a filter whose impulse response has a support M×M=X2×Y2=3×3 (Figs. 1b and c), the number of multiplications as indicated in processor1 of Fig. 2 is 36. If we apply the doubly circulant matrix 2-D filtering technique given by Eq. 11 [7] in Fig. 1, the number of multiplications would be constant and equal to nine per output sample. For an output matrix of 5×5 it reach 225 operations and for our case the output matrix 3x3 requires 81 operations, so the gain in simplicity more than 2.

$$\begin{bmatrix} y0 \\ y1 \\ y2 \\ y3 \\ y4 \end{bmatrix} = \begin{bmatrix} H0 & H4 & H3 \\ H1 & H0 & H4 \\ H2 & H1 & H0 \\ H3 & H2 & H1 \\ H4 & H3 & H2 \end{bmatrix} \begin{bmatrix} x0 \\ x1 \\ x2 \end{bmatrix} \tag{11}$$

In Eq. 11 H0 to H4 are basic matrices of dimension 3×3, and.x0, x1, x2 are input column vectors of length 3 stacked to give column–ordered vector length 9. Output vectors y0 to y4 are columns of length 5 and represent 2-D array size 5×5.

Noise labeled with convolution can be reduced by well-known linear or nonlinear techniques. Nonlinear methods often provide good noise cleaning technique in which each pixel is compared to the average of its eight neighbors. According to Eq. 12.

$$[C - 1/8\sum_{i=1}^{8} S_i\,] > T, \quad \text{then} \quad C = 1/8\sum_{i=1}^{8} S_i \tag{12}$$

Each core pixel C is compared to the average of its eight neighbors S_i. If the level of the difference is greater than some threshold level **T**, the pixel is judged to be noisy, and it is replaced by its neighborhood average.

Pseudo code below (Fig. 3) shows how we exploit previous partitioning for implementation block -convolution.

```
Clc; clear;
[I]=imread ('original image');
Ip=I(control size of input image);
A=double(Ip);
[X,Y]=size(A);
H=6;V=6;%size of sub-block1
p=2;%overlap elements
m=floor((X-p)/(H-p));%number of sub-blocks in horizontal
n=floor((Y-p)/(V-p));%number of sub-blocks in vertical
yot = cell(m, n);
for(i=1:m) % Global search(for all)
  for(j=1:n)
     for(h=1:H) % making sub-image 6x6 (Local sub-block1-overlapped)
         for(v=1:V)
            x=h+(i-1)*H-p*(i-1);
            y=v+(j-1)*V-p*(j-1);
            z{i,j}(h,v)=A(x,y);
            ss=z{i,j};%cell to array
          end
     end
    hf=[1 1 1;1 1 1;1 1 1];%any system
    x1=ss(1:3,1:3);A1 = newconv2DO(x1,hf);dad{1,1}=A1;
    x2= ss(1:3,4:6);A2=newconv2DO(fliplr(x2),fliplr(hf));dad{1,2}=A2;
    x3= ss(4:6,1:3);A3=newconv2DO(flipud(x3),flipud(hf));dad{2,1}=A3;
    x4=ss(4:6,4:6);A4=newconv2DO(flipud(fliplr(x4)),flipud(fliplr(hf)));dad{2,2}=A4;
    yo = calloutO(dad{1,1},dad{1,2},dad{2,1},dad{2,2});yot{i,j}=yo;
   end
end
y=conv2(Ip,h);%Matlab conv
L=cell2mat(yot);F=max(min(L));N=L/F;F2=max(min(y));
subplot(1,2,1), imshow(N);subplot(1,2,2), imshow(y/F2);
```

FIGURE3: Partitioning and Convolution Matlab code

4. EXPEREMENTAL RESULTS and PERFORMANCE EVALUATION

Grayscale image (possibly color image layers) of size 1206×1206 is used in the experiments of this paper (see Fig. 3). An original image as shown in Fig. 4a is subjected after partitioning to filtering with folded impulse response hf=[1, 2, 1; 0, 0, 0;-1, -2, -1] and without partitioning (Matlab convolution using either double circulant matrix or sliding window) with impulse response hm=[-1, -2, -1; 0, 0, 0; 1, 2, 1]. For enhancing output image point, gray level modification is undertaken i.e. dividing output image by F=max (min (output image matrix)). The results of extracting horizontal Sobel high pass edge components of original image are shown in Figs. 4b and 4c.

Fig.4c (right) results due to the conventional convolution, whereas Fig.4b results due to the proposed block convolution. Comparing the two results, it is observed, that the image in Fig. 4b shows better local features than that in Fig4.c where local edge detail (inner borders) has been lost, and a distortion as global features (outer borders) of $\Delta x \times \Delta y$ = hf -1×hf -1 has been introduced.

Our aim here is to evaluate the results of both two-convolution methods shown before by using two objective evaluation criteria: spatial quality correlation coefficient [4,14]; the peak signal-to-noise ratio (PSNR) and the corresponding Mean Square Error (MSE), computed using Matlab functions below:

a - Orignal input image

b

c

FIGURE4: Image with a Sobel Horizontal edge detector
b-Output image due to proposed method c-Output image due to classical methods

$$c = corr2\,(y1\_n,\, y) \text{ and } [psnr,\, mse] = psnr\_mse\_maxerr\,(y1\_n\,,\, y) \tag{13}$$

Here ***y*** and ***y1_n*** are variables that represent images before addition of Gaussian noise (Fig4-c) and after it, respectively. Same process is repeated for the results of the proposed algorithm (Fig4-b). For illustration, the results of evaluation are shown together in Table-2 and Fig. 5 .

Higher PSNR of image means better quality of the measured image and vice versa to MSE. The PSNR results by the proposed method are higher than that by classical methods. The difference is around 7dB, this is why images resulted from proposed method are visually distinguishable better. Regarding this evaluation, we can say that there is a matching between objective and subjective evaluations.

Spatial quality of the proposed method is higher than that of the conventional convolution techniques mentioned above this page. Under the effect of white Gaussian, SNR from 0 db to 30 db, the performance of the proposed algorithm does not change and it preserves the same properties attained from noise free source. Moreover, the proposed technique could also generate high quality images when the noisy images de-noised with thresholding (see Eq, 12).The proposed algorithm guarantees that the quality of the resulting convolved image is always satisfactory for noise free source, noisy and de-noised convolved images.
The presented partitioning algorithm can be useful in many applications such as image classification, fusion [14] and identifications [9] and others.

N	**0**	**30**
psnr- conventional method	19.61	19.63
psnr-proposed method [db]	26.16	26.3
mse- conventional method	712.1	708.5
mse-proposed method	157.2	152.4

TABLE2: Evaluation results (min and max) from PSNR and MSE

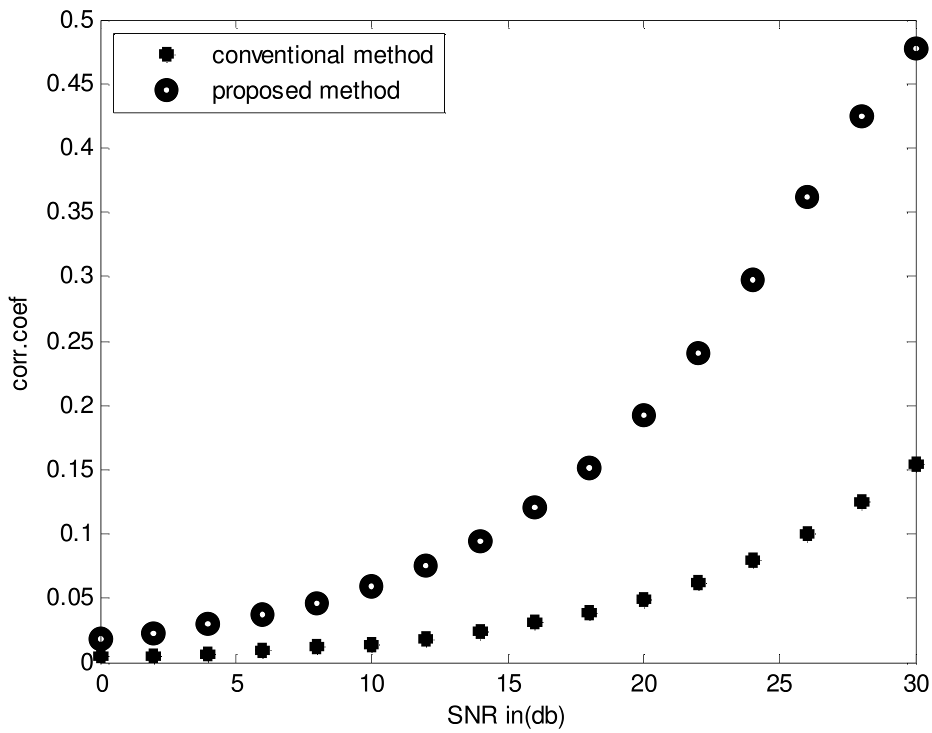

FIGURE 5 : Correlation coefficient for conventional and proposed methods

5. DISCUSSION and CONCLUSION

In this paper, an algorithm has been designed for decreasing complexity and minimizing number of operations enough to reconstruct the input image. We have presented and implemented a 2-D algorithm for the sub-image (block) digital filtering or cross correlation measuring based on double partitioning of large 2-D image into smaller sub-images. Moreover, the decomposition of computation into parallel processing as depicted in processor1 has reduced the complexity. The spatial filtering has been used to implement filtering stage. Since the convolution and correlation are two related operations, so this algorithm is common for them except that in correlation a mask needs not to be rotated by 180 degree prior correlations with the input image.

Comparison of this algorithm with Matlab convolution function or cross-correlations functions gives zero error in gray-level values with small different in sizes**.** This algorithm is designed so that the side effects of convolution at the boundaries of image are reduced. This is accomplished by neglecting a surrounding frame of size 2x2, which is equal to the filter size minus one (M-1). A positive effect of this is translated by a reduction in complexity, which raises the advantages of this algorithm and gives an example of zooming-in of an entire image as shown in Fig. 4b of Appendix . This can be interpreted by the fact that a local scale reduction of each small sub-images results in a change of distance from the sensor.
The proposed algorithm can be used in various applications as follows:

- Hiding information and specially with hiding technology based on block-partitioning strategy.
  We can embed hiding process before or after convolution stage. Possibly, with some arrangement both convolution, hiding processes and geometric transformation may be implemented in parallel. It is possible to approximate complex geometric transformation by partitioning an image.
- Image registration: corner points are specific points determined by all pixels of input sub image and have maximum number of operations, so they can be used as control points.

These above mentioned applications constitute the objects for future research work.

6. APPENDIX

Matlab-Functions: new convolution (newconv2DO) and call out results of convolution (callout O).

```
Function [y1] =newconv2DO(x1, hf)
p11=sum (sum (x1.*hf)); %1st o/p convolved sample of 1st row
xr=x1(1:3, 2:3); %omit 1st column [x1c] from [x1] (see Fig.2)
hr=hf (1:3, 1:2); %omit 3ed column [h3c] from [hf]
p12=sum (sum (xr.*hr)); %2ed o/p convolved sample
xrr=xr (1:3, 2); %again omit column2 from previous matrix [xr]
hrr=hr (1:3, 1); %again omit column2 from previous matrix [hr]
p13=sum (sum (xrr.*hrr)); %3ed o/p convolved samples
X1= [p11 p12 p13]; %call results row1
xr2=x1(2:3, 1:3); %omit1st row [x1r] from original [x1]
hr2=hf (1:2, 1:3); %omit 3ed row [h3r] from original [hf]
p21=sum (sum (xr2.*hr2)); %1st o/p convolved sample of 2nd row
xr3=xr2 (1:2, 2:3); %omit 1st column [x1c] from previous matrix [xr2]
hr3=hr2 (1:2, 1:2); %omit 3ed column [h3c] from previous matrix [hr2]
p22=sum (sum (xr3.*hr3));
xr4=xr3 (1:2, 2);
hr4=hr3 (1:2, 1);
p23=sum (sum (xr4.*hr4));
X2= [p21 p22 p23]; %call results row2
xr5=xr2 (2, 1:3); %omit 2nd row [x2r] from previous matrix [xr2]
hr5=hr2 (1, 1:3); %omit 2nd row [h2r] from previous matrix [hr2]
p31=sum (sum (xr5.*hr5));
xr6=xr5 (1, 2:3); hr6=hr5 (1, 1:2);
p32=sum (sum (xr6.*hr6));
xr7=xr6 (1, 2); hr7=hr6 (1, 1);
p33=sum (sum (xr7.*hr7));
X3= [p31 p32 p33]; %call results row3
y1=[X1; X2; X3]; %O/P of 1st processor
End
```

```
Function [yo] =calloutO (y1, y2, y3, y4)
Yf=zeros (6, 6);
Yf (1:3, 1:3) =y1;
Yf (1:3, 4:6) =y2;
yf (4:6, 1:3) =y3;
yf (4:6, 4:6) =y4;
yf(1:6,2)=yf(1:6,2)+yf(1:6,6);%hor
Yf (1:6, 3) =yf (1:6, 3) +yf (1:6, 5);
yf(2,1:4)=yf(2,1:4)+yf(6,1:4);%ver.
Yf (3, 1:4) =yf (3, 1:4) +yf (5, 1:4);
Yo=yf (1:4, 1:4);
End
```

7. REFERENCES

[1] S. E Umbaugh. *Computer Imaging digital image analysis and processing*. CRC press Book, 2005, PP.659.

[2] A. R. Weeks. *Fundamentals of Electronic Image Processing*. New Delhi: University of Central Florida and Prentice-Hall of India, 2005, pp.570.

[3] A. K. Jain. *Fundamentals of Digital image processing*. New Delhi: Prentice-Hall of India, 1997, PP.569.

[4] R. C. Gonzalez, R. E. Woods , S. L. Eddins. *Digital image processing using Matlab*. Pearson Education. Inc, 2005(2002), PP.310.

[5] K. Berberidis. An efficient partitioning-based scheme for 2-D convolution and application to image restoration. *IEEE*, PP.843-846, 2002.

[6] L. Ismail , D. Guerchi. "Performance Evaluation of convolution on the cell broad band engine processor". *IEEE Transactions on parallel and distributed systems*, vol. 22, no.2, pp.337-351. 2011.

[7] T. C. Lu, S. R. Liao, P.L. Chen, C.C. Chang, Z.H. Wang. "Information hiding technology based on block-partitioning strategy". ISECS international colloquium computing, communication, control, and management, pp.500-505. 2009.

[8] E. D. Gelasca."Full-reference objective quality metrics for video watermarking, video partitioning and 3D model watermarking". PhD thesis, PP.187, 2005.

[9] R. Cuccchiara, M. Piccardi. "Exploiting image processing locality in cache pre-fetching .In High Performance Computing". 2002.

[10] Li Ma, Y. Wang, T. Tan**.** "Iris recognition based on multichannel Gabor filtering". The 5th Asian Conference on Computer Vision, pp.1-5, 2002.

[11] **D**. Agrawal, D. Ali, J.Singhai**.** "A modified partition fusion technique of mulltifocus images for improved image quality". Conference - *Bioinformatics and image*, special issue on ICIT, vol. 4 No. 3, pp.658-663, 2009.

[12] T. Bose. *Digital signal and image processing*. John Wiley & sons.inc, pp.706, 2004.

[13] M. Sonka, V. Hlavac, R. Boyle. *Image processing analysis and machine vision*. Chapman and Hall, PP-555, 1994.

[14] M. Holia, V.K.Thakar. **"**Image registration for recovering affine transformation using Nelder Mead Simplex method for optimization." *IJIP-Journal*, vol. 3, Iss.5, pp.218-228, 2011.