

IMPULSE NOISE REMOVAL ON AN EMBEDDED, LOW MEMORY SIMD PROCESSOR

Jongmyon Kim¹, Soojung Ryu¹, Antonio Gentile², Linda M. Wills¹, and D. Scott Wills¹

¹Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0250
{jmkim, soojung, linda.wills, and
scott.wills}@ece.gatech.edu

²Dip. Ingegneria Automatica e Informatica
Università di Palermo
V.le delle Scienze, 90128 Palermo, Italy
antonio.gentile@unipa.it

Abstract: Vector median filters efficiently reduce noise while preserving image details. However, their high computational complexity for color images makes them impractical for real-time systems. We propose new computationally efficient filtering algorithms, called index mapping filters (IMFs). These filtering algorithms are accelerated by implementing them on a massively data parallel processor array. In addition to greater computational efficiency, these algorithms result in robust noise reduction of corrupted color images. Analyses of mean square error, signal-to-noise-ratio, and visual comparison metrics indicate that IMFs are competitive with the vector median filter (VMF) in their ability to correct impulse noise in color images. These algorithms are implemented on a SIMD processor array being developed for high efficiency, high-performance portable products. Executing on a 4,096 node SIMD chip operating at 50MHz, IMFs 3x3 window applied to a 256x256 color image would take 442 microseconds (22,104 clock cycles) for Index Mapping Distance Filter (IMDF) and 408 microseconds (20,415 clock cycles) for Index Mapping Median Filter (IMMF).

1. INTRODUCTION

Vector median filter (VMF) is widely used to remove impulse noise for color images because of its edge preserving capability and ability to reduce impulse noise [1]-[4]. However, its high computational complexity has limited its use in most real world applications [5]. We propose a new class of index mapping filter (IMF) algorithms to efficiently restore the grayscale and color images corrupted by impulse noise while significantly reducing computation time. Since the correlation among multi-channel signal components (e.g. RGB) is not exploited, the construction of the IMF algorithms is based on separating the color pixel data into the luminance and chromaticity channels. The YCbCr (Y: luminance and Cb, Cr: chromaticity) color space allows more flexible processing of luminance data independent of chromaticity. It is also correlated, including luminance data from all the RGB components. This can simplify the IMF algorithms by reducing median or 1-norm distance filtering computation for Cr and Cb while restoring color image efficiently.

Moreover, this paper proposes a parallel implementation of IMFs algorithms using a low memory, fine grain single instruction stream multiple data streams (SIMD) pixel processing array, called SIMPil [6]. The SIMPil array consists of a large number of small processing elements (PEs), each concurrently executing instructions issued from the controller. Taking advantage

of massive data parallelism, it achieves much higher performance in execution time of the parallel IMFs implementation.

2. THE SIMD PIXEL PROCESSING ARRAY (SIMPil)

Massive data parallel architectures have been applied to image processing for almost three decades [7]. Early SIMD machines (ILLIAC IV, Goodyear MPP) were limited by technology [8]. Later machines (e.g., MarPar [9], TMC Connection Machine CM-5 [10]) were designed for general sets of applications, including scientific workloads. They achieved generality by sacrificing efficiency and parallelism. The SIMPil architecture is a low memory, portable multimedia system [11]. Figure 1 shows the micro-architecture of a SIMPil processing element, along with its interconnection network. Each processing element has a reduced instruction set computer (RISC) datapath with specialized units for SIMD operation and mechanisms for the area I/O data streams [6].

Each processor incorporates an analog to digital converter to convert light intensities that are incident on the color filter array (CFA) [12] into digital values. SIMPil PEs concurrently execute the instructions broadcast by a controller. Each PE includes 256 words of local memory, an arithmetic unit, barrel shifter, communication unit (NEWS network), and multiply-

accumulator (MACC). Following processing of the frame, the output can be transformed to RGB output.

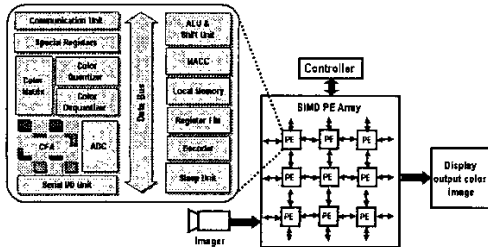


Figure 1: Block diagram of SIMPil processing element.

3. INDEX MAPPING FILTERING ALGORITHM

The construction of the index mapping filtering (IMF) algorithms is based on separating the color pixel data into the luminance and chromaticity channels. Although the RGB color format is more commonly used, the YCbCr color space enables more flexible processing of luminance data independent of chromaticity. IMF algorithms enhance true details, limit the overshoot near sharp edges and attenuate impulse noise in flat areas.

Although vector median filter (VMF) is widely used to remove impulse noise for color images because of its edge preserving capability and performance of impulse noise, its huge computational complexity has prevented its use in most real world applications [5]. Therefore, we propose more computationally efficient IMF algorithms.

3.1. Index Mapping Median Filter Algorithm

The pseudocode of the index mapping median filter (IMMF) algorithm is given in Figure 2, along with a pictorial description. The IMMF algorithm is applied to only luminance (Y) values while obtaining index values of the median Y position for chromaticity components (Cb, Cr). When the image size increases, the computation time of IMMF algorithm becomes considerably faster than that of scalar median filter. Moreover, the IMMF algorithm implemented on SIMPil provides an extraordinary performance gain (e.g. in execution time).

Given a processing window of size w , the sorting algorithm takes $w(w-1) = O(w^2)$ time-complexity at each window location.

Let the image size in pixels be M (i.e., M is the number of rows times the number of columns). The computation time complexity can be defined as

$$\begin{aligned} \text{TC(Sequential IMMF)} &= O(M \times w^2) \\ \text{TC(Data-parallel IMMF)} &= O(\text{Sequential IMMF})/\text{PE}, \end{aligned}$$

where PE is the number of processing elements.

Since the frequently used window size is relatively small (w is usually $3 \times 3 = 9$), the dominant factor will be the image size M . As M increases, the computation time of the sequential IMMF algorithm is significantly increased. However, in the case of the data-parallel IMMF algorithm, the computation time is reduced by a factor of the number of processors.

```

For all image pixels, (Y, Cb, Cr) = Transform (R, G, B);
REPEAT for all rows
  REPEAT for all columns
    //Median filtering on Y values for a window size, w
    Sort Y values inside the window
    For Cb and Cr,
      get the indexes of median Y values (med_row, med_col)
    //Replace with median using Index
    (Y,Cb,Cr)[current_row][current_col] = (Y,Cb,Cr)[med_row][med_col];
  END repeat
END repeat

```

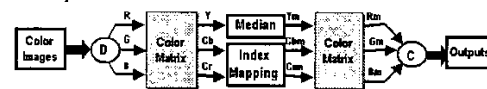


Figure 2: IMMF algorithm. D = decomposer. C=composer.

3.2. Index Mapping Distance Filter Algorithm

Given N luminance (Y) values $\{Y_1, Y_2 \dots Y_N\}$, the output of the 1-norm (Manhattan distance) median operator is defined by the equations

$$\begin{cases} Y_{M1} \in \{Y_1, Y_2 \dots Y_N\} \\ \sum_{i=1}^N \|Y_{M1} - Y_i\|_1 \leq \sum_{i=1}^N \|Y_j - Y_i\|_1, \quad j = 1 \text{ to } N \end{cases} \quad (1)$$

where Y_i is the i^{th} component value of Y.

The pseudocode of IMDF algorithm is given in Figure 3, along with a pictorial description.

Similar to the IMMF algorithm, the filtering of the IMDF algorithm is applied to only luminance (Y) values while obtaining index values of median Y position for chromaticity components (Cb, Cr). This result provides more computational time reduction than that of VMF. It performs to find the minimum distances out of the sum of the distances to all the window's Y components.

In addition, the IMDF algorithm implemented on SIMPil provides a huge computation time reduction. The computation time complexity of IMDF algorithm for image size of M and window size of w can be defined as

$$\begin{aligned} \text{TC(Sequential IMDF)} &= O(M \times w^2) \\ \text{TC(Data-parallel IMDF)} &= O(\text{Sequential IMDF})/\text{PE}, \end{aligned}$$

where PE is the number of processing elements. Indeed, the data-parallel IMDF algorithm again provides a huge computational efficiency to that of sequential one by the factor of $1/\text{PE}$.

```

For all image pixels, (Y, Cb, Cr) = Transform (R, G, B);
REPEAT for all rows
  REPEAT for all columns
    Initialize Sum_Distance;
    //Compute the sum of the distances to other Ys for a window
    size w = k x k
    REPEAT for all i=1 to k
      REPEAT for all j=1 to k
        Sum_Distance += ||Yi - Yj||
      END repeat
    END repeat
    //Get the index of minimum Sum_Distance
    Index(median_row, median_col) = Min(Sum_Distance);
    //Replace with median using Index
    (Y,Cb,Cr)(current_row)(current_col) = (Y,Cb,Cr)(med_row)(med_col);
  END repeat
END repeat

```

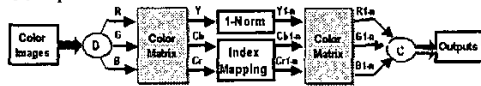


Figure 3: IMDF algorithm. D = decomposer. C = composer.

Note that the median and 1-norm filtering for a gray image is identical to that for only luminance Y channel in the color image.

4. EXPERIMENTS AND RESULTS

We have implemented the IMF algorithms, scalar median filter (SMF), and VMF in Microsoft C on a PC workstation with a Pentium II 266 MHz microprocessor, running Window 2000. In addition, parallel IMFs, SMF, and VMF have been tested on the SIMPIL Simulator running on a PC workstation with a Pentium II 266 MHz microprocessor. The evaluation is performed on a 256x256 RGB color image *TANK* (shown in Figure 6). The test image is corrupted with 10% 'salt & pepper' noise and a 3x3 filtering mask is used. The goal of efficient IMF algorithms is twofold: robust image restoration from corrupted color image and computational effectiveness.

The mean squared error (MSE) and peak signal to noise ratio (PSNR) are used as quantitative measures for evaluation purposes. The MSE and PSNR are defined as

$$MSE = \frac{\sum_{i=1}^m \sum_{j=1}^n [I_{ref}(i, j) - I_{pre}(i, j)]^2}{m \cdot n} \quad (2)$$

$$PSNR = 10 \log \left[\frac{255^2}{\frac{1}{m \cdot n} \sum_{i=1}^m \sum_{j=1}^n [I_{ref}(i, j) - I_{pre}(i, j)]^2} \right] [dB] \quad (3)$$

where $m \times n$ is the size of the image, $I_{ref}(i, j)$ is an arbitrary pixel of an original image and $I_{pre}(i, j)$ is an arbitrary pixel of a processed image.

Figure 4 and Figure 5 show the MSE values and PSNRs for the corrupted image and filtered images with SMF, IMMF, VMF, and IMDF, respectively. Analyses of the MSE and PSNR indicate that the performance of IMMF and IMDF is competitive to that of SMF and VMF.

In addition to analyses of the quantitative evaluation presented above, a qualitative evaluation is necessary because the visual assessment of the processed image is the best subjective measure to determine the efficiency of the method [2]. Figure 6-(a) shows the color *TANK* image corrupted with 10% 'salt & pepper' noise. Figure 6-(b-e) show the restored images using SMF, IMMF, VMF, and IMDF, respectively. All of four filters restore the color image efficiently from corruption by impulse noise while preserving details.

In addition to the efficient color image restoration, the execution time for each filter should be emphasized. Table 1 shows the execution times for SMF, IMMF, VMF, and IMDF with a fixed 256 x 256 color image respectively. First, we compare the execution times of non-data-parallel algorithms for IMMF, IMDF, SMF, and VMF. The results show that the execution times for IMMF, IMDF are much faster than that of SMF and VMF. IMFs, SMF, and VMF implemented on SIMPIL, execute significantly faster. In addition, IMFs require fewer functional units and achieve faster execution than SMF and VMF. Executing on a 4,096 node SIMPIL chip operating at 50MHz, IMFs, SMF, and VMF with 3x3 window applied to a 256x256 color image requires 442 microseconds (22,104 clock cycles) for IMDF, 408 microseconds (20,415 clock cycles) for IMMF, 880 microseconds (44,017 clock cycles) for SMF, and 1,313 microseconds (65,650 clock cycles).

The color version of this paper can be obtained from <http://www.ece.gatech.edu/~jmkim/icdsp/icdsp.ps>.

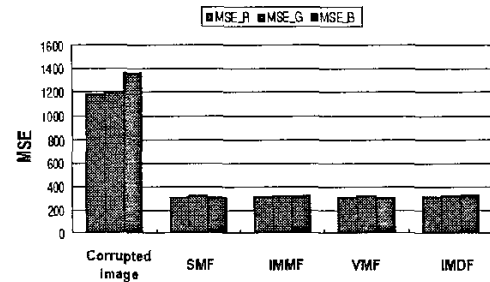


Figure 4: MSE of IMFs as compared with SMF and VMF.

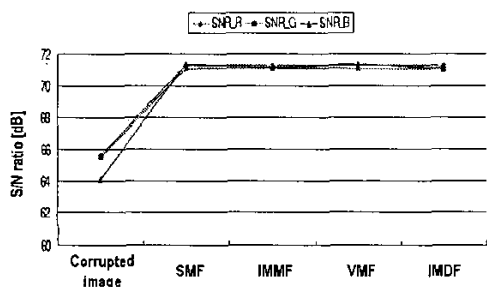


Figure 5: PSNR of IMFs as compared with SMF and VMF.

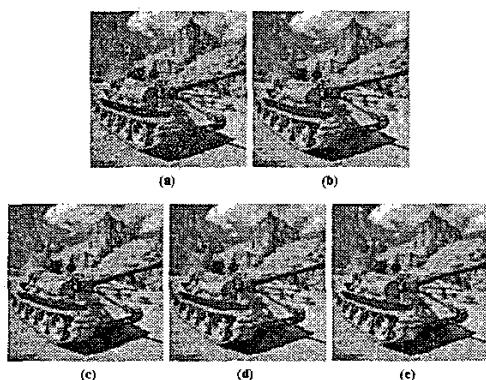


Figure 6: Restoration results of IMFs (3x3 window) as compared with SMF and the VMF. (a) Tank image distorted by 10% 'salt and pepper' noise. Restored images: (b) Using SMF. (c) Using IMMF. (d) Using VMF. (e) Using IMDF.

Table 1
Execution times (sec) comparison of selected IMFs implementations as compared with SMF and VMF.

Operator (3x3 window)	Platform	Clock Rate	Execution Time (sec)
SMF	Pentium II	266 MHz	0.861
IMMF	Pentium II	266 MHz	0.661
VMF	Pentium II	266 MHz	5.758
IMDF	Pentium II	266 MHz	2.4
SMF	4096 node SIMPii	50 MHz	0.000880
IMMF	4096 node SIMPii	50 MHz	0.000408
VMF	4096 node SIMPii	50 MHz	0.001313
IMDF	4096 node SIMPii	50 MHz	0.000442

5. CONCLUSION

A new class of impulse noise removal algorithms, Index Mapping Filters, for color image on a low memory, embedded SIMD pixel processor has been presented. Application performance and experimental simulation results have demonstrated that the proposed filtering algorithms enable robust filtering and computational effectiveness. Moreover, IMF algorithms implemented on SIMPii are considerably faster than that of the classical filters such as SMF and VMF. Future research will address processor datapath modifications to improve computational efficiency and area efficiency on data parallel processors.

REFERENCES

- [1] J. Astola, P. Haavisto, Y. Neuvo, "Vector Median Filters." *Proceedings of the IEEE*, vol. 78, no. 4, pp. 678-689, 1990.
- [2] P. E. Trahanias, A. N. Venetsanopoulos, "Vector Directional Filters: A new class of multichannel image processing filters," *IEEE Trans. on Image Processing*, vol. 2, no. 4, pp. 528-534, 1993.
- [3] E. Abreu, et al, "A New Efficient Approach for the Removal of Impulse Noise from Highly Corrupted Images," *IEEE Trans. on Image Processing*, vol. 5, pp. 1012-1025, 1996.
- [4] P. S. Windyga, "Fast Impulsive Noise Removal," *IEEE Trans. on Image Processing*, vol. 10, no. 1, pp. 173-178, 2001.
- [5] M. Barni, V. Cappellini, A. Mecocci, "Fast Vector Median Filter Based on Euclidean Norm Approximation," *IEEE Signal Processing Letters*, vol. 1, no. 6, pp. 92-94, 1994.
- [6] A. Gentile, H. Cat, F. Kossentini, F. Sorbello, D. Scott Wills, "Real-Time Vector Quantization-Based Image Compression On the SIMPii Low Memory SIMD Architecture," *IEEE Intl. Performance, Computing, and Communications Conference (IPCCC-97)*, pp. 10-16, 1997.
- [7] K. Diefendorff, R. Dubey, "How Multimedia Workloads Will Change Processor Design," *IEEE Computer*, vol. 30, no. 9, pp. 43-45, 1997.
- [8] T. Nesson, "Randomized, Oblivious, Minima Routing Algorithms for Multicomputers." *TR-24-95*, 1995.
- [9] MarPar (MP-2) System Data Sheet, MarPar Corporation, 1993.
- [10] L. W. Tucker and G.G. Robertson, "Architecture and applications of the connection machine," *IEEE Computer*, vol. 21, no. 8, pp. 26-38, 1988.
- [11] R.B. Lee, M.D. Smith, "Media Processing: A New Design Target," *IEEE Micro*, vol. 16, no. 4, pp. 6-9, 1996.
- [12] B. E. Bayer, "Color Imaging Array," *U.S. Patent 3,971,065*, 1976.