Reducing the Area on a Chip Using a Bank of Evolved Filters

Zdenek Vasicek and Lukas Sekanina

Faculty of Information Technology, Brno University of Technology
Božetěchova 2, 612 66 Brno, Czech Republic
vasicek@fit.vutbr.cz, sekanina@fit.vutbr.cz

Abstract. An evolutionary algorithm is utilized to find a set of image filters which can be employed in a bank of image filters. This filter bank exhibits at least comparable visual quality of filtering in comparison with a sophisticated adaptive median filter when applied to remove the salt-and-pepper noise of high intensity (up to 70% corrupted pixels). The main advantage of this approach is that it requires four times less resources on a chip when compared to the adaptive median filter. The solution also exhibits a very good behavior for the impulse bursts noise which is typical for satellite images.

1 Introduction

As low-cost digital cameras have entered to almost any place, the need for high-quality, high-performance and low-cost image filters is of growing interest. In this paper, a new approach is proposed to the impulse noise filters design. The aim is to introduce a class of simple image filters that utilize small filtering windows and whose performance is at least comparable to existing well-tuned algorithms devoted to common processors. Furthermore, an area-efficient hardware implementation is required because these filters have to be implemented on the off-the-shelf hardware, such as field programmable gate arrays (FPGA).

In most cases, impulse noise is caused by malfunctioning pixels in camera sensors, faulty memory locations in hardware, or errors in the data transmission (especially in satellite images [1]). We distinguish two common types of impulse noise: the *salt-and-pepper noise* (commonly referred to as intensity spikes or speckle) and the *random-valued shot noise*. For images corrupted by salt-and-pepper noise, the noisy pixels can take only the maximum or minimum values (i.e. 0 or 255 for 8-bit grayscale images). In case of the random-valued shot noise, the noisy pixels have an arbitrary value. We will deal with the salt-and-pepper noise in this paper.

Traditionally, the salt-and-pepper noise is removed by median filters. When the noise intensity is less than approx. 10% a simple median utilizing 3×3 or 5×5-pixel window is sufficient. Evolutionary algorithms (EA) have been applied to the image filter design problems in recent years [2,3,4]. EA is utilized either to find some coefficients of a pre-designed filtering algorithm or to devise a complete structure of a target image filter. As the first approach only allows tuning existing designs, the use of the second approach has led to introducing completely new filtering schemes, unknown so far [4]. The images filtered by evolved filters are not as smudged as the images filtered by median filters. Moreover, evolved filters occupy only approx. 70% of the area needed to implement the median filter on a chip.

When the intensity of noise is increasing (10-90% pixels are corrupted), simple median filters are not sufficient and more advanced techniques have to be utilized. Various approaches were proposed (see a survey of the methods, e.g. in [5]). Among others, adaptive medians provide good results [6]. However, they utilize large filtering windows and additional values (such as the maximum and minimum value of the filtering window) have to be calculated. This makes them expensive in terms of hardware resources. Others algorithms are difficult to accelerate in hardware for real-time processing of images coming from cameras.

Unfortunately, the evolutionary design approach stated above which works up to 10% noise intensity does not work for higher noise intensities. The method proposed in this paper combines simple evolved filters with human-designed components to create a bank of 3×3 filters which provides a sufficient filtering quality for high noise intensities (up to 70%), and simultaneously a very low implementation cost in hardware.

2 Conventional Image Filters

Various approaches have been proposed to remove salt-and-pepper noise from grayscale images [7,8,9,5]. As linear filters have inclination to smoothing, most of proposed approaches are based on a nonlinear approach. The median filter is the most popular nonlinear filter for removing the impulse noise because of its good denoising power, computational efficiency and a reasonably expensive implementation in hardware [10]. The median filter utilizes the fact that original and corrupted pixels are significantly different and hence the corrupted pixels can easily be identified as non-medians. However, when the noise level (the number of corrupted pixels) increases, some pixels remain corrupted and unfiltered [11]. Median filters which utilize larger filtering windows are capable of removing noise of high intensity but filtered images do not exhibit a sufficient visual quality.

The adaptive median filters produce significantly better resulting images than convential medians [6]. The filter operates with a kernel of $S_{max} \times S_{max}$ pixels. The kernel is divided into subkernels of size $3 \times 3, 5 \times 5, \ldots, S_{max} \times S_{max}$ inputs. For each subkernel, the minimum, maximum and median value is calculated. In order to obtain the filtered pixel, the calculated values are processed by the algorithm described in [6].

With the aim to visually compare images filtered by the convential median filter and adaptive median filter, Figure 1 provides some examples for the 40% salt-and-pepper noise (PSNR states for the peak signal-to-noise ratio). We can observe that there are many unfiltered shots in the image obtained by the median filter. We used the 3×3 kernel in order to easily compare the results described is next sections. Note that the use of larger kernels implies that many details

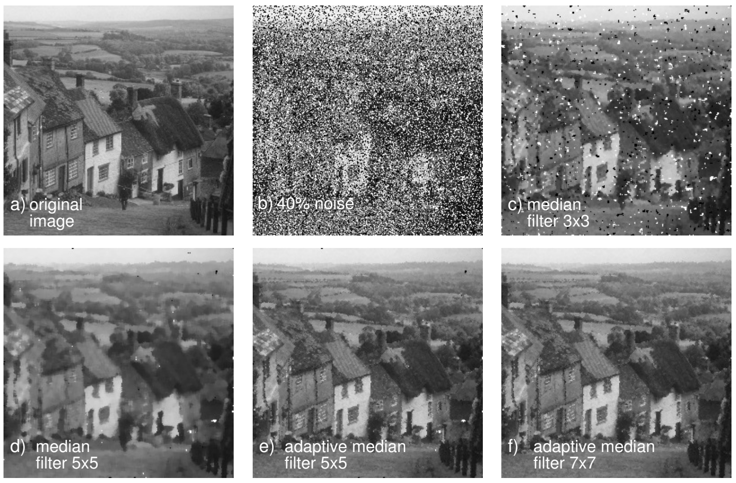


Fig. 1. Images obtained by using conventional filters. a) Original image b) Noise image corrupted by 40% Salt-and-Pepper noise, PSNR: 9.364 dB c) Filtered by median filter with the kernel size 3×3, PSNR: 18.293 dB d) Filtered by median filter with the kernel size 5×5, PSNR: 24.102 dB e) Filtered by adaptive median with the kernel size up to 5×5, PSNR: 26.906 dB f) Filtered by adaptive median with the kernel size up to 7×7, PSNR: 27.315 dB.

are lost in the image. On the other hand, the image obtained by the adaptive median filter is sharp and preserves details. However, the adaptive median (with 5x5 filtering window) costs approx. eight times more area on a chip in comparison to a conventional 3x3 median filter. Even better visual results can be achieved by using more specialized algorithms [9], but their hardware implementation leads to area-expensive and slow solutions.

3 Evolutionary Design of Image Filters

3.1 The Approach

This section describes the evolutionary method which can be utilized to create innovative 3×3 image filters [4]. These filters will be utilized in the proposed bank of filters. Every image filter is considered as a function (a digital circuit in the case of hardware implementation) of nine 8-bit inputs and a single 8-bit output, which processes grayscale (8-bits/pixel) images. As Fig. 2 shows, every pixel value of the filtered image is calculated using a corresponding pixel and its eight neighbors in the processed image. In order to evolve an image filter which removes a given type of noise, we need an original (training) image to measure the fitness values of candidate filters. The goal of EA is to minimize the difference between the original image and the filtered image. The generality of the evolved filters (i.e., whether the filters operate sufficiently also for other images of the same type of noise) is tested by means of a test set.

3.2 EA for Filter Evolution

The method is based on Cartesian Genetic Programming (CGP) [12]. A candidate filter is represented using a graph which contains n_c (columns) $\times$ n_r (rows) nodes placed in a grid. The role of EA is to find the interconnection of the programmable nodes and the functions performed by the nodes. Each node represents a two-input function which receives two 8-bit values and produces an 8-bit output. Table 1 shows the functions we consider as useful for this task [4]. We can observe that these functions are also suitable for hardware implementation (i.e. there are not such functions as multiplication or division). A node input may be connected either to an output of another node, which is placed anywhere in the preceding columns or to a primary input. Filters are encoded as arrays of integers of the size $3 \times n_r \times n_r + 1$. For each node, three integers are utilized which encode the connection of node inputs and function. The last integer encodes the primary output of a candidate filter.

Table 1. List of functions implemented in each programmable node

code	function	description	code	function	description
0	255	constant	8	$x \gg 1$	right shift by 1
1	x	identity	9	$x \gg 2$	right shift by 2
2	$255 - x$	inversion	A	$swap(x, y)$	swap nibbles
3	$x \vee y$	bitwise OR	B	$x + y$	+ (addition)
4	$\bar{x} \vee y$	bitwise $\bar{x}$ OR y	C	$x +^S y$	+ with saturation
5	$x \wedge y$	bitwise AND	D	$(x + y) \gg 1$	average
6	$\overline{x \wedge y}$	bitwise NAND	E	$max(x, y)$	maximum
7	$x \oplus y$	bitwise XOR	F	$min(x, y)$	minimum

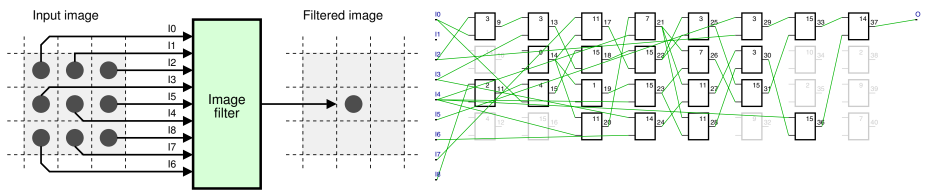


Fig. 2. The concept of image filtering using a 3×3 filter (left). An example of evolved filter (right).

EA uses a single genetic operator – mutation – which modifies 5% of the chromosome (this value was determined experimentally). No crossover operator is utilized in this type of EA because no suitable crossover operator has been proposed so far [13]. Mutation modifies either a node or an output connection. The EA operates with the population of λ individuals (typically, $\lambda = 8$). The initial population is randomly generated. Every new population consists of a parent (the fittest individual from the previous population) and its mutants. In case that two or more individuals have received the same fitness score in the previous generation, the individual which did not serve as the parent in the previous population will be selected as a new parent. This strategy was proven to be very useful [12]. The evolution is typically stopped (1) when the current best fitness value has not improved in the recent generations, or (2) after a predefined number of generations.

3.3 Fitness Function

The design objective is to minimize the difference between the filtered image and the original image. Usually, *mean difference per pixel* (*mdpp*) is minimized. Let u denote a corrupted image and let v denote a filtered image. The original (uncorrupted) version of u will be denoted as w. The image size is $K \times K$ (K=128) pixels but only the area of 126×126 pixels is considered because the pixel values at the borders are ignored and thus remain unfiltered. The fitness value of a candidate filter is obtained as

$$fitness = 255.(K-2)^2 - \sum_{i=1}^{K-2} \sum_{j=1}^{K-2} |v(i,j) - w(i,j)|.$$

3.4 Design Examples

This approach was utilized to evolve efficient image filters for Gaussian noise and 5 % salt-and-pepper noise and to create novel implementations of edge detectors [4]. Examples of filtered images for the 40 % salt-and-pepper noise are given in Fig. 3. When compared with the common median filter (see Fig. 1 and PSNR), evolved filters preserve more details and generate sharper images. Note that these filtered images represent the best outputs that can be obtained by a single 3×3-input filter evolved using described method for the 40% noise.

Figure 2 shows an example of evolved filter. We can observe that EA can create only a combinational behavior and the filter utilizes only 3×3 pixels at the input. These filters are not able to compete to adaptive median filters which sophistically operate with larger kernels. A way to improve evolved filters could be to increase the kernel size; however, this will lead to smoothing and loosing details in images.

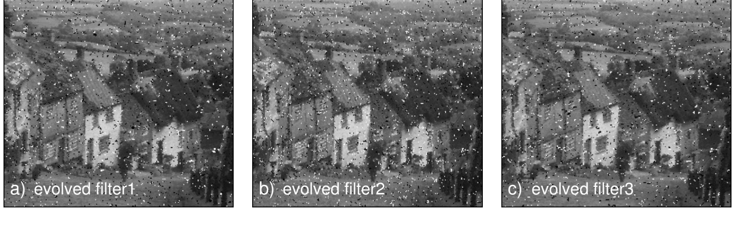
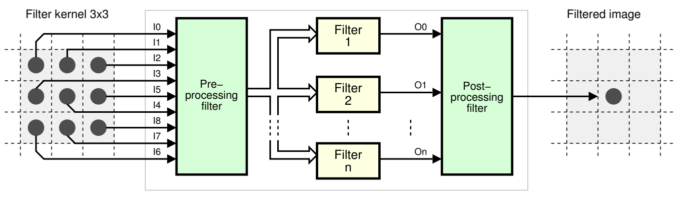


Fig. 3. A corrupted image (see Fig. 1) filtered by evolved filters a) evf1 (PSNR: 18.868 dB), b) evf2 (PSNR: 18.266 dB) and c) evf3 (PSNR: 18.584 dB)

4 Proposed Approach

In order to create a salt-and-pepper noise filter which generates filtered images of the same quality as an adaptive median filter and which is suitable for hardware implementation, we propose to combine several simple image filters utilizing the 3×3 window that are designed by an evolutionary algorithm according to previous Section 3. As Figure 4(a) shows the procedure has three steps: (1) the reduction of a dynamic range of noise, (2) processing using a bank of n filters and (3) deterministic selection of the best result.

We analyzed various filters evolved according to description in Section 3 and recognized that they have problems with the large dynamic range of corrupted pixels (0/255). A straightforward solution of this problem is to create a component which inverts all pixels with value 255, i.e. all shots are transformed to have a uniform value.



(a)

(b)

Fig. 4. a) Proposed architecture for salt-and-pepper noise removal and b) training image

Preprocessed image then enters a bank of n filters that operate in parallel. Since we repeated the evolutionary design of salt-and-noise filters (according to Section 3) many times, we have gathered various implementations of this type of filter. We selected n *different* evolved filters which exhibit the best filtering quality and utilized them in the bank. Note that all these filters were designed by EA using the same type of noise and training image and with the same aim: to remove 40% salt-and-pepper noise.

Finally, the outputs coming from banks $1 \ldots n$ were combined by n-input median filter which can easily be implemented using comparators [14]. As the proposed system naturally forms a pipeline, the overall design can operate at the same frequency as a simple median filter when implemented in hardware.

5 Experimental Results

5.1 Quality of Filtering

The filters utilized in the bank were evolved using the method described in Section 3. These filters use the size of kernel 3×3 pixels and contain up to 8×4 programmable nodes with functions according to Table 1.

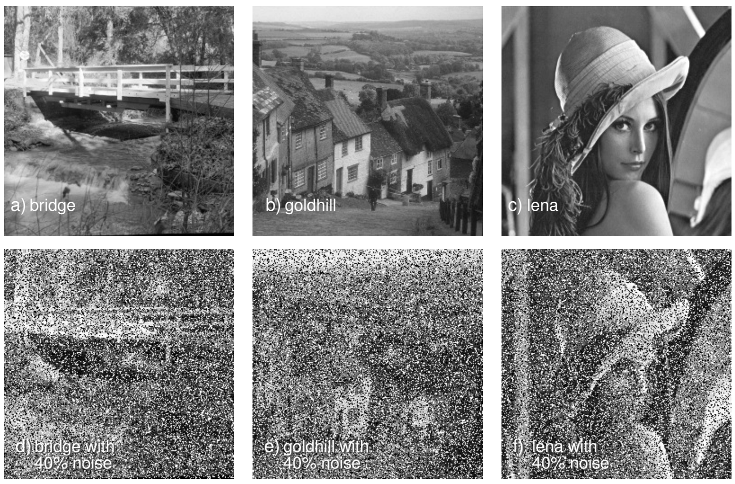


Fig. 5. Examples of test images

Table 2. PSNR for adaptive median filter with the kernel size up to 5×5 and 7×7

kernel size	$\mathbf{5 \times 5}$					$\mathbf{7 \times 7}$				
image/noise	10%	20%	40%	50%	70%	10%	20%	40%	50%	70%
goldhill	31.155	30.085	26.906	24.290	15.859	31.155	30.085	27.315	25.961	20.884
bridge	29.474	28.064	24.993	22.567	14.781	29.474	28.058	25.177	23.710	19.060
lena	33.665	31.210	27.171	24.433	15.468	33.655	31.207	27.529	25.984	20.455
pentagon	32.767	31.460	28.235	25.217	16.315	32.767	31.460	28.621	27.175	21.654
camera	30.367	28.560	25.145	22.675	14.973	30.367	28.560	25.298	23.852	19.242

A training 128×128-pixel image was partially corrupted by 40% salt-and-pepper noise (see Fig. 4(b)). EA operates with an eight-member population. The 5% mutation is utilized. A single run is terminated after 196,608 generations. Results will be demonstrated for 5 test images of size 256×256 pixels which contain the salt-and-pepper noise with the intensity of 10%, 20%, 40%, 50% and 70% corrupted pixels. Figure 5 shows some examples of test images. Table 2 summarizes results obtained for the adaptive median filter which serves as a reference implementation. All results are expressed in terms of

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\frac{1}{MN} \sum_{i,j} (v(i,j) - w(i,j))^2}$$

where $N \times M$ is the size of image.

Table 3 summarizes results for the images filtered using the bank of size 3 and 5. The output pixel is calculated by a 3-input (5-input, respectively) median circuit. Surprisingly, only three filters utilized in the bank are needed to obtain a bank filter which produces images of at least comparable visual quality to

Table 3. PSNR for the bank filter

filter	**3-bank**					**5-bank**				
image/noise	10%	20%	40%	50%	70%	10%	20%	40%	50%	70%
goldhill	33.759	30.619	27.716	25.867	19.091	34.392	31.131	27.966	25.965	19.079
bridge	31.458	28.992	25.83	24.282	18.333	32.321	29.714	26.124	24.441	18.327
lena	30.304	28.162	25.684	24.137	18.324	30.393	28.424	25.881	24.203	18.314
pentagon	34.631	31.89	28.681	26.577	18.437	35.201	32.411	28.945	26.683	18.435
camera	30.576	28.185	25.284	23.72	17.85	31.091	28.74	25.576	23.919	17.845

Table 4. Result of synthesis for different filters

filter	**optimal median filter**			**adaptive median**		**evolved filters utilized in bank**					**proposed filter bank**	
	3×3	5×5	7×7	5×5	7×7	fb1	fb2	fb3	fb4	fb5	3-bank	5-bank
no. of slices	268	1506	4426	2024	6567	156	199	137	183	148	500	843
area [%]	1.1	6.4	18.7	8.6	27.8	0.7	0.8	0.6	0.8	0.6	2.1	3.6
f_{max} [MHz]	305	305	305	303	298	316	318	308	321	320	308	305

the adaptive median filter. This fact is demonstrated by Figure 6 where the visual quality of the images filtered by the adaptive median and 3-bank filter is practically undistinguishable.

5.2 Implementation Cost

In order to compare the implementation cost of median filters, adaptive median filters, evolved filters and the bank of filters, all these filters were implemented in FPGA [15]. Results of synthesis are given for relatively large Virtex II Pro XC2vp50-7 FPGA which contains 23616 slices (configurable elements of the FPGA). This FPGA is available on our experimental Combo6x board. Table 4 shows that proposed bank filters require considerably smaller area on the chip in comparison to adaptive median filters whose implementation is based on area-demanding sorting networks.

In order to implement the proposed 3-bank filter in a small and cheap embedded system, a smaller FPGA, XC3S50, is sufficient (it contains 768 slices). However,a larger and more expensive XC3S400 FPGA (containing 3584 slices) has to be utilized to implement the adaptive median filter.

5.3 Other Properties of Evolved Bank Filters

Figure 7 shows another interesting feature we observed for the bank of evolved filters. This kind of filters is relatively good in removing the *impulse bursts noise*; much better than the adaptive median filters. The impulse bursts usually corrupt images during the data transmission phase when the impulse noise occurs. The main reason for the occurrence of bursts is the interference of frequency

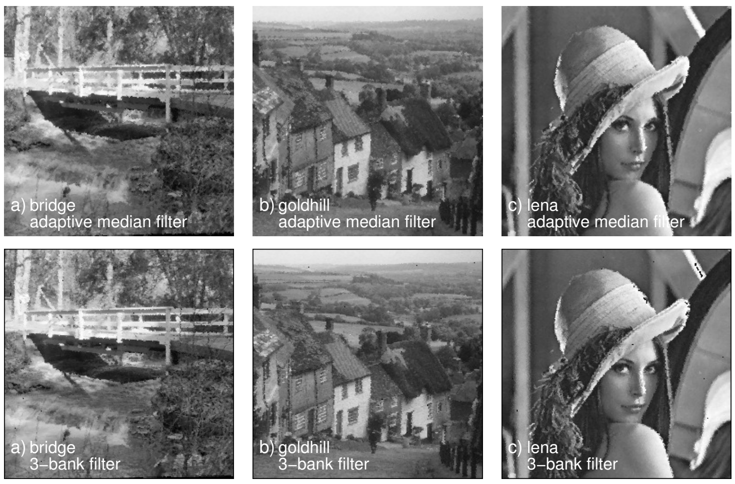


Fig. 6. Comparison of resulting images filtered using the adaptive median filter with kernel size up to 7×7 (a, b, c) and 3-bank filter (d, e, f)

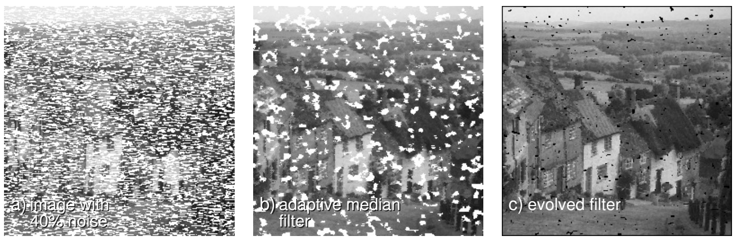


Fig. 7. a) Image corrupted by 40% impulse noise (bursts), images filtered using b) adaptive median with kernel size up to 5×5 (PSNR: 11.495 dB) and c) 3-bank filter (PSNR: 22.618 dB)

modulated carrying signal with the signals from other sources of emission. Reliable elimination of this type of noise by means of standard robust filters can be achieved only by using sliding windows that are large enough. However, e.g., the 5x5 median filter leads to significant smearing of useful information [1]. Note that images shown in Figure 7 were obtained by the bank filter which was not trained for the impulse bursts noise at all. This solution represents a promising area of our future research.

6 Discussion

The proposed approach was evaluated on a single class of images. Future work will be devoted to testing the proposed filtering scheme on other types of images. Anyway, results obtained for this class of images are quite promising from the

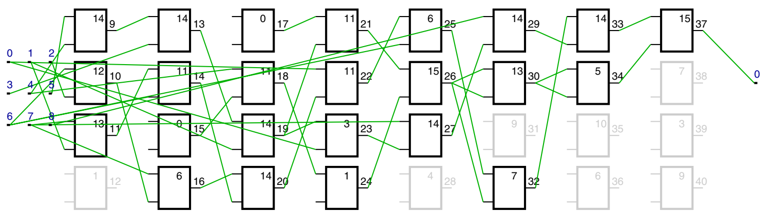


Fig. 8. Example of an evolved filter utilized in the 3-bank filter

application point of view. We can reach the quality of adaptive median filtering using a 3-bank filter; however, *four times less* resources are utilized. This can potentially lead to a significant reduction of power consumption of a target system. Moreover, Table 4 does not consider the implementation cost of supporting circuits (i.e. the FIFOs) needed to correctly read the filtering windows from memory. This cost can be significant since adaptive median filters require larger filtering windows than the bank filter.

Currently we do not exactly know why three (five, respectively) filters evolved with the aim of removing 40%-salt-and-pepper noise are able to suppress the salt-and-pepper noise with the intensity up to 70%. Moreover, none of these filters does work sufficiently in the task which it was trained for (the 40% noise). We can speculate that although these filters perform the same task, they operate in a *different* way. While a median filter gives as its output one of the pixels of the filtering window, evolved filters sometime produce new pixel values. By processing these n-values in the n-input median, the shot can be suppressed. We tested several variants of evolved filters in the bank but never observed a significant degradation in the image quality.

The existence of several filters in the bank offers an opportunity to permanently evolve one of them while the remaining ones could still be sufficient to achieve a correct filtering. A possible incorrect behavior of the candidate filter will not probably influence the system significantly. Therefore, this approach could lead to on-line adaptive filtering, especially in the case when EA can modify different filters of the bank. Note that a solution which uses only a single filter cannot be utilized in the on-line adaptive system in which the image processing must not be interrupted.

7 Conclusions

In this paper a new class of image filters was introduced. The proposed bank filter consists of a set of evolved filters equipped with a simple preprocessing and post-processing unit. Our solution provides the same filtering capability as a standard adaptive median filter; however, using much fewer resources on a chip. The solution also exhibits a very good behavior for the impulse bursts noise which is typical for satellite images. In particular, evolutionary design of image filters for this type of noise will be investigated in our future research.

Acknowledgements

This research was partially supported by the Grant Agency of the Czech Republic under No. 102/07/0850 *Design and hardware implementation of a patent-invention machine* and the Research Plan No. MSM 0021630528 – *Security-Oriented Research in Information Technology.*

References

1. Koivisto, P., Astola, J., Lukin, V., Melnik, V., Tsymbal, O.: Removing Impulse Bursts from Images by Training-Based Filtering. EURASIP Journal on Applied Signal Processing 2003(3), 223–237 (2003)
2. Dumoulin, J., Foster, J., Frenzel, J., McGrew, S.: Special Purpose Image Convolution with Evolvable Hardware. In: Oates, M.J., Lanzi, P.L., Li, Y., Cagnoni, S., Corne, D.W., Fogarty, T.C., Poli, R., Smith, G.D. (eds.) EvoIASP 2000, EvoWorkshops 2000, EvoFlight 2000, EvoSCONDI 2000, EvoSTIM 2000, EvoTEL 2000, and EvoROB/EvoRobot 2000. LNCS, vol. 1803, pp. 1–11. Springer, Heidelberg (2000)
3. Porter, P.: Evolution on FPGAs for Feature Extraction. PhD thesis, Queensland University of Technology, Brisbane, Australia (2001)
4. Sekanina, L.: Evolvable components: From Theory to Hardware Implementations. Natural Computing. Springer, Heidelberg (2004)
5. Schulte, S., Nachtegael, M., Witte, V.D., Van der Weken, D., Kerre, E.E.: Fuzzy impulse noise reduction methods for color images. In: Computational Intelligence, Theory and Applications International Conference 9th Fuzzy Days in Dortmund, pp. 711–720. Springer, Heidelberg (2006)
6. Hwang, H., Haddad, R.A.: New algorithms for adaptive median filters. In: Tzou, K.-H., Koga, T. (eds.) Proc. SPIE, Visual Communications and Image Processing '91: Image Processing, vol. 1606, pp. 400–407 (1991)
7. Yung, N.H., Lai, A.H.: Novel filter algorithm for removing impulse noise in digital images. In: Proc. SPIE, Visual Communications and Image Processing '95, vol. 2501, pp. 210–220 (1995)
8. Bar, L., Kiryati, N., Sochen, N.: Image deblurring in the presence of salt-and-pepper noise. In: Scale Space, pp. 107–118 (2005)
9. Nikolova, M.: A variational approach to remove outliers and impulse noise. J. Math. Imaging Vis. 20(1-2), 99–120 (2004)
10. Ahmad, M.O., Sundararajan, D.: A fast algorithm for two-dimensional median filtering. IEEE Transactions on Circuits and Systems 34, 1364–1374 (1987)
11. Dougherty, E.R., Astola, J.T.: Nonlinear Filters for Image Processing. SPIE/IEEE Series on Imaging Science & Engineering (1999)
12. Miller, J., Job, D., Vassilev, V.: Principles in the Evolutionary Design of Digital Circuits – Part I. Genetic Programming and Evolvable Machines 1(1), 8–35 (2000)
13. Slany, K., Sekanina, L.: Fitness landscape analysis and image filter evolution using functional-level cgp. In: Proc. of European Conf. on Genetic Programming. LNCS, vol. 4445, pp. 311–320. Springer, Heidelberg (2007)
14. Knuth, D.E.: The Art of Computer Programming: Sorting and Searching, 2nd edn. Addison-Wesley, Reading (1998)
15. Vasicek, Z., Sekanina, L.: An area-efficient alternative to adaptive median filtering in fpgas. In: Proc. of the 17th Conf. on Field Programmable Logic and Applications, pp. 1–6. IEEE Computer Society Press, Los Alamitos (to appear, 2007)