UDC 621.397
DOI: 10.15587/2706-5448.2021.225257
Article type «Reports on Research Projects»

Oleg Vasylchenkov,
Igor Liberg,
Mykhailo Mozhaiev,
Dmytro Salnikov

CONVEYORIZED IMPLEMENTATION OF ASWM IMAGE FILTER ON PLD

The object of research is the adaptive switching weighted median image filter (ASWM) algorithm. This algorithm is one of the most effective in the field of impulse noise suppression. The computational complexity and algorithmic features of this adaptive nonlinear filter make it impossible to implement a filter that works in real time on modern PLD microcircuits.

The most problematic areas of the algorithm are the weight coefficient estimation cycle, which has no limit on the number of iterations and contains a large number of division operations. This does not allow implementing the filter on PLDs with a sufficiently effective method.

In the course of the research, the programming model of the filter in Python was used. The performance of the algorithm was assessed using the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) metrics.

Modeling made it possible to find out empirically the number of iterations of the cycle for estimating the weight coefficients at different levels of noise density and to estimate the effect of artificial limitation of the maximum number of iterations on the filter performance. Regardless of the intensity of the noise impact, the algorithm performs less than 40 iterations of the evaluation cycle. Let's also simulate the operation of the algorithm with different variants of the division module implementation. The paper considers the main of them and offers the most optimal in terms of the ratio of accuracy/hardware costs for implementation. Thus, a modified algorithm was proposed that does not have these disadvantages.

Thanks to modifications of the algorithm, it is possible to implement a pipelined ASWM image filter on modern PLDs. The filter is synthesized for the main families of Intel PLDs. The implementation, which is not inferior in terms of SSIM and PSNR metrics to the original algorithm, requires less than 65,000 FPGA logical cells and allows filtering of monochrome images with FullHD resolution at 48 frames/s at a clock frequency of 100 MHz.

Keywords: *adaptive filter, nonlinear filter, median filter, impulse noise, Peak Signal-to-Noise Ratio, Structural Similarity Index Measure.*

Received date: 08.10.2020
Accepted date: 02.12.2020
Published date: 30.12.2020

1. Introduction

The process of obtaining graphic data from various sensors and transferring it to image processing subsystems is always accompanied by data corruption or even loss of data.

Electromagnetic noise or transmission errors often result in random pixels in the output image. This type of noise is called impulse noise. It can be seen in gray scale images as random white and black pixels or groups of pixels that distort basic image information.

The median filter is currently one of the most effective at filtering impulse noise. The filter is applied to all pixels without exception. During median filtering, some information is lost, which leads to smoothing of edges and blurring [1].

To eliminate this drawback, a weighted median filter WM [2] was proposed, which allows to preserve more of the original image details due to the weights for each pixel inside the filtering window.

The ideal filter should not alter pixels that are not damaged by noise. In [3], the popular types of filters that have been proposed to combat the loss of image details due to anti-aliasing are considered.

In particular, the following are proposed:
– noise detectors for the classification and detection of noisy pixels [4, 5];
– hybrid filters [6, 7];
– adaptive «switching» filters [8–10], which did not apply median filtering for pixels without noise.

The most effective is the adaptive switching median filter (ASWM) [11]. Nevertheless, its algorithm is difficult to implement on PLDs and modern processors, which complicates its use in real-time tasks and confirms the relevance of this study. Thus, *the object of research* is the adaptive switching weighted median image filter (ASWM) algorithm.

The aim of research to find methods for implementing ASWM that allow implementing a pipelined filter on most of the programmable logic integrated circuits available on the market.

2. Methods of research

The ASWM image filter uses a noise detector based on the apparatus of ordinal statistics.

The work algorithm is shown in Fig. 1.

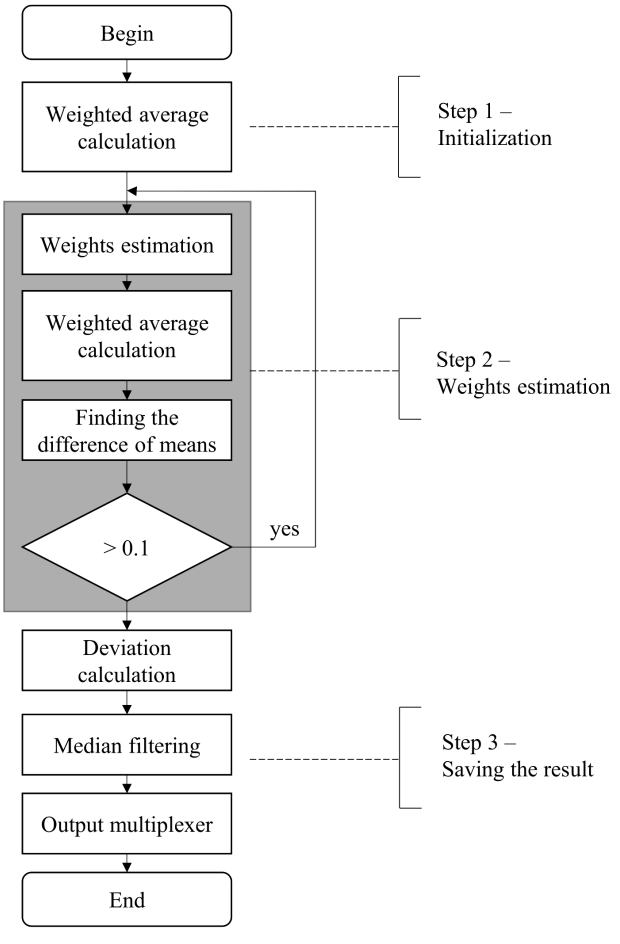

Fig. 1. Algorithm of ASWM filter operation

Step 1 – Initialization. Weights $w_{k,l}$ are initialized to 1.0.

In each image window, a weighted average value is calculated for a pixel with coordinates (i, j) in accordance with the equation:

$$M_w(i,j)=\frac{\sum_{k,l} w_{k,l} X_{i+k,j+l}}{\sum_{k,l} w_{k,l}}, \quad (1)$$

where $w_{k,l}$ – weighting factors, in accordance with equation (2); $X_{k,l}$ – elements of the filter window.

Step 2 – Weights estimation.

$$w_{k,l}=\frac{1}{\left|X_{i+k,j+l}-M_w(i,j)\right|+\delta}, \quad (2)$$

where $X_{k,l}$ – elements of the filter window; M_w – weighted average; δ – preset small value to avoid division by zero.

Then a new value $M_w(i,j)$ is calculated with the new calculated weights.

Step 3 – Saving the result. If $\left|M_w(i,j)^t - M_w(i,j)^{t-1}\right| < \varepsilon$, where ε – specified threshold, the loop ends, otherwise go to step 1. Then the standard deviation is calculated according to the equation:

$$\delta_w(i,j)=\sqrt{\frac{\sum_{k,l} w_{k,l}(X_{i+k,j+l}-M_w(i,j))^2}{\sum_{k,l} w_{k,l}}}, \quad (3)$$

where $w_{k,l}$ – weighting factors calculated according to equation (2); $X_{k,l}$ – filter window elements; M_w – weighted average according to equation (1).

The conclusion about pixel noise is made on the basis of the product of the standard deviation and the threshold value. Finally, the ASWM filter can be described as an equation:

$$Y_{i,j}=\begin{cases} m_{i,j}, \text{ if } \left|X_{i,j}-M_w(i,j)\right| > \alpha \times \delta_w(i,j) \\ X_{i,j}, \text{ otherwise} \end{cases}, \quad (4)$$

where $m_{i,j}$ – median value in the current window; $X_{i,j}$ – initial value of the pixel; $Y_{i,j}$ – value of the filtered pixel; $M_w(i,j)$ – weighted average of the window α – predetermined threshold; $\delta_w(i,j)$ – standard deviation.

As can be seen from the algorithm, the process of finding the coefficients is iterative. The number of iterations is not known in advance. The noise level of the current pixel is determined by the value of the standard deviation and the amount of deviation of the pixel value from the weighted average. This allows filtering with high noise reduction performance over a wide range of noise levels.

The disadvantages of this filter can be summarized as follows:

– large number of division operations;
– presence of a cycle of non-deterministic length;
– sensitivity to a decrease in the bit depth of fixed-point calculations;
– inability to parallelize the algorithm due to the disadvantages described above.

These shortcomings do not allow implementing the pipelined filtering algorithm ASWM on modern PLDs and vector processors, which makes the task of processing images in real time almost impossible.

3. Research results and discussion

3.1. Development of a modified ASWM filter. As mentioned above, the disadvantage is the complexity of the weight estimation routine. The basis of the program module for estimating weights is a loop with an indefinite number of iterations. On the basis of the constructed software model, a study of the filter operation at various levels of impulse noise was carried out.

As a result, histograms of the distribution of the number of cycle iterations were obtained (Fig. 2) at different noise densities. As it is possible to see, the overwhelming majority of calls perform less than 25 iterations, i. e., to implement the filter without significant performance losses, 25 loop iterations are enough.

Since the aim of research is to implement the fastest filter option, it was decided to implement a chain of N filtering blocks that can be «bypassed» (using a bypass signal) to simulate an exit from the loop when the early exit condition is met. If this condition is not reached throughout all filtration blocks, the weights of the last block will be used (Fig. 3).

ISSN 2664-9969

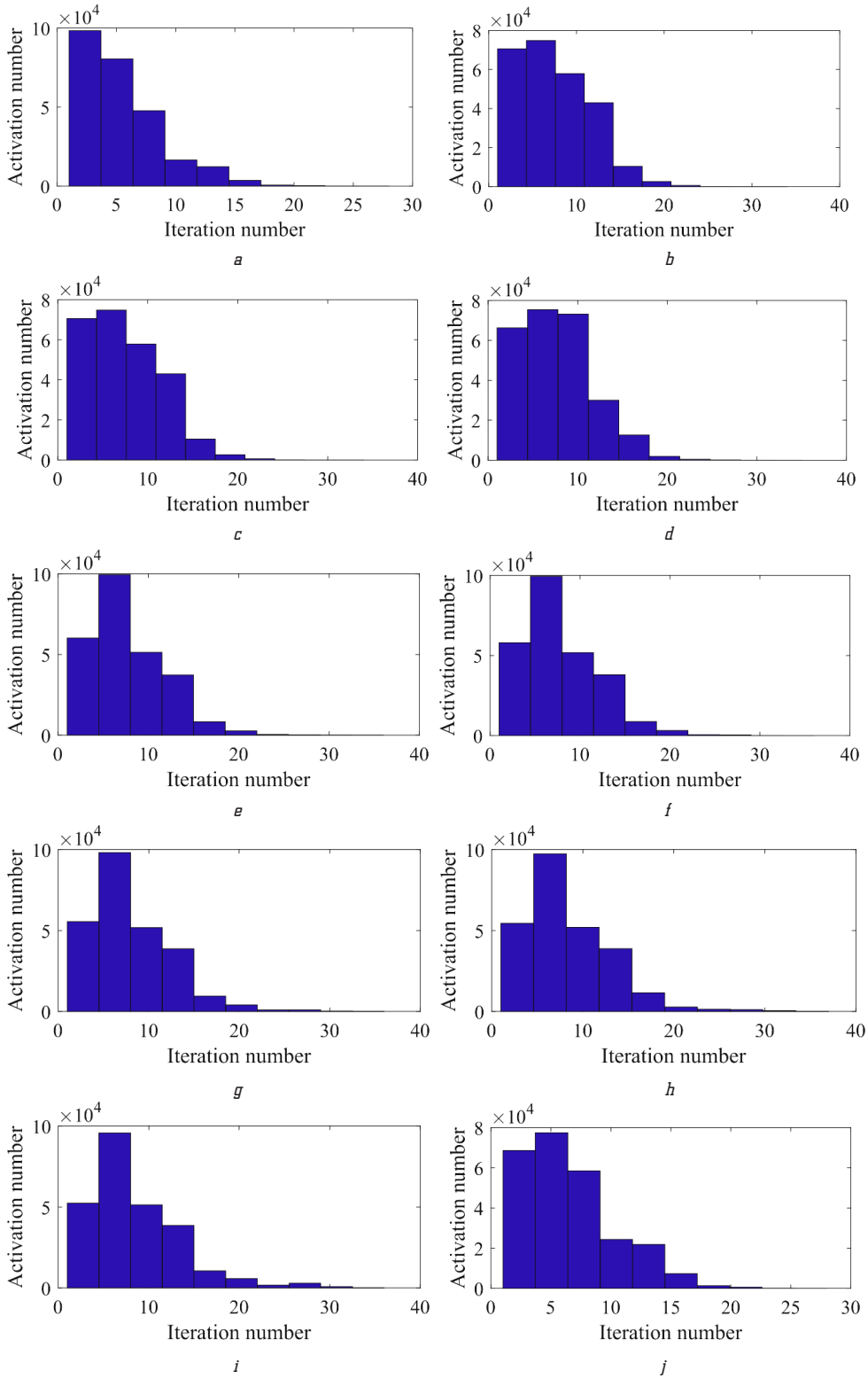

Fig. 2. Distribution of the number of iterations of the algorithm at different levels of noise density:
a – 5 %; *b* – 15 %; *c* – 20 %; *d* – 30 %; *e* – 35 %; *f* – 45 %; *g* – 50 %; *h* – 60 %; *i* – 65 %; *j* – 75 %

3.2. Optimization of the weight estimation block. The entire loop that estimates the weights for deviation-finding operations can be broken down into smaller submodules. The weight estimation block consists of a weight calculation sub-block and logic, which consists of comparison operations and a multiplexer (Fig. 4).

This approach allows pipelining the filter, limiting its length (the number of loop iterations) without significant loss of filtering quality. Fig. 5 and Fig. 6 shows the performance metrics of the original ASWM model (maximum number of iterations – 100) and modified in the described way (number of evaluation units (iterations) – 40, 20, 10, and 5).

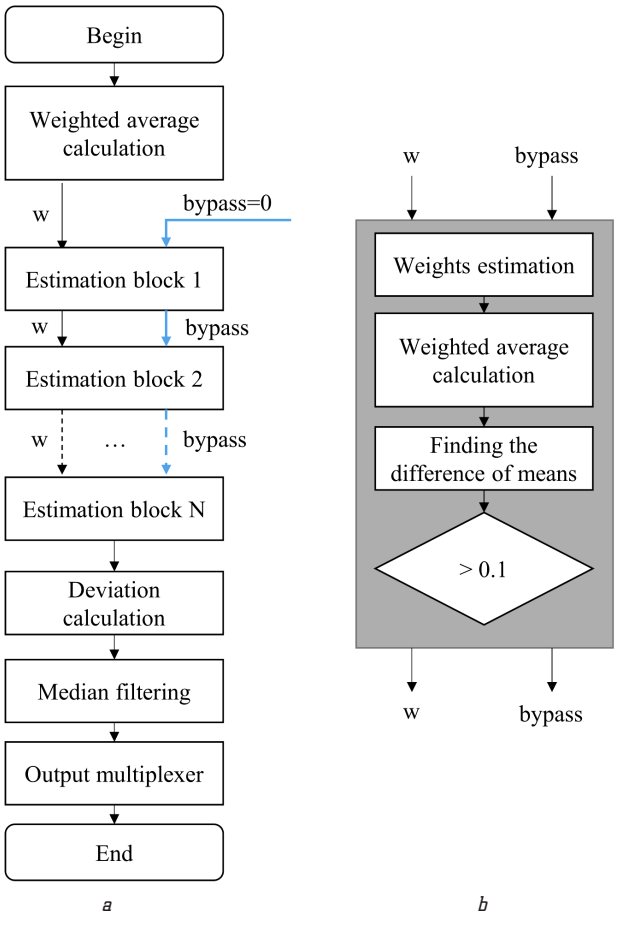

Fig. 3. Modified ASWM filtering algorithm:
a – filter algorithm; *b* – structure of the weight estimation block

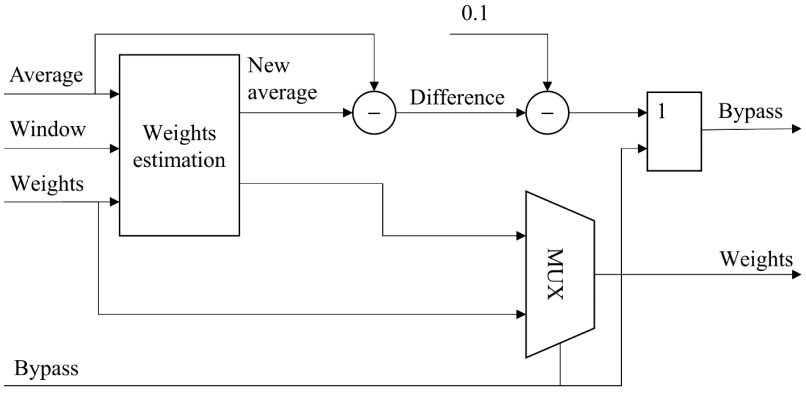

Fig. 4. Block diagram of the weight estimation module

It should be noted that a significant part of the resources used to implement the entire filter is consumed by the weight estimator, which, as mentioned above, contains nine 32-bit divisions, each of which consumes more than 1000 PLD lookup tables.

As seen from Fig. 5 and Fig. 6, the PSNR (peak signal-to-noise ratio) and SSIM (structural similarity index measure) indices [12] for filters with the number of weight estimation blocks from 100 to 20 practically do not decrease. The performance of filters with 5–10 units is significantly reduced.

Modern PLD manufacturers offer solutions [13] with a total number of LUTs sufficient for pipeline implementation of an ASWM filter with 15 weight estimation modules. To ensure the quality of filtration, 25 is required.

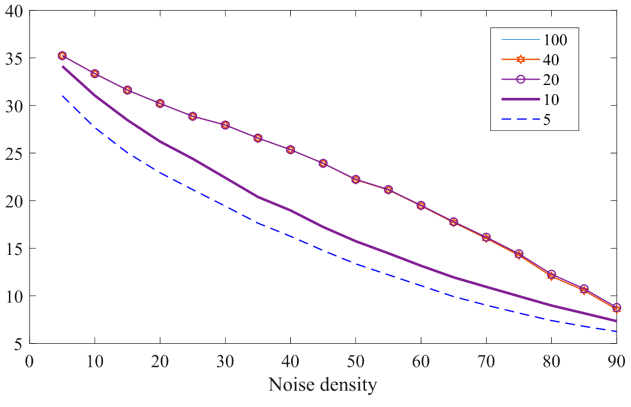

Fig. 5. PSNR filter with different number of weight estimation blocks

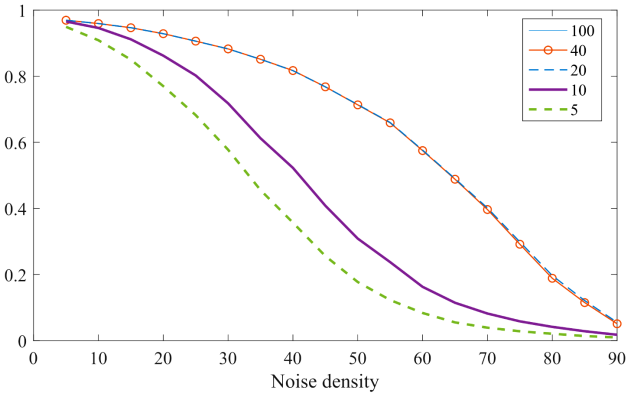

Fig. 6. SSIM filter with different number of weight estimation blocks

It is necessary to use 32-bit division in the weight estimation module.

The implementation of division is expensive for PLDs.

3.3. Division module optimization. The bit width of the numerator and denominator affects the consumption of hardware resources for the implementation of the division unit. Studies have shown that with a division capacity of less than 32 bits, the required filtering quality indicators cannot be obtained.

Thus, the next stage of research is the development of an efficient fission module with reduced resource consumption.

Below are the main methods of division with less resource consumption.

As a result of the filter simulation, it was determined that divisors greater than the value of 5 have a significant effect on the quality.

This simplifies the approximation task due to the truncation of the area with high nonlinearity.

A study of several options for replacing the division module was carried out on the filter software model.

One option is to approximate the division by a polynomial. A polynomial of the 5th degree allows to implement a function that requires 4 clock cycles to get the result (Fig. 7).

ISSN 2664-9969

$$y = -1.3724e^{-13}x^5 + 1.9381e^{-10}x^4 - 1.0196e^{-7}x^3 + 2.4567e^{-5}x^2 - 0.0027x + 0.1093. \quad (5)$$

The deviation does not exceed 10 %. The accuracy of the work does not meet the requirements.

Approximation by a polynomial of degree 10 (Fig. 8).

$$y = 1.7852e^{-24}x^{10} - 4.7764e^{-21}x^9 + 5.5142e^{-18}x^8 - 3.5953e^{-15}x^7 + 1.4556e^{-12}x^6 - 3.7901e^{-10}x^5 + 6.3589e^{-8}x^4 - 6.7055e^{-6}x^3 + 0.0004x^2 - 0.0142x + 0.2147. \quad (6)$$

As it is possible to see, increasing the degree of the polynomial significantly improves the quality of the algorithm. The deviation does not exceed 5 %.

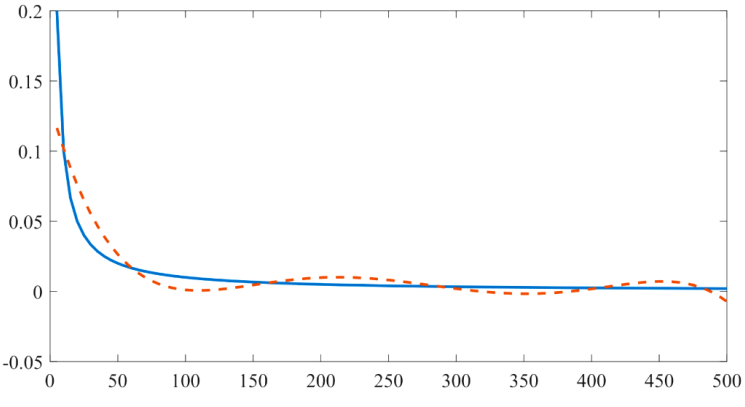

Fig. 7. Comparison of the division function and approximated by a polynomial of degree 5

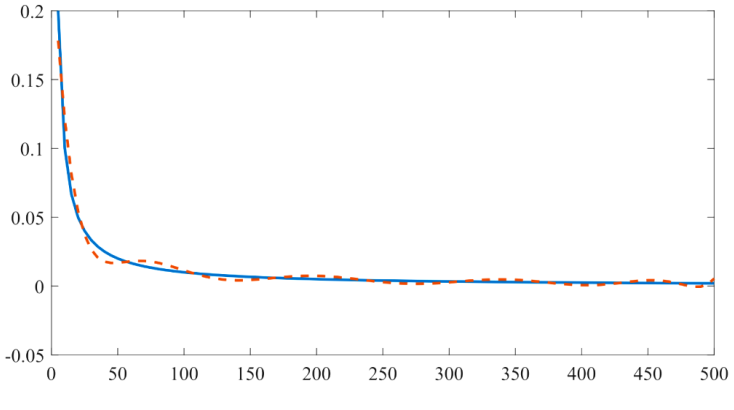

Fig. 8. Comparison of the division function and the approximated polynomial of degree 10

The third stage of the study was associated with the use of the tabular method. Since the dividend is constant, and the divisor is a fractional number not exceeding 255 in absolute value, it makes sense to consider tabular methods.

A tabular method using linear interpolation to improve accuracy can be represented by the following equations:

$$\mathrm{tg}\alpha = table[n] - table[n+1], \quad (7)$$

$$y = table[n] - \mathrm{tg}\alpha\{x\}, \quad (8)$$

where $table[n]$ and $table[n+1]$ – tabular division values; $\{x\}$ – fractional part x; n – integer part of x.

The deviation in this case does not exceed 5 %.

For this method, it is necessary to store tables to load the current and next value.

Further modeling showed that the fractional part of the number in this case can be neglected, while maintaining the high quality of filtration.

According to the filter algorithm, the values of X in the cycle of the weight estimation cannot exceed 28. However, 8-bit division cannot be applied because the filtering quality is significantly reduced.

Neglecting the fractional part of the divisor does not make significant changes (the drop in PSNR and SSIM indicators within 0.5 %) while maintaining the bit depth of the result.

Thus, the most acceptable option is to use a 32-bit integer division result table.

Such architecture can be implemented on PLD using a read only memory unit with an 8-bit address bus and a 32-bit data bus. The memory consumption for the implementation of the evaluation unit is 8 Kbit, which makes it possible to implement up to 85 evaluation units in modern devices.

The following shows the consumption of hardware resources for the implementation of the weight estimator with a modified FPGA module.

The described approach allows to implement division without using additional logic, but using PLD memory. Memory consumption allows implementing the required number of blocks in modern PLD Altera and Xilinx. The described solution can be further minimized by further truncating the divisor.

3.4. Resource consumption calculation and performance evaluation. The total consumption of PLD resources required to implement the filter can be estimated as the sum of the components:

$$V_{sum} = V_{mean} + V_{deviation} + V_{loop} \cdot N, \quad (9)$$

where V_{mean} – PLD resources for the unit of the weighted average; $V_{deviation}$ – FPGA resources for the deviation block; V_{loop} – PLD resources per unit of evaluation; N – the number of evaluation units.

The total number of LUTs depends on the number of evaluation units in the filter. Table Figure 1 shows the number of LUTs required to implement an N unit filter for Intel PLDs.

The length of the filter conveyor is also the sum of the length of its components. The length of the conveyor for a different number of evaluation units is presented in Table 2.

Using the described method in existing PLD microcircuits, it is possible to implement an ASWM filter containing up to 50 weight estimation units, which is sufficient to obtain an acceptable filtering quality.

The described structure at a clock frequency of 100 MHz allows filtering 48 frames of monochrome FullHD images per second, which is sufficient for real-time operation.

Table 1

PLD resource consumption

Resources	*N*	Cyclone V	Cyclone IV E	Cyclone 10
Total logic elements	10	5071	29792	29792
	15	5071	38377	38377
	20	5071	46962	46962
	25	5071	55547	55547
	30	5071	64132	64132
Total registers	10	47054	18368	18368
	15	68614	25733	25733
	20	90174	33098	33098
	25	111734	40463	40463
	30	133294	47828	47828
Total memory bits	10	776960	788193	788193
	15	1165440	1182138	1182138
	20	1553920	1576083	1576083
	25	1942400	1970028	1970028
	30	2330880	2363973	2363973
Multiplier 9-bit	–	150	178	66

Table 2

PLD conveyor length depending on the number of blocks

N	Total pipeline length
10	80
20	130
25	155
30	180

4. Conclusions

In this work, a software model of the ASWM filter is developed. The analysis of the filter operation with a limited number of iterations of the weight estimation submodule is carried out. Methods for implementing the division module are considered. An algorithm has been developed that is optimal in terms of the ratio of hardware costs to accuracy.

A method for implementing a fully pipelined ASWM image filter is proposed. The techniques used are considered, which make it possible to significantly reduce the PLD hardware resources required for implementation without critical losses in the filtering quality. The filter is synthesized for the main families of Intel PLD and requires less than 65,000 PLD logic elements to implement the most efficient version. At the same time, at a clock frequency of 100 MHz, the filter allows processing up to 48 frames/s.

References

1. Gonzalez, R C. (2003). *Digital Image Processing*. Beijing: Publishing Hourse of Electronics Industry, 123–124.
2. Brownrigg, D. R. K. (1984). The weighted median filter. *Communications of the ACM, 27 (8)*, 807–818. doi: http://doi.org/10.1145/358198.358222
3. Goyal, P., Chaurasia, V. (2017). Application of median filter in removal of random valued impulse noise from natural images. *2017 International Conference of Electronics, Communication and Aerospace Technology (ICECA), 1*, 125–128. doi: http://doi.org/10.1109/iceca.2017.8203657
4. Konieczka, A., Balcerek, J., Dabrowski, A. (2018). Method of adaptive pixel averaging for impulse noise reduction in digital images. *2018 Baltic URSI Symposium (URSI)*. Poznan, 221–224. doi: http://doi.org/10.23919/ursi.2018.8406738
5. Dawood, H., Dawood, H., Guo, P. (2015). Removal of random-valued impulse noise by Khalimsky grid. *2015 Asia Pacific Conference on Multimedia and Broadcasting*. doi: http://doi.org/10.1109/apmediacast.2015.7210268
6. Darus, M. S., Sulaiman, S. N., Isa, I. S., Hussain, Z., Tahir, N. M., Isa, N. A. M. (2016). Modified hybrid median filter for removal of low density random-valued impulse noise in images. *2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. Batu Ferringhi, 528–533. doi: http://doi.org/10.1109/iccsce.2016.7893633
7. Zhang, X., Liao, H., Du, X., Xu, B. (2018). A Fast Hybrid Noise Filtering Algorithm Based on Median-Mean. *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. Changchun, 2120–2125. doi: http://doi.org/10.1109/icma.2018.8484392
8. Hsieh, C., Huang, P., Zhao, Q. (2018). Impulse Noise Replacement With Adaptive Neighborhood Median Filtering. *2018 International Conference on Machine Learning and Cybernetics (ICMLC)*. Chengdu, 491–496. doi: http://doi.org/10.1109/icmlc.2018.8527058
9. Akkoul, S., Ledee, R., Leconge, R., Harba, R. (2010). A New Adaptive Switching Median Filter. *IEEE Signal Processing Letters, 17 (6)*, 587–590. doi: http://doi.org/10.1109/lsp.2010.2048646
10. Kang, C.-C., Wang, W.-J. (2009). Modified switching median filter with one more noise detector for impulse noise removal. *AEU – International Journal of Electronics and Communications, 63 (11)*, 998–1004. doi: http://doi.org/10.1016/j.aeue.2008.08.009
11. Akkoul, S., Ledee, R., Leconge, R., Harba, R. (2010). A New Adaptive Switching Median Filter. *IEEE Signal Processing Letters, 17 (6)*, 587–590. doi: http://doi.org/10.1109/lsp.2010.2048646
12. Wang, Z., Bovik, A. C., Sheikh, H. R., Simoncelli, E. P. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing, 13 (4)*, 600–612. doi: http://doi.org/10.1109/tip.2003.819861
13. *Intel FPGA Product catalog, Version 20.3*. Available at: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/sg/product-catalog.pdf

Oleg Vasylchenkov, *PhD, Professor, Department of Automation and Control in Technical Systems, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine, ORCID: http://orcid.org/0000-0002-0969-2248, e-mail: oleh.vasylchenkov@khpi.edu.ua*

Igor Liberg, *PhD, Professor, Department of Automation and Control in Technical Systems, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine, ORCID: http://orcid.org/0000-0002-2404-5620, e-mail: i_liberg@ukr.net*

Mykhailo Mozhaiev, *PhD, Head of Department of Computer-Technical and Telecommunication Research, National Scientific Center «Hon. Prof. M. S. Bokarius Forensic Science Institute», Kharkiv, Ukraine, ORCID: http://orcid.org/0000-0003-1566-9260, e-mail: mikhail.mozhayev@hniise.gov.ua*

Dmytro Salnikov, *Assistant, Department of Automation and Control in Technical Systems, National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine, ORCID: http://orcid.org/0000-0002-0490-4061, e-mail: dmytro.salnikov@khpi.edu.ua*