Smoothing Filtering of Images in the Residue Number System

Nikolai I. Chervyakov[1], Pavel A. Lyakhov[2], Nikolai N. Nagornov
Department of Applied Mathematics and Mathematical Modeling
North-Caucasus Federal University
Stavropol, Russia
[1]k-fmf-primath@stavsu.ru, [2]ljahov@mail.ru

Dmitry I. Kaplun[1], Alexander S. Voznesenskiy, Danil V. Bogayevskiy[2]
Department of Automation and Control Processes
St. Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
[1]dikaplun@etu.ru, [2]dan4ezz94@gmail.com

***Abstract*— We propose a new method for image smoothing using the Residue Number System (RNS) in this paper. The essence of the approach is to replace the computationally complex division operation in RNS by multiplying all the fractional numbers by a power of two and rounding off. All subsequent calculations are performed only on fixed-point numbers as a result of these actions. The carried out theoretical and practical studies have shown that the error resulting from rounding doesn't a significant influence the result of image filtering, when a certain accuracy of calculations is reached. It is opens the possibility of effective hardware implementation on FPGA and ASIC.**

Keywords— residue number system; digital image processing; smoothing filter

I. Introduction

Methods of digital image processing are widely used in various fields of science and technology: medicine, biology, physics, astronomy, as well as in the industrial, defense and law enforcement fields [1]. The Residue Number System (RNS) has a great potential for increasing the efficiency of digital imaging systems [2]. Its inherent small-scale representation of numbers and the possibility of independent parallel processing of data [3] make it possible to significantly improve the efficiency of computations in applications with the prevailing number of modular operations of addition, subtraction and multiplication due to the optimal use of integrated circuit resources, in particular FPGA [4]. Digital image processing [1] is one of such applications.

The cleaning of image noise which is random changes in pixel values is one of the actual tasks of digital image processing [5]. Smoothing filters are used for solving this problem in practice [6, 7]: Gaussian, median, binomial etc. The operating principle of smoothing filters is based on calculating the average value among several neighboring pixels that leads to necessity to perform a division operation. Division is difficult in RNS and new methods and algorithms to quickly perform this operation are actively developed in recent years [6–8].

Now, various studies have been carried out to improve the efficiency of computations with the use of smoothing filters for image processing in RNS. In [6], a method is described, according to which there is a separation of calculations between RNS and the binary number system (BNS). The addition, subtraction and multiplication operations are performed in RNS, while the division operation is performed in BNS. In [7], a modification of this method was proposed, which realizes the division into RNS, but imposes restrictions on the bases of RNS.

We propose a new method for image smoothing using RNS. The main idea of this approach is to replace the computationally complex division operation in RNS by multiplying all fractional numbers by a factor of a certain value and then rounding off. All subsequent computations are performed only over integer numbers as a result of these actions. This allows for an effective hardware implementation on FPGA [9].

II. Introduction to RNS

Numbers in RNS are represented as a set of residues from division $(a_1, a_2, ..., a_n)$ into a basis of pairwise prime numbers $\{p_1, p_2, ..., p_n\}$, are called moduli of RNS. The product of all RNS moduli $P = \prod_{i=1}^{n} p_i$, is called dynamic range of the system. Any integer number $0 \le A < P$ can be uniquely represented in RNS as a tuple $A = (a_1, a_2, ..., a_n)$, where $a_i = |A|_{p_i} = A \bmod p_i$ [2].

Addition, subtraction and multiplication of numbers in BNS are equivalent to adding, subtracting and multiplying the residues of these numbers in RNS by the appropriate moduli:

$$A \pm B = (|a_1 \pm b_1|_{p_i}, |a_2 \pm b_2|_{p_2}, ..., |a_n \pm b_n|_{p_n}), \quad (1)$$

$$A \cdot B = (|a_1 \cdot b_1|_{p_i}, |a_2 \cdot b_2|_{p_2}, ..., |a_n \cdot b_n|_{p_n}). \quad (2)$$

Reverse conversion from RNS to BNS based on Chinese Remainder Theorem [13]:

$$A = \left| \sum_{i=1}^{n} \left| \left| P_i^{-1} \right|_{p_i} \cdot a_i \right|_{p_i} \cdot P_i \right|_P , \qquad (3)$$

where $P_i = \dfrac{P}{p_i}$ and $\left| P_i^{-1} \right|_{p_i}$ means a multiplicative inverse of P_i modulo p_i.

III. Smoothing Filtering Optimization in RNS

Image A consisting of X rows and Y columns as a function $A(x, y)$, where $0 \le x \le X - 1$ and $0 \le y \le Y - 1$ are spatial coordinates and the amplitude A at any point with a pair of coordinates (x, y) is the intensity of the image color at that point. The elements $A(x, y)$ are called pixels of image A. Filtering of the image may be represented as:

$$A_2(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} A_1(x+i, y+j) \cdot f_{i,j} \qquad (4)$$

for all pairs of values (x, y), where A_1 – original image, A_2 – filtered image, and $f_{i,j}$ – coefficients of filter with dimensions $(2k+1) \times (2k+1)$:

$$F = \frac{1}{d} \cdot \begin{pmatrix} f_{-k,-k} & \cdots & f_{-k,k} \\ \cdots & \cdots & \cdots \\ f_{k,-k} & \cdots & f_{k,k} \end{pmatrix}, \qquad (5)$$

where the sum of all filter coefficients is equal to unity and d is the averaging coefficient, defined by the formula:

$$d = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f_{i,j} . \qquad (6)$$

We need first convert the original filter coefficients to use this method. Multiplication of all the filter coefficients $f_{i,j}$ by the factor 2^n is the first step. This multiplier is effective from the point of view of the hardware implementation, since the execution of the operations of multiplication and division in the binary number record corresponds to a comma shift by n digits to the right or to the left respectively.

All the filter coefficients $2^n f_{i,j}$ are rounding up further. We get rid of fractional values and all subsequent operations on the integrated circuit are performed only on fixed-point numbers as a result of these actions. The degree n of the factor 2^n is the bit-width of the filter coefficients in this case.

The filter coefficients are transferred from BNS to RNS with the selected system of moduli $\{p_1, p_2, ..., p_m\}$ after that. The scheme of the smoothing filtering in RNS is shown in Fig. 1.

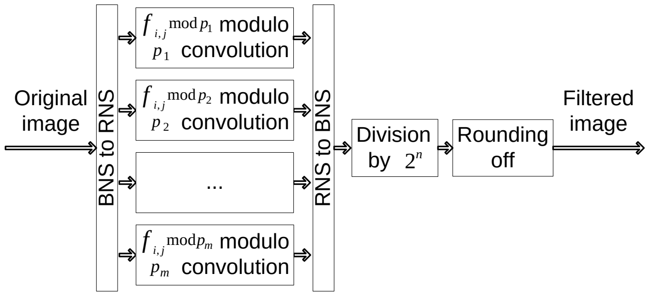

Fig. 1. The scheme of the smoothing filtering in RNS

The brightness values of the incoming digital image are converting from BNS to RNS initially. After that, the convolution with the converted filter coefficients is performed for each modulus of the system $\{p_1, p_2, ..., p_m\}$. The result of this operation converting back from RNS to BNS according to the formula (3). The values obtained are divided by 2^n and rounding down further.

There is a loss of information as a result of the rounding operation, resulting in an error. The question arises as to the magnitude of this error and its effect on the image filtration result. The accuracy of calculations increases with increasing the bit-width n of the filter coefficients. It is necessary to find out what the bit-width should be used to ensure that the calculation error does not a significant influence the final result of image filtration. We used Peak Signal to Noise Ratio ($PSNR$) between two images (obtained in BNS and in RNS) for quantify the image processing quality [5].

IV. Theoretical Analysis of the Maximum Error of the Method of Smoothing Filtering

The error initially arises when the filter coefficients are rounded up, then it increases when the convolution operation is performed, as is the rounded down after division into BNS.

We introduce the following notation from [10]: LAE_1 – the limited absolute error (LAE) of the filter coefficients rounded up; LAE_2 – the LAE of the normalized (divided by 2^n) convolution results; $AE_2 \in [0, LAE_2]$ – the absolute error (AE) of the normalized convolution results ; LAE_3 – the LAE of the normalized convolution results rounding down; $\lambda \in [0,1)$ – the fractional part of the exact convolution results; LAE_4 – the LAE of the normalized convolution results rounded down. We carry out theoretical calculations for an estimation of the maximum error of the method of smoothing filtering for Gaussian filters F_1, F_2 and F_3 (with dimensions 3×3, 5×5 and 7×7 respectively). We compute LAE of the filter coefficients rounded up at first.

$$LAE_1 = \sum_{i=-k}^{k} \sum_{j=-k}^{k} \left(\left\lceil 2^n f_{i,j} \right\rceil - 2^n f_{i,j} \right) \qquad (7)$$

We define LAE of the normalized convolution results after this.

$$LAE_2 = \frac{LAE_1 \cdot M}{2^n}. \tag{8}$$

Then we calculate LAE of the normalized convolution results rounding down.

$$LAE_3 = LAE_2 + \lambda - \lfloor LAE_2 + \lambda \rfloor. \tag{9}$$

The exact value of the convolution results is rarely an integer. Thus, LAE_3 depends not only on LAE_2, but also on λ. The resulting error is LAE of the convolution results rounded down.

$$LAE_4 = |LAE_2 - LAE_3|. \tag{10}$$

LAE_3 partially compensates LAE_2 according to the formula (10). Two of these errors are commensurable at a certain stage of increasing the bit-width n. In practice, the resulting error LAE_4 of method has the smallest value at this stage. We have determined the formula for finding LAE_4. However, the value of λ can't be determined in the course of theoretical calculations. It must be excluded from the formula, given the influence it has on the result of the calculations. We express in the formula (10) LAE_3 in terms of LAE_2 and λ according to the formula (9) for this purpose.

$$LAE_4 = |LAE_2 - (LAE_2 + \lambda - \lfloor LAE_2 + \lambda \rfloor)| = \\ = |\lfloor LAE_2 + \lambda \rfloor - \lambda|. \tag{11}$$

We consider two cases:

1. $\lfloor LAE_2 + \lambda \rfloor - \lambda > 0 \Rightarrow \lfloor LAE_2 + \lambda \rfloor \geq 1$. The larger $\lfloor LAE_2 + \lambda \rfloor$, the greater LAE_4 in this case. Thus, $\lfloor LAE_2 + \lambda \rfloor = \lfloor LAE_2 \rfloor + 1$ and λ represents the addition of the fractional part of LAE_2 to one, that is $\lambda = \lfloor LAE_2 \rfloor + 1 - LAE_2$. We substitute the resulting expression in the formula (11).

$$LAE_4 = \lfloor LAE_2 + \lfloor LAE_2 \rfloor + 1 - LAE_2 \rfloor - \\ - (\lfloor LAE_2 \rfloor + 1 - LAE_2) = LAE_2. \tag{12}$$

2. $\lfloor LAE_2 + \lambda \rfloor - \lambda \leq 0 \Rightarrow \lfloor LAE_2 + \lambda \rfloor \leq \lambda \Rightarrow \lfloor LAE_2 + \lambda \rfloor = = 0 \Rightarrow LAE_4 = |0 - \lambda| = \lambda$. The larger λ, the greater LAE_4 in this case. But $\lfloor LAE_2 + \lambda \rfloor = 0 \Rightarrow LAE_2 + \lambda = 1 - \varepsilon \Rightarrow \lambda = = 1 - \varepsilon - LAE_2$. Using AE_2 instead of LAE_2, and equating its value to zero is the optimal decision in this situation. Thus, then formula (11) takes the following form.

$$LAE_4 = |\lfloor 0 + 1 - \varepsilon \rfloor - (1 - \varepsilon)| = 1 - \varepsilon. \tag{13}$$

It is advantageous to use formula (12) for $LAE_2 > 1 - \varepsilon \geq 1$, since LAE_4 represents the limited error value. Thus, it is possible to determine uniquely LAE_4 depending on LAE_2 and independently of λ from formulas (12) and (13).

$$LAE_4 = \begin{cases} LAE_2, LAE_2 \geq 1, \\ 1 - \varepsilon, LAE_2 < 1. \end{cases} \tag{14}$$

$PSNR$ is calculated according to formula in this case:

$$PSNR = 10 \log_{10} \frac{M^2}{LAE_4{}^2}, \tag{15}$$

where $MSE = LAE_4{}^2$.

The values $PSNR$ were obtained as a result of theoretical calculations for 13 various a bit-widths n $(n = 1,...,13)$ of the Gaussian filters F_1, F_2 and F_3 coefficients and $M = 255$ shown in Table 1.

TABLE I. THE RESULTS OF THEORETICAL CALCULATIONS (DB)

n	F_1	F_2	F_3
1	-10.881	-21.214	-27.421
2	-1.938	-14.403	-21.023
3	2.499	-7.044	-14.403
4	6.021	0.561	-7.739
5	13.201	7.180	-1.493
6	22.144	15.296	5.242
7	26.581	23.059	11.774
8	30.103	29.080	18.917
9	37.283	31.907	25.886
10	46.227	37.927	31.579
11	48.131	43.948	39.378
12	48.131	48.131	43.304
13	48.131	48.131	48.131

From Table 1 we can conclude:

1. The size of the filter is increased by adding coefficients at the edges. The further the coefficient from the middle of the filter, the less influence it has on the calculated value. Therefore, the smaller its eigenvalue and the greater the error in its rounding. Which leads to increasing the bit-width of the filter coefficients to maintain the same level of accuracy.
2. The result of the filtering doesn't contain significant distortion $(PSNR \geq 40)$ when using a bit-widths $n = 10$, $n = 11$ and $n = 12$ for processing with filters of sizes 3×3, 5×5 and 7×7 respectively. A value of 40 dB describes the difference between the two images that is not visible to the viewer. Thus, we can determine the minimum value of the bit-width n of the filter with size $(2k + 1) \times (2k + 1)$ coefficients at which the result of processing doesn't contain significant distortion:

$$n = 9 + k. \tag{16}$$

3. The result of the filtering does contain minimum distortion $(PSNR \approx 48.131)$ when using a bit-widths

$n=11$, $n=12$ and $n=13$ for processing with filters of sizes 3×3, 5×5 and 7×7 respectively. Thus, we can determine the minimum value of the bit-width n of the filter with size $(2k+1)\times(2k+1)$ coefficients at which the result of processing contains minimum distortion:

$$n = 10 + k. \tag{17}$$

Further, we carry out simulation of this method to be confirmed with the results of the theoretical calculations.

V. Simulation and Discussion of the Results

The simulation was carried out in MatLab software version R2015b for an 8-bit image in grayscale: «Lena» (Fig. 2a). RNS with moduli $\{2^r-1, 2^r, 2^r+1\}$ [2] with $r=8$ and dynamic range $P=16776960$ was used. A discrete white Gaussian noise of 5, 10, ..., 50 dB is applied to the original image using the wgn command. Further initial and obtained images are filtered in BNS and in RNS with a bit-widths $n=1,...,13$ of the Gaussian filters F_1, F_2 and F_3 coefficients with the imfilter command. The convolution results in BNS are also rounded down.

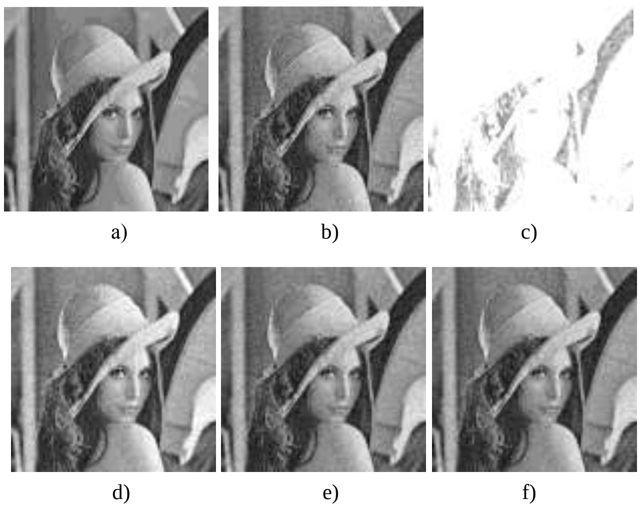

Fig. 2. The results of simulation image «Lena» with using filter F_3: a) original image; b) noised image with wgn of 30 dB; the results of filtration: c) RNS, $n=4$, $PSNR=-2.005$ dB; d) RNS, $n=8$, $PSNR=23.214$ dB; e) RNS, $n=12$, $PSNR=27.858$ dB; f) BNS, $PSNR=27.844$ dB.

An example of the simulation results is shown in Fig. 2. Fig. 2 shows that the quality of image filtering in RNS is gradually improved with the increasing the bit-width n of the filter coefficients: only the darkest places are visible at $n=4$; the small distortions are visible at $n=8$; the result of the filtering in RNS is indistinguishable from the filtering result in BNS at $n=9+k=9+3=12$.

The values $PSNR$ were obtained as a result of image processing by Gaussian filter F_3 are presented in Table 2. The bit-widths of the filter coefficients in RNS are chosen according to formulas (16) and (17). As shown in Table 2, the results of image processing with filter F_3 in RNS with a bit-width $n \geq 9+k=12$ of the filter coefficients don't contain significant distortions compared to the results of filtering in BNS. Thus, the results of simulation confirm the results of the theoretical calculations. We can do the following conclusions,

TABLE II. The results of simulationob with filter F_3 (dB)

Noise	Noised image	BNS	RNS		BNS – RNS	
			$n=12$	$n=13$	$n=12$	$n=13$
original	∞	33.524	33.560	33.571	-0.037	-0.048
5	43.028	33.488	33.521	33.532	-0.033	-0.045
10	38.099	33.406	33.439	33.448	-0.033	-0.042
15	33.103	33.153	33.188	33.196	-0.035	-0.043
20	28.138	32.480	32.503	32.515	-0.022	-0.034
25	23.123	30.773	30.790	30.797	-0.017	-0.024
30	18.285	27.844	27.858	27.858	-0.014	-0.014
35	13.753	23.956	23.970	23.965	-0.014	-0.009
40	10.152	19.896	19.915	19.906	-0.020	-0.010
45	7.918	16.722	16.729	16.726	-0.007	-0.004
50	6.760	14.845	14.844	14.846	0.001	0.000

based on the results of theoretical calculations and simulation:

1. The bit-width n of the filter coefficients at which the filtration result doesn't contain significant distortions $(PSNR \geq 40)$, can be found by formula (16).

2. You can achieve a reduction in the resources used in the hardware implementation of this method, since the highest bits of the coefficients are zero.

VI. Conclusion

We propose a new method for image smoothing using RNS. The main idea of this approach is to replace the computationally complex division operation in RNS by multiplying all fractional numbers by the factor of a certain value and then rounding off. All subsequent computations are performed only over integer numbers as a result of these actions.

The results of theoretical calculations and simulation showed that if the bit-width of the filter coefficients $n=9+k$, where the value k is determined by the dimensions $(2k+1)\times(2k+1)$ of the smoothing filter, the calculation error resulting from rounding doesn't a significant influence the result of image filtering $(PSNR \geq 40)$. This allows for an effective hardware implementation on FPGA.

References

[1] A.C. Bovik, Handbook of image and video processing, 2nd ed. San Diego, Elsevier Academic Press, 2005.

[2] C.-H. Chang, A.S. Molahosseini and L.S. de Sousa, Embedded Systems Design with Special Arithmetic and Number Systems. Cham, Springer International Publishing, 2017.

[3] C.-H. Chang, A.S. Molahosseini, A.A.E. Zarandi and T.F. Tay, "Residue Number Systems: A New Paradigm to Datapath Optimization for Low-Power and High-Performance Digital Signal processing Applications,"

IEEE Circuits and Systems Magazine, vol. 15, no. 4, 2015, pp. 26-44, doi: 10.1109/MCAS.2015.2484118.

[4] G. Bailey, Design for embedded image processing on FPGAs. Singapore, Wiley-IEEE Press, 2011.

[5] N.I. Chervyakov, P.A. Lyakhov and M.G. Babenko, "Digital Filtering of Images in a Residue Number System Using Finite-Field Wavelets, " Automatic Control and Computer Sciences, 2014, vol. 48, no. 3, pp. 180-189. doi: 10.3103/S0146411614030031.

[6] E. Vasalos, D. Bakalis and H.T. Vergos, "RNS Assisted Image Filtering and Edge Detection," IEEE 18th International Conference on Digital Signal Processing (DSP), July 2013, pp. 1–6, doi: 10.1109/ICDSP.2013.6622821.

[7] N.I. Chervyakov, P.A. Lyakhov, A.S. Ionisyan and M.V. Valueva, "High-Speed Smoothing Filter in the Residue Number System," Digital Information Processing, Data Mining, and Wireless Communications (DIPDMWC), 2016, pp. 121–126. doi: 10.1109/DIPDMWC.2016.7529375.

[8] C.-C. Chang and Y.-P. Lai, "A division algorithm for residue numbers," Applied Mathematics and Computation, vol. 172, no. 1, 2006, pp. 368-378, doi: 10.1016/j.amc.2005.02.008.

[9] F. Talbi, F. Alim, S. Seddiki, I. Mezzah and B. Hachemi, "Separable Convolution Gaussian Smoothing Filters on a Xilinx FPGA platform, " IEEE Fifth International Conference on Innovative Computing Technology (INTECH), 2015, pp. 112-117. doi: 10.1109/INTECH.2015.7173372.

[10] R.L. Burden, J.D. Faires and A.M. Burden, Numerical Analysis, 10th ed. Boston, Cengage Learning, 2016.