Real-Time Wavelet-Spatial-Activity-Based Adaptive Video Enhancement Algorithm for FPGA

V. Zlokolica[1], M. Katona[1], M. Juenke[2], Z. Krajacevic[1], N. Teslic[1], and M. Temerinac[1]

[1] MicronasNIT, Fruskogorska 11, 21000 Novi Sad, Serbia
[2] Micronas GmbH, Hans-Bunte-Strasse 19, 79108 Freiburg Germany
zlokolica@micronas.com

Abstract. In this paper we present a wavelet-based video enhancement algorithm designed for highly optimized dedicated ICs. The proposed algorithm is implemented on FPGA platform with target being real-time video processing. The main application of the proposed scheme is a high definition (HD) TV, where we consider visibly annoying video coding artifacts and noise (assumed as white Gaussian).

In the proposed denoising scheme each video frame is processed independently, i.e., only spatial filtering is performed. Specifically, two-dimensional (2D) non-decimated wavelet transform is applied to the frame, after which the proposed activity-adaptive shrinkage operation on the wavelet coefficients is done. Finally, the denoised image is reconstructed by inverse wavelet transform. The main contribution of the paper is the proposed (i) hardware-friendly scheme for the wavelet decomposition - reconstruction framework with full parallelism and reduced memory resources required and (ii) efficient and low computationally expensive activity-adaptive shrinkage algorithm for denoising.

The designed framework is verified in SystemC and on FPGA platform with WXGA Panel. The annoying artifacts and noise are shown to be efficiently removed with small or no visible reduction in spatial resolution.

1 Introduction

Video sequences are often corrupted by noise and/or coding artifacts. Noise reduction in image sequences is used for various purposes, e.g., for visual improvement in video surveillance and broadcast. Recently, since the introduction of HD video, the reduction of noise and artifacts in general has become even more important; this is due to the increase of video resolution, where the artifacts/noise become more apparent and annoying.

In general denoising is usually achieved through some form of linear or non-linear operation on correlated picture elements. A number of techniques for still image [1, 2, 3, 4] and video [5, 6, 7] denoising have been proposed. Some of these techniques are of fairly low complexity and as such are suitable for real-time applications in hardware; other more complicated methods provide better denoising

J. Blanc-Talon et al. (Eds.): ACIVS 2008, LNCS 5259, pp. 182–193, 2008.

results for the cost of an increased complexity, which is usually a bottleneck in a real-time hardware implementations. In the past, several hardware-oriented video denoising algorithms were proposed [5,8], which were of relatively low complexity. Nowadays, there are more and more sophisticated and complex techniques implemented in hardware, e.g., wavelet-based [9] schemes for video denoising; this is due to the increased amount of memory resources and computational power available.

The wavelet representation naturally facilitates the construction of the spatio-temporally adaptive denoising algorithms: it compresses the essential information in a video sequence into relatively few, large coefficients (corresponding to noise-free image details) and provides relatively large number of small coefficients (corresponding to noise or uniform image regions). Due to this property, wavelet transform is used effectively for image compression and denoising by simply disregarding small coefficients (setting them to zero) and leaving the high amplitude wavelet coefficients as they are. This approach for image denoising is usually, in the literature, referred to as hard thresholding [10]. However, for a superior denoising performance one has to employ more sophisticated spatial and/or temporal adaptive thresholding [11,3,12,4]. On the other hand, wavelet transform, on the implementation side, usually requires significant memory resources and computational burden. Hence, for the wavelet-based denoising methods aimed at a real-time processing in hardware, certain design optimizations must be done in a way which reduces the algorithm performance to the least extent.

In this paper we propose a wavelet-based video enhancement algorithm, designed for a real-time processing on platforms with restricted resources. These specific implementation conditions are still regarded by design of the algorithm. The framework for the wavelet transform (direct and inverse) is designed to maximally reduce required memory resources and increase the parallelism in the processing. The proposed algorithm for video enhancement is a spatial-activity-based approach that applies wavelet-shrinkage function, which is dependent on the estimated local spatial and intra-scale activity; this activity is proportional to the amount of high amplitude wavelet coefficients in a local neighborhood.

The proposed algorithm is optimized, implemented and verified on FPGA platform. The noise and video coding artifacts (ringing and blocking) are shown to be efficiently removed in real time, with relatively low computation effort and memory required. The paper is organized as follows. First a general description of the framework is given in Section 2. We explain in detail the wavelet transform design scheme for FPGA in Section 2.1 and noise-adaptive denoising algorithm in Section 2.2. Finally, we present implementation details and evaluation results in Section 3 and conclude the paper in Section 4.

2 The Proposed Framework

The proposed video enhancement algorithm is implemented and designed for the video system shown in Fig. 1. As illustrated, the input of our system is

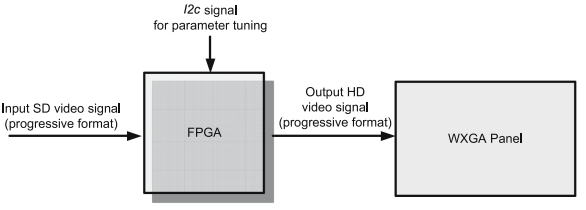

Fig. 1. System platform

noisy, standard definition (SD), progressive format video. This signal is fed into the FPGA which performs noise reduction and subsequently up-scaling. The processed video is then displayed on a WXGA panel.

The general description of the proposed video denoising framework for the FPGA is as follows: (i) the input video is in YUV format, where the processing is only performed on the luminance (Y) channel. The other two, chrominance channels C_R and C_B, are only delayed for the time needed for processing the Y component in order to synchronize in time all three components.[1](ii) Each video frame is processed independently, i.e., only spatial filtering is performed. This is due to the imposed restrictions on memory resources in our video system. Another reason for considering only spatial filtering is because such an algorithm can be independently tested and placed in different parts of the video chain.

(iii) The algorithm for video enhancement is a spatial-activity-based approach which estimates the local spatial activity by calculating the number of significant wavelet coefficients at the same orientation and different resolution scales in a particular spatial neighborhood. (iv) The spatial activity is then used as an input for piece-wise linear shrinkage function, which is applied to wavelet coefficients in order to perform denoising, i.e., artifact reduction. The proposed scheme uses similar ideas as [4]; the main difference is that in our approach we do not employ connectivity indicator and we use a specific weighting map for activity estimation. Additionally, we apply a piece-wise linear function on the estimated spatial local activity in order to obtain the shrinkage function. The latter enables independent tuning of noise/artifact reduction and tunable sharpness enhancement.

2.1 Wavelet Transform Framework

We perform non-decimated wavelet transform and use Haar wavelet basis in our approach for the sake of low complexity and low memory requirements. We have found experimentally through simulations that the choice of wavelet basis does not influence the performance of our denoising algorithm significantly.We apply a 2D (spatial) wavelet transform [13] to each video frame and denote *wavelet bands*

[1] The reason for not processing the chrominance channels is a well known fact that a human eye is much more sensitive to the luminance than to the chrominance components.

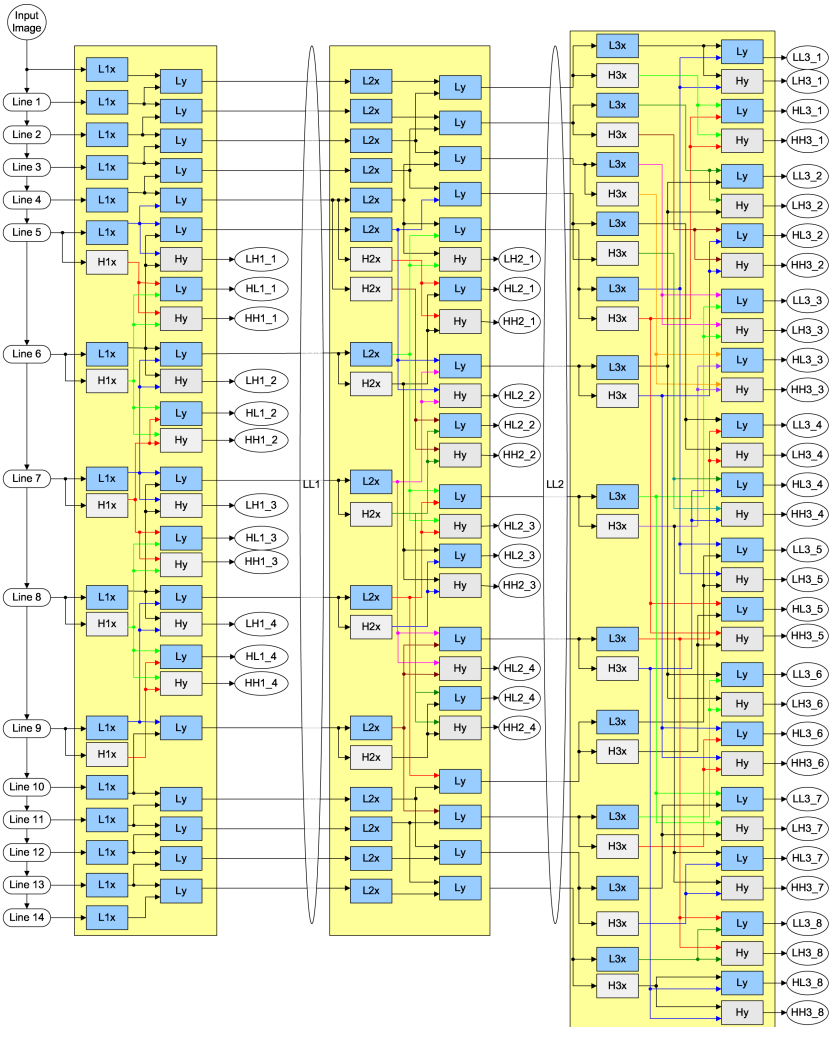

Fig. 2. Direct wavelet transform

of this spatial wavelet transform by WB=LL,LH,HL,HH for the low-pass (approximation), horizontal, vertical and diagonal orientation bands, respectively. Specifically, three resolution scales are computed, with three differently oriented wavelet bands.

The decomposition of the designed wavelet transform is shown in Fig. 2. Note that all major memory resources are put in front of the block for the direct wavelet transform (and in fact all the processing) where multiple computations are done in parallel; this includes the computation of the wavelet coefficients (decomposition) for multiple lines in the image in parallel and immediate reconstruction (inverse wavelet transform) of picture elements from the processed (shrinked) wavelet

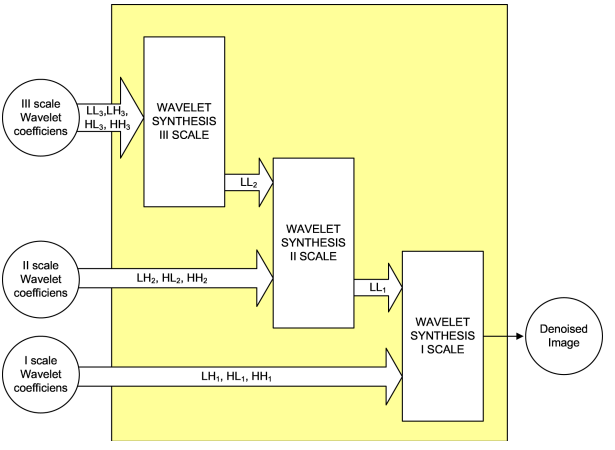

Fig. 3. Inverse wavelet transform

coefficients. This is in contrast to a common approach for wavelet decomposition [9], where the memory lines are inside the decomposition and reconstruction block and where the intermediate results have to be stored at separate memory locations. These separate memory locations usually use external SDRAM or DDR2 memory resources, while in the proposed approach no external memory is used. The advantage of this approach is two fold: firstly less memory is needed and secondly full parallelism in processing is enabled with minimum delay and minimum memory access, which results in reduced time for processing. In addition, in the proposed scheme we have at our disposal a concurrent access to the spatial neighborhood (at a particular time instant) of the wavelet coefficients at different scales and orientations, without any additional memory requirements.

An internal memory consists of 14 line memories, which are designed as a (FIFO) shift-buffers. The top line memory ("line 1") receives the data from the input video frame. After that the data is forwarded from one memory line to the next in a sequential order. Besides the memory lines, other blocks in decomposition step (illustrated in Fig. 2) stand for the simple filtering operations: multipliers and adders. Specifically, $L1x$, $L2x$, $L3x$, $H1x$, $H2x$ and $H3x$ denote low-pass and high-pass filters for corresponding resolution scales ($l = 1, 2, 3$) in the horizontal direction. These filters consist only of small delay buffer (its length depends on the resolution scale - the higher the scale the longer the buffer is) and one addition.[2] Additionally, Ly and Hy stand for the low-pass and high-pass filters in the vertical direction where only one addition is used, since the data is coming in parallel.

The available neighborhood in the vertical direction is shown in Fig. 2 as the output of the decomposition block for each decomposition step in circles. The first number next to the orientation band denotation corresponds to the decomposition step and the second to the vertical coordinate. The neighborhood

[2] The coefficients are 1 and -1 in case of Haar wavelet.

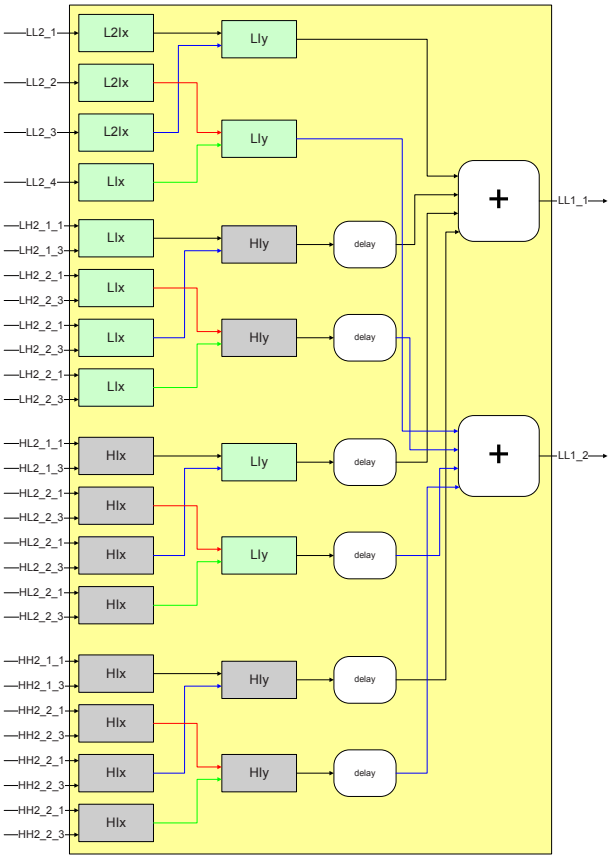

Fig. 4. Second reconstruction step. "WBk_j_i" corresponds the the processed wavelet band at scale "k", horizontal position "i" and vertical coordinate "j"(WB=LH,HL,HH)

in the horizontal direction is defined analogously, using delay operators. Note that the construction of the direct wavelet transform as illustrated in Fig. 2 provides also vertical and horizontal alignment of image features (represented by wavelet coefficients) between different scales; we use this property for denoising (as described in Section 2.2).

The general scheme for the inverse wavelet transform is depicted in Fig. 3. The processed (denoised) wavelet coefficients are fed in parallel (meaning vertically and horizontally) to the reconstruction block. An example of the second reconstruction step is shown in Fig. 4. The reconstruction (synthesis) is thus performed without long delays. The only introduced delays introduced come from additional alignment needed; this is because some operations have to be done sequentially: LL_2 and LL_1 have to be computed before performing synthesis for the second and the first scale, respectively. All other operations are done in parallel.

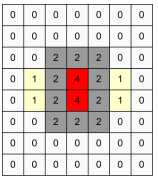

Fig. 5. Neighborhood for the wavelet bands at first scale

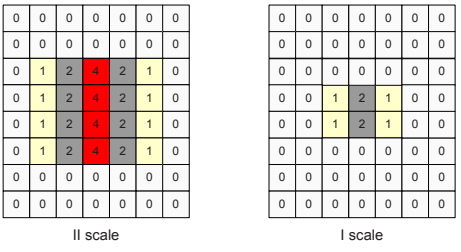

Fig. 6. Neighborhood for the wavelet bands at second scale

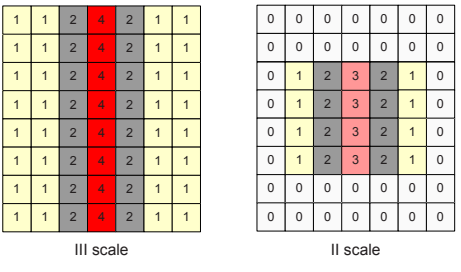

Fig. 7. Neighborhood for the wavelet bands at third scale

2.2 Spatial Denoising

In the proposed denoising method we exploit spatial and interscale correlations in wavelet bands in order to estimate the local spatial activity. Non-decimated wavelet transform is used because of higher spatial correlation of the wavelet coefficients which facilitates more accurate activity estimation map and hence superior noise reduction. Additionally, the inter-scale dependencies between the wavelet coefficients of the same orientation and at different resolution scales are supposed to be large in case of significant noise-free image features; the feature represented by the corresponding wavelet coefficients should normally progress from the highest scale to the lower. Note, however that these features spread more in space for lower resolution scales, i.e., for the higher decomposition levels. Consequently, larger local spatial neighborhood should be investigated in case of lower resolution scales.

The spatial and inter-scale correlation ensures higher reliability for local spatial activity estimation because the similar amplitude coefficients are expected to be spread in spatial and inter-scale domain. In our denoising algorithm we combine the information concerning the spatial (intra-scale) and inter-scale *activity* for determining the shrinkage function for each spatial position and each wavelet band separately. The activity is estimated by taking into account the amplitude of the wavelet coefficients from a specifically defined spatial and scale-parent neighborhood surrounding the wavelet coefficient to be shrinked. The proposed neighborhood for the first, second and third scale and their corresponding weights assigned to each position are depicted in Fig. 5-7.

The estimated activity θ for the local spatial area $R^{(l)}$, in the wavelet band WB at resolution scale l, is computed as follows:

$$\theta(R^{(l)}) = \sum_{\mathbf{r}_k \in R^{(l)}} \beta(\mathbf{r}_k, l) q_{\mathbf{r}_k} \tag{1}$$

where $\beta(\mathbf{r}_k, l)$ stands for the weights given for the neighborhood $R^{(l)}$ as shown in Fig. 5-7.[3] Additionally, $q_{\mathbf{r}_k}$ is defined as follows:

$$q_{\mathbf{r}_k} = \begin{cases} 1\ , & |WB_{\mathbf{r}_k}| > T_a \\ 0\ , & \text{otherwise} \end{cases} . \tag{2}$$

T_a denotes the threshold based on which is determined if the wavelet coefficient is of significant amplitude or not; this threshold is experimentally found and depends on the estimated Gaussian noise level and/or estimated strength of video coding artifacts.

Based on the estimated activity θ the following shrinkage function γ is defined:

$$\gamma = \begin{cases} 0\ , & \theta < T_n^{(l)} \\ \alpha_l \theta\ , & T_n^{(l)} < \theta < T_s^{(l)} \\ \alpha_p\ , & \theta > T_s^{(l)} \end{cases} . \tag{3}$$

where α_l, $T_n^{(l)}$, $T_s^{(l)}$ $(l = 1, 2, 3)$ and α_p are enhancement parameters that determine the denoising and sharpness strength.

This shrinkage function is subsequently applied to the group of wavelet coefficients with weights given value 4 (denoted with red), in Fig. 5-7; after that the shrinked wavelet coefficients are fed into the output buffers and sent to the block for the inverse wavelet transform.[4]

3 Implementation and Evaluation

The proposed method for the wavelet-based denoising was first designed and evaluated in SystemC, where a special attention was payed to operations and

[3] Denotation for the wavelet band WB is excluded from equation 1 for the sake of clarity.

[4] The length of the output buffers used in the denoising block corresponds to the size of the horizontal filters needed for the reconstruction step.

Table 1. Objective results for "Bicycle" sequence corrupted with Gaussian noise ($\sigma = 15$) and with coded bit rate 500kB/s

	$\sigma = 15$			Bit-rate 500kB/s		
measure	**PSNR**	**FROS**	**SSIM**	**PSNR**	**FROS**	**SSIM**
Noisy	24.2	9.1	0.54	28.7	7.4	0.91
Balster [4]	28.9	9.4	0.87	29.1	7.6	0.92
Proposed method	29.3	10.2	0.85	29.2	8	0.93

Table 2. Objective results for "CrossPosint" sequence corrupted with Gaussian noise ($\sigma = 10$) and with coded bit rate 500kB/s

	$\sigma = 10$			Bit-rate 500kB/s		
measure	**PSNR**	**FROS**	**SSIM**	**PSNR**	**FROS**	**SSIM**
Noisy	28.2	8.8	0.8	27.2	5.8	0.91
Balster [4]	31.6	8.3	0.94	28	6.3	0.92
Proposed method	31.5	8.9	0.95	28.5	6.7	0.93

memory resources that are suitable for the FPGA implementation. Namely, we reduce the number of memory resources at the cost of a small increase in the concurrent (parallel) processing.

The denoising performance was initially tested in SystemC simulation on the noisy video sequences where noise (white Gaussian) was added to the noise-free video sequence. Subsequently, we tested the algorithm on coded video sequences containing blocking and ringing artifacts. The results showed that Gaussian-like noise and ringing artifacts were efficiently removed, while the blocking artifacts were partially removed. Additionally, since the proposed algorithm enables sharpening as well, we have also evaluated this aspect. The results have shown that in some cases we get improvement in video sharpness, while in other cases it has not brought any improvement or has introduced new artifacts. Finally, the spatial resolution of the video sequence after processing has in most cases remained the same; in cases of highly textured images the visual appearance has shown the decrease in terms of spatial resolution to some extent.

We evaluated the denoising results visually and also using objective quality assessment measures: PSNR, Structural Similarity (SSIM) [14] and Full Reference Objective Score (FROS) [15].[5] In Table 1 and Table 2 we show the objective results for two sequences: "Bicycle" with added Gaussian noise $\sigma = 10$ and coded video on 500kB/s, and "CrossPoint" sequence with added Gaussian noise $\sigma = 15$ and coded video on 500kB/s. The results of the proposed video denoising method implemented in FPGA was compared to the algorithm of [4]. This was done since both algorithms are based on similar ideas, where the proposed method is of relatively lower complexity and of different structure; this is due

[5] In all three measures higher values indicate better results: e.g., in case of SSIM 10 corresponds to very good visual result and for FROS max value is 1.

(a) Noisy (coding Bit-rate 500kB/s) (b) Denoised by the proposed method

Fig. 8. Visual results for 'CrossPoint' sequence

to limitations required by FPGA real-time processing with restricted memory resources. The results in Table 1 and Table 2 show that denoising results are similar or mostly slightly better than the method of [4] and especially for coded video sequences. Namely, in terms of PSNR, for the case digitally coded video sequence, it is always better while in case of added Gaussian noise it is similar or slightly better. In case of FROS measure, the proposed method shows always better results, while in case of SSIM approach it is sometimes worse and sometimes better. In Fig. 8 visual results are shown for digitally coded video sequence and the denoised sequence by the proposed method.

For a real-time evaluation we have used CHIPit Gold Edition FPGA platform [16] with two Xilinx Virtex II FPGAs XC2V6000-5 [17]. The analog input video stream in 720p60 format was connected to the first FPGA via the ADC input board. Complete denoising in the wavelet domain is implemented within one FPGA. The processed stream in YUV format is sent to the second FPGA where the signal adjustment for the LVDS connection with the WXGA panel is performed.[6] Structure of this FPGA is quite simple and it is not analyzed in this paper.

SystemC implementation of the described algorithm was converted to the appropriate Verilog RTL description. The principles for writing synthesizable SystemC described in [18] are used during system description. The Synopsys SystemC compiler is used for automatic SystemC conversion to the synthesizable Verilog RTL description. This module was included in the FPGA wrapper containing clock generation unit, I2C slave module for parameter configuration and appropriate interfaces. The Synplicity SynplifyPro and Xilinx tool chain were used for RTL synthesis and FPGA place and route, respectively. The synthesized wavelet denoising is using approximately 62% of the available Slices in the FPGA. Additional resources include internal FPGA memory (28% of available BlockRAMs) and 93 dedicated multipliers (64%). Complete FPGA1 is

[6] Signal adjustment for Full HD panel connected to the output signal is also supported.

occupying slightly more Slices (72%, 24,346 out of 33,792) and multipliers (77%, 97 out of 144). The number of BlockRAMs is significantly bigger with 97% in total (111 out of 144). This is due to internal frame splitting to 3 horizontal parts which can be independently processed. In our case only one third of complete frame is denoised, while other two thirds are only bypassed. Such a configuration is convenient for the evaluation of denoising results in comparison to the input noisy video stream.

Wavelet processing and denoising in wavelet domain are running on video front-end clock of 27MHz. The RTL architecture is designed for frequencies above 45MHz, meaning that the proposed algorithm can also be used in video back-end data paths. If necessary, the clock frequency can be further increased with data pipeline re-design in the denoising block, where the longest combinatorial path is located.

4 Conclusion and Future Work

In this paper we have proposed a new method for video enhancement, designed for a real-time FPGA implementation. The designed FPGA processes the video sequence in real time with low memory consumption (no external memory resources are used). Based on the algorithm evaluation we concluded that the reduction of noise and ringing artifacts is satisfactory, while further algorithm improvement is needed for superior blocking artifact reduction.

References

1. Donoho, D.L.: De-noising by soft-thresholding. IEEE Trans. Information Theory 41, 613–627 (1995)
2. Chang, S., Yu, B., Vetterli, M.: Adaptive wavelet thresholding for image denoising and compression. IEEE Trans. on Image processing 9 (2000)
3. Pižurica, A., Philips, W., Lemahieu, I., Acheroy, M.: A joint inter- and intrascale statistical model for bayesian wavelet based image denoising. IEEE Trans. on Image processing 11, 545–557 (2002)
4. Balster, E., Zheng, Y., Ewing, R.: Feature-based wavelet shrinkage algorithm for image denoising. IEEE Transactions on Image Processing 14, 2024–2039 (2005)
5. Jostschulte, K., Amer, A., Schu, M., Schroder, H.: Perception adaptive temporal tv-noise reduction using contour preserving prefilter techniques. IEEE Trans. on Consumer Electronics 44, 1091–1096 (1998)
6. Pižurica, A., Zlokolica, V., Philips, W.: Noise reduction in video sequences using wavelet-domain and temporal filtering. In: SPIE Conference on Wavelet Applications in Industrial Processing, Providence, RI, USA, vol. 5266, pp. 48–59 (2003)
7. Zlokolica, V., Pižurica, A., Philips, W.: Wavelet-domain video denoising based on reliability measures. IEEE Trans. on Circuits and Systems for Video Technology 16, 993–1007 (2006)
8. De Haan, G.: Ic for motion-compensated de-interlacing, noise reduction and picture rate conversion. IEEE Trans. on Consumers Electronics 45, 617–623 (1999)

9. Katona, M., Pižurica, A., Teslic, N., Kovacevic, V., Philips, W.: A real-time wavelet-domain video denoising implementation in fpga. EURASIP Journal on Embedded Systems 2006, 1–12 (2006)
10. Donoho, D., Johnstone, I.: Adapting to unknown smoothness via wavelet shrinkage. Journal of American Statist. Assoc. 90, 1200–1224 (1995)
11. Sendur, L., Selesnick, I.: Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependences. IEEE Trans. on Image processing 50, 2744–2756 (1999)
12. Selesnick, W.I., Li, K.Y.: Video denoising using 2d and 3d dual-tree complex wavelet transforms. In: Proc. SPIE on Wavelet Applications in Signal and Image Processing, San Diego, CS, USA, vol. 5207, pp. 607–618 (2003)
13. Mallat, S.: A wavelet tour of signal processing, 2nd edn. Academic Press, London (1999)
14. Wang, Z., Bovik, A.C., Sheik, H.R., Simoncelli, E.P.: Image quality assessment: From error measurement to structural similarity. IEEE Transactions on Image Processing 13 (2004)
15. Wang, Z., Sheik, H.R., Bovik, A.C.: No-reference perceptual quality assessment of jpeg compressed images. In: IEEE International Conference on Image Processing, Rochester, New York, USA (2002)
16. Chipit gold edition, Technical Features and Architecture (2006), `http://www.uchipit.com`
17. Virtex-ii platform fpga, Product Specification, DS031 (v3.4) (2005)
18. Katona, M., Teslic, N., Krajacevic, Z.: Fpga design with systemc. In: 10th International Conference Mixed Design of Integrated Circuits and Systems (MIXDES), Lódz, Poland (2003)