Stick Based Speckle Reduction for Real-Time Processing of OCT Images on an FPGA

H. Luecken, G. Tech, R. Schwann, G. Kappen

This paper presents an FPGA based real-time implementation of an adaptive speckle reduction algorithm. Applied to the log-compressed image of a high-resolution optical coherence tomography (OCT) system, all related signal processing steps from envelope detection to VGA video signal generation are executed on a single chip. Images from measured OCT data show that the chosen algorithm produces a smooth, detailed image with fewer image artifacts than comparable approaches. An estimation of the hardware effort, the possible throughput rate and the resulting image frame rate is given for different window sizes used here in speckle reduction.

Keywords: optical coherence tomography, real-time processing, speckle reduction, FPGA.

1 Introduction

Optical coherence tomography (OCT) is a rather new imaging technique based on broadband interferometry. Like ultrasonic imaging, OCT generates a cross-sectional image of the scanned tissue. The penetration depth of OCT ranges from about 1–2 mm in opaque tissue (e.g. skin) to 2 cm in transparent tissue (e.g. eye). The achievable lateral resolution mainly depends on the focusing optics, while the axial resolution is based on the light source of the OCT scanner. Currently, resolutions down to a few micrometers are possible, allowing OCT images to compete with histological examination of tissue. The main drawback of histology is that for each observation a tissue sample has to be taken surgically. Thus, the main advantages of OCT are its non-invasive nature and real-time feasibility.

Similar to ultrasonic images, OCT speckle arises from constructive and destructive interference of light backscattered from reflectors smaller than a wavelength. Speckle noise degrades contrast, and makes it difficult to identify small reflectors. Additionally, post-processing algorithms (such as deconvolution or segmentation) that are used to enhance image resolution and quality may suffer from the presence of speckle in OCT images. Thus, prior to image enhancement algorithms, a speckle reduced image is available that can be blended with the original image to enhance viewing convenience.

This paper is organized as follows: In section 2 the basic principles of OCT are described and the high resolution OCT system is presented in detail. Section 3 deals with the theory of speckle reduction algorithms and gives an overview of the algorithmic complexity. Based on these investigations, section 4 evaluates real time implementation feasibility on an FPGA.

2 High resolution OCT

This section gives details about the OCT system used throughout this work, and a flexible digital post processing realized on an FPGA is presented.

2.1 OCT principle

A general OCT system is depicted in Fig. 1. The light from a broadband light source is split into a reference beam and a probe beam directed onto the object of interest. By combining the backscattered light with the reference beam, an interference pattern is created and detected by a photodetector. The strength of the interference is dependent on the optical path difference and on the reflection. The envelope of the interference pattern shows the reflectivity in a certain depth of the probe [1].

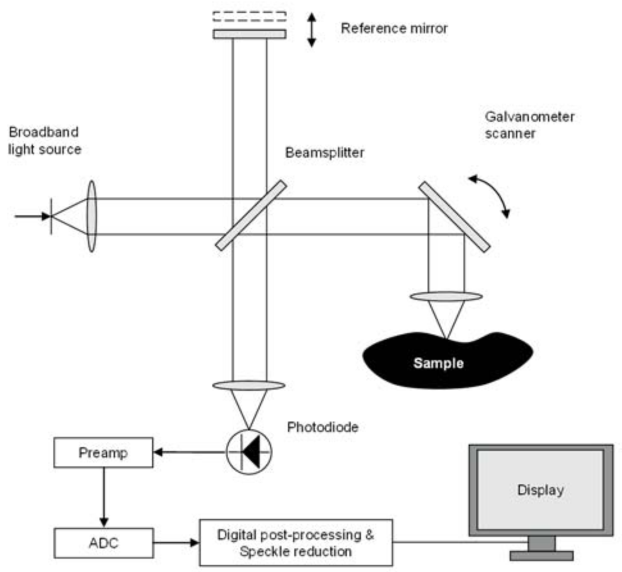

Fig. 1: OCT principle

Hence, by altering the length of the reference beam, a reflectivity profile of the probe can be recorded, called A Scan. The photodetector signal is amplified and analog-to-digital converted to enable digital filtering, demodulation and advanced post processing. By deflecting successively the probe beam in lateral direction, multiple scans can be combined to a 2-dimensional B-Scan, showing a high resolution cross-sectional view of the probe.

2.2 OCT system

Interference of the detected light depends on the difference of the optical length of the reference arm and probe arm. Unlike narrowband light sources, broadband light sources show interference only for small differences up to the coherence length. The applied source emits light with a spectral bandwidth of 255 nm at a center wavelength of 800 nm, which leads to a coherence length of 1.6 μm. The coherence length is the limit for the axial resolution of the system.

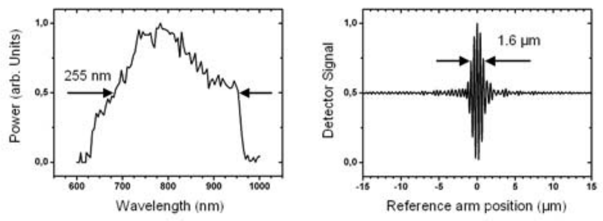

Fig. 2: Spectrum of the broadband light source and interference pattern of an ideal mirror

Resolution in the lateral direction of the OCT-System depends basically on the focusing optics and lenses. The system considered here provides up to 0.01 degrees of lateral resolution.

The length of the reference arm can be varied on a micrometer scale with a piezoelectric crystal. Adaptive control enables a linear movement of the reference mirror at constant speed. Therefore the recorded signal shows the interference pattern as a sine-wave carrier signal. The depth-dependent reflectivity of the probe shapes a superimposed envelope. The carrier frequency depends on the mirror velocity and center frequency of the light source, and is about 350 KHz for the considered system, mainly limited by the mechanical set-up of the reference mirror.

2.3 Digital post-processing

After amplifying the photodiode signal, it is sampled and converted to the digital domain. A standard analog-to-digital converter is used, providing precision of 12 Bits at a sampling frequency of 1 MHz. This data is fed to an Altera Stratix II FPGA, which serves as a test-bed to evaluate different speckle reduction algorithms and the real-time abilities of the system. Fig. 3 shows a block diagram of the architecture.

Standard signal processing for an OCT system consists of filtering and amplitude demodulation to determine the signal's envelope. The implemented system starts with a bandpass filter. Subsequently a Hilbert transformation is chosen to demodulate the signal by adding the imaginary part to

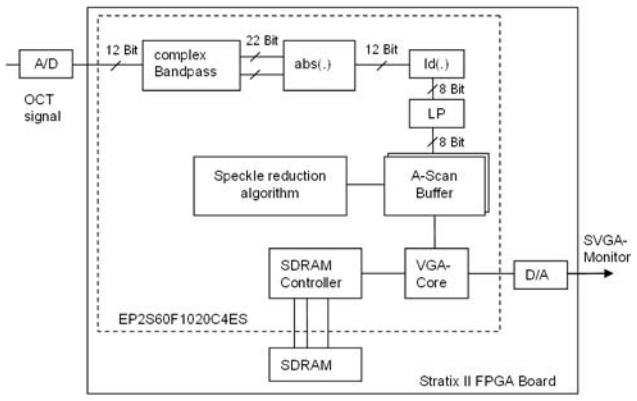

Fig. 3: Digital post-processing of OCT data

the OCT signal and taking the absolute value. The bandpass and Hilbert transform can be performed in one step, in Fig. 3 denoted as complex bandpass. To compute the absolute value, i.e. the squared sum of the imaginary and real part of the signal, an efficient coordinate rotation (CORDIC) algorithm has been implemented as proposed in [5].

Subsequent to envelope detection, the signal is compressed to 8 Bit by computing the logarithm. To adapt the sampling rate to the axial resolution given by the coherence length, down-sampling is done after low-pass filtering to avoid aliasing.

The demodulated and compressed OCT data of each scan line is written to a memory buffer, and the adjacent scans are combined in lateral direction to a B-Scan. A VGA protocol converter allows visualization of the OCT picture on a standard VGA display with a resolution of 1024×768 pixels at 75 Hz. The architecture makes use of a 2-stage memory structure (Table 1). For advanced processing and speckle reduction, up to 63 adjacent A Scans can be stored in a buffer and can be processed in both the axial and the lateral direction. A 32 MByte SDRAM is used as frame buffer by the video controller.

Table 1: Memory usage of OCT post-processing

Block	A-Scan Buffer	SDRAM
Memory Size	524,288 Bits	804,864 Bytes

The required Arithmetical Logical Units (ALUTs) for an FPGA implementation of this post processing scheme are summarized in Table 2.

Table 2: Resource requirements of basic OCT post-processing blocks

Block	Band -pass	Abs	Log	LP	VGA	SDRAM-Controller	Σ
ALUTs	936	469	109	763	1562	429	4268

As can be seen, only 9 % of the overall FPGA (48,352 ALUTs) are used in this post processing approach, leaving sufficient space for extension and enhancements.

3 Speckle reduction algorithm

This section details with the speckle reduction technique considered here.

Speckle reduction aims to reduce noise-like speckle distortion in B-Scans, and it should also generally preserve structures like edges, borders and small features of interest. A large variety of speckle reducing filters has been published, particularly from research in ultrasonic image processing. Approaches using region growing, diffusion and wavelet decomposition have been proposed.

This paper considers the adaptive speckle filter introduced in [3] using asymmetric sticks, called ASF. Similar to other speckle reduction techniques, the basic concept is to compute the output from windows which are adapted to the local content of the source image. In contrast to algorithms working with rectangular windows and/or weights [6, 7, 8]

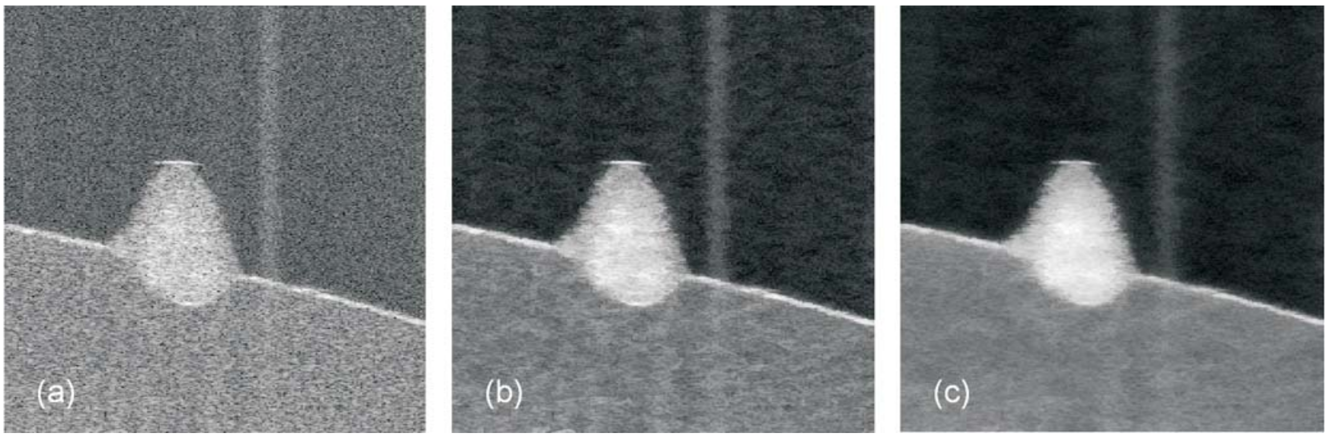

Fig. 4: High resolution OCT image of a spheroid: (a) Original image, (b) Asymmetric stick filter (11×11 window), (c) Asymmetric stick filter (21×21)

which depend on the distance to the center of the window, ASF resolves the window in angular steps. This method shows superior results at the expense of high computational load and is impracticable to implement on a general purpose processor for real-time processing. The algorithm will now be explained in detail.

The filter uses directional features (called sticks, Fig. 5) and allows iterative application, since it behaves like a non-linear diffusion model. The method is based on a weighted sum of means, which are calculated in the stick direction. The weights are given by the reciprocal of the local variance in each direction.

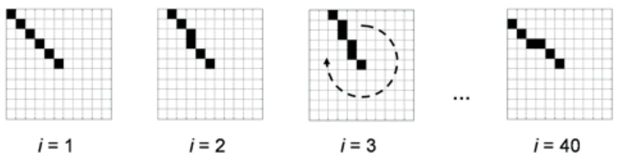

Fig. 5: Kernels $H_i(x,y)$ showing sticks for a window size $N = 11$

According to [3], the filter algorithm is given by

$$\hat{I} = \frac{1}{W}\sum_{i=1}^{4N-4} g(V_i)\bar{I}_i$$

and

$$W = \sum_{i=1}^{4N-4} g(V_i)\,,$$

where $\hat{I}$ denotes the output for each pixel. V_i and $\bar{I}_i$ are the variance and mean of the original image data I along the i^{th} stick pointing from the center of the $N\times N$ filter window in one of $4N-4$ different directions. The sticks can be constructed by hardware, making efficient use of Bresenham's line algorithm [9]. Given the i^{th} stick as a filter kernel $H_i(x, y)$, the mean and variance are computed by

$$\bar{I}_i(x, y) = I_i(x, y) * H_i(x, y)$$

and

$$V_i(x, y) = \left[I^2(x, y) * H_i(x, y)\right] - \bar{I}_i^2$$

with * denoting a 2-dimensional correlation. As variation function $g(\cdot)$ this paper applies

$$g(V_i) = \begin{cases} k\dfrac{255}{V_i} & \text{if } V_i \ge k \\ 255 & \text{else.} \end{cases}$$

Sticks across an edge show high variance and therefore the mean in this direction is weighted to have less influence on the computed output. Hence, smoothing only happens in homogeneous directions whereas it is suppressed across edges. In homogeneous speckle regions, all sticks have approximately the same variance and weight, and as a result all directions are smoothed evenly.

Because of the high quantization of angular steps, ASF is able to adapt its kernel very flexibly to the image content. Inhomogeneities with various shapes can not only be retained, but can be smoothed without blurring edges. Fig. 4 shows a measured OCT image of a spheroid placed on an agar substrate and the enhanced images for two different stick lengths. The speckle pattern and additive noise are significantly reduced.

In [3], it is stated that due to the overlapping sticks the algorithm works like a Gaussian smoothing filter in homogeneous regions. However, a closer look at the filter kernel in these regions shows that the weights are

$$w(r) = \begin{cases} c\cdot(4N-4) & \text{for } r = 0 \\ c\cdot\dfrac{4N-4}{8r} & \text{else.} \end{cases}$$

(Assuming V_i equal for all directions i, with r denoting the number of stick pixels from the window center and a constant c)

The kernel is not Gaussian, but decreases with $1/(8r)$. For better smoothing, the influence of the center pixel can be reduced. A straightforward way to do this is to subtract a fraction of the original pixel value from the output

$$\hat{I}' = (1+\alpha)\hat{I} - \alpha I\,.$$

An empirically found value of α is 0.125. This adjustment is especially important if only one iteration is applied.

To compare the image quality of ASF to an existing FPGA implementation of the ARGF-based [8] algorithm by Mazumdar et al. [4], both filters have been implemented in C++ and Matlab and adjusted to OCT-image data. Fig. 6 shows the resulting images.

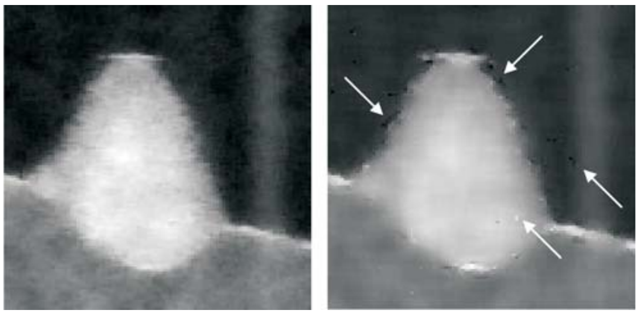
Fig. 6: Comparison of speckle reduction algorithms (left Asymmetric Stick Filter, right ARGF-based filter)

The right image is slightly degraded by artifacts (marked with arrows). These are caused by hard switching between square filter kernels and outliers found in speckle statistics. They do not appear in the left image, because of the soft thresholding and high angular quantization of ASF. Furthermore, ASF needs only one parameter k, which can be adjusted very easily. The filter introduced in [4] needs 4 parameters, which are difficult to adjust empirically.

4 Real-time implementation

This section describes the real-time FPGA implementation of the ASF algorithm introduced in section 3. This work focuses on meeting real-time requirements. Therefore, the timing requirements for the ASF algorithm are derived based on the A-Scan rate and pixels per A-Scan. Subsequently the architecture of the ASF algorithm will be shown and the throughput rate for a serial and a parallel implementation will be estimated.

In the second part, implementation results are presented for a Stratix II FPGA for different sets of parameters of the algorithm. Here, the dependency of different stick-lengths corresponding to varying window sizes as well as the number of sticks on the required hardware will be shown, measured in Logic Elements (LEs). A possible parallel execution of the ASF algorithm to achieve highest throughput rate will be described.

4.1 Signal processing concept

Fig. 7 shows the signal processing concept of the ASF algorithm. As can be seen, the algorithm is divided into three blocks that realize different steps of the algorithm at different sample rates. This architecture allows pipelining of the algorithm to increase the throughput rate. In Fig. 7, the first block receives a window of $N \times N$ pixels and calculates the sum along each stick and the sum of the squared elements for N_S sticks. Because the $N \times N$ window pixels are prefetched and stored in an array, they are available without latency.

In each clock cycle, a serial implementation performs one of the N_S summations over N_L stick elements. This leads to a required number of N_S processing cycles for the first block and thus one summation value is calculated each clock cycle. In a parallel implementation, all these summations are calculated simultaneously leading to a calculation time of one clock cycle for all summation results.

The processing cycles of the second block are mainly affected by the summation over N_S sticks, leading also to a number of about N_S cycles, and hence one calculated value per clock cycle. Here, the parallel implementation calculates all results in one clock cycle.

Finally, block III realizes a divider which does not affect the throughput rate of this pipelined architecture.

The number of processing cycles required to perform the ASF algorithm realized by the three-block approach considered here is equal to the maximum number of processing cycles of block I and II. Thus, a serial implementation requires about N_S processing cycles if the pipeline has been filled up. In contrast to this, a fully parallel implementation requires three clock cycles to determine a speckle reduced pixel value.

Based on the A-Scan rate and the number of pixels per A-Scan, the upper limit of processing cycles to calculate a new pixel value can be derived from

$$\mathrm{cyc} = \frac{f_{\mathrm{clk}}}{f_{\text{A-Scan}} \cdot l},$$

where l is the A-Scan length (in pixels), f_{clk} and $f_{\text{A-Scan}}$ are the clock frequency and the A-Scan rate in Hz respectively. With an A-Scan rate of 50 Hz and an A-Scan length of 768 pixels, approximately 2600 cycles are acceptable for the calculation of each speckle reduced pixel. As can be seen from the processing cycle calculation above, serial as well as parallel ASF implementations easily meet this requirement for reasonable window sizes.

4.2 FPGA implementation

Fig. 8 shows the number of required ALUTs for a serial implementation of the ASF algorithm. The increasing number of ALUTs is mainly caused by the first block. As can be seen in Fig. 8, the bit widths of the second and third block's input

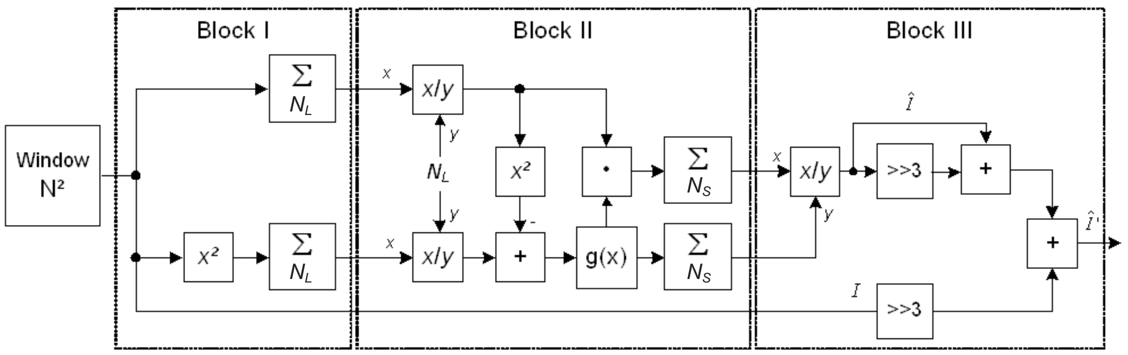

Fig. 7: Signal processing architecture of the speckle reduction algorithm

parameters are just slightly increasing by ld(N_L) and ld(N_S) respectively, leading to nearly constant hardware resources for these blocks. The number of required ALUTs of the first block is approximately a quadratic function of N.

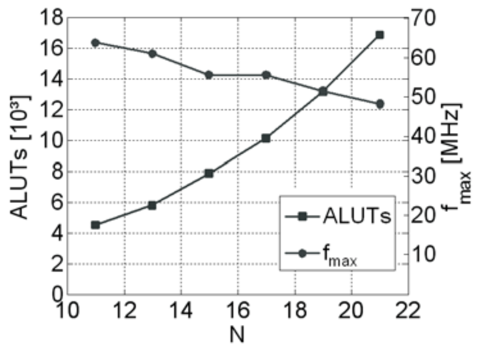

Fig. 8: Resource usage and maximum clock frequency of ASF as a function of window size

Furthermore, the maximum clock frequency for an implementation on the Stratix II FPGA (EP2S60F1020C4) is given in Fig. 8. As can be seen for reasonable values of N, the maximum clock frequency is greater than 50 MHz.

4.3 Performance evaluation

To determine the maximum window size for real time calculation, the equation for the upper limit of available cycles per pixel in section 4.1 can be rewritten. For a stick length of $N_L = (N+1)/2$, using the minimum number of required cycles $N_S = 4N - 4$ for the serial implementation yields:

$$N = \frac{f_{\text{clk}}}{4l \cdot w \cdot f_{\text{img}}} + 1\,.$$

This form allows a calculation of the window size depending on A-Scan length (l), image rate (f_{img}) and width (w) as well as clock frequency (f_{clk}).

For a second available OCT system based on Fourier-domain image reconstruction, an A-Scan length of 1000 pixels, an image rate of 5 Hz and a width of 250 lines is required. In this case, a window size of $N = 11$ can be processed in real-time.

5 Conclusion

A generic design approach allows for a flexible implementation of the ASF algorithm. Thereby, algorithm parameters like window size, number of sticks, and weighting function can be configured at compile time. The performance of the serial implementation meets the requirements of the broadband OCT system used here.

Even for FFT-based fast OCT systems, the presented hardware implementation can apply real-time speckle reduction to the full image size. For even higher demands, parallelization of the algorithm is straightforward using the proposed generic design approach. Thus, increased throughput rate can be traded for hardware effort regarding the number of required ALUTs.

The introduction of a slight modification of the algorithm shows that a single run provides sufficient smoothing of the image.

Acknowledgments

Parts of this work were funded by the Deutsche Forschungsgesellschaft (DFG). The work was supported by Felix Spöler and the Institute of Semiconductor Electronics at RWTH Aachen University.

The authors would like to thank Prof. Tobias G. Noll for supervising this work at the Chair of Electrical Engineering and Computer Systems, RWTH Aachen University.

References

[1] Schmitt, J. M.: Optical Coherence Tomography (OCT): A Review, In *IEEE Journal of Selected Topics in Quantum Electronics*, Vol. **5** (1999), No. 4, July/August 1999.

[2] Schmitt, J. M., Xiang, S. H., Yung, K. M.: Speckle in Optical Coherence Tomography, In *Journal of Biomedical Optics*, Vol. **4** (1999), No. 1, January 1999.

[3] Xiao, C. Y., Zhang, S., Chen, Y. Z.: A Diffusion Stick Method for Speckle Suppression in Ultrasonic Images, In *Pattern Recognition Letters*, 2004, No. 25.

[4] Mazumdar, B., Mediratta, A., Bhattacharyya, J., Banerjee, S.: A Real Time Speckle Noise Cleaning Filter for Ultrasound Images, In *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems*, 2006.

[5] Schwann, R., Kappen, G.: Cordic Based Postprocessing of Ultrasound Beamformer Data, In *Proceedings 9th International Student Conference on Electrical Engineering* "POSTER 2005", 2005.

[6] Loupas, T., McDicken, W. N., Allan, P. L.: Adaptive Weighted Median Filter for Speckle Suppression in Medical Ultrasonic Images, In *IEEE Transactions on Circuits and Systems*, Vol. **36** (1989), p. 129–135.

[7] Koo, J. I., Park, S. B.: Speckle Reduction with Edge Preservation in Medical Ultrasonic Images Using a Homogeneous Region Growing Mean Filter (HRGMF), In *Ultrasonic Imaging*, Vol. **13** (1991), p. 211–237.

[8] Chen, Y., Yin, R., Flynn, P., Broschat, S.: Aggressive Region Growing for Speckle Reduction in Ultrasound Images. In *Pattern Recogn. Lett*. Vol. **24** (Feb. 2003), No. 4–5, p. 677–691.

[9] Bresenham, J.: Algorithm for Computer Control of a Digital Plotter, In *IBM Systems Journal*, Vol. **4** (1965), No. 1, p. 25–30.

Heinrich Luecken

Gerhard Tech

Dipl.-Ing. Robert Schwann
e-mail: schwann@eecs.rwth-aachen.de

Dipl.-Ing. Götz Kappen
e-mail: kappen@eecs.rwth-aachen.de

Chair of Electrical Engineering and Computer Systems
RWTH Aachen University
Schinkelstrasse 2
Aachen, Germany