A SYSTEMATIC APPROACH TO REAL-TIME IMAGE SEGMENTATION IN FPGA DEVICES

Franci Kopač, Andrej Trost

Faculty of Electrical Engineering, University of Ljubljana, Ljubljana, Slovenia

Key words: image processing, image segmentation, FPGA

Abstract: Image segmentation is an important step in image processing and recognition. Processing in real time introduces additional demands, which constrain our ability to choose between segmentation techniques and hardware architectures. The existing literature only describes very specialized image segmentation solutions. This paper introduces a systematic approach to processing block design, which facilitates the selection of a suitable segmentation technique and its adaptation to an FPGA implementation. Research into the published segmentation techniques and existing FPGA solutions enabled us to propose a system architecture suitable for effective implementation of most segmentation techniques. We also present a methodology to search an optimal technique for real time segmentation. The proposed approach has the advantage that it is not constrained to a specific image segmentation area. This wide design space enables us to use the methodology to find an optimal solution for each specific implementation example.

Sistematični pristop k segmentaciji slike v realnem času z vezji FPGA

Kjučne besede: obdelava slik, segmentacija slike, FPGA

Izvleček: Segmentacija slike je pomemben korak pri postopkih avtomatske obdelave in razpoznavanja slike. Pri obdelavi slik v realnem času imamo dodatne zahteve, ki močno zožijo izbor pomembnih tehnik segmentacije in strojne arhitekture. V obstoječi literaturi najdemo rešitve za segmentacijo slike v realnem času za ozko določena področja uporabe. V članku je predstavljen sistematičen pristop k načrtovanju procesnih blokov, ki olajšuje izbiro segmentacijske tehnike in njeno prilagajanje implementaciji z FPGA vezji. Na osnovi pregleda segmentacijskih tehnik in obstoječih rešitev v FPGA vezjih smo določili arhitekturo sistema, ki omogoča učinkovito implementacijo večine segmentacijskih tehnik. Predstavljena je metodologija iskanja optimalne tehnike za izvedbo segmentacije v realnem času. Prednost prikazanega pristopa je v tem, da ni omejen na specifično področje uporabe segmentacije. Ker imamo širok načrtovalski prostor, lahko z uporabo metodologije iskanja pridemo do optimalne rešitve za vsak konkreten primer uporabe.

1 Introduction

Image segmentation has been around for about 40 years and can be considered a relatively mature part of the computer vision. A multitude of segmentation techniques exists, suitable for everything from handwriting recognition to automated vehicle guidance. But there is a rift between research and practical use: the more sophisticated techniques seem to be rarely used in real life /1/. This results in very strict requirements for the imaging part of computer vision systems and limits introduction of computer vision to everyday life.

While several problems are to blame for this situation, one seems to be crucial: advanced image segmentation techniques require impractical amounts of conventional computing power to produce results on traditional CPUs in real time - they have reached the limits of the Von Neumann bottleneck /2/. Designing in the domain of FPGA circuits provides a way to break this barrier /3/: most segmentation techniques can effectively be parallelized in hardware.

A systematic approach to implementation of the image segmentation techniques on FPGA systems does not yet exist. The majority of existing papers only deals with implementation of a particular image segmentation technique as a tailored part of the system (e.g. autonomous robot), and only relatively simple segmentation techniques are used (see /4/, /5/, /6/). Furthermore, these implementations usually aren't reusable. We therefore propose a systematic approach which represents a framework for future implementations and takes into account existing approaches and hardware/software co-design methodologies.

2 Classification of image segmentation techniques

2.1 General overview

Table 1 presents an overview of the segmentation techniques divided into classes.

The technique classes are not sharply delineated and very frequently, customized combinations of techniques are used to achieve best performance for a given application. The presented overview shows:

- image segmentation techniques have pronounced advantages and drawbacks,
- most technique classes already have been efficiently mapped to FPGA devices.

This work was supported by Ministry of Education, Science, and Sport of Republic of Slovenia as part of research program P2-0246 "Algorithms and Optimizations in Telecommunications".

Table 1: Overview of image segmentation techniques

Segmentation technique class	**Advantages**	**Drawbacks**	**Applications**	**Efficient FPGA implementations exist**
Global thresholding	very fast, lowest resource consumption	sensitive to uneven illumination, limited application area, no spatial information considered, sensitive to noise	document imaging, fingerprint recognition	yes
Local thresholding	fast, low resource consumption, less sensitive to uneven illumination	limited application area, sensitive to noise	document imaging, fingerprint recognition	yes
Watersheds	correct placement of boundaries, fast	usually requires subsequent region merging step to reduce over-segmentation, sensitive to noise	medical imaging, face detection, content-based image retrieval (CBIR), surveillance	yes
Region growing	tolerant to noise, connected edges, custom similarity criteria can be used	computationally expensive, sensitive to seed selection	medical imaging, image compression, motion estimation, CBIR, face detection	yes
Split-and-merge	segments objects on different scales (pyramid structure)	requires post-processing, slow convergence, computationally expensive	range images, aerial photographs, radiography	no
Clustering	custom similarity criteria can be used, can detect small variations	sensitive to noise, no spatial information considered, user must specify number of clusters in advance, computationally expensive	range images, satellite images, medical imaging, CBIR, robot vision, non-destructive testing (NDT), scene analysis	yes
Model based	tolerant to noise, takes into account a-priori information	very computationally expensive	medical imaging	no
Statistical	extremely general, tolerant to noise	very computationally expensive	CBIR, medical imaging, road segmentation, face recognition, robot vision, satellite imaging, NDT	yes
Neural network based	fast, tolerant to noise	large sample set required for training, computationally expensive	medical imaging, CBIR, NDT, road segmentation, face recognition, robot vision	yes
Fuzzy set based	tolerant to noise	very computationally expensive	medical imaging, CBIR, road segmentation, NDT, robot vision	yes
Physics based	proper segmentation of highlights	very computationally expensive, restricted set of materials	satellite images, visual inspection	no

2.2 Existing classifications

The papers describing the image techniques usually lack comprehensive comparisons with other techniques, especially regarding the execution efficiency. Only few papers offer direct comparison and classification of image segmentation techniques:

In an early comparison /7/, desirable characteristics of segmentation are defined: uniformity and homogeneity of regions, simple regions without many holes, clear differences between adjacent regions and spatially accurate, non-ragged boundaries.

In /8/, techniques are classified and compared. Objective evaluation of segmentation techniques is discussed and some general criteria for quality of segmentation (correlation, uniformity and entropy) are given.

In /9/, color representations are discussed, with emphasis on perceptually uniform color spaces. The segmentation techniques are divided into feature-space (clustering and thresholding approaches), image-domain (split-merge, region growing and edge based approaches), neural network classification, and physics based techniques (using reflection models), with discussion on individual techniques in the different classes.

In /10/, color segmentation techniques (including texture segmentation) are overviewed and pre- and post-processing is discussed. An array of techniques is then implemented in software, tested on natural images, and carefully compared.

We propose the creation of a classification of image segmentation techniques, where the performance of these techniques for different application areas is evaluated. This will allow the designer to objectively compare different techniques and choose the one most suitable for his application.

2.2.1 Technique comparison

The scope of usability of a technique is usually already defined by the author of the technique. Detailed comparison should therefore be limited to techniques that are a priori suitable for a certain application area, using a performance metric to make an objective comparison.

In /11/, the evaluation methods for image segmentation techniques are divided into two types: the analytical and empirical methods. The analytical methods are not very common, due to the lack of formal image segmentation theory, so only few examples are given. The empirical methods are divided into goodness and discrepancy methods. The goodness methods evaluate desirable properties of segmentation technique, while the discrepancy methods compare the actual segmentation results with "ideal" segmentation, usually defined by a human. Different empirical methods are then compared and the discrepancy methods are found to be more powerful.

We propose that discrepancy metrics, customized for different application areas, are used to compare the segmentation techniques. After evaluating a particular segmentation technique, its strong and weak points should be clearly outlined. This will enable the designer to select the optimal technique for the proposed application.

3 Processing structure of segmentation techniques

Most image segmentation techniques can be divided into three main steps: preprocessing, segmentation and post-processing (see Figure 1). While the segmentation steps usually differ, many techniques share the same pre- and post-processing steps.

A typical preprocessing step is Gaussian filtering of the image to reduce the impact of noise or to construct different scale spaces of the image. Similar 2D convolution process is also used for edge detection (Laplacian, Sobel) or texture filtering. This is a time consuming task on architectures with a single processor, but also one that can be accelerated using an FPGA /12/.

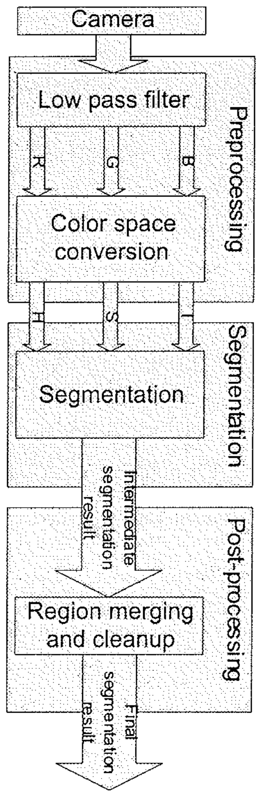

Figure 1: Image segmentation processing steps

Another common preprocessing step is color space transformation. The normal RGB color space is not optimal for color image segmentation and many features have much better contrast if a proper color space is used. This is a time consuming operation involving matrix multiplication on every pixel, especially when performed twice (e.g. RGB to HSI and back). It can be effectively parallelized (each pixel can be processed independently) and is a good candidate for FPGA implementation. An interesting implementation can be found in /13/, where the trigonometric functions were replaced with simple arithmetic.

Post-processing tasks include morphological operations, region merging or removal of unwanted segments. Efficient mapping of morphological operations in FPGAs has been demonstrated in /14/ and region merging is being looked into /15/. Nevertheless, post-processing tasks are not always well suited to FPGA implementation and are usually implemented in software, facilitated by the reduced amount of data due to preceding segmentation.

3.1 Architecture of an image segmentation block

An analysis of the preprocessing→segmentation→post-processing workflow enables us propose a general architecture of image segmentation blocks.

The preprocessing steps are suitable for implementation with FPGA devices. The segmentation process is usually less regular and the use of a processor with an additional FPGA coprocessor is needed to implement the segmentation algorithm effectively. The FPGA coprocessor computes the regular computationally expensive parts of the algorithm, while the processor processes these results on a higher level. The same combination can also be used for post-processing tasks, since they usually aren't suitable for a pure FPGA implementation.

The proposed architecture consists of a preprocessing FPGA block, processor and an FPGA coprocessor with external image and segmentation data buffers, as shown in Figure 2. The link between the processor and the FPGA coprocessor can be implemented in different ways, as custom instructions (especially usable for soft-core processors), as a coprocessor or through a dedicated bus. The FPGA devices containing hard-core processors (Xilinx Virtex II Pro) or soft-core processors (Altera Nios) are most suitable for this implementation.

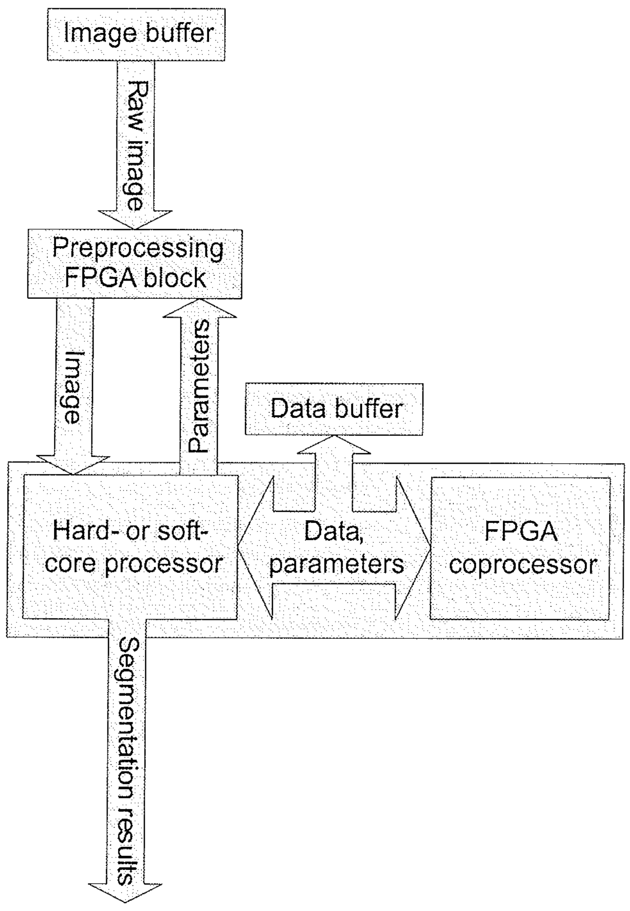

Figure 2: General image segmentation block architecture

4 Proposed design flow

An integral part of the proposed systematic approach to image segmentation is the image segmentation technique implementation workflow. It consists of two main parts, technique selection (using the classification of image segmentation techniques) and adaptation of technique to make it suitable for implementation on an FPGA-based system.

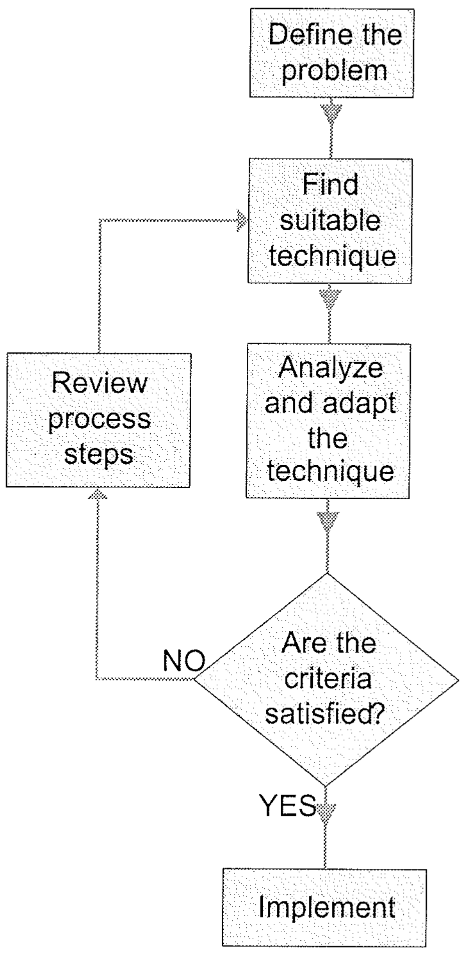

Figure 3: Design flow

4.1 Technique selection

The definition of the problem consists of the definition of the input image or video properties (source of image, expected image content, resolution, frame rate, grayscale or color, color space, amount and type of noise, etc.) and the definition of the environment, performance criteria and goals of segmentation (static or temporal segmentation, well defined or natural objects, stable or variable lighting, expected number of objects, tolerance of erroneous segmentation results, time to process one image, real-time requirements, etc.). The hardware must be defined in terms of resources that are available to the image segmentation block (architecture, free processor cycles, free memory, free FPGA blocks, and power limitations).

Once the properties of the desired system are known, classification of image segmentation techniques is used to determine the adequate technique. The application is matched to one of the application types in the classification. The recommended techniques are compared and the most suitable is chosen. Next, the preprocessing steps to make it compatible with input image properties must be defined. This can include filtering and/or color space conversion. Also, any necessary post-processing should be defined (e.g. morphological operations, etc.)

4.2 Adaptation

Once the general structure of the segmentation block is defined, adaptation of the segmentation block for FPGA implementation must be performed, using the following criteria:

- numerical precision,
- complexity of mathematical operations,
- parallelization,
- data throughput and memory usage.

4.2.1 Numerical precision

Floating point algorithms are generally not suitable for FPGA implementation, due to their increased complexity and silicon footprint. Usually, the image segmentation techniques are robust enough to convert them to fixed point computation without major changes. If this is not possible, the technique must be adapted to fixed point precision.

4.2.2 Complex mathematical operations

Complex arithmetic operations can be defined as operations that are not easily broken down into the basic arithmetic operations already built-in or easily implemented in FPGA-s: addition, subtraction and multiplication. Some examples are: division, roots, trigonometric functions and logarithms.

These operations are generally hard to implement efficiently and precisely at the same time. To implement them, approximations /13/ or look-up tables /16/ must be used and a careful analysis of the tradeoff between needed precision and allowable complexity must be completed. A better approach is to simplify the operations by reorganizing data structures and by mathematical simplification of required operations.

4.2.3 Parallelization

FPGA implementations usually depend on parallelism. In contrast to processors, where one task of essentially unlimited complexity can be processed at a time, FPGA devices are better suited to parallel execution of many simpler tasks.

Most image processing and segmentation techniques are of a local nature i.e. they work on one or a small range of pixels at a time. An array of parallel processing units can be used to accelerate computation by working on many pixels at a time. If this is not possible, other techniques to exploit parallelism exist, e.g. loop unrolling.

4.2.4 Data throughput and memory usage

FPGA devices have a small amount of fast internal RAM, which is not large enough to buffer the whole image or segmentation results. An effective implementation should minimize accesses to external RAM, reusing the data already transferred into the internal memory as much as possible, while using the smallest practical amount of the internal memory. This can be achieved by data structures reorganization and pipelining. Careful buffering will keep the time spent waiting for data to a minimum. The local processing techniques are advantageous in this respect.

4.3 Evaluation

After the adaptation process, the proposed image segmentation block should be evaluated using design-specific criteria like real-time performance, silicon area, power consumption etc. If the criteria are not fulfilled, processing steps should be redefined and adaptation repeated. If this is not enough, a different segmentation technique must be chosen or the problem must be redefined.

5 Case study

For our short case study, we decided to analyze, adapt and implement the image binarization technique described in /17/.

5.1.1 Analysis

The technique consists of 9 processing steps in the following order: 5x5 mean smoothing of the original 8-bit grayscale image, activity calculation, Laplacian calculation, label image generation, label image processing, label image thresholding, 3x3 mean smoothing, gradient magnitude calculation and removal of false print pixels.

An analysis of these steps shows that six steps are good candidates for hardware implementation: the 5x5 mean smoothing, the 3x3 mean smoothing, the gradient magnitude calculation, the Laplacian calculation, the label image generation and the Sobel filtering step. These steps are suitable for hardware implementation (fixed point precision, local processing) and can work concurrently, while the remaining tasks are better suited for software implementation. The mathematical operations are limited to addition and constant multiplication. The local nature of processing enables reuse of data in internal buffers, reducing the data throughput.

5.1.2 Adaptation

We can adapt the mean smoothing blocks by using simple shifting instead of multiplication. The errors thus introduced are constant (21.9% for 5x5 and 43.8% for 3x3 mean smoothing) and can be entirely compensated for by changing the two thresholds used by the technique. Multiplier stages are no longer needed, at the cost of slightly reduced dynamic range. The Laplacian calculation and Sobel filtering hardware blocks require only multiplication by powers of two, also enabling us to use shifting instead of multipliers.

Furthermore, the blocks can be grouped into stages, where two or more blocks can work in parallel: First, 5x5 and 3x3 mean smoothing blocks process the input image. Second, activity, Laplacian and label image calculation is performed on 5x5 mean smoothed image, while the last stage performs Sobel filtering on 3x3 mean smoothed image.

5.1.3 Implementation

We decided to implement the hardware blocks on an Altera Apex prototype board with an Apex EP20K200EFC484-2X device. The implementation used a 100 x 100 pixel image, stored in external SRAM memory, and a state machine to read pixel data, compute the results of the processing steps and write them back to the SRAM memory. The design was done in VHDL in Altera Quartus II 4.1 IDE.

The external SRAM memory represents a bottleneck, because data can only be read one pixel at a time. We therefore divided the processing into a pipeline consisting of three stages: First 27 states of the state machine were used to concurrently compute the 5x5 and 3x3 mean smoothing results for the given pixel and store them into SRAM (25 pixels read, 2 written). Next 19 states were used to concurrently compute the activity, Laplacian and label image for the given pixel of the 5x5 smoothed image and store the label image in the SRAM memory (18 pixels read, 1 written). The final 10 states were used to compute Sobel filtered result of the given pixel of the 3x3 mean smoothed image and write it into SRAM (9 pixels read, 1 written). When processing reaches the last pixel of the input image, it resumes at the first pixel, refreshing the results regularly. The pipeline design means that three passes through the data are needed to obtain all the results for the first frame and afterwards each subsequent pass returns the results for subsequent frames.

The worst case propagation delay in the design amounts to 146 ns, SRAM read and write delays included, allowing an 6.8 MHz system clock, which results in a refresh rate of about 12 fps. The design uses 3300 logic elements (39% of the device) and 6144 memory bits (5% of the device). There are enough resources remaining to enable us to include a Nios soft core processor into the design to perform the remaining processing tasks.

This implementation, while not optimized for performance, is intended as a demonstration of the second part of the image segmentation implementation design flow, outlining the basic tasks needed to implement image segmentation in FPGA devices. It shows that analysis and adaptation of the technique in accordance with the proposed guidelines are vital for a successful implementation, enabling us to save silicon and improve performance.

6 Future work and conclusion

We have shown that a systematic approach to image segmentation on modern FPGA platforms is needed. To make the proposed systematic approach to image segmentation practically useful, we plan to research a comprehensive overview of different types of segmentation techniques regarding their performance for particular application types.

Our proposal consists of an image segmentation technique comparison, criteria for adaptation of techniques for implementation on FPGA devices, an architecture for image segmentation blocks and a workflow to efficiently select and adapt a suitable technique for image segmentation on an FPGA device.

The approach was illustrated by analyzing, adapting and implementing a sample image segmentation technique in accordance with the given criteria, resulting in reduced implementation complexity and the ability to perform some tasks in parallel.

7 References

/1/ M. Turk, "Computer vision in the interface", Communications of the ACM, Volume 47, 60-64, 2004

/2/ R. Hartenstein, "The digital divide of computing", Proceedings of the first conference on computing frontiers, Ischia, Italy, 357 - 362, 2004

/3/ B. A. Draper, J. R. Beveridge, A. P. W. Bohm, C. Ross, M. Chawathe, "Accelerated image processing on FPGAs", IEEE Transactions on Image Processing, 1543- 1551, 2003

/4/ T. Morie, T. Nakano, "A face/object recognition system using coarse region segmentation and dynamic-link matching", International Symposium on Bio-Inspired Systems, Part IV, Brain-inspired Information Technology (BrainIT 2004), Kitakyushu, 2004.

/5/ V. Bonato, A. K. Sanches, M. Fernandes, J. M. P. Cardoso, E. Simões, E. Marques, "A Real Time Gesture Recognition System for Mobile Robots", International Conference on Informatics in Control, Automation, and Robotics (ICINCO'04), 207-214, 2004

/6/ E. Ashari, R. Hornsey, "FPGA Implementation of Real-Time Adaptive Image Thresholding", Photonics North 2004, 2004

/7/ R. Haralick, L. Shapiro, "Survey: Image segmentation techniques", Computer Vision, Graphics and Image Processing 29, 100-132, 1985

/8/ N.R. Pal, S.K. Pal, "A review of image segmentation techniques", Pattern Recognition, Vol. 26, 1277-1294, 1993

/9/ L. Lucchese, S.K. Mitra, "Color Image Segmentation: A State-of-the-Art Survey", Image Processing, Vision, and Pattern Recognition, Proc. of the Indian National Science Academy, Vol. 67, A, No. 2, 207-221, 2001

/10/ M. Sharma, "Performance Evaluation of Image Segmentation and Texture Extraction Methods in Scene Analysis", M.S. Thesis, University of Exeter, 2001

/11/ Y. J. Zhang, "A Survey on Evaluation Methods for Image Segmentation", Pattern Recognition, Vol. 29, No. 8, 1335-1346, 1996

/12/ S. Perri, M. Lanuzza, P. Corsonello, G. Cocorullo, "SIMD 2-D Convolver for Fast FPGA-based Image and Video Processors", Proceedings of the Military and Aerospace Programmable Logic Devices (MAPLD) International Conference, Washington, USA, 2003

/13/ V. Bonato, A. K. Sanches, M. Fernandes, J. M. P. Cardoso, E. Simões, E. Marques, "A Real Time Gesture Recognition System for Mobile Robots," International Conference on Informatics in Control, Automation, and Robotics (ICINCO'04), INSTICC, 207-214, 2004

/14/ K. S. Hemmert, B. L. Hutchings, A. Malvi, "An Application-Specific Compiler for High-Speed Binary Image Morphology", Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines, 2001.

/15/ J. D. Kranthi Kumar, S. Srinivasan, "A Novel VLSI Architecture to Implement Region Merging Algorithm for Image Segmentation", Euromicro Symposium on Digital System Design (DSD'04), 620-623, 2004

/16/ P. Hung, H. Fahmy, O. Mencer, M. J. Flynn, "Fast Division Algorithm with a Small Lookup Table", Asilomar Conference on Signals, Systems, and Computers, 1999

/17/ O.D. Trier, T. Taxt. "Improvement of Integrated Function Algorithm for Binarization of Document Images", PRL, Vol. 16, No. 3, 277-283, 1995

univ. dipl. inž. Franci Kopač
Univerza v Ljubljani, Fakulteta za elektrotehniko
Tržaška cesta 25, 1000 Ljubljana
Tel.: 01 4768351, Faks: 01 4264630
E-mail: franci.kopac@fe.uni-lj.si

doc. dr. Andrej Trost
Univerza v Ljubljani, Fakulteta za elektrotehniko
Tržaška cesta 25, 1000 Ljubljana
Tel.: 01 4768350, Faks: 01 4264630
E-mail: andrej.trost@fe.uni-lj.si

Prispelo (Arrived): 17.01.2005 *Sprejeto (Accepted): 15.03.2005*