Tomasz Kryjak*, Marek Gorgoń**

REAL-TIME IMPLEMENTATION OF MOVING OBJECT DETECTION IN VIDEO SURVEILLANCE SYSTEMS USING FPGA

The article presents the concept of real-time implementation computing tasks in video surveillance systems. A pipeline implementation of a multimodal background generation algorithm for colour video stream and a moving objects segmentation based on brightness, colour and textural information in reconfigurable resources of FPGA device is described. System architecture, resource usage and segmentation results are presented.

Keywords: *background generation, background subtraction, image processing, hardware acceleration, FPGA*

IMPLEMENTACJA DETEKCJI OBIEKTÓW RUCHOMYCH W CZASIE RZECZYWISTYM W SYSTEMACH NADZORU WIZYJNEGO Z WYKORZYSTANIEM UKŁADÓW FPGA

W artykule zaprezentowano koncepcję implementacji zadań obliczeniowych wykorzystywanych w systemach nadzoru wizyjnego w czasie rzeczywistym. Opisano implementację wielomodalnej metody generacji tła dla sekwencji wideo zarejestrowanych w kolorze oraz segmentację obiektów ruchomych z wykorzystaniem informacji o jasności, kolorze i teksturze w zasobach rekonfigurowalnych układów FPGA. Zaprezentowano architekturę systemu, zużycie zasobów i przykładowe rezultaty segmentacji.

Słowa kluczowe: *generacja tła, odejmowanie tła, przetwarzanie obrazów, akceleracja sprzętowa, układy FPGA*

1. Introduction

Modern, advanced surveillance systems make greater use of visual information, which gives the fullest opportunity to assess the situation in locations monitored by the control system. Such a system usually consists of multiple cameras (analogue or digital),

* AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, IT and Electronics, Department of Computer Science, al. Mickiewicza 30, 30-059 Krakow, Poland, kryjak@agh.edu.pl

** AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, IT and Electronics, Department of Computer science, al. Mickiewicza 30, 30-059 Krakow, Poland, mago@agh.edu.pl

each of which generates a data stream. Information from the cameras is transmitted to the computer centre, where the fundamental phase of image processing and analysis takes place and also all video data is stored. Because of the computational complexity of algorithms, high-power computing units, among others supercomputers and, additionally, hardware accelerators, e.g. GPU are used. The type and nature of the calculations on the visual data enables also the use of reconfigurable computers based on FPGA devices. This paper addresses the problem of increasing of computing performance in digital video surveillance. There are many factors that determine the demand for processing power in such systems. These include:

- complexity of the system (e.g., number of cameras in the system, the extent of supervised facility, etc),
- range of computational tasks in the vision system,
- selection of appropriate algorithms and their efficient implementation,
- requirements on the quantity and quality archiving of video data,
- required response time of the vision system,
- encoding, encryption and transmission of video data,

Currently implemented digital video surveillance systems use a local area network infrastructure to transmit video data. Scheme of an advanced video surveillance system is presented in Figure 1.

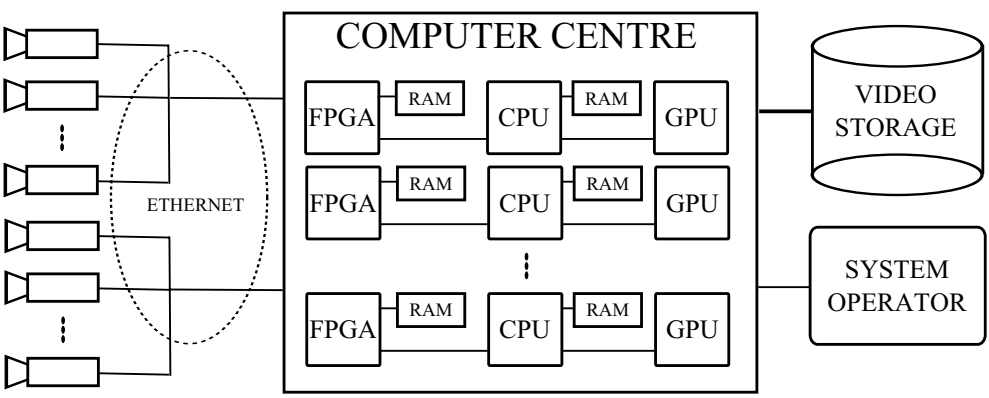


Fig. 1. Scheme of an advanced surveillance system

Computational tasks performed in a distributed digital video surveillance system are shown in Figure 2.

Tasks, which implementation in FPGA devices is possible and justified, are highlighted. These include compression and decompression of data (such as H.264) encryption and decryption of data and pre-processing as well as some elements of the image analysis. The use of FPGA can be considered in two elements of the vision system. Firstly, by performing an integration of image pre-processing in a smart camera. For example, in [19] a reconfigurable system was used for the selection of relevant data

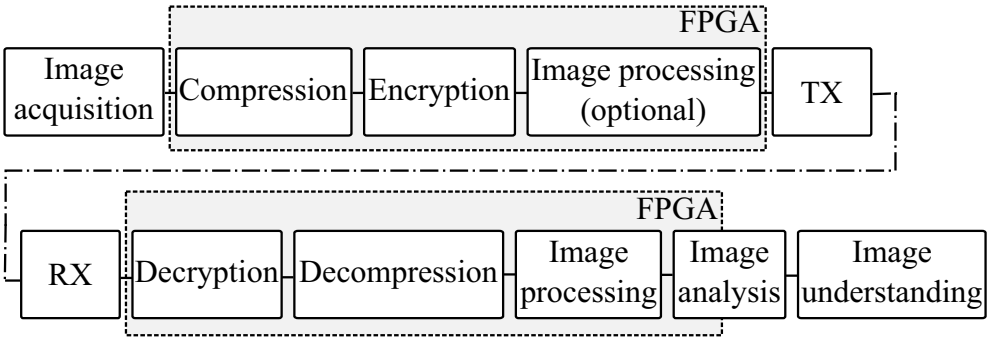


Fig. 2. Computing tasks in a video surveillance system

from the video stream. More often, FPGA are used at the computer centre, as a device that captures the video stream (i.e. frame-grabber with FPGA). Such a placement allows accelerating selected stages of pre-processing and image analysis and reduces the data transmission overhead between CPU and FPGA, because only the processing results are transferred to the CPU. The main task of advanced video surveillance is to detect unusual situations that constitute a potential source of danger. These include: abandoned objects (e.g. suitcases, bags), violations of the forbidden zones, the disappearance of certain objects from the scene (theft), monitoring of people's behaviour, people identification based on facial image, etc. Virtually all of them require the detection of movement in the visual scene. Motion detection of objects in real-time systems is usually performed by subtraction of a background model from the current frame, which can be assigned to the image pre-processing phase. Effectiveness of image recognition and understanding largely depends on the manner and quality of background subtraction. Background represents the static part of the scene. Due to different lighting conditions and changes in the observed scene, the background image is variable in time. The changes are characterized by a large spread dynamics. One can distinguish fast changes in the background, such as the rapid changes in sunlight and slow ones, for example, change of the time of day. Hence the resulting difficulty in determining the correct background image. Background generation operation can be assimilated to an image pre-processing methods. The manner and quality of its operation to a large extent affects the effective implementation of recognition and understanding of the image content. In this paper, background generation and segmentation of moving objects was implemented in FPGA. At the outset it was assumed that all operations are performed on colour image. A number of background generation algorithms has been cited and analysed in Chapter 2. Hardware-based implementations of background generation methods, described in the literature, are given in Chapter 3. Based on the analysis, an algorithm was selected (Chapter 4), and then implemented in FPGA (Chapter 5). Chapter 6 describes the hardware implementation of the moving object segmentation algorithm. Chapter 7 describes the integration of the background

generation module with segmentation module and contains the concept of the whole algorithm implementation on the RASC RC 100 platform. In Chapter 9 the results with additional discussion of further directions of development of the system are presented.

2. Background generation methods – a short survey

Methods based on background generation are the most commonly used to detect motion, assuming that the image is recorded by a static camera. They form the foundation of many advanced surveillance systems. The overall concept is the detection of movement on the basis of subtraction of the current image frame (from the camera) from the background reference model, which is the result of the background generation algorithm.

Over 20 years of research in this field led to a number of algorithms and their modifications. A very good overview of the methods is presented in the work [6]. Below, the most important were shortly described. Background generation methods can be divided into non-recursive and recursive.

Non-recursive techniques require storing a buffer of N previous frames and use a sliding window approach for background estimation. These include: mean pixel value from last N frames, median of pixel values from last N frames, W4 method [8], linear predictive filter (Wiener filter) and KDE (Kernel Density Estimators) [5]. These methods are characterized by high adaptability, and do not depend on prior history of N frames. The main disadvantage of these methods is a relatively high memory complexity (i.e. a buffer for 30 frames at resolution 720×576 in RGB colour space (24 bit/pixel) requires about 35 MB memory).

In the recursive techniques the background model is updated only on the basis of the current frame. The main advantage is a very low memory complexity, and main disadvantage is a low immunity to errors in the model (they are suppressed rather slowly). Recursive methods include: approximated median filtering [14] (also referred to as sigma-delta method), Single Gaussian method (SG) [20], Kalman filter, Mixture of Gaussians (MOG) [19], clustering [4], codebook [11], HMM (Hidden Markov Models). It is noteworthy that the methods can be divided into unimodal, which store a single background model (i.e. sigma-delta, SG) and multimodal, which store multiple background models - clusters (i.e. MOG, clustering, codebook). Multimodal methods perform better in conditions of rapidly changing lighting, for example: shadows cast by clouds, can handle small movement in the background (i.e. weaving leaves, flowing water) and solve the initialization of the background in the presence of moving objects on the scene.

3. Implementation of background generation in FPGA devices

Implementation of the background generation algorithm in FPGA resources allows to offload the computer's CPU, which computational power can be used to implement

further stages of an advanced video surveillance system (object tracking, object classification, analysis of behaviour, etc.) or create a smart camera system in which the detection of moving objects will be done in the camera, and the results will be used, for example, in intelligent compression [18]. In addition, in the case of multimodal methods of background generation the architecture of the FPGA allows to parallelize operations associated with updating the model.

In literature some work related to the implementation of background generation in FPGA can be found. In the paper [2] an implementation of a method based on MOG and temporal low-pass filtering [13] with taking into consideration the specific character of calculations performed in FPGA device is described. For each pixel there are K clusters stored (few possible background representations). A single cluster consists of a central value c_k (represented by 11 bits: 8 bits integer part, 3 bits fractional part). and weight w_k (6 bits). The following versions of the method were described: a grey-scale, unimodal model (K=1), a unimodal, RGB colour space model (K=1) and a bi-modal model in greyscale (limited by the 36-bit data bus to external memory available on the target platform). The module was implemented on a RC 300 board with Virtex II 6000 FPGA and 4 banks of ZBT SRAM ($4 \times 2\text{M} \times 36$-bits).

In the paper [9] an implementation of the MOG method in FPGA resources (Virtex 2V1000) was presented. The primary problem with hardware implementation of this algorithm is access to external RAM – a single Gaussian cluster stored with the assumed precision requires 124 bits. Therefore, in the paper, a compression scheme based on the assumption that adjacent pixels have similar distribution is proposed. Simulation results show that the method will allow to reduce the demand for bandwidth to RAM by approximately 60%. Implementation of background generation methods in FPGA are also described in the works [1], [7], [10], [16].

4. Description of the implemented background generation algorithm

Analysis of recent papers and conducted initial study showed, that the main problem in implementing a background generation algorithm in reconfigurable resources is the external RAM throughput. Therefore the starting point for the algorithm selection was an analysis of the necessary RAM access presented in Table 1.

When comparing the methods presented in Table 1 the following assumptions were made. Image resolution was 720×576. A single grey-scale pixel or single colour component is represented as a fixed-point 10-bit number (8-bits integer part, 2 bits fractional part). For the multimodal methods the cluster's weight was set as a 6-bit number and the number of clusters was set as $K = 3$. Additional bits inform of the standard deviation for the models of RGA and MOG. The most important parameter from the implementation point of view is the column 'Location Model', which shows how wide is the word describing the background model for a given pixel location. Taking into account the additional assumptions: background model should

Table 1
Analysis of the required RAM access for selected background generation methods

Background model	Colour model	Pixel model [b]	Weight [b]	Additional [b]
Running Gaussian Average	grey-scale	10	–	–
	colour	30	–	–
Runing Gaussian Average (with standard deviation)	grey-scale	10	–	10
	colour	30	–	30
Clustering ($K = 3$)	grey-scale	10	6	–
	colour	30	6	–
Mixture of Gaussians ($K = 3$)	grey-scale	10	6	10
	colour	30	6	30

Background model	Single cluster [b]	Location model [b]	The whole model [MB]
Running Gaussian Average	10	10	0,49
	30	30	1,48
Runing Gaussian Average (with standard deviation)	20	20	0,99
	60	60	2,97
Clustering ($K = 3$)	16	16	0,79
	36	108	5,34
Mixture of Gaussians ($K = 3$)	26	26	1,29
	66	198	9,79

be in colour and multimodal, and the target implementation platform is RASC RC 100, where the width of the memory access is 128-bits, it was decided to implement the clustering method .

The clustering algorithm [4], [12] is similar to MOG, but without the assumption that a pixel (colour component) has a Gaussian distribution. The advantage of this method is the lack of complex mathematical operations (i.e. absence of divisions, square roots, exponentials) – which consume many FPGA resources – and requires a significantly lower memory throughput than MOG. The main disadvantage is the lack of automatic method of setting a threshold, like in the MOG method. It can therefore be concluded that the chosen algorithm is a compromise between ease of implementation, and performance.

For purposes of implementing this algorithm in reconfigurable resources, several modifications have been done. The first change concerns the use of colour space. It was decided to use the CIE Lab space, as more suitable for segmentation, including the detection and removal of shadows (on the basis of work [3]). In the CIE Lab colour space the luminance component L is a number between 0 – 100 (7 bits), and the chrominance components (ab) are numbers between -127 to 127 (8 bits). Starting from a value of 128 bits (available bus width to external RAM memory on the RC 100 platform) the number of clusters was set $K = 4$. For a single cluster the following

representations were assumed: luminance component L: 9 bits (7 bits integer + 2 bits fractional), chrominance components a and b: 8 bits each, weight 6 bits. Thus, a cluster requires 31 bits, and four clusters (model for a single location in the image) 124 bits.

For the first frame in the video sequence the method is initialized – the actual frame becomes the first background cluster. For the subsequent frames, the following operations are performed (for each pixel independently):

a) Calculating the distance between the new pixel, and each of the clusters. Distances are calculated separately for the luminance and chrominance components, depicted by L and C respectively, on the basis of formulas:

$$dL = |L_F - L_{Mi}| \tag{1}$$

$$dC = |Ca_F - Ca_{Mi}| + |Cb_F - Cb_{Mi}| \tag{2}$$

where: L_F, Ca_F, Cb_F – pixel from the current frame, L_{Mi}, Ca_{Mi}, Cb_{Mi} – pixel from the i-th background cluster.

b) Determination of a background cluster which is the nearest to the current pixel and verification whether dL and dC values are smaller than the defined thresholds ($luminanceTh$ and $colourTh$).

c) If the cluster meets the assumptions of point B), it is updated in accordance with the formula:

$$M_{new} = \alpha_1 \cdot F + (1 - \alpha_1) \cdot M \tag{3}$$

where: M – background cluster, M_{new} – updated background cluster, F- current frame, α_1 – learning rate.

It's weight is also increased. Due to the 6-bit representation of the weight assumed, it can reach a maximum value of the 63. Then the clusters are sorted according to their weights. It is worth noting that the following simplification is possible: the cluster, whose weight has been incremented, could swap places only for the cluster, which is directly in front of him. In the vast majority of cases, the assumption is true, and can significantly simplify the sorting (both in software and hardware).

d) If none of the clusters meets the assumptions of point B), the cluster with the lowest weight is replaced by the current pixel and its weight is set to zero. During preliminary tests it was noticed that this approach resulted in too rapid inclusion of foreground elements (i.e. people who stopped for a moment) into the background model. It was therefore decided to introduce a modification in the form of using an update scheme according to equation (3) with the parameter α_2 rather than a direct assignment. In this way, foreground elements can become a part of the background only if they remain in the location long enough to integrate with the last cluster of the background.

The original algorithm assumed operation in two phases – background initialization (learning) and actual background subtraction. In the initialization phase, the

algorithm works as described above. Then, if the sum of weights of the first B clusters is larger than the threshold (tB) for all the pixels, a background model is created (composed of B clusters). On its basis further moving object recognition is conducted. At this stage it is not possible to add new elements to the background. Only an update according to formula (3), in order to compensate for slight changes in lighting is performed. In the described implementation it was decided to constantly run in the initialization phase, so that the background model can adapt to changes in the scene. In the future, a feedback loop from the further image processing steps and image understanding will be added. This should allow a better control of the background model.

Another modification is the omission of the mechanism for determining moving objects proposed in the original algorithm. A mask of moving objects is created in another module (described in Chapter 6) and the background generation module only provides background pixel value for every localization in the video frame. For a correct background value is considered that one which cluster weight exceeds a certain threshold ($weightTh$). The last cluster (with the smallest weight), on the grounds that it constitutes a buffer between the current frame and the background model is not considered as candidate for valid background.

5. Hardware implementation of background generation

Diagram of the designed background generation module are presented in Figure 3.

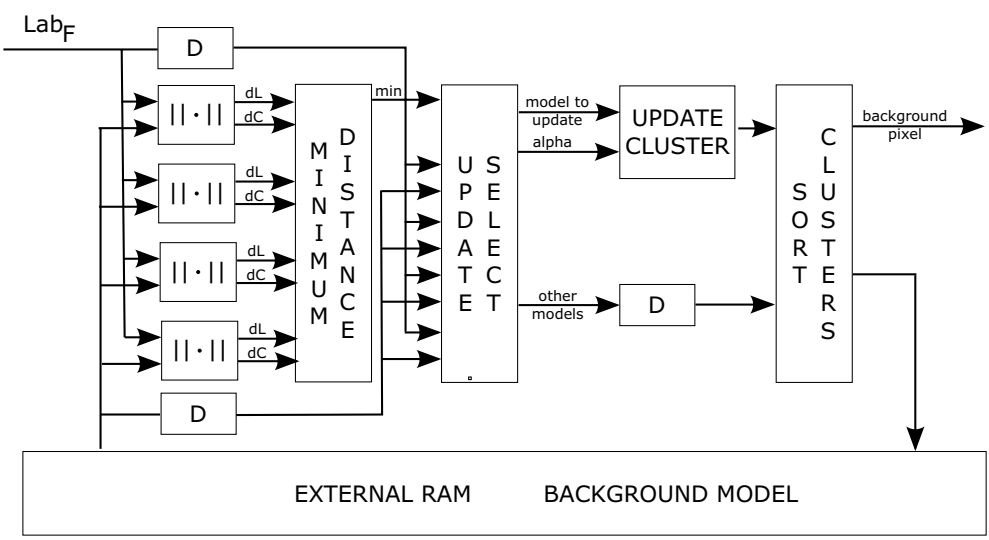


Fig. 3. Scheme of the background generation module

Used components:

- $||\cdot||$ – determination of the distance between the current pixel, a background cluster (point A),
- D – delay (in order to maintain piepelining),
- MINIMUM DISTANCE – choosing the cluster with the smallest distance from the current pixel (point B),
- UPDATE SELECT – choosing of cluster to update,
- UPDATE CLUSTER – implementation of equation (3) and weight update. For parameter α, which is in the range $[0, 1)$ a 10-bit fixed-point representation was selected. Multiplication was carried out using the available in Virtex 4 FPGA hardware multiplyiers (DSP48 blocks),
- SORT CLUSTERS – clusters sorting and the selection of the current representation of the background.

The module was described in VHDL with the use of IP-Core hardware modules (multiplication, delay lines). Synthesis and implementation for the Virtex 4 LX 200 device (available on the RASC RC 100 platform) was done using Xilinx ISE 11.1. Behavioural and post place and route simulations conducted in the ModelSim 6.5c showed full compliance of the hardware module with the one created in the Matlab 2009b. Results of the static timing analysis indicate, that the module should run at 100 MHz clock. This value is completely sufficient for a video stream with a resolution of 720×576 at 25 frames per second (for those parameters a pixel clock is equal 15 MHz). FPGA resource usage is presented in Table 2.

Table 2
FPGA resources used in the implementation of background generation module (Virtex 4 LX 200)

	Used	Available	Utilization
FF	893	178176	1%
LUT 4	1493	178176	1%
SLICE	799	89088	1%
DSP48	6	96	6%

It is worth noting that the module consumes only a small proportion of the resources available in the Virtex 4 LX200 FPGA, which allows implementation of further stages of image processing in the same device or alternatively to use a low-cost FPGA device (i.e. Xilinx Spartan series).

6. Moving object detection

Detection and tracking of objects is essential for many applications, in particular, is one of the most important elements of an intelligent video surveillance system. In the method proposed in this article it was decided to use three sources of information:

luminance (component L from CIE Lab colour space), chrominance (components ab from the Lab colour space) and SILTP (Shift Invariant Local Ternary Pattern) [17] texture descriptor. On the basis of normalized values, combination of three measures was proposed:

$$LCT = w_L \cdot dNL + w_c \cdot dN(ab) + w_T \cdot SILTP_N \tag{4}$$

where: w_L, w_C, w_T – weights – during experiments respectively 1, 3 i 2, dNL – normalized luminance difference, $dN(ab)$ normalized chrominance difference, SILTPN normalized SILTP texture descriptor. In the final step of the algorithm the LCT coefficient was thresholded with a selected threshold (0.9375 in the experiments). The final processing stage was a binary median filtering with mask size of 5×5.

Scheme of the whole segmentation module is presented in Figure 4.

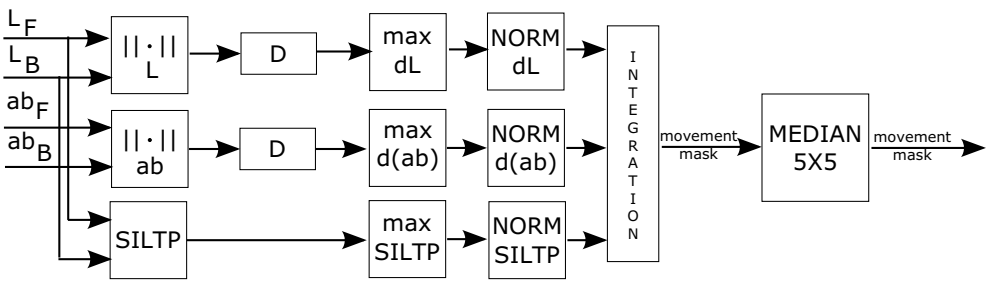


Fig. 4. Scheme of the segmentation module

Used components:

- $||L||$ and $||ab||$ – calculating the distance between the current frame and the background in accordance with formulas 1 and 2,
- SITLP – calculating the SILTP descriptor,
- D – delay (to allow the whole system run in pipeline),
- max dL, $d(ab)$, SILTP – modules, calculating the maximum value in the previous frame (for normalization),
- NORM – normalization to the range $[0 : 1]$,
- INTEGRATION – module responsible for the integration of information about the brightness, colour and texture and the final thresholding (decision foreground/background),
- MEDIAN 5×5 – binary median with window size 5×5.

The segmentation module, like the described earlier background generation module, was described in VHDL with the use of IP-Core hardware modules (multiplication and delay lines). Synthesis and implementation for the Virtex 4 LX 200 device (available on the RASC RC 100 platform) was done using Xilinx ISE 11.1 and simulations were performed with ModelSim 6.5 software. The tests showed compatibility of the hardware module with software model (described in Matlab 2009b). The image size

was assumed 720×576. It has an impact on the length of delay lines used in the module (SILTP, median) and final consumption of resources and delay (latency) introduced by the module. Results of the static timing analysis indicate, that the module should run at 144 MHz clock. FPGA resource usage is presented in Table 3.

Table 3
FPGA resources used in the implementation of the segmentation module. (Virtex 4 LX 200)

	Used	Available	Utilization
FF	2914	178 176	1%
LUT 4	4003	178 176	2%
SLICE	2804	89 088	3%
DSP48	10	96	10%

7. Integration of background generation and object segmentation

The background generation and segmentation modules were combined, with a colour (CIE Lab) 5×5 median filtering between them. The median was implemented using a bitonic merge sorter [15]. At this stage, the median is realized for each Lab component separately. In the future, correlation between colour components could be considered. Integrated background generation and foreground segmentation is presented in Figure 5.

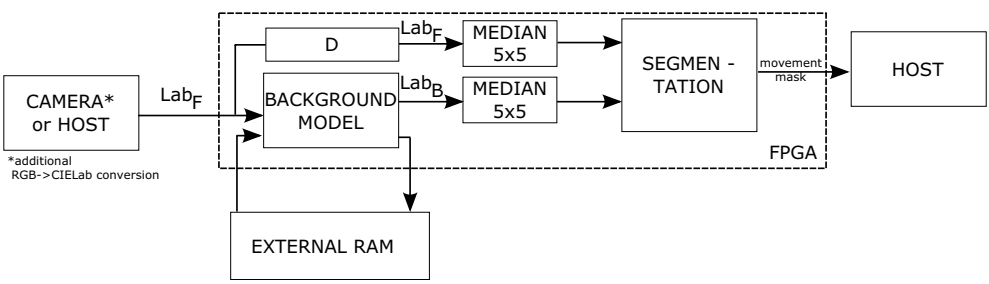


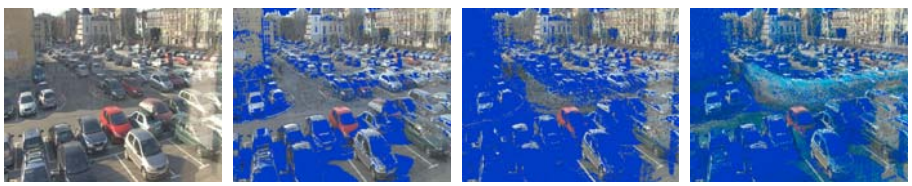
Fig. 5. Integrated background generation and object segmentation

The described modules were designed and implemented for Virtex-4 LX 200 used by RASC RC 100 platform. These modules could also by ported to more powerful FPGA boards: Pico M-503 and Pico M-502 equipped with Virtex-6 or Spartan-6. Both type of platforms are available at ACK Cyfronet AGH Kraków.

8. Results and discussion

The proposed and implemented method of background generation in terms of quality of results exceeds the methods based only on a single background model (RGA), but is

not as good as the MOG method. The algorithm was tested at challenging sequences in which very rapid sunlight changes were present. The method created two different models of the background, one for strong solar lightning and second for lightning during cloudiness (Figure 6, Cluster 1 and 2), and was able to switch between them.



(a) Cluster 1 (b) Cluster 2 (c) Cluster 3 (d) Cluster 4

Fig. 6. Background clusters created by the algorithm. Cluster 1 (sunlight), Cluster 2 (shadow), a blue pixel – cluster weight less than threshold

Tests of the segmentation module showed that there are several situations in which the integration of information about the luminance, colour and texture improves the performance of the algorithm. Many methods and systems for segmentation of moving objects are using only information about brightness. However, there are situations (for example in Figure 7) when the colour information allows to improve segmentation results. Similarly, additional information, particularly on the edges, can be obtained by analyzing the texture (Figure 7, e). In addition, an integrated approach, makes the system less sensitive to the selection of the final binarization threshold.

(a) (b) (c) (d) (e) (f) (g)

Fig. 7. Foreground segmentation. a – actual frame, b – background model, c – normalized luminance difference, d – normalized chrominance difference, e – normalized SILTP texture descriptor, f – combination of three measures, g – thresholded image

Further works could include implementing the MOG background generation method (which requires solving the high memory throughput issue), designing an automatic thresholding method as well as detailed test of the method on different video sequences.

9. Conclusion

The article presents the concept of accelerating computing tasks in an advanced video surveillance system and the implementation of background generation and segmentation of moving objects module in a reconfigurable device. Research has shown that

the implementation of this type of processing is entirely possible, does not require large FPGA resources and allows to offload the computer's CPU whose processing power can be used in later stages of image analysis. In addition, the results show that the use of colour images, even though it requires the execution of approximately three times more calculations and use of three times more memory, can improve the performance of the segmentation of moving objects.

Acknowledgements

The authors are grateful to the Academic Computer Center Cyfronet AGH for support of this work and access to hardware resources.

The work presented in this paper was supported by the Ministry of Science and Higher Education of the Republic of Poland, under the project 'SIMPOZ' (Grant No. 0128/R/t00/2010/12).

References

[1] Abutaleb M. M., Hamdy A., Abuelwafa M. E., Saad E. M.: *FPGA-based object-extraction based on multimodal sigma – delta background estimation.* [in:] 2nd International Conference on Computer, Control and Communication, 2009. IC4 2009., Feb. 2009, pp. 1–7.

[2] Appiah K., Hunter A.: *A single-chip FPGA implementation of real-time adaptive background model.* [in:] IEEE International Conference on Field-Programmable Technology, 2005. Proceedings., Dec. 2005, pp. 95–102.

[3] Benedek C., Szirányi T.: *Study on color space selection for detecting cast shadows in video surveillance.* Int. J. Imaging Syst. Technol., 17, Oct. 2007, pp. 190–201.

[4] Butler D., Sridharan S., Bove V. M. Jr.: *Real-time adaptive background segmentation.* [in:] IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03)., volume 3, vol. 3, Apr. 2003, pages III – 349–52.

[5] Elgammal A., Harwood D., Davis L.: *Non-parametric model for background subtraction.* [in:] FRAME-RATE WORKSHOP, IEEE, 2000, pp. 751–767.

[6] Elhabian S. Y., El-Sayed K. M., Ahmed S. H.: *Moving Object Detection in Spatial Domain using Background Removal Techniques – State-of-Art.* Recent Patents on Computer Science, 1, 2008, pp. 32–34.

[7] Gorgon M., Pawlik P., Jablonski M., Przybylo J.: *FPGA-based road traffic videodetector.* [in:] Proc. of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools, Washington, DC, USA, 2007. IEEE Computer Society, pp. 412–419.

[8] Haritaoglu I., Harwood D., Davis L. S.: *W4: Who? when? where? what? a real time system for detecting and tracking people.* [in:] Third Face and Gesture Recognition Conference, Apr. 1998, pp. 222–227.

[9] Jiang H., Ardo H., Owall V.: *Hardware accelerator design for video segmentation with multi-modal background modelling.* [in:] IEEE International Symposium on Circuits and Systems, 2005. ISCAS 2005., , vol. 2, May. 2005, pp. 1142–1145.

[10] Juvonen M. P. T., Coutinho J. G. F., Luk W.: *Hardware architectures for adaptive background modelling.* [in:] 3rd Southern Conference on Programmable Logic, 2007. SPL '07., Feb. 2007, pp. 149–154.

[11] Kim K., Chalidabhongse T. H., Harwood D., Davis L.: *Real-time foreground-background segmentation using codebook model.* Real-Time Imaging, 11, Jun. 2005, pp. 172–185.

[12] Li Q-Z., He D-X., Wang B.: *Effective moving objects detection based on clustering background model for video surveillance.* [in:] Proc. of the 2008 Congress on Image and Signal Processing, vol. 3, CISP '08, Washington, DC, USA, 2008. IEEE Computer Society, pp. 656–660.

[13] Makarov A.: *Comparison of background extraction based intrusion detection algorithms.* [in:] International Conference on Image Proc., 1996. Proc., vol. 1, Sep. 1996, pp. 521–524.

[14] McFarlane N. J. B., Schofield C. P.: *Segmentation and tracking of piglets in images.* Machine Vision and Applications, 8, 1995, pp. 187–193. 10.1007/BF01215814.

[15] Mueller R., Teubner J., Alonso G.: *Data processing on FPGAs.* [in:] *Very Large Data Bases Conference*, Lyon, 2009.

[16] Oliveira J., Printes A., Freire R. C. S., Melcher E., Silva I. S. S.: *FPGA architecture for static background subtraction in real time.* [in:] Proc. of the 19th annual symposium on Integrated circuits and systems design, SBCCI '06, New York, NY, USA, 2006, pp. 26–31. ACM.

[17] Qin R., Liao S., Lei Z., Li S. Z.: *Moving cast shadow removal based on local descriptors.* [in:] 20th International Conference on Pattern Recognition (ICPR), 2010, pages 1377–1380, Aug. 2010.

[18] Salem M. A. M., Klaus K., Winkler F., Meffert B.: *Resolution mosaic-based smart camera for video surveillance.* [in:] Third ACM/IEEE International Conference on Distributed Smart Cameras, ICDSC 2009., Sep. 2009, pp. 1–7.

[19] Stauffer C., Grimson W. E. L.: *Adaptive background mixture models for real-time tracking.* [in:] IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999, volume 2, 1999, pp. (xxiii+637+663).

[20] Wren C. R., Azarbayejani A., Darrell T., Pentland A. P.: *Pfinder: real-time tracking of the human body. IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), Jul. 1997, pp. 780–785.