A novel Background Subtraction Scheme for in-Camera Acceleration in Thermal Imagery

Antonis Nikitakis,
Technical University of Crete
School of Electronic and Computer Engineering,
Chania, Crete, Greece
anikita@mhl.tuc.gr

Ioannis Papaefstathiou
Synelixis Solutions Ltd,
Farmakidou 10,Chalkida, Greece
ygp@ synelixis.com

Konstantinos Makantasis
Institute of Communication and Computer Systems,
Athens, Greece
kmakantasis@isc.tuc.gr

Anastasios Doulamis
National Technical University of Athens
School of Rural and Surveying Engineering,
Athens, Greece
adoulam@cs.ntua.gr

***Abstract*— Real-time segmentation of moving regions in image sequences is a very important task in numerous surveillance and monitoring applications. A common approach for such tasks is the "background subtraction" which tries to extract regions of interest from the image background for further processing or action; as a result its accuracy as well as its real-time performance is of great significance. In this work we utilize a novel scheme, designed and optimized for FPGA-based implementations, which models the intensities of each pixel as a mixture of Gaussian components; following a Bayesian approach, our method automatically estimates the number of Gaussian components as well as their parameters. Our novel system is based on an efficient and highly accurate on-line updating mechanism, which permits our system to be automatically adapted to dynamically changing operation conditions, while it avoids over/under fitting. We also present two reference implementations of our Background Subtraction Parallel System (BSPS) in Reconfigurable Hardware achieving both high performance as well as low power consumption; the presented FPGA-based systems significantly outperform a multi-core ARM and two multi-core low power Intel CPUs in terms of energy consumed per processed pixel as well as frames per second. Moreover, our low-cost, low-power devices allow for the implementation, for the first time, of a highly distributed surveillance system which will alleviate the main problems of the existing centralized approaches.**

Keywords— Background subtraction; Gaussian mixture model; FPGA; low power; surveillance; monitoring.

I. Introduction

The task of background subtraction is widely used for locating moving objects, and visual attention modeling. Thus, it constitutes a key component for many high level vision based applications such as video surveillance and monitoring systems. Furthermore, as modern surveillance systems incorporate hundreds of cameras all feeding tons of image data for processing to a central video control system, this central node can easily be the bottleneck in terms of either network bandwidth and/or computation resources.

In this work we propose a novel hardware scheme for background segmentation which is optimized to be implemented next to the -camera using low cost and low power reconfigurable hardware. Our novel distributed implementation achieves real-time performance and low power consumption while it alleviates the central node bottleneck problem.

II. Related Work

A common method for adaptive background subtraction is based on the temporal averaging of images which is expressed as a single Gaussian average [1] or a Temporal median filter [2], [3]. The algorithm presented in [4] is known as GMM or Mixture of Gaussians and it was probably the most efficient scheme when originally presented as it utilized multiple Gaussians instead of a single one. The model was further improved in [5] known as the "MOG algorithm". A relative recent review of those schemes can be found in [6] which presents certain performance metrics.

Considering hardware accelerators, there are numerous FPGA-based implementations but almost all of them utilize the GMM algorithm not the MOG or any better successor of it. In the work in [7] and the later improvement in [8] the authors propose a real-time video segmentation/surveillance system based on GMM which incorporates a novel memory reduction scheme. Other approaches such as the work in [9] propose OpenCV-compatible accelerators of the GMM algorithm in reconfigurable hardware offering 24 fps for HD video. The same authors claim an even better performance of 91 fps in HD video in their improved work in [10], using a Xilinx Virtex 4 device. However, the authors focus on the processing only and they have not considered the memory bandwidth needed in order to achieve this processing rate; it seems that in a real system the memory bandwidth is becoming the bottleneck. In particular, the requested memory bandwidth in order to trigger the full performance listed is at least 8GB/sec where at the same time the low-power FPGA boards usually utilize a 64bit-wide DRAM that is clocked from 200-300MHz; as a result a typical memory sub-system can support only one fifth of the requested memory bandwidth. In this work, following a distributed computing approach, we demonstrate, that low cost, low power reconfigurable units can be connected with the camera module offering better accuracy than the existing systems, real-time performance, lower power consumption and less memory-related problems.

III. Algorithm and Hardware Architecture

The algorithm of the proposed scheme and the accuracy evaluation is described in detail in our previous work [11][1]. In this section we describe the hardware architecture of the

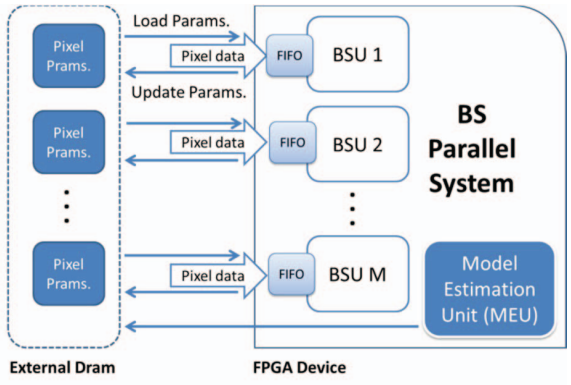
[1] Reader may refer also to http://arxiv.org/abs/1506.08581 if [11] isn't available on-line at the date of publication of this work.

proposed Background Subtraction Parallel System (BSPS). BSPS extends the algorithm in [11] by efficiently parallelizing it in custom hardware in order to take advantage of the vast reconfigurable resources of today's FPGA devices .

A. High Level Architecture

Since the Gaussian Mixture Model makes no assumptions for the pixel relationships, all pixels can be computed independently. Thus our hardware architecture consists of many parallel cores (4-16) in a scalable configuration, each processing a single pixel. In the next section we demonstrate two proposed configurations; one low cost featuring a 4-core BSPS Engine and a second one featuring a 16-cores BSPS Engine processing 16 pixels in parallel.

Fig. 1. The data loading process of the Background Subtraction Parallel System (BSPS).



Each of the cores is connected to a shared bus in order to get the processing data from the external DRAM (or memory mapped camera module) of a host System. The data loading is performed in batches of up to 16 pixels as shown in Figure 1.

All operations are per pixel with no dependencies between them. Thus by using a buffering scheme, utilizing simple FIFOs, we can hide the latency of the external DRAM and make our scheme working seamlessly as a streaming accelerator. Along with each pixel data a description of its current background model is loaded. The data loading and write-back operations are fully pipelined. More details regarding the bandwidth demands are given in Section V. The output of each core is a binary segmentation for the corresponding pixel (background or foreground) and the updated parameters of the background model. The Model Estimation Unit is depicted in Figure 2 and is described in more detail in subsection C.

B. System's Organization

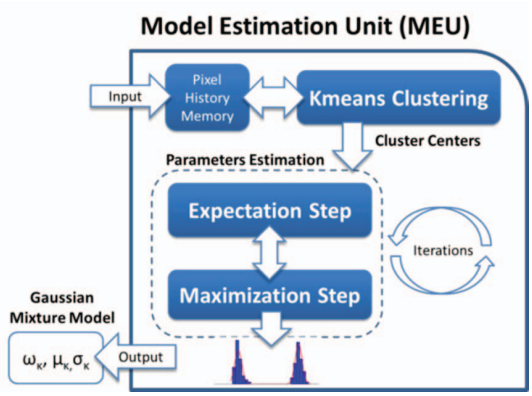
The BSP system comprises of two basic sub-modules: the **Model Estimation Unit (MEU)** which is depicted in Figure 2 and the **Background Subtraction Unit (BSU)** depicted in Figure 3. The MEU is activated just once at the initialization process of the system. It is responsible for building an accurate Gaussian mixture model for the background at all pixel locations. It uses a small history of pixel values (~100) and automatically estimates the appropriate number of Gaussian Components. After the model is built the MEU stores to the external DRAM the model parameters for each pixel i at location (x, y). The model parameters for each image pixel i are the Gaussian weight ϖ_{ik} , the mean μ_{ik} and the standard deviation σ_{ik}, where i corresponds to the specific pixel location and k to the *k-th Gaussian* component of that pixel location.

Then and during the normal operation of the system, only the **Background Subtraction Unit (BSU)** is activated. The BSU takes as input the pixel data stream (e.g. from a CCTV feed) along with its pixel parameters and gives as an output the segmentation of each pixel (Background or Foreground) while also updating the pixel's parameters (i.e its Gaussian Mixture Model -*GMM*). In this fashion for each pixel location (x, y) in the frame a GMM_{xy} value is maintained and updated which is utilized for the classification of all the new incoming pixels from the data stream at that specific pixel location (x, y).

C. The Model Estimation Unit (MEU)

One of the key advantages of the proposed scheme is that it doesn't require any prior knowledge about the background or any parameters to be set from the user in order to achieve an optimal operation. The Model Estimation Unit (MEU) is responsible for this task which is to build an accurate model for the specific background and its inherent temporal characteristics. It takes as input a small history of a pixel's values (~100) at a specific pixel location (x, y) and outputs an accurate Gaussian Mixture Model for this pixel. Two basic algorithms are utilized in this module: the k-means clustering algorithm and the Expectation-Maximization algorithm (EM). The k-means algorithm takes as input the pixel history data (~100 values) and the maximum number of the cluster centers which define the maximum number of the components of the Gaussian Model at the specific pixel location; the output is the cluster centers of the EM algorithm. Figure 2 gives an overview of the organization of the MEU module.

Fig. 2. The Model Estimation Unit Organization.

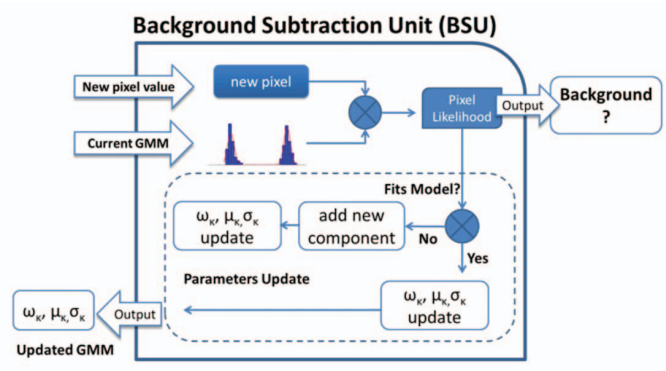


D. The Background Subtraction Unit (BSU):

The Background Subtraction Unit (BSU) is responsible for classifying the incoming pixels in the two available classes (Foreground/Background) and also updating the background model according to the new pixel and the current model. For this reason the **BSU** takes as input a new pixel value (x_{new}) and the current Gaussian mixture for this pixel (which is stored outside the chip) and gives as output the updated Gaussian mixture as well as the binary classifications of the incoming pixel. This unit also implements a novel scheme to

automatically adapt to new observed data. Based on the observation of a new pixel value x_{new}, there are two cases; either the new pixel is successfully modeled by the trained model, or not. To estimate if the new sample is successfully modeled, we find the closest component to the new sample, using the Mahalanobis distance [12]. More details considering the model adaptation process are given in [11]. An overview of the organization of the BSU module is depicted in Figure 3.

Fig. 3. The organization of the Background Subtraction Unit



IV. HARDWARE COST

The Background Subtraction Parallel System is completely scalable in terms of parallel pixel cores so as to accommodate the exact cost and performance needs of the target application. We primarily demonstrate our system in a low cost Xilinx Artix7 FPGA device (xc7a200tfbg484-3) to support the case of a low cost, next-to camera accelerator. In order to demonstrate that our system can support much higher frame rates and/or analysis , we also implemented it in a more powerful Virtex7 device (xc7vx550tffg1158-3); as the performance results clearly show our novel architecture seamlessly scales to a much larger device utilizing up to 16 parallel cores while the performance increases almost linearly with the silicon footprint .

TABLE I. TYPICAL HARDWARE COST ON A). LOW COST, LOW POWER XILINX ARTIX7 DEVICE (XC7A200TFBG484-3). IMPLEMENTING 4-BSU CORES/1-MEU CORE., B). XILINX VIRTEX 7 DEVICE (XC7VX550TFFG1158-3). IMPLEMENTING 16-BSU CORES/1-MEU CORE.

Logic Utilization	*ARTIX7*	*VIRTEX7*
Number of Flip Flops	53%	35%
Number of Slice LUTs	92%	78%
Number of DSP48E	68%	41%
Number of Block RAM_18K	2%	1.2%

For the code synthesis and bitstream generation we used Xilinx Vivado and Vivado HLS. For validation and proof only purposes our system was also implemented in a low end Zedboad evaluation platform [13] powered by a small Xilinx Zynq device. Table I shows the hardware utilization for the Artix7 device when implementing 4 BSU cores and 1 MEU core. The reason behind implementing only 1 MEU is that this unit operates only for the initialization and parameter estimation of the system and thus its performance is not critical. It also shows the hardware utilization of the Virtex 7 device when implementing 16 BSU cores and 1-MEU core.

Our scheme can also take advantage of FPGA's reconfigurable resources and seamlessly support different ratios of BSU/MEU cores, depending on the application. For example when the BSPS is re-initialized frequently more MEU cores may be needed. Thus different bistreams can be loaded on-the-field containing the desired number of BSU and MEU cores.

V. PERFORMANCE AND POWER EVALUATION

In order to demonstrate the performance of our system we compare it with a) a quad-core ARM Cortex A9 CPU (Exynos4412 SoC) clocked at 1.7 GHz with 2GB RAM b) a low power mobile Intel i5 (2450M) Processor clocked at 2.5Ghz with 8GB RAM and c) an ultra-low power Intel i3 (4005U) CPU with 4GB RAM, clocked at 1.7 GHz[2]. Both Intel processors contain two physical cores with hyper threading capability (4 threads in total). All the aforementioned CPUs have been selected for references in our evaluation as they offer very high computation power per watt.

For the Intel i3 and i5 the software compiler platform used was Microsoft Visual Studio 2012 and our code was optimized for maximum speed (i.e. the maximum offered -O2 optimization level). For the ARM A9 platform the code has been compiled with g++ compiler using -O2 and -O3 optimization level (O3 wasn't improving performance at all). In all software reference implementations OpenMP (-fopenmp) was used in order to utilize all the available cores/threads of the underlying platforms. For the FPGA we measured the exact number of clock cycles needed for segmenting a single pixel by a single core including loading and write back cycles. For this purpose we used the Zedboard evaluation board [13]. The exact clock cycles measured were also verified for the proposed Artix7 and Virtex7 devices using post-place & route timing simulation.

TABLE II. COMPARISON TABLE BETWEEN A XILINX ARTIX7 DEVICE @210MHZ, A XILINX VIRTEX7 DEVICE @ 222 MHZ, AN INTEL I5 @2.5GHZ AN ARM CORTEX A9 @1.7GHZ AND A DSP @ 600MHZ

Image frame	**Artix7 4-cores**	**Virtex7 16-Cores**	**ARM A9 4-cores**	**Intel i3 2-cores/ 4-threads**	**Intel i5 2-cores/ 4-threads**	**MOG *[14]* Blackfin BF-537 DSP**
320x240 (fps)	28.15	112.61	8.27	22.45	31.0	3.57
640x480 (fps)	7.04	28.15	2.07	5.61	7.75	-
µJ/pixel	2.13	2.15	4.7-6.2	4.52	9.28	-

Considering the I/O latency between the DRAM (or a memory-mapped Camera Module) and the FPGA, it is completely hidden as the operations for each core depend only on a single pixel and its corresponding background model. All this information is encoded in 256 bits in average, thus a buffering scheme using simple 256-wide FIFOs is adequate for fully pipelining the data transfers with the processing. The bandwidth demands of the device from the DRAM is no more than 250 MB/sec for 25fps at 640x480 resolution which is easily achievable even by low-end FPGA and DRAM devices.

[2] Intel i5 is implemented at 32nm, Intel i3 at 22 nm, and Virtex/Artix7 devices are implemented at 28nm

Table II shows that the 4-cores in the Artix7 low-cost device can achieve a processing rate of 28.15 fps at 320x240 exceeding by far the capabilities of a modern thermal camera (such as the FLIR A-315 one). The 4-core FPGA design outperformed by far the ARM A9 quad core CPU since it achieves more than 3x higher fps rate. In terms of power Artix7 consumes 4.6 watts based on Vivado's Power analysis tool while the quad core ARM A9 consumes from 3.5 to 4 watts[3]. The Intel i5 utilizing 2 physical cores (4-threads) heavily outperforms the ARM CPU while being only slightly ahead of the Artix7 device; its power consumption is 22.1watts[4] and this refers only to the actual CPU consumption.

Moving to the Intel i3, its fps performance is between the Artix 7 and the Intel i5 while its consumption is 7.8 Watts. In both Intel's CPUs the hyper threading capability showed very little overall performance improvement in terms of fps. Moving to the 16-core Virtex7 implementation, it offers the best performance of all the examined platforms, outperforming by 3x the Intel i5 in terms of fps, while the Virtex7 consumes 18.6 Watts. Looking at the energy metric of μJ/pixel in Table II, both FPGAs trigger a similar value of this metric which is more than 4 times lower than that of Intel's i5 CPU and more than 2 times lower than that of Intel's i3 CPU. For the ARM A9 this metric is expressed as a range as it is based mostly on specs3; in our evaluation experiments we measured the total dynamic power of the board using ODROID smart Power [15] but it wasn't possible to accurately measure only the CPU core modules. The last column in Table II refers to the work of [14] which implements the original MOG algorithm in an in-camera DSP processor (Blackfin BF-537). Even though it is hard to have a direct comparison, we see how challenging for embedded processors is to keep up with the demanding task of background segmentation, even for a less accurate and less CPU demanding scheme such as MOG.

Moving to the **Model Estimation Unit (MEU)**, both reference FPGA implementations utilize a single core of this module. The parameter estimation phase takes from several seconds to up to a couple of minutes for the FPGA implementations while the same applies for the ARM A9. As this process is performed only once upon initialization of the system it is certainly not in the critical path of the application therefore we did not try to accelerate further this module. Additionally, in very low-end devices it can be fully omitted and the model parameters can be computed and stored from an embedded CPU which will reside outside the chip.

VI. Conclusions

In this work a novel scheme for background subtraction in thermal imagery is presented which is designed and optimized for reconfigurable implementation. The presented FPGA-based implementations significantly outperform one ARM and two Intel low power mutli-core CPUs both in terms of energy per processed pixel and in fps performance. Our modular novel architecture can be efficiently implemented from a low-end FPGA to a very large one while supporting different ratios of processing elements according to the application demands. To the best of our knowledge this is the first time that the very demanding task of background subtraction can be executed next to the camera sensor in real-time and at a low power budget, which allows for a distributed new approach that avoids the bottlenecks of the existing centralized solutions.

[3] No accurate measurement for the CPU cores was possible as the Odroid U3 evaluation board used does not offer on-chip power meters.

[4] Measurements were performed using Intel's Power Gadget 3.0 Tool utilizing Intel's on-chip power meters.

Acknowledgment

The research leading to these results has received funding from the European Union's FP7, under grant agreement n.313161, eVACUATE Project (www.evacuate.eu).

References

[1] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: real-time tracking of the human body," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, Jul. 1997.

[2] B. P. L. Lo and S. A. Velastin, "Automatic congestion detection system for underground platforms," in *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2001*, 2001, pp. 158–161.

[3] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.

[4] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 1999, vol. 2, p. -252 Vol. 2.

[5] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*, 2004, vol. 2, pp. 28–31 Vol.2.

[6] S. Y. Elhabian, K. M. El-Sayed, and S. H. Ahmed, "Moving object detection in spatial domain using background removal techniques – State-of-art," *Comput. SCI*, vol. 1, pp. 32–54, 2008.

[7] F. Kristensen, H. Hedberg, H. Jiang, P. Nilsson, and V. Öwall, "An Embedded Real-Time Surveillance System: Implementation and Evaluation," *J. Signal Process. Syst.*, vol. 52, no. 1, pp. 75–94, Aug. 2007.

[8] H. Jiang, H. Ardo, and V. Öwall, "A Hardware Architecture for Real-Time Video Segmentation Utilizing Memory Reduction Techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 2, pp. 226–236, Feb. 2009.

[9] M. Genovese and E. Napoli, "FPGA-based architecture for real time segmentation and denoising of HD video," *J. Real-Time Image Process.*, vol. 8, no. 4, pp. 389–401, Dec. 2011.

[10] M. Genovese and E. Napoli, "ASIC and FPGA Implementation of the Gaussian Mixture Model Algorithm for Real-Time Segmentation of High Definition Video," *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 22, no. 3, pp. 537–547, Mar. 2014.

[11] K. Makantasis, A. Doulamis, and K. Loupos, "Variational Inference for Background Subtraction in Infrared Imagery," in *11th International Symposium on Visual Computing*, 2015, vol. [Accepted to appear].

[12] P. Mahalanobis, "On the generalised distance in statistics," presented at the Proceedings National Institute of Science, India, 1936, vol. 2, pp. 49–55.

[13] "Zedboard Evaluation Board." [Online]. Available: http://zedboard.org/.

[14] Y. Shen, W. Hu, J. Liu, M. Yang, B. Wei, and C. T. Chou, "Efficient Background Subtraction for Real-time Tracking in Embedded Camera Networks," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, New York, NY, USA, 2012, pp. 295–308.

[15] "ODROID | Hardkernel." [Online]. Available: http://www.hardkernel.com/main/products/prdt_info.php?g_code=G137361754360. [Accessed: 09-Sep-2015].