Performance Evaluation of SoC-FPGA Based Floating-Point Implementation of GMM for Real-Time Background Subtraction

Luis Javier Morantes-Guzmán[1,2](✉), Cristian Alzate[1], Luis Castano-Londono[1], David Marquez-Viloria[1], and Jesus Francisco Vargas-Bonilla[2]

[1] Faculty of Engineering, Instituto Tecnológico Metropolitano ITM, Medellín, Colombia
{luismorantes,luiscastano,davidmarquez}@itm.edu.co, cristianalzate224500@correo.itm.edu.co

[2] Faculty of Engineering, Universidad de Antioquia UdeA, Medellín, Colombia
jesus.vargas@udea.edu.co

Abstract. The embedded systems continue to display as solutions of smart surveillance systems. Background subtraction using Gaussian Mixture Model (GMM) is often portrayed as a common step for video processing. This work discusses the implementation of an embedded vision system on system-on-a-chip (SoC) device that integrates both a processor and an FPGA (Field Programmable Gate Array) architecture. The conventional Register Transfer Level (RTL) design, typically used for FPGA programming is slow, and the use of floating-point arithmetic is complex. However, the use of High-Level Synthesis (HLS) tools allows describing algorithms using high-level programming languages. Three background subtraction algorithms with floating-point arithmetic were developed using a hardware-software co-design methodology. The paper presents the details of the implementation on a ZedBoard Zynq Evaluation and Development Kit, considering requirements as hardware resources and power consumption used. Also, performance comparisons among a PC-based, ARM, FPGA and SOC-FPGA implementations are presented. The results showed that frame rates needed for real-time video processing were reached.

Keywords: Background subtraction · Gaussian Mixture Model (GMM) · Field Programmable Gate Array (FPGA) · System-on-a-chip (SoC) · High-Level Synthesis (HLS)

1 Introduction

The video surveillance applications over embedded vision systems have had a rapid growth in recent years. These systems make use of background subtraction as one of the key techniques for automatic video analysis. The state of the

J. C. Figueroa-García et al. (Eds.): WEA 2019, CCIS 1052, pp. 126–134, 2019.
https://doi.org/10.1007/978-3-030-31019-6_11

art methods make use of GMM estimators as background subtraction approach but it has a high computational cost for real-time implementation [6,7]. Hardware implementations of GMM algorithm on FPGA has been proposed in some works to deal with the computational burden requirements. An architecture for real-time segmentation and denoising of HD video on Xilinx and Altera FPGAs is presented in [5]. The authors describe the implementation and performance evaluation of OpenCV based GMM algorithm and morphological operations applied to a sequences of 1920×1080 frame size. In a latter work [6], authors present implementation and evaluation of architectures based on the OpenCV GMM algorithm for HD video over FPGA and ASIC technologies. In both cases, the FPGA based implementations were developed using VHDL. Fixed-point representation is used to achieve required performance for HD video resolution. An FPGA based implementation of GMM for offline background subtraction using fixed point arithmetic is presented in [1]. The block diagrams of the implemented system modules are shown. Moreover, qualitative results for a group of 360×240 frames selected from a sequence of 1250 are presented. The implementation of other FPGA based approaches of background subtraction techniques are found in some works. A performance evaluation of two multimodal background subtraction algorithms implemented on a ZedBoard is presented in [3]. A performance comparison of GMM, ViBE, and PBAS algorithms implementation on CPU, GPU, and FPGA is presented in [2]. In [8], is presented a summary of several FPGA based implementations of background modelling algorithms, developed by these authors in previous works. Additionally, the authors present the evaluation of the PBAS implementation using the SBI dataset. An FPGA based implementation of the codebook algorithm on a Spartan-3 FPGA is presented in [9]. The authors describe the system architecture and performance for sequences of 768×576. Hardware acceleration for real-time foreground and background identification based on SoC FPGA is presented in [10]. The proposed architecture was implemented on a Zynq-7 ZC702 Evaluation Board and is evaluated using datasets of real-time HD video stream.

The main contribution of this paper is a implementation that allows major flexibility in comparison to previous works with fixed-point representation and conventional RTL description. In this paper is presented an FPGA based implementation of the GMM algorithm on a ZedBoard using floating-point arithmetic. The OpenCV GMM function code is adapted for the Vivado-HLS, and parallelization directives are used for optimization. Taking advantage of the SoC architecture of the Artix-7 FPGA device, the generated HLS custom IP core is integrated with a Zynq processing system allowing the development of a complete embedded vision system. The hardware resources and power consumption are presented for the HLS custom IP core and the complete embedded vision system. Moreover, performance comparison with CPU software-based implementation for sequences at three different resolutions is presented in frames per second (fps). The experimental results show that the developed system is suitable for real-time at 768×576 resolutions with low-power consumption. The paper is organized as follows. In Sect. 2 the implemented algorithms and system architecture are described. Results and performance analysis are presented in Sect. 3. Finally, conclusions are drawn in Sect. 4.

2 Proposed Method

The algorithm implementations were developed using a hardware-software co-design methodology. The first stage is the software design that starts with the coding of the algorithms using C++ language and OpenCV libraries to take advantage of the rapid prototyping, visualization, and easy re-coding. The BGSLibrary (Background Subtraction Library) [13], covers a collection of algorithms from the literature. We selected three classic background subtraction algorithms: Frame Difference, Gaussian Mixture Model (GMM1) [11], and Efficient Gaussian Mixture Model (GMM2) [14]. The common steps of the GMM methods may be summarized in Algorithm 1.

Algorithm 1. Gaussian Mixture Model

Input: Frame Image I. Max. Gaussian distributions = 3
1: **for** each I **do**
2:   **for** each pixel j in I **do**
3:     Select number of Gaussians
4:     Represent j by sum of weighted Gaussian distributions
5:     Update all distributions
6:     Check for match with current pixel
7:   **end for**
8: **end for**
Output: Foreground Mask, Background Model.

The hardware implementation stage proposes two steps. The first step is the acceleration of the Algorithm 1 using Vivado HLS. Parallelization directives are applied in the code to improve the performance of the algorithm following the diagram showed in Fig. 1. Finally, HDL code generated by Vivado HLS is

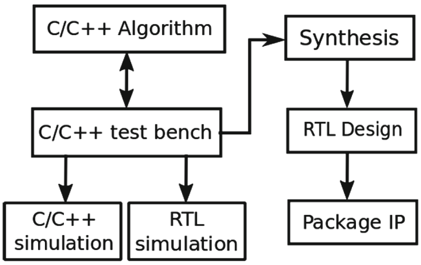

Fig. 1. HLS flow design used for FPGA implementations of the algorithms.

synthesized and downloaded to the FPGA. Figure 2 shows the code generated with the name HLS Custom IP Core inside of the SoC-FPGA architecture.

The algorithms were implemented on a ZedBoard Zynq Evaluation and Development Kit, which is a heterogeneous architecture based on FPGA. The integrated development environment (IDE) for HDL synthesis named Vivado® Design Suite was used for the synthesis of each design. Vivado® High-Level Synthesis (VHLS) is included in the suite, and it is responsible for transformation of the code written in C/C++ to HDL using a software-based approach, see Fig. 1.

The Operating System (OS) named PetaLinux was used on the ARM. This OS is a Linux distribution customized for SoC Kit boards and it facilitates the management of the peripherals on the development board such as Ethernet, USB, and HDMI ports. The communication between the processor and the FPGA is performed using AMBA AXI4-stream protocol. This is a data flow handler that offers several end-to-end stream pipes for the data transport of the applications. AXI4-stream works as a data transmission channel between the processing system (PS) and the programmable logic (PL). In the PS side, it works as a memory access control executed from a C++ function in PetaLinux. In the PL side, the communication is done using AXI Master. AXI Master maps the memory to stream conversion, and it performs the low-level transmission tasks, allowing to the designer to read/write the DRAM in Linux, and read/write from FPGA to DRAM using a high level approach. AMBA AXI4-Lite is an interface for data transmission between PS and PL for simple communication because it has a low-throughput memory-mapped communication and this interface is used for the control signals and status registration of the data, see Fig. 2.

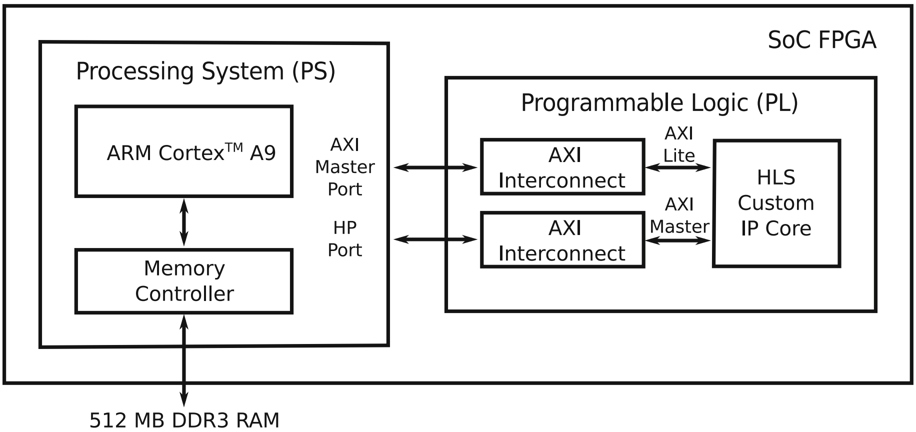

Fig. 2. Architecture of embedded vision system on SoC-FPGA.

3 Experiments and Results

We compared implemented algorithm quantitatively and qualitatively on the Wallflower dataset [12]. The results show that the proposed architecture can compute the foreground of the scenarios in the first column of the Fig. 3.

Fig. 3. Qualitative results. From left to right: original image, ground truth, Frame Difference, GMM1, GMM2. From top to bottom: bootstrap (BS), camouflage (CA), foreground aperture (FA), lightswitch (LS), time of day (TD), waving trees (WT).

Quantitative results were calculated with three quality metrics: Precision, Recall and F-score, as shown in Eqs. 1, 2 and 3, which are based on the amount of false positives (FP), false negatives (FN), true positives (TP) and true negatives (TN). The results in Table 1 are consistent with the found results in the state of the art.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F\text{-}score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{3}$$

Table 1. Performance using Wallflower dataset [12].

Algorithm	Sequence						
	Metrics	BS	CA	FA	LS	TD	WT
Frame Difference	Precision	0.834	0.721	0.754	0.888	0.823	0.626
	Recall	0.686	0.557	0.601	0.630	0.622	0.643
	F-score	0.753	0.629	0.669	0.737	0.709	0.635
GMM1 [11]	Precision	0.654	0.836	0.579	0.213	0.967	0.911
	Recall	0.675	0.807	0.602	0.389	0.796	0.945
	F-score	0.664	0.822	0.591	0.276	0.873	0.928
GMM2 [14]	Precision	0.626	0.895	0.839	0.310	0.895	0.913
	Recall	0.676	0.897	0.736	0.322	0.550	0.938
	F-score	0.652	0.896	0.784	0.316	0.682	0.925

The complete embedded vision system proposed is based on a heterogeneous architecture composed of a SoC-FPGA, as seen in Fig. 2. The complete system transmits information between ARM and FPGA, for this reason the maximum performance of the different implementations is limited for the maximum bandwidth of communication channels. However, the complete system facilitates management of peripherals and data using the operating system. Moreover, HLS custom IP vision core running in the standalone FPGA can be used in applications with direct connection to the FPGA. Table 2 shows both main hardware resources and power consumption in the algorithm implementations. This table compares the data for the HLS custom IP vision core on FPGA and the complete embedded vision system on the SoC-FPGA. The latter uses an amount greater of hardware resources due primarily to the drivers for the management and the transmission of data provided by the ARM.

Table 2. Hardware resources required on a ZedBoard FPGA after place and route. The whole system includes processing modules.

Algorithm	LUT	Flip Flop	Slice	DSP48	Power (W)
HLS custom IP vision core (FPGA)					
Frame Difference	1667/53200	2341/106400	701/13300	4/220	0.025
GMM1 [11]	18452/53200	56265/106400	9473/13300	25/220	0.446
GMM2 [14]	18872/53200	57698/106400	9623/13300	47/220	0.375
Complete embedded vision system (SoC-FPGA)					
Frame Difference	4452/53200	5275/106400	1739/13300	4/220	1.593
GMM1 [11]	32336/53200	68875/106400	12860/13300	25/220	2.102
GMM2 [14]	34056/53200	72059/106400	13249/13300	47/220	2.046

The performance of the SoC-FPGA and standalone FPGA were compared against a PC, as can be seen in Table 3. The ZedBoard Zynq is a SoC that contains a dual core ARM Cortex-A9 and one Artix-7 FPGA, with a FPGA clock period of 10 ns (100 MHz). The PC is equipped with a processor AMD Quad-Core A10-9620P running to 2.5 GHz. The average frame rate in PC, ARM and SoC-FPGA is computed as in Eq. 4. For FPGA we need to compute a frame rate as in Eq. 5. The performance measures for SoC-FPGA are better than the measured for the PC, and SoC-FPGA allows the real-time implementation in all cases. The standalone FPGA has the best performance because it does not have the limitation of the communication channels. Additionally, a comparison against the standalone ARM of the SoC-FPGA is included. An ARM processors is one of a family of CPUs based on the RISC architecture that is typically used over microprocessor boards and mobile devices used for real-time embedded system applications. The performance of FPGA in the most of the cases exceeds by 10× the performance of ARM, achieving over 40× in the best cases.

$$fps = \frac{1}{(\text{elapsed time per frame})} \tag{4}$$

$$fps = \frac{1}{(\text{FPGA cycles per frame}) \cdot (\text{clock period})} \tag{5}$$

Table 3. Performance comparison for sequences at three resolutions, in fps.

Resolution	Algorithm	PC	ARM (PS)	FPGA (PL)	SoC-FPGA (PS+PL)
160 × 120 [12]	Frame Difference	2459	597	5019	4947
	GMM1 [11]	284	30	1295	519
	GMM2 [14]	623	77	1728	613
352 × 288 [13]	Frame Difference	483	137	1277	1269
	GMM1 [11]	63	6	246.3	128
	GMM2 [14]	152	14	328	153
768 × 576 [4]	Frame Difference	141	28	224	220
	GMM1 [11]	19	2	56	22
	GMM2 [14]	38	3	75	26

4 Conclusions

This work presented implementation of three background subtraction algorithms in real-time using floating-point arithmetic. The HLS implementations permit a fast design and implementation of several architectures with different parallelization directives, in this way, it is possible to improve the performance of complex algorithms with a standard floating-point precision. The performance measures of the proposed architecture shown better computational times compared to PC-based implementation, and the parallelization of the GMM algorithms reached the frame per seconds needed for real-time video processing with a low power consumption. For these reasons, the heterogeneous architectures based on FPGA shown to be an effective tool for the video surveillance applications over embedded vision systems.

Acknowledgements. This study was supported by the AE&CC research Group COL0053581, at the Sistemas de Control y Robótica Laboratory, attached to the Instituto Tecnológico Metropolitano. This work is part of the project "Improvement of visual perception in humanoid robots for objects recognition in natural environments using Deep Learning" with ID P17224, co-funded by the Instituto Tecnológico Metropolitano and Universidad de Antioquia. L. J. Morantes-Guzmán is under grants of "Convocatoria de Doctorados Nacionales 617 - COLCIENCIAS 2013".

References

1. Arivazhagan, S., Kiruthika, K.: FPGA implementation of GMM algorithm for background subtractions in video sequences. In: Raman, B., Kumar, S., Roy, P.P., Sen, D. (eds.) Proceedings of International Conference on Computer Vision and Image Processing. AISC, vol. 460, pp. 365–376. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-2107-7_33
2. Bulat, B., Kryjak, T., Gorgon, M.: Implementation of advanced foreground segmentation algorithms GMM, ViBE and PBAS in FPGA and GPU – a comparison. In: Chmielewski, L.J., Kozera, R., Shin, B.-S., Wojciechowski, K. (eds.) ICCVG 2014. LNCS, vol. 8671, pp. 124–131. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11331-9_16
3. Cocorullo, G., Corsonello, P., Frustaci, F., Guachi, L., Perri, S.: Multimodal background subtraction for high-performance embedded systems. J. Real-Time Image Proc. (2016). https://doi.org/10.1007/s11554-016-0651-6
4. Ferryman, J., Shahrokni, A.: Pets 2009: dataset and challenge. In: 2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, pp. 1–6, December 2009. https://doi.org/10.1109/PETS-WINTER.2009.5399556
5. Genovese, M., Napoli, E.: FPGA-based architecture for real timesegmentation and denoising of HD video. J. Real-Time Image Proc. **8**(4), 389–401 (2013). https://doi.org/10.1007/s11554-011-0238-1
6. Genovese, M., Napoli, E.: 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. **22**(3), 537–547 (2014). https://doi.org/10.1109/TVLSI.2013.2249295
7. Goyal, K., Singhai, J.: Review of background subtraction methods using gaussian mixture model for video surveillance systems. Artif. Intell. Rev. **50**, 241–259 (2017). https://doi.org/10.1007/s10462-017-9542-x
8. Kryjak, T., Gorgon, M.: Real-time implementation of background modelling algorithms in FPGA devices. In: Murino, V., Puppo, E., Sona, D., Cristani, M., Sansone, C. (eds.) ICIAP 2015. LNCS, vol. 9281, pp. 519–526. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23222-5_63
9. Rodriguez-Gomez, R., Fernandez-Sanchez, E.J., Diaz, J., Ros, E.: Codebook hardware implementation on FPGA for background subtraction. J. Real-Time Image Proc. **10**(1), 43–57 (2015). https://doi.org/10.1007/s11554-012-0249-6

10. Safaei, A., Wu, Q.M.J., Yang, Y.: System-on-a-chip (SoC)-based hardware acceleration for foreground and background identification. J. Frankl. Inst. **355**(4), 1888–1912 (2018). https://doi.org/10.1016/j.jfranklin.2017.07.037. Special Issue on Recent advances in machine learning for signal analysis and processing
11. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: 1999 Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No. PR00149), vol. 2, p. 252 (1999). https://doi.org/10.1109/CVPR.1999.784637
12. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: principles and practice of background maintenance. In: Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 1, pp. 255–261 (1999). https://doi.org/10.1109/ICCV.1999.791228
13. Vacavant, A., Chateau, T., Wilhelm, A., Lequièvre, L.: A benchmark dataset for outdoor foreground/background extraction. In: Park, J.-I., Kim, J. (eds.) ACCV 2012. LNCS, vol. 7728, pp. 291–300. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37410-4_25
14. Zivkovic, Z., van der Heijden, F.: Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recogn. Lett. **27**(7), 773–780 (2006). https://doi.org/10.1016/j.patrec.2005.11.005