Improved automated parallel implementation of GMM background subtraction on a multicore digital signal processor

Smail Bariko[1], Abdessamad Klilou[2], Abdelouahed Abounada[1], Assia Arsalane[3]
[1]Automation and Energy Conversion Team, Faculty of Sciences and Technologies, Sultan Moulay Slimane University, Beni Mellal, Morocco
[2]Microelectronic, Embedded Systems and Telecommunications Team, Faculty of Sciences and Technologies, Sultan Moulay Slimane University, Beni Mellal, Morocco
[3]Laboratory of Engineering and Applied Technologies, High School of Technologies, Sultan Moulay Slimane University, Beni Mellal, Morocco

Article Info

Article history:

Received Nov 30, 2023
Revised Mar 28, 2024
Accepted May 4, 2024

Keywords:

Background subtraction
C6678 DSP
Gaussian mixture model
Open multiprocessing
Parallel processing

ABSTRACT

Scene segmentation is an essential step in a wide range of video processing applications, for instance, object recognition and tracking. The Gaussian mixture model (GMM) for background subtraction (BS) has gained widespread usage in scene segmentation, despite its known computational intensity. To tackle this challenge, we propose a practical solution to accelerate processing through a parallel implementation on an embedded multicore platform. In this paper, we present an improved automated parallel implementation of the GMM algorithm using the Orphan directive provided by open multiprocessing (OpenMP). Experimental assessments conducted on the eight cores of the C6678 digital signal processor (DSP) demonstrate significant advancements in parallel efficiency, particularly when handling high-resolution frames, including high-definition (HD) and full-HD resolutions. The achieved parallel efficiency surpasses the results obtained with classical OpenMP scheduling modes, encompassing dynamic, static, and guided approaches. Specifically, the parallel efficiency reaches approximately 82% for full-HD resolution frames and, 99.3% for low-resolution frames, respectively.

Corresponding Author:

Smail Bariko
Automation and Energy Conversion Team, Faculty of Sciences and Technologies
Sultan Moulay Slimane University
Beni Mellal, Morocco
Email: bariko_smail@yahoo.fr

1. INTRODUCTION

At the core of the moving object detection process, background subtraction (BS) is considered as critical step responsible for extracting moving objects [1]. Over the past two decades, this field has experienced significant algorithmic advancements, with researchers proposing various techniques that differ in their approach to updating and generating the background model. Comprehensive literature reviews of these techniques are presented by Garcia-Garcia *et al.* [2]. According to [2], BS algorithms can be classified into four distinct categories, based on the characteristics of the background pattern: i) mathematical concepts: fuzzy models [3], statistical models [4] and Dempster–Schafer models [5]; ii) machine learning techniques: support vector machines [6] and neural networks [7]; iii) signal processing techniques: Wiener filter [8] and Kalman filter [9]; and iv) classifications models: clustering algorithms [10].

While recent BS methods provide satisfactory accuracy, the use of complex models is computationally demanding. An efficient algorithm should balance accuracy with a low computational load. According to [11], Gaussian mixture model (GMM) is a promising candidate for addressing these constraints, as it represents an optimal compromise between accuracy and performance. Consequently, researchers have proposed statistical methods to manage and overcome challenges in background scenes, such as illumination and noise. For real-world applications, researchers have employed basic techniques like GMM [4], codebook [10] and visual background extractor (ViBe) [12]. This choice is driven mainly by memory and time requirements required by new BS methods [2].

In the literature, GMM is acknowledged as one of the predominant statistical methods [4], [13], [14]. Stauffer and Grimson [4] presented an adaptive model for real-time tracking, where each pixel is characterized by a mixture of Gaussians. This representation is dynamically updated in real-time through the incorporation of new input frames. Zivkovic [15] enhanced the GMM algorithm by proposing dynamic updates of K Gaussians for each pixel. As a result, K is adjusted dynamically to the multimodality of every pixel in accordance with scene evolution.

The fields of image and video processing have experienced a significant surge in challenges and complexity to meet the demands of real-time applications. This is explained by the market demand for images with high resolution (i.e.: full-high-definition (full-HD) 1920×1080 and HD 1280×720), in various application areas, including the detection of traffic violations, surveillance of national borders, and monitoring critical government infrastructure. Consequently, video processing has become both bandwidth and computationally intensive. To address this challenge and limitation, parallel processing techniques are essential to achieve high computational performance and fulfill real-time requirements. In this paper, the computational platform chosen is the multicore C6678 digital signal processor (DSP) from Texas instruments (TI), selected for its advantageous features, including high computing performance and low power consumption [16].

Over the years, several studies have examined automated parallel implementations based on open multiprocessing (OpenMP) for the GMM BS algorithm, with the aim of enhancing its computational performance and parallel efficiency. Szwoch *et al.* [17] suggested a parallel implementation of the GMM BS using a supercomputer comprising 192 nodes connected with an InfiniBand network. Each computing node consisted of two six-core CPUs in the Xeon EM64T architecture. OpenMP was used, and both static and dynamic scheduling techniques were evaluated. The achieved speedup value could not exceed 3.75 for medium frame resolution and 2.7 for HD resolution frames when twelve threads were utilized. Mabrouk *et al.* [18] proposed a parallel implementation of the GMM BS on a multicore platform, which included two Intel Xeon(R) CPU E5-2670 8-core processors. The distribution of processing across the multicore platform was accomplished through the application of OpenMP, resulting in a speedup of 11.6 for the HD resolution frame when sixteen cores were enabled. In our previous work [19], we evaluated OpenMP classical scheduling (OCS) modes (e.g., dynamic, static, and guided), and found that only dynamic scheduling provided a high speedup compared to other scheduling modes, such as guided and static. The maximum speedup achieved with eight enabled cores was 3.6 for HD resolution frames.

The main contribution of this paper is the parallel efficiency improvement of GMM BS algorithm on multicore DSP platform. This is achieved by selecting a suitable OpenMP directive: OpenMP orphan directive (OOD). Indeed, the OOD approach proves particularly advantageous, simplifying the task of implementing coarse-grain parallel algorithms [20], in which very large program regions are parallelized. The overall results demonstrate a significant improvement in speedup, even in the case of full-HD and HD resolution frames. The paper is structured as follows: section 2 introduces the GMM BS algorithm, describes the experimental setup, and outlines the proposed parallel implementation approach. Section 3 presents the experimental findings. Finally, a conclusion is provided in section 4.

2. MATERIAL AND METHOD

2.1. Gaussian mixture model for background subtraction

GMM has gained prominence in the field of BS. The pioneering work of Friedman and Russell [21] introduced a probabilistic model, wherein each pixel was characterized by a weighted sum of a limited number of Gaussian distributions. Subsequently, Stauffer and Grimson [4] made significant contributions by presenting an advanced GMM, accommodating K Gaussian models per pixel, typically K takes value within the range of 3 to 5 [4]. This advancement marked a significant stride in refining the GMM technique for BS. The formulation of the probability associated with the current pixel value, as illustrated in (1), underscores the inherent probabilistic foundation of this methodology:

$$P(x_t)=\sum_{i=1}^{K} \omega_{i,t} * \eta(x_t , \mu_{i,t}, \Sigma_{i,t}) \quad (1)$$

where: $\sum_{i,t}$=denotes the covariance matrix; $\mu_{i,t}$=the mean value; $\omega_{i,t}$=represents the weight of i^{th}Gaussian in the mixture at time t; $\eta(X_t, \mu_{i,t}, \sum_{i,t})$=specifies the Gaussian probability density function.

Comparative is conducted between the incoming pixel and the GMM to identify the pixel in proximity to 2.5 standard deviations. Two distinct scenarios are encountered: scenario 1: a match is established, prompting the adjustment of both the mean and the variance for the corresponding Gaussian distribution; and scenario 2: if no match is identified, the new incoming pixel substitutes the least probable component within the mixture.

In (2) outlines the process for updating the weights of the K distributions.

$$\omega_{k,t}=(1-\alpha)\omega_{k,t-1}+\alpha(M_{k,t}) \tag{2}$$

Where: α is the learning rate and $M_{k,t}$ equals 1 for the matched model, and 0 otherwise.

The parameters update of the matched distribution is defined by (3) and (4).

$$\sigma^2_t=(1-\rho)\sigma^2_{t-1}+\rho\left(x_t-\mu_t\right)^T(x_t-\mu_t) \tag{3}$$

$$\mu_t=(1-\rho)\mu_{t-1}+\rho x_t \tag{4}$$

Where: x_t represents new input frame pixel value; μ_{t-1} and σ_{t-1} represent the last mean and variance values of the matched Gaussian. The (5) represents the second learning rate, denoted as ρ.

$$\rho=\alpha*(x_t|\mu_k,\sigma_k) \tag{5}$$

The last step encompasses the estimation of the background, involving the sorting of Gaussians based on the ω/σ ratio. The initial ranked distributions B, with a cumulative weight sum surpassing the specified threshold (*Th*), are identified as the background, as described by the (6).

$$B=argmin_b\left(\sum_{k=1}^{b}\omega_k>Th\right) \tag{6}$$

Where: *Th* represents the minimum threshold of the background model.

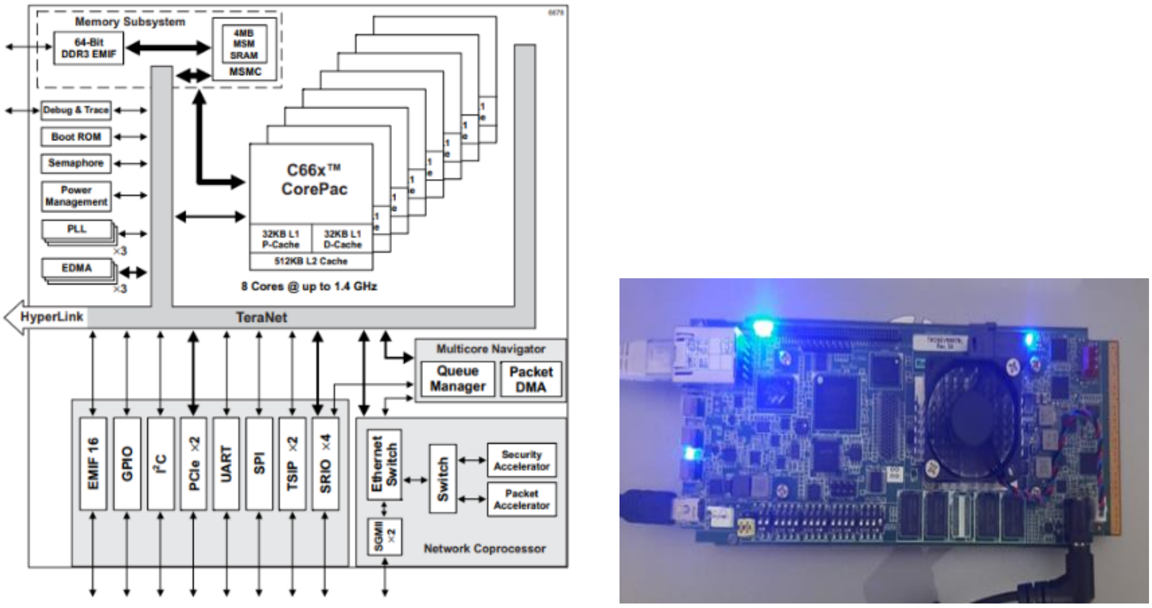
2.2. TMS320C6678 evaluation module overview

The TMS320C6678 evaluation module was used as the experimental platform, featuring a single C6678 DSP chip and 512 MB of dynamic random-access memory (DRAM) memory. The C6678 chip comprises eight DSP cores, each operating at a clock frequency of 1 GHz and delivering a computing performance of sixteen giga floating-point operations per second (GFLOPS). Notably, the architecture of the C66x DSP cores is based on very long instruction word (VLIW) design [16], [22]. The memory structure of the C6678 DSP is hierarchically organized into various levels, with the on-chip memory (L1) representing level 1, ensuring expedited CPU access compared to the external memory.

Detailed view of the TMS320C6678 DSP components is shown in Figure 1. A comprehensive functional block diagram of the C6678 board is depicted in Figure 1(a), while Figure 1(b) illustrates the C6678 evaluation module. These capabilities have inspired numerous research communities to develop real-time applications using this hardware platform [23]–[26]. Throughout our implementation, we utilized version 8.3.7 of the C6000 TI compiler.

2.3. Parallelization method

The OpenMP serves as an application programming interface (API) that facilitates parallel programming on multicore platforms characterized by homogeneous processors and shared memory architectures [27], [28]. It facilitates the handling of parallel implementations by offering directives that specify to the compiler the parallel regions within the code. Users are also required to select appropriate scheduling techniques to effectively distribute processing tasks among different cores. The choice of scheduling type significantly influences the overall performance outcomes. On the other hand, a deep understanding of the algorithm structure and the nature of the algorithm's workload loop is considered a key factor in identifying accurate OpenMP scheduling. Indeed, in the case of the GMM BS algorithm, the workload is considered irregular. This irregularity arises from the dynamic nature of the algorithm and its dependence on the complexity of the scene. The irregular workload can be attributed to several factors, including: i) varying background complexity: Different pixels in an image may have varying complexities in the background due to changes in lighting or object movements and ii) adaptive model updating: GMM models need to be continuously updated based on the characteristics of the scene. Pixels with more dynamic backgrounds or variations would require more frequent updates, resulting in a heavier workload.



(a) (b)

Figure 1. Overview of the TMS320C6678 DSP (a) functional block diagram of the TMS320C6678 DSP and (b) EVM6678 development board

Due to the irregular nature of the GMM algorithm, we chose the OOD approach, which offers significant advantages in simplifying the implementation of coarse-grain parallel algorithms [20]. Coarse-grained parallelism proves to be a suitable strategy for irregular loop algorithms, where the workload per iteration exhibits substantial variations. This approach entails dividing the overall task into larger units of work, with each unit representing a significant portion of the total workload. Coarse-grained parallelism effectively accommodates the irregularities in workload, reducing synchronization overhead compared to fine-grained approaches. This makes it particularly effective for scenarios where the computational requirements of different iterations vary widely. Overall, the OOD empowers users with more nuanced control over parallelization, leading to enhanced performance and improved stability in parallel programs, especially in cases where nested parallel regions are involved.

The Algorithm 1 shows the pseudocode of our proposed implementation using OOD approach. In this case, the "omp for" directive in Background_SubtractorGMM function is considered as an orphan directive. The utilization of an orphan directive in this context highlights a key aspect of our design strategy, emphasizing the parallelization of the BS process. Figure 2 shows CDnet 2012 highway sub-dataset. Figure 2(a) presents a visual representation of the input frame, providing a clear snapshot of the raw data processed from CDnet dataset [29]. Additionally, Figure 2(b) complements this by illustrating the generated mask, derived from our enhanced parallel implementation of the GMM algorithm.

Algorithm 1. Parallel implementation of GMM BS using OOD approach

```
//Main function
omp_set_num_threads (8);
GmmModel defined by {Mean(mk), Weight(ωk) and variance(σk);
Perform all GMM model initialization;
Get current input frame;
Begin
   #pragma omp parallel
   Call Background_SubtractorGMM (In InputFrame,
                                                    inout GMMModel,
                                                    Out MaskFrame);
end function

//Background_SubtractorGMM function
#pragma omp for
```

```
For each (PixelIndex: = [0…SizeOfFrame [; PixelIndex++)
   Begin
     pixel= InputFrame[PixelIndex];
     For each k Gaussian
        Begin
          diff(k) = abs (mk -pixel);
          if (diff[k] < Tmatch) then
          Update GMMModel {Mean(mk), Weight(ωk) and variance(σk)};
          else
          Update GMMModel {Weight(ωk)};
          end if
     end for
  Normalization of Weight (ωk).
  For each k Gaussian
     Begin
     Rank and sort all Gaussians by the ratio ωk ⁄σk;
  end for
  Retain the first B componants whose weight is greater than threshold (Th);
  if (pixel does not match background model) then
      mark pixel in MaskFrame as foreground;
   else
      mark pixel in MaskFrame as background;
end for
end function
```

(a) (b)

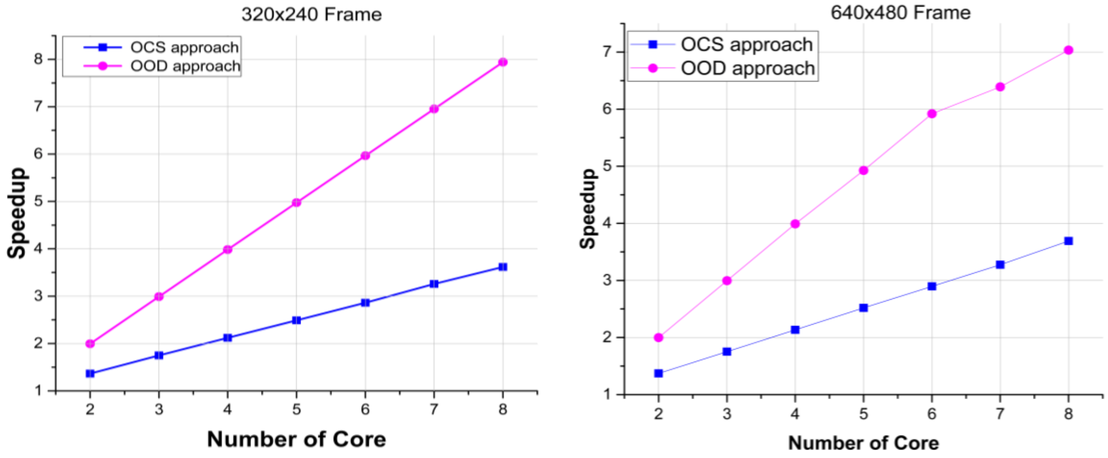
Figure 2. CDnet 2012 highway sub-dataset of (a) input frame and (b) generate mask [29]

3. RESULTS AND DISCUSSION

The new parallel approach proposed in this paper, based on OOD, was applied to different frame resolutions to validate the effectiveness of this method. The results obtained for various resolution frames (Figures 3 and 4), as presented in Figure 3(a) for low-resolution frames, Figure 3(b) for medium-resolution frames, Figure 4(a) for HD frames, and Figure 4(b) for full-HD frames, demonstrate that our proposed method yields improved results compared to OCS modes presented in prior work [19]. The dynamic scheduling with a chunk size equal to 128 provides the best speedup results, as presented in [19]. However, in the current work, our new approach based on OOD outperforms the OCS methods.

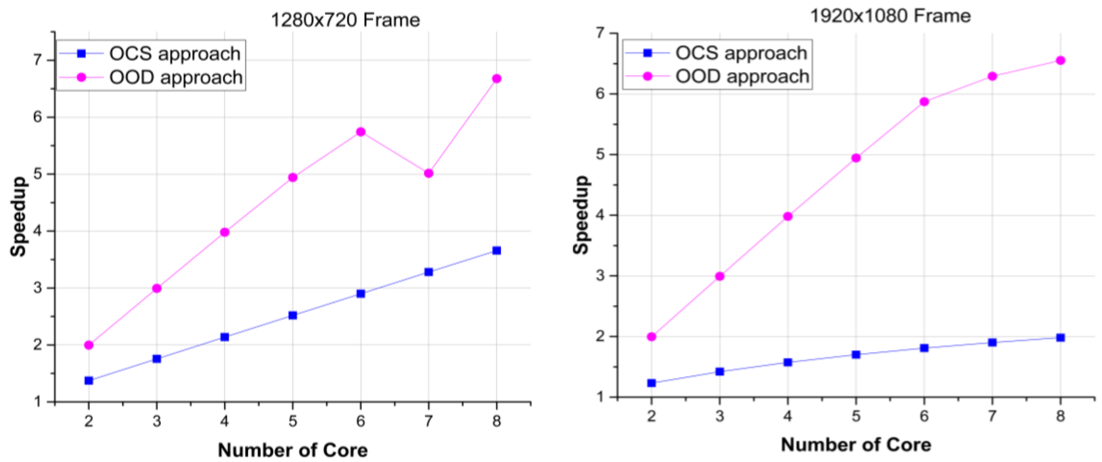
Linear speedup was achieved for the low-resolution frame, as shown in Figure 3(a). However, we observed a decrease in speedup for medium resolution frame, as illustrated in Figure 3(b), starting from the seventh core. For HD and full-HD frames, a speedup decrease was noticed when using more than six activated cores as illustrated Figure 4(a) and Figure 4(b), respectively. This reduction in speedup can be attributed to the limitation of the DRAM memory bandwidth. Access to the DRAM is restricted to a single core at a time, utilizing a 64-bit interface [16].

By strategically aligning the OOD with the specific demands of the GMM BS algorithm, we succeeded in optimizing the algorithm's parallelization. As shown in Figure 5, the OOD provides the best parallel efficiency performance compared to the conventional OpenMP scheduling methods presented in [19]. The adoption of the OOD, subsequent to code reallocation, represents a strategic move to bolster the parallel efficiency of the GMM BS algorithm.



(a) (b)

Figure 3. Comparison of obtained speedup between OOD and OCS approaches in (a) 320×240 frame resolution and (b) 640×480 frame resolution



(a) (b)

Figure 4. Comparison of obtained speedup between OOD and OCS approaches in (a) 1280×720 frame resolution and (b) 1920×1080 frame resolution

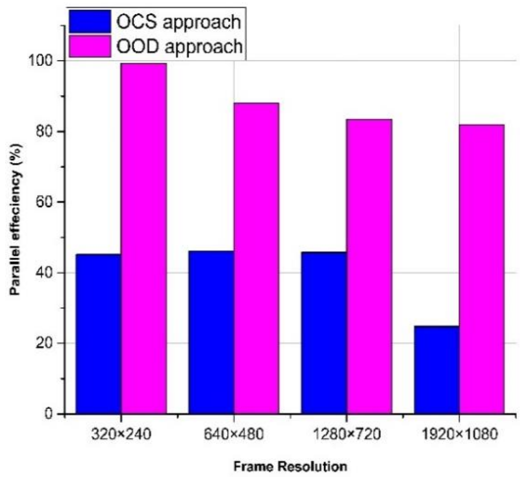


Figure 5. Parallel efficiency comparison between OOD and OCS approaches

4. CONCLUSION

In this paper, we demonstrated the parallel implementation of the GMM BS algorithm on a multicore DSP platform using OpenMP. After conducting a comprehensive analysis of the GMM BS algorithm's workload and structural intricacies, we recognized the need for an adaptive approach due to the irregular workload per pixel processing. In this context, the integration of the orphan directive from the OpenMP API played a crucial role in achieving optimal performance, surpassing alternative scheduling modes such as dynamic, static, and guided. Indeed, we enhanced the GMM BS algorithm's processing capabilities, resulting in significant improvements in parallel efficiency. Specifically, we achieved 82% parallel efficiency for full-HD resolution frame and a linear speedup (i.e., 99.3% parallel efficiency) for low-resolution frame when all eight DSP cores were enabled. Looking ahead, our future work aims to expand the parallel implementation of the GMM BS algorithm to 16 DSP cores, followed by the parallel implementation of vehicle tracking processing chain.

REFERENCES

[1] S.-C. S. Cheung and C. Kamath, “Robust background subtraction with foreground validation for urban traffic video,” *EURASIP Journal on Advances in Signal Processing*, vol. 2005, no. 14, p. 726261, Dec. 2005, doi: 10.1155/ASP.2005.2330.

[2] B. Garcia-Garcia, T. Bouwmans, and A. J. R. Silva, “Background subtraction in real applications: challenges, current models and future directions,” *Computer Science Review*, vol. 35, p. 100204, Feb. 2020, doi: 10.1016/j.cosrev.2019.100204.

[3] W. Kim and C. Kim, “Background subtraction for dynamic texture scenes using fuzzy color histograms,” *IEEE Signal Processing Letters*, vol. 19, no. 3, pp. 127–130, Mar. 2012, doi: 10.1109/LSP.2011.2182648.

[4] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, pp. 246–252, doi: 10.1109/CVPR.1999.784637.

[5] O. Munteanu, T. Bouwmans, E.-H. Zahzah, and R. Vasiu, “The detection of moving objects in video by background subtraction using Dempster-Shafer theory,” *Editura Politehnica*, 2015.

[6] H.-H. Lin, T.-L. Liu, and J.-H. Chuang, “A probabilistic SVM approach for background scene initialization,” in *Proceedings. International Conference on Image Processing*, vol. 1, pp. 893–896, doi: 10.1109/ICIP.2002.1039116.

[7] D. Culibrk, O. Marques, D. Socek, H. Kalva, and B. Furht, “Neural network approach to background modeling for video object segmentation,” *IEEE Transactions on Neural Networks*, vol. 18, no. 6, pp. 1614–1627, Nov. 2007, doi: 10.1109/TNN.2007.896861.

[8] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: principles and practice of background maintenance,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999, pp. 255–261 vol.1, doi: 10.1109/ICCV.1999.791228.

[9] S. Messelodi, C. M. Modena, N. Segata, and M. Zanin, “A Kalman filter based background updating algorithm robust to sharp illumination changes,” In: Roli, F., Vitulano, S. (eds) *Image Analysis and Processing – ICIAP 2005. ICIAP 2005. Lecture Notes in Computer Science*, vol 3617. Springer, Berlin, Heidelberg, 2005, doi: 10.1007/11553595_20.

[10] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Background modeling and subtraction by codebook construction,” in *2004 International Conference on Image Processing, 2004. ICIP ’04.*, vol. 5, pp. 3061–3064, doi: 10.1109/ICIP.2004.1421759.

[11] Y. Benezeth, P. M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, “Review and evaluation of commonly-implemented background subtraction algorithms,” in *2008 19th International Conference on Pattern Recognition*, Dec. 2008, pp. 1–4, doi: 10.1109/ICPR.2008.4760998.

[12] O. Barnich and M. V. Droogenbroeck, “ViBE: A powerful random technique to estimate the background in video sequences,” in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, Apr. 2009, pp. 945–948, doi: 10.1109/ICASSP.2009.4959741.

[13] M. Hofmann, P. Tiefenbacher, and G. Rigoll, “Background segmentation with feedback: the pixel-based adaptive segmenter,” in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2012, pp. 38–43, doi: 10.1109/CVPRW.2012.6238925.

[14] A. B. Chan, V. Mahadevan, and N. Vasconcelos, “Generalized stauffer–grimson background subtraction for dynamic scenes,” *Machine Vision and Applications*, vol. 22, no. 5, pp. 751–766, Sep. 2011, doi: 10.1007/s00138-010-0262-3.

[15] Z. Zivkovic, “Improved adaptive Gaussian mixture model for background subtraction,” in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, 2004, pp. 28-31 Vol.2, doi: 10.1109/ICPR.2004.1333992.

[16] “TMS320C6678 multicore fixed and floating-point digital signal processor,” Texas Instruments 2014. Accessed Jun. 28, 2020 [Online]. Available: https://www.ti.com/lit/ds/symlink/tms320c6678.pdf.

[17] G. Szwoch, D. Ellwart, and A. Czyżewski, “Parallel implementation of background subtraction algorithms for real-time video processing on a supercomputer platform,” *Journal of Real-Time Image Processing*, vol. 11, no. 1, pp. 111–125, Jan. 2016, doi: 10.1007/s11554-012-0310-5.

[18] L. Mabrouk, S. Huet, S. Belkouch, D. Houzet, Y. Zennayi, and A. Hamzaoui, “Performance and scalability improvement of GMM background segmentation algorithm on multi-core parallel platforms,” in *Proceedings of the 1st International Conference on Electronic Engineering and Renewable Energy (ICEERE 2018)*, 2019, pp. 120–127, doi: 10.1007/978-981-13-1405-6_16.

[19] S. Bariko, A. Arsalane, A. Klilou, and A. Abounada, “Efficient parallel implementation of Gaussian Mixture model background subtraction algorithm on an embedded multi-core digital signal processor,” *Computers and Electrical Engineering*, vol. 110, p. 108827, Sep. 2023, doi: 10.1016/j.compeleceng.2023.108827.

[20] H. Jin, M. Frumkin, and J. Yan, “Automatic generation of OpenMP directives and its application to computational fluid dynamics codes,” in *High Performance Computing. ISHPC 2000*, 2000, pp. 440–456, doi: 10.1007/3-540-39999-2_42.

[21] N. Friedman and S. Russell, “Image segmentation in video sequences: a probabilistic approach,” *arXiv:1302.1539*, 2013.

[22] Texas Instruments, “TMDSEVM6678L EVM technical reference manual version 2.01,” *Advantech*, 2012. Accessed Sep. 26, 2021, [Online]. Available: https://www.farnell.com/datasheets/1691442.pdf.

[23] A. Klilou, F. Bourzeix, O. Bourja, Y. Zennayi, L. Mabrouk, and S. Belkouch, “Real-time parallel implementation of road traffic radar video processing algorithms on a parallel architecture based on DSP and ARM processors,” in *2015 15th International*

Conference on Intelligent Systems Design and Applications (ISDA), Dec. 2015, pp. 183–188, doi: 10.1109/ISDA.2015.7489222.
[24] A. Arsalane, N. El Barbri, K. Rhofir, A. Tabyaoui, and A. Klilou, "Building a portable device based on DSP for meat discrimination," in *2016 International Conference on Engineering & MIS (ICEMIS)*, Sep. 2016, pp. 1–6, doi: 10.1109/ICEMIS.2016.7745327.
[25] S. Bariko, A. Klilou, A. Abounada, and A. Arsalane, "Efficient implementation of background subtraction GMM method on digital signal processor," in *2020 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, Nov. 2020, pp. 1–5, doi: 10.1109/ISAECT50560.2020.9523671.
[26] A. Klilou and A. Arsalane, "Parallel implementation of pulse compression method on a multi-core digital signal processor," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 6, pp. 6541-6548, Dec. 2020, doi: 10.11591/ijece.v10i6.pp6541-6548.
[27] R. van der Pas, E. Stotzer, and C. Terboven, *Using OpenMP—the next step: affinity, accelerators, tasking, and SIMD*, MIT Press, 2017.
[28] R. Chandra, R. Menon, L. Dagum, D. Kohr, D. Maydan, and J. McDonald, *Parallel programming in OpenMP*, Morgan Kaufmann, 2000.
[29] "Changedetection.net (CDNET), A video database for testing change detection algorithms," Changedetection.net, 2014, Accessed Jun. 13, 2020, [Online]. Available: http://changedetection.net/.

BIOGRAPHIES OF AUTHORS

Smail Bariko was born in Marrakech, Morocco. He received the engineer's degree in electronics and embedded systems from ENSA-Marrakech in 2011. Since 2011, he worked on different industrials field: aeronautics, semi-conductor and automotive companies like Zodiac Aerospace, STMicroelectronics, Altran, Lear corporation and Stellantis. His-scientific research interests include real-time implementation of image and video processing in parallel architecture platform. He can be contacted at email: bariko_smail@yahoo.fr.

Abdessamad Klilou was born in Marrakech, Morocco, in 1987. He received an engineer's degree in 2010 and a Ph.D. degree in 2016 from the University of Cady Ayyad, Marrakech, Morocco. Since 2017, he is a full professor at the Department of Electrical Engineering in the Faculty of Sciences and Technology, Sultan Moulay Slimane University, Beni Mellal, Morocco. His area of research is focused on parallel and real time optimization of signal processing algorithms on multi-core and multi-processors parallel machine. He can be contacted at email: a.klilou@usms.ma.

Abdelouahed Abounada was born in Sidi Slimane, Morocco. He received his Ph.D. degree from Cadi Ayyad University in 1992. In addition, he received the HDR degree in 2013 from Sultan Moulay Slimane University. His-research interests include advanced methods for numerical automatic and renewable energy. He can be contacted at email: a.abounada@gmail.com.

Assia Arsalane received an engineer's degree in electrical engineering from the National School of Applied Sciences of Khouribga in 2014 and a Ph.D. degree in 2019 from the University of Hassan I, Settat, Morocco. She is a full professor in the Department of Mechatronics, High School of Technologies, Sultan Moulay Slimane University, Beni Mellal, Morocco. Her area of research includes artificial intelligence, machine vision, image processing and embedded systems. She can be contacted at email: arsalan.assia@gmail.com.