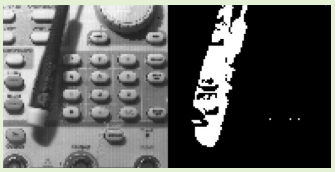
# HOPBAS10K: A 98×98 Pixels CMOS Vision Sensor for Background Subtraction

D. García Lesta, D. Cabello, P. López, V.M. Brea

***Abstract*— Background subtraction is one of the first visual tasks in many video processing applications. In this work, we introduce a hardware adaptation of a top-ranked rule-based algorithm, the Pixel-Based Adaptive Segmenter (PBAS), implemented on an integrated circuit with on-focal plane processing. On average, our hardware-oriented PBAS (HO-PBAS) proposal features a similar algorithm performance to that of the original PBAS with the benefit of a reduced number of samples of the background model and linear equations, and thus a simpler overall model. This algorithm was implemented as a 98×98 pixels full-custom mixed-signal 180 nm standard CMOS vision sensor. This solution features in-pixel processing with resource sharing strategies and 47 $\mu$m pixel pitch. The in-pixel processing includes the whole algorithm data path along with image pre-processing. The assessment of our implementation through F-Measure metrics with images captured by our chip from the public dataset *changedetection* shown on a PC screen results in a decrease of performance of only 6.7% with respect to the software version of PBAS.**



***Index Terms*— Background subtraction, CMOS vision sensors, PBAS, Mixed-signal**

## I. Introduction

In the last decades the growing number of cameras made unfeasible to analyze all their data by human operators. Thus, if computer-based video analysis is not applied, a huge amount of relevant information is being thrown away without taking full advantage of it. Classical solutions of video analysis are based on commercial cameras capturing scenes and sending raw data elsewhere to be processed. This approach, however, commonly faces two main challenges: time constraints and power consumption [1]–[3].

There are different solutions to address transmission and processing speed, as it can be typically solved by improving the infrastructure with a better data connection or a more powerful computer unit, depending on the specific bottleneck of the application. Nevertheless, this strategy commonly amplifies the difficulty of addressing the power requirements as high data bandwidth interfaces and powerful general-purpose computing devices, such as CPUs or GPUs, are energy hungry. In this work, we focus on low-power edge-computing devices, with the processing being executed as close to the sensor as possible, reducing the energy required for data transmission, which are no longer raw images but information extracted from them [4]. An additional benefit from local video processing is the relaxation of privacy measures, such as face or car plates blurring, as only features like objects' positions or sizes in the image are transmitted.

The main goal of this work is to build a system able to perform real-time video processing with low power consumption for background subtraction, or equivalently, foreground detection, which consists in the detection of moving objects in a scene [5]. Foreground detection is an attention mechanism in still camera scenarios that helps to increase the accuracy of higher level algorithms such as object recognition, classification or activity analysis, making the later steps more efficient [6]. Background subtraction is used in applications as traffic monitoring [7], video surveillance [8] or industry [9].

In regard to background subtraction for real applications compared with those developed in fundamental research, the existing gap between them is remarkable [10]. This difference widens when state-of-the-art algorithms are compared with silicon implementations, that are often based on frame differencing with some refinement or feedback mechanisms [11], [12]. This gap usually comes from the hardware requirements of modern algorithms, which lead to design complexity, which, in turn, would result in large pixel pitches and high power consumption. Therefore, the common trend in the literature is to run less elaborate motion detectors on CMOS vision sensors than top-ranked algorithms listed in image processing or computer vision contests like *changedetection* [13], trading image processing accuracy for the benefit of hardware metrics like power consumption or spatial resolution [12], [14], [15].

The motivation of our work is to fill this gap, addressing the challenge of the complexity of state-of-the-art computer vision tasks by leveraging PBAS- a high performance background subtraction algorithm listed in *changedetection* [13]- on a mixed-mode CMOS hardware architecture with pixel-level parallelism to still yield competitive hardware metrics, i.e.,

This work has received funding from projects PID2021-128009OB-C32, from the MCIN/AEI/10.13039/501100011033 and FEDER; and the European Union (European Regional Development Fund): from the Xunta de Galicia-Consellería de Cultura, Educación e Ordenación Universitaria Accreditation 2019–2022 ED431G-2019/04 and Reference Competitive Group Accreditation 2021–2024, GRC2021/48.

Authors are with the Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela, Santiago de Compostela, 15782, Spain. (e-mail: {daniel.garcia.lesta; diego.cabello; p.lopez; victor.brea}@usc.es).

low power consumption and processing time.

The rest of this article is organized as follows. Section II describes the algorithm implemented for the background subtraction, a modification of PBAS, that we have called HO-PBAS, specifically designed for an efficient hardware implementation as CMOS vision sensors. Section III addresses the design of such a CMOS vision sensor, the so-called HOPBAS10K. Section IV discusses the main experimental results. Finally, conclusions are drawn in Section V.

## II. HO-PBAS Design

One of the first steps in computer-based video analysis is to detect the regions of interest (ROI) of the scene and then use this information for many different tasks, i.e., visual surveillance of human activities, visual observation of natural environments [16], or content-based video coding, just to name a few. One of the possible techniques for ROI detection is background subtraction, which identifies the parts of the image different from that of a reference frame. Algorithms based on non-parametric models of the background, such as PBAS [17], gained a great popularity due to their great results in contests with public datasets [13].

In order to avoid large pixels and low fill-factors, the PBAS algorithm needs to be simplified with the lowest possible degradation of quality metrics. A hardware-oriented version of PBAS, HO-PBAS, was developed by authors in [18], where a set of modifications on the algorithm pipeline were carried out for a more feasible hardware implementation in the analog domain. This was accomplished by changing the complex equations in the processing pipeline by linear operations, reducing the neighborhood interaction from 8- to 4-connectivity or decreasing the number of samples stored in the background model.

In HO-PBAS the background model $\mathbf{B}(x_i)$ of pixel $x_i$ comprises $N$ previous samples of the scene for this pixel:

$$\mathbf{B}(x_i) = \{B_1(x_i), ..., B_k(x_i), ..., B_N(x_i)\} \tag{1}$$

This background model is compared against the incoming pixel value $I(x_i)$. This comparison is performed by counting how many samples within the range $(I(x_i) - R(x_i), (I(x_i) + R(x_i))$, assessing if this number is below a fixed value $\#_{min}$, where $R(x_i)$ is the pixel-dependant segmentation threshold. This procedure could be summarized as follows:

$$S_n(x_i) = \begin{cases} 1, & \#\{dist(I(x_i), B_k(x_i)) < R(x_i)\} < \#_{min} \\ 0, & \text{else} \end{cases} \tag{2}$$

where $S_n(x_i)$ is the segmentation result for the frame $n$ at pixel $x_i$. This result will be 1, i.e., foreground, if the number of samples inside the range does not exceed the minimum value $\#_{min}$, or 0, i.e., background, if otherwise. The background model $\mathbf{B}(x_i)$ is updated with two different mechanisms. The first one is meant to adjust the model to slow changes, such as the global illumination evolution during the day. It works by replacing the $kth$-sample of the background model $\mathbf{B}(x_i)$ with the new pixel value $I_n(x_i)$. This process is carried out with a pixel-dependent update probability $p(x_i)$, which affects only pixels that were segmented as background. Unlike the conventional first-in-first-out approach [19], a random sample of the background model is selected to be replaced with the new value. The update probability $p_n(x_i)$ is calculated through (3), where $p_{inc}$ and $p_{min}$ are fixed parameters, and $d(x_i)$ is the background model dynamics estimation. The result of the update probability obtained from (3) is thresholded with the maximum allowed value $p_{max}$.

$$p_n(x_i) = \begin{cases} p_{min}, & \text{if } S_n(x_i) = 1 \\ p_{n-1}(x_i) + [1 - d(x_i)] \cdot p_{inc}, & \text{else} \end{cases} \tag{3}$$

As explained above, $R(x_i)$ is adjusted per pixel according to the pixel dynamics, i.e., the variety of the values stored in the background model. This behavior is modeled through:

$$R(x_i) = d(x_i) \cdot R_{scale} \tag{4}$$

where $R_{scale}$ is a fixed parameter and $d(x_i)$ is the background dynamics estimator, defined by:

$$d(x_i) = \beta \cdot [max_k(B_k(x_i)) - min_k(B_k(x_i))] \tag{5}$$

where $\beta$ is a constant scaling parameter optimized through image simulation to set $d(x_i)$ in a proper range.

## III. CMOS Vision Sensor

The system floorplan of our CMOS vision sensor that implements HO-PBAS is shown in Fig. 1. It comprises the core circuitry, which integrates the Processing Elements (PE), and the periphery circuits shared by all the pixels. The core circuitry is formed by a combination of different active circuits, such as amplifiers, buffers or comparators. This is required to execute the HO-PBAS algorithm, which is more elaborated than background subtraction algorithms usually implemented on state-of-the-art mixed-signal integrated circuits, which are commonly based on frame differencing with diverse sorts of feedback mechanisms [11], [14], [15]. Furthermore, differently from HO-PBAS, these implementations are not evaluated in the *changedetection* benchmark. This will have a large impact on the power consumption and area requirements. To mitigate this effect, processing circuitry sharing will be applied as much as possible, and all the active circuits will be activated only when required, controlled by power-gating signals.

### A. Core Circuitry

The core circuitry is formed by a matrix of 49×49 PEs, as seen in Fig. 1, where each PE incorporates four pixels and one Processing Unit (PU), which features the processing circuitry that implements the HO-PBAS functional blocks which can be shared among neighboring pixels. This strategy allowed to shrink the pixel pitch from 60 $\mu$m to 47 $\mu$m, by multiplexing the PU operation in time. The rest of the algorithm blocks, whose functionality cannot be shared or when doing so would imply an excessive increase on the design complexity, are integrated in the four pixels of the PE.

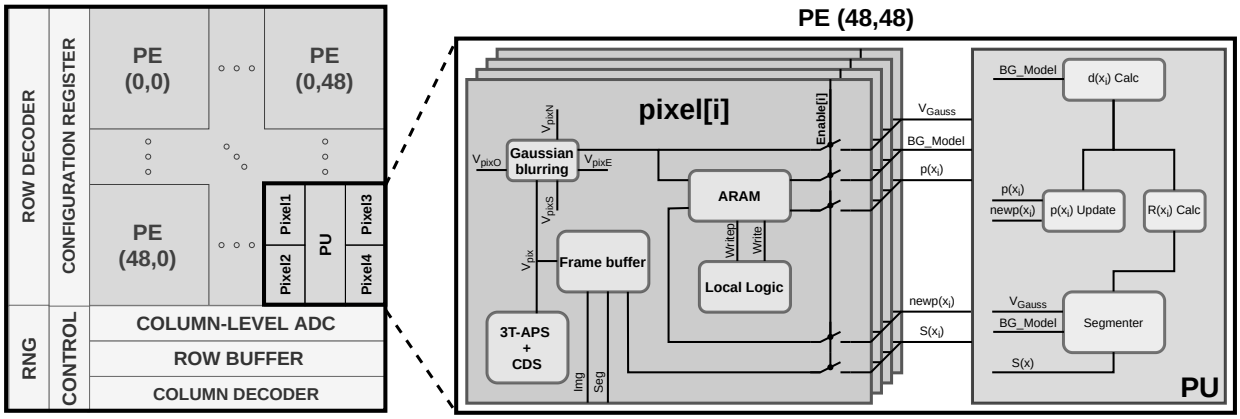



Fig. 1: HOPBAS10K floorplan (left), and detailed PE schematic, formed by four pixels and one shared PU (right).

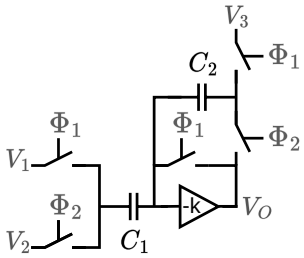



Fig. 2: Schematic of the arithmetic unit implemented in the processing blocks of HOPBAS10K.

All functions required by the HO-PBAS algorithm are implemented near the sensor in the analog domain, with the circuits controlled by digital signals generated in the periphery of the pixel array. In this work, in-pixel means a photodiode and its associated circuits which only operate on values from its corresponding photodiode. These circuits are: 3 transistors active pixel sensors (3T-APS) for image capturing, Correlated Double Sampling (CDS) for reset noise removal, the frame buffer to hold the captured image until it is read out, the Gaussian diffusion block for image pre-processing, local logic circuits and the Analog Random Access Memories (ARAM) for the background model storage. Fig. 1 shows the simplified schematic of every PE. All the circuits that can be shared by different pixels are time multiplexed in every PU. Thus, when the image processing is being carried out one pixel at a time is connected to this unit. This shared PU is formed by the blocks that calculate $d(x_i)$, $R(x_i)$ and the pixel update probability $p(x_i)$, as well as the block that takes the segmentation decision according to (2), i.e., the segmenter.

*1) Pixel:* As shown in Section II, HO-PBAS introduces several modifications to the PBAS algorithm to make it feasible for an analog hardware implementation while maintaining the accuracy of the algorithm. One of the most important simplifications was to linearize the equations in the datapath, so that they are easier to run on switched-capacitor (SC) circuits. The circuit shown in Fig. 2, which we call here arithmetic unit, will be replicated in all the processing blocks described below. Its operation relies on a cascode inverter amplifier with an open-loop gain k = 81 dB, which produces an average error in the arithmetic unit output of 1.2 mV, smaller than the LSB of an 8-bit ADC with $V_{dd}$ = 1.8 V as power supply. With this amplifier and capacitors $C_1$ and $C_2$, a differential amplifier with adjustable offset and gain is implemented, controlled by the two non-overlapped clock signals $\Phi_1$ and $\Phi_2$. Assuming that the amplifier gain is infinite, the output voltage $V_o$ can be obtained as:

$$V_o = V_3 + \frac{C_1}{C_2}(V_1 - V_2) \tag{6}$$

A detailed schematic and timing diagram for image capturing and spatial low filtering can be seen in Fig. 3. For the image acquisition stage, an 8×8 $\mu$m$^2$ NWELL photodiode over p-type substrate operating in integration mode was used with a 3T-APS as conditioning circuit. A global shutter acquisition is carried out by a sample and hold circuit placed next to the CDS block, realized with the arithmetic unit of Fig. 2, working as in-pixel frame buffer, implemented with a switched-capacitor buffered by an operational amplifier with a negative feedback loop. This frame buffer also features a digital register to store the segmentation result.

After the CDS operation, a low pass spatial filter is applied to the captured image to reduce noise, improving the algorithm performance [18]. The filter involves the convolution of the input image $I(x, y)$ with the Gaussian kernel $G(x, y, \sigma)$ resulting in the filtered image:

$$I_{filt} = I(x, y) * \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{7}$$

The physical implementation of the filter is based on the solution of reference [20]. In that work the result of that convolution for each pixel $I_{i,j}$ is obtained as the solution for the first-order differential equation

$$\frac{dI_{i,j}}{dt} = D(I_{i+1,j} + I_{i-1,j} + I_{i,j+1} + I_{i,j-1} - 4I_{i,j}) \tag{8}$$

where D is the constant diffusion coefficient. In the case of the Gaussian diffusion, the degree of blurring is controlled by D through the expression $\sigma = \sqrt{2Dt}$, where t is the time. In this implementation, image values will be stored in capacitors $C$ as voltages $V_{i,j}$, and connected to the four neighbors through

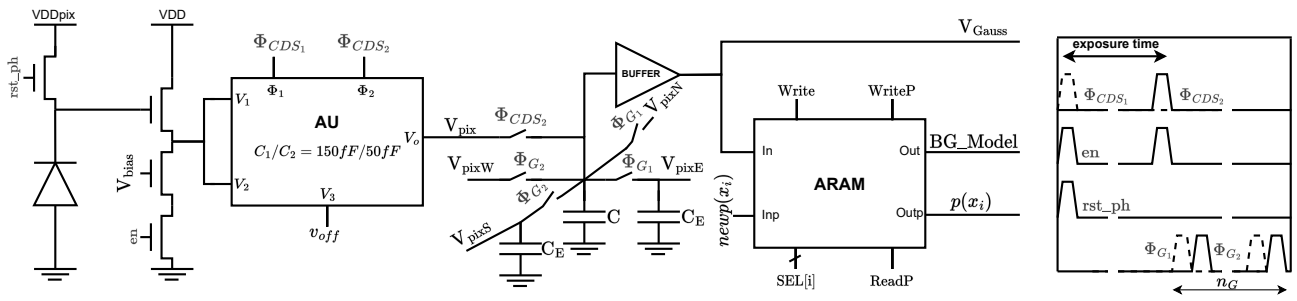



Fig. 3: Pixel schematic with timing diagram. This operation includes image capture and Gaussian filtering and it is executed in parallel for all the pixels.

a resistive network with resistance $R$. Then, (8) becomes:

$$C\frac{dV_{i,j}}{dt} = \frac{V_{i+1,j} + V_{i-1,j} + V_{i,j+1} + V_{i,j-1} - 4V_{i,j}}{R} \quad (9)$$

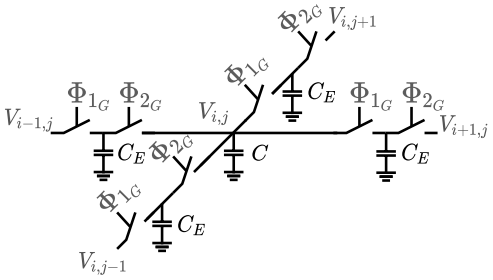
from where $D = 1/RC$, obtaining a filter spread factor $\sigma = \sqrt{2t/RC}$. This kind of system can be realized with different approaches, such as transistors in the linear region or polysilicon resistances. The first option might introduce nonlinearities and reduce the possible operation range, whereas the latter requires a large silicon area for in-pixel processing circuitry. In addition, complementary logic is required to control the transient evolution of the network, effectively controlling $\sigma$ magnitude.

These problems can be solved by replacing resistances by a switching capacitor (SC) circuit that mimics its behavior. This can be obtained by the circuit shown in Fig. 4a, where the image values are stored in capacitors $C$, and the resistive network is implemented through the exchange capacitor $C_E$ switched by transistors controlled by the non-overlapping signals $\Phi_{1G}$ and $\Phi_{2G}$, in what is called the single-Euler configuration, resulting in:

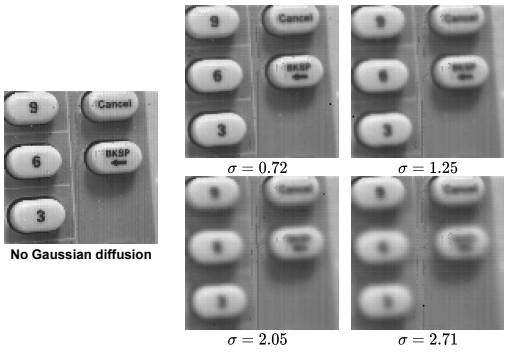
$$\sigma = \sqrt{\frac{2n_G C_E}{C}} \quad (10)$$

where $n_G$ is the number of complete cycles of the clock signals $\Phi_{1_G}$ and $\Phi_{2_G}$. In our implementation, both the stage capacitor $C$ and the exchange capacitor $C_E$ are MIM devices to ensure the linearity of the circuit with capacitances of 133 fF and 35 fF, respectively. These capacitors were sized to feature $\sigma$ ranging from 0 to 6 (validated through image simulations to be enough by a large margin) with a number of clock cycles $n_G$ below 63. Each pixel will write the captured voltage $V_{ij}$ into the stage capacitor $C$ and the Gaussian diffusion will be performed by charge sharing through the exchange capacitors $C_E$. Finally, to perform a non-destructive reading of the result, a buffer is incorporated between the exchange capacitor and the next circuit in the data pipeline. Experimental results are shown in Fig. 4b for different configurations with the same input image, showing that larger $\sigma$ values lead to more diffusion or blurring.

An in-depth study of the effects of our ARAM implementation on the HO-PBAS accuracy was addressed in [18]. In that work, different types of memories were compared, studying their electrical performance: write and read errors, long-term




(a) Single-Euler configuration of the SC network that performs the low pass spatial filter of the captured image.




(b) Experimental results for different configurations of $\sigma$.

Fig. 4: Gaussian diffusion network designed for image low-pass filtering on HOPBAS10K.

degradation due to the off resistances of the access transistors and value corruption after many readings. The conclusion of that work is that a shared input buffer with individual source follower for each memory as the output buffer is the one with the best performance, and hence, it is the one included on the HOPBAS10K chip. The output buffers will have an enable transistor with a twofold purpose: to cut down the power consumption when they are not used, and to connect or disconnect them from the output bus, as shown in the schematic of the ARAM of Fig. 6. All the memory capacitors are implemented with MIM devices of 150 fF. The rationale behind MIM capacitors instead of MOS capacitors is that MOS capacitors are expected to suffer from parasitic light

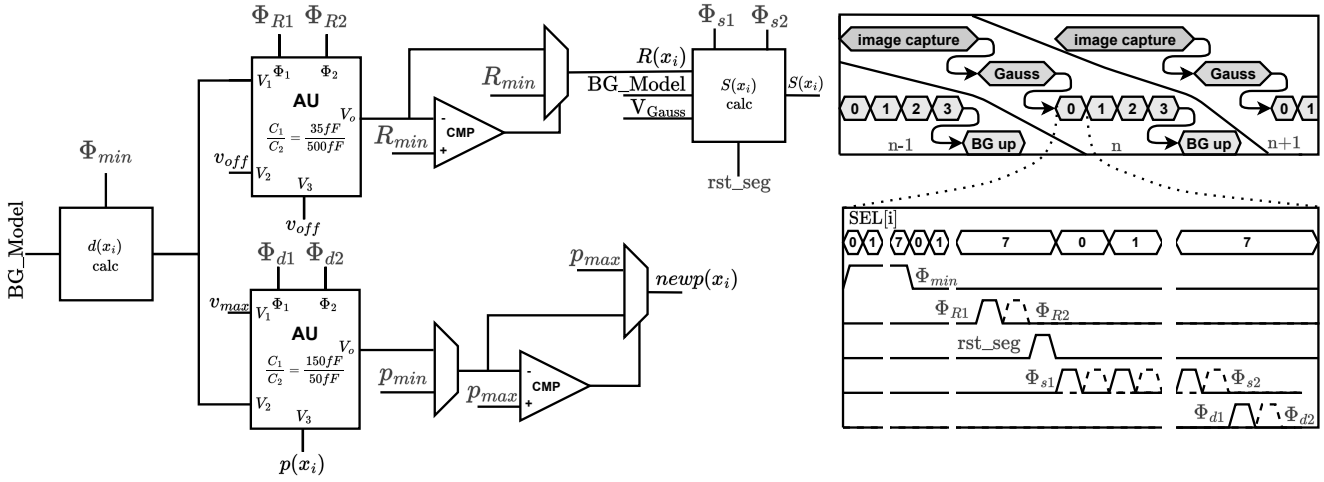



Fig. 5: PU schematic with timing diagram, presenting the last stages of the processing of frame $n-1$, the full acquisition and processing of frame $n$, and the acquisition and processing of frame $n+1$. It can be seen that the system executes acquisition of a new frame in parallel with the processing of the previous captured image. Also, the signals required for the processing of each pixel from the group of four pixels in a PE are shown.

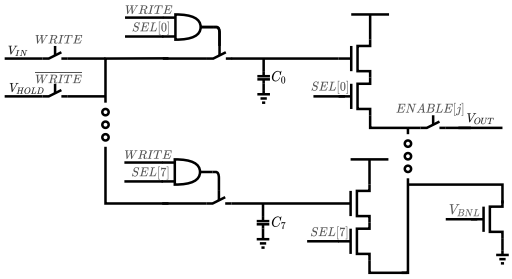



Fig. 6: Schematic of the HOPBAS10K ARAM.

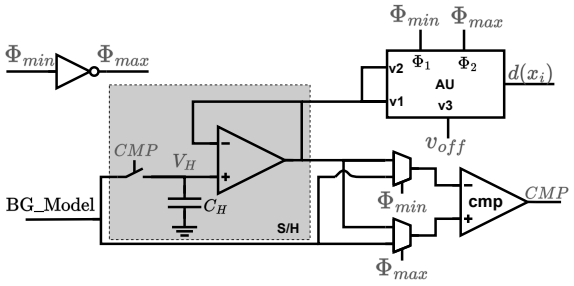



Fig. 7: Schematic of the $d(x_i)$ update block.

sensitivity, which comes as leakage currents from the incident light as the stored charge in the transistor channel will generate a photocurrent due to the pn junction of parasitic diodes [21].

*2) Processing Unit:* As explained in Section II, both $p(x_i)$ and $R(x_i)$ depend on the background dynamics estimator $d(x_i)$, which needs to be calculated for each pixel using (5). This value is extracted from the background model with the circuit of Fig. 7, which works as follows. First, signal $\Phi_{min}$ is set to low connecting the inputs of comparator $cmp$ in a configuration where the maximum value is searched. All the values stored in the ARAM are provided one by one, multiplexed in time, to the BG_MODEL node, and they are compared with the current value in the sample and hold (S/H) circuit. For each value above $V_H$ the comparator output $CMP$ triggers the S/H input switch and updates the stored value. Once all the ARAM values have been compared to each other, the S/H output will be $max_k(B_k(x_i))$, which is sampled by the arithmetic unit of Fig. 2. Then, $\Phi_{min}$ is set to high and the process is repeated. As the comparator inputs are swapped, the S/H will output the minimum value at the end of the cycle $min_k(B_k(x_i))$. This value is the input of the arithmetic unit, resulting in an output equal to $d(x_i)$.

The full schematic of the PU can be seen in Fig. 5. The background model dynamics estimator $d(x_i)$ is used in the circuit that calculates the segmentation sphere radius $R(x_i)$ through (4), implemented with an arithmetic unit. Additionally, to check if the resulting value is larger than the fixed parameter $R_{min}$, an additional stage with a comparator is added to the arithmetic unit. The estimation of the background model dynamics is also used to calculate the new update probability $p_n(x_i)$ for each pixel. Similarly to the block that calculates $R(x_i)$, an additional stage is added to the arithmetic unit to threshold the result with the fixed parameter $p_{max}$.

The last circuit required for image processing is the one responsible for the segmentation decision. Based on the previous image samples stored in the ARAM and the incoming pixel value, this circuit must calculate the one-to-one absolute difference between the incoming pixel and the ARAM values, and count how many are inside the segmentation sphere. Fig. 8 shows the schematic of the segmenter, where the different stages can be seen. The first one is the absolute difference circuit. It is implemented with an arithmetic unit and configured with unity gain and with a multiplexer for each input controlled by a comparator in such a way that the highest input is always connected to $V_1$; see (6). With this setup, the result of the difference will always be a positive value plus

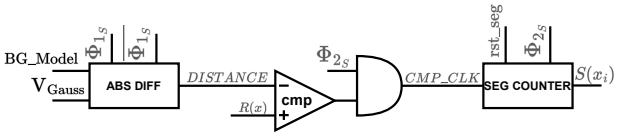



Fig. 8: Schematic of the circuit responsible for the segmentation result in the HOPBAS10K.

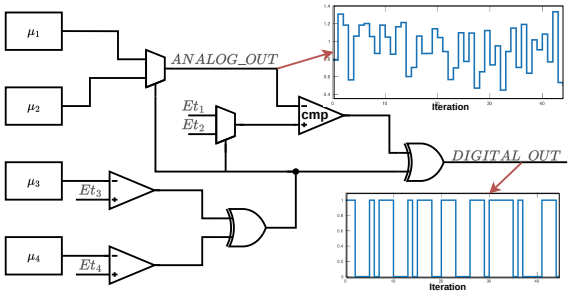



Fig. 9: Combination of RNG cells to reduce the output temporal correlation and improve the bitstream quality and corresponding output for both analog and digital parts from experimental tests.

the offset voltage.

Once the absolute difference between the background model sample and the input image is obtained, the result is compared with the radius of the sphere $R(x_i)$ with the circuit of Fig. 8. Thus, if the difference is smaller than the radius $R(x_i)$ it will mean that the sample is inside the sphere. To avoid the effect of transient voltages, an *AND* gate is added to the output of the comparator. Thereby, the counter will only receive pulses when $\Phi_{2_G}$ is high, preventing the effect of glitches to cause false positives.

As depicted in the timing diagram presented in Fig. 5, the processing of information captured by each pixel of the PE occurs sequentially, requiring four processing steps to handle the entire image. Following the foreground detection and update of algorithm parameters, each pixel updates in parallel its background model based on the outcome of the preceding step (labeled in Fig. 5 as *BG up*). Additionally, data processing takes place concurrently with the acquisition of the new frame, rendering the frame rate independent of processing time and solely reliant on sensor quality and scene characteristics.

### B. Array Periphery

The periphery of the pixel array comprises the blocks responsible for the digital control signals generation, which were designed with digital synthesis tools; the circuits designed for the system readout (both the captured image with a single-slope ADC and the segmentation result); and the random number generator, which feeds analog and digital output streams into the pixel array, required by the random model update mechanism of HO-PBAS.

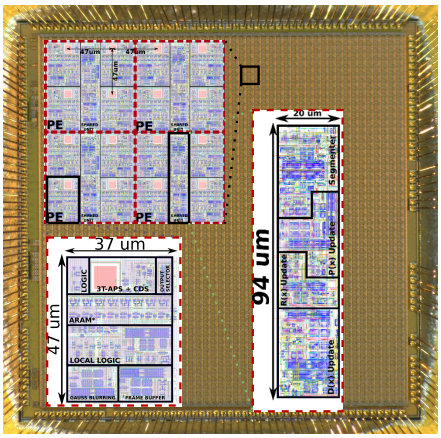



Fig. 10: HOPBAS10K microphotograph with the layout of 4 PEs and PE area breakdown.

Different options can be found for the ADC architecture in the literature, each of them with its benefits and drawbacks. SAR and cyclic ADCs commonly offer the best figure of merit [22], [23]. However, in this work a single slope ADC (SS-ADC) was selected because of its control simplicity and great linearity. It was implemented in a per-column approach, making the conversion time only proportional to the number of rows and not to the whole array size. The voltage ramp, generated externally, and ADC counter can be shared between all the ADC cells, requiring only one comparator and an 8-bit register per column, resulting in a negligible area compared with that of the PE.

True random number streams are of crucial importance as they are responsible for the proper update mechanism of the background model. Switched-capacitor circuits allow us to implement random number generators (RNGs) based on chaotic maps with a great level of robustness under mismatch and process variation [24]. These circuits work with a non-linear transfer function, which can have different shapes, to produce output streams using as input the output of the previous iteration. As the chaotic maps are built *ad hoc* for our design, they feature the appropriate specifications in terms of output voltage range or bit rate, among others. In this work, a variation of the solution reported in [25] was designed. In our work, different chaotic maps are implemented with an analog arithmetic unit, explained in the previous section. These circuits need to operate with an offset voltage to be compatible with the voltage range of subsequent circuits. Thus, the output voltage $V_{j+1}$ of iteration $j+1$ is obtained from $V_j$ as:

$$V_{j+1} = \begin{cases} \mu(V_j - V_{ref}) + V_{ref}, & \text{if } V_j < Et + V_{ref} \\ \frac{\mu}{\mu-1}(V_{ref2} - V_j) + V_{ref}, & \text{else} \end{cases} \tag{11}$$

where $V_{ref}$ is the offset voltage, $V_{ref2} = 1 + V_{ref}$, $\mu$ is the parameter that controls the position of the skew peak and $Et = 1/\mu$. Even when the analog output is chaotic by construction, a strong temporal correlation exists. For reducing this correlation and improving the output quality of the bitstream, a combination of four different skew-tent

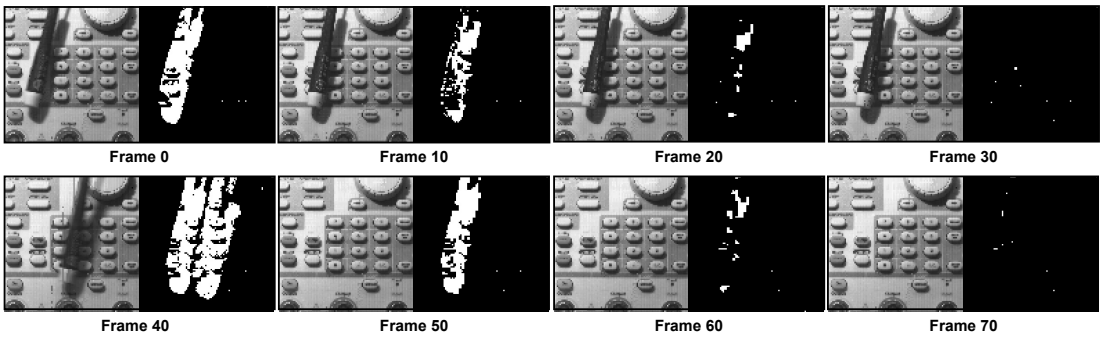



Fig. 11: HOPBAS10K qualitative tests, where a screwdriver is detected over the background of an oscilloscopes's button panel, then included into the model and finally removed from the scene.

TABLE I: Results for the tests of HOPBAS10K analysis with the *changedetection* dataset compared with those from the software versions of HO-PBAS and PBAS with a background model of 8 samples.

| | **Software** | | **Hardware** |
|---|---|---|---|
| | PBAS | HO-PBAS | HOPBAS10K |
| *shadow* | 0.6864 | 0.7367 | 0.3422 |
| *badWeather* | 0.5492 | 0.6987 | 0.4325 |
| *PTZ* | 0.0447 | 0.1220 | 0.0248 |
| *dynamicBackground* | 0.1555 | 0.4994 | 0.3427 |
| *cameraJitter* | 0.2387 | 0.5585 | 0.2847 |
| *thermal* | 0.6791 | 0.3694 | 0.4352 |
| *intermittentObjectMotion* | 0.4107 | 0.2793 | 0.2197 |
| *turbulence* | 0.0625 | 0.6718 | 0.1435 |
| *baseline* | 0.7512 | 0.7052 | 0.4837 |
| *lowFramerate* | 0.3725 | 0.5078 | 0.0718 |
| *nightVideos* | 0.2482 | 0.4009 | 0.1958 |
| ***Overall*** | 0.3863 | 0.4981 | 0.3604 |

maps was developed. The schematic of the RNG system implemented on our chip can be seen in Fig. 9 accompanied with experimental results for both analog and digital output, showing a random behavior for both cases good enough to provide segmentation results with performance metrics in the same order as that of the software version of original PBAS over the *changedetection* dataset, as it will be shown in the next section.

## IV. Experimental Results and Analysis

All designs described in the previous section were integrated into a standard CMOS 180 nm IC, resulting in a 98×98 pixel array with internal/external control signals generation. The chip microphotograph is shown in Fig. 10. Qualitative results for both the imager and the segmentation result can be seen in Fig. 11, where the process of a static foreground object removal can be observed.

As it can be seen in Fig. 11, the qualitative analysis of HOPBAS10K shows the expected response, which is to detect the screwdriver at frame 0, then absorbing it into the background model through frames 10 to 30, and finally detecting it again when it is moved at frame 40, and also including the ghost into the model through frames 50 to 70. This example shows the proper behavior of the system. Nevertheless, a different background model update speed could

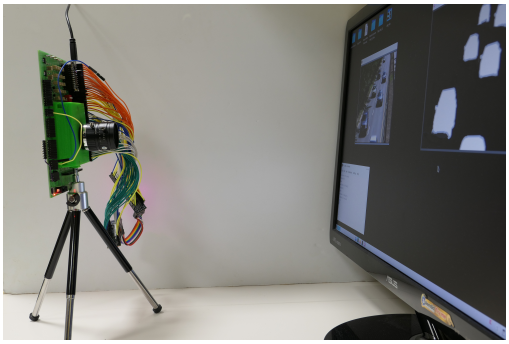


Fig. 12: Experimental setup with the camera prototype capturing the images of the *changedetection* benchmark from the PC screen.

be obtained by modifying some algorithm parameters, i.e., $p_{min}$ and $p_{inc}$ of (3). To further validate the proper operation of the system in more challenging situations, assessing that the complex functional blocks that make HO-PBAS obtain a better figure of merit than other algorithms in the state-of-the-art, a quantitative test was executed. The experimental setup of this test is displayed on Fig. 12, where the camera prototype was capturing and processing the images from the *changedetection* benchmark at 50 fps. The setup platform comprises, in addition to the HOPBAS10K IC and the lens system, an FPGA feeding control signals and reading the results, a separate microcontroller that writes HOPBAS10K configuration registers, and a PC to read the results stored on the FPGA. The segmentation result of each frame was then compared against the manually annotated groundtruth of the dataset, extracting the F-Measure figure of merit for each video.

The results for each category can be seen in Table I, along with the corresponding values for the software version of HO-PBAS and PBAS for a background model of 8 samples, where the algorithm is fed with the ideal images from the dataset. It can be seen that in many categories our hardware implementation achieves better performance than the original software version of PBAS. However, due to typical hardware errors, the software version of HO-PBAS outperforms HOPBAS10K, as expected. Overall, a decrease in performance of 27.6% with respect to the HO-PBAS, and 6.7% to the PBAS is

TABLE II: Comparison with the state-of-the art in terms of hardware metrics. Right column labeled with * for the power consumption and FoM of this work refers to estimated values through post-layout simulations with the problem of the power gating of the comparator in the segmenter block solved. FD: Frame differencing, BS: Background subtraction, DT-DBS: Double threshold dynamic background subtraction

| | [15] | [11] | [14] | [12] | This Work | |
|---|---|---|---|---|---|---|
| Process | 180 nm 1P6M | 350 nm 2P3M | 110 nm 1P4M | 180 nm 1P6M | 180 nm 1P6M | |
| Supply A/D (V) | 0.8 | 3.3 | 1.2/3.3 | 1.2/0.8 | 1.8/3.3 | |
| Pixel Size ($\mu m^2$) | 10 × 10 | 26 × 26 | 4 × 4 | 7.9 × 7.9 | 47 × 47 | |
| Fill factor (%) | 20 | 12 | 49 | 33.7 | 2.9 | |
| Pixel array | 132 × 104 | 64 × 64 | 640 × 480 (160× 120) | 256 × 216 (128× 108) | 98 × 98 | |
| Motion detection | FD | DT-DBS | DT-DBS | FD + BS | BS | |
| Domain | Mixed | Mixed | Digital | Mixed | Mixed | |
| Processing architecture | Pixel | Pixel | Column | Column | Group of pixels | |
| Processing framerate (fps) | 510 | 13 | 8 | 15 | 100 | |
| Processing power ($\mu$W) | 74.4 | 33 | 344 | 2.36 (14FD+1BS) | 15372 | 750* |
| FoM (pJ/pixel·frame) | 35.6 | 620 | 2240 | 11.4 | 16005 | 780* |
| Validation with dataset (F-Measure) | - | - | - | - | 0.36 | |

observed, where it has to be taken into account not only the performance decrease from hardware inaccuracies but also from the experimental setup itself with images being captured from a PC screen.

Regarding the comparison with the state-of-the-art in terms of hardware metrics, we performed a study on published works with motion detectors with on focal plane processing. It has to be noticed that all of them implement less elaborated algorithms when compared with HO-PBAS, commonly based on frame differencing, which can not deal with challenging situations for foreground detectors as for example, dynamic background. Also, it should be noted that none of them appear in the background subtraction contest *changedetection* [13], and that their hardware evaluation regarding algorithm performance was mainly qualitative, and not against a public benchmark. That explains for instance the large pixel pitch and small fill-factor caused by the number of functional blocks required in the HO-PBAS datapath shown in Table II. This also has a large impact on power consumption, as it can be seen in the FoM, which is larger than others. To improve these numbers several strategies could be applied. For instance, more advance technology nodes would significantly reduce the required area of the processing circuitry. Furthermore, techniques such as 3D CMOS stacking or backside illuminated IC could be of a great help to improve its figure-of-merit. Also, considering that some margin is still available with respect to the maximum processing speed, more in-pixel processing circuit sharing could be applied, reducing the architecture parallelism but also the required area for these circuits.

Even taking into account the gap in the algorithm complexity compared with the state-of-the-art, the measured power consumption seemed too high. To detect the source of such a large power consumption, a power break-down was performed using simulation results. After its results were analyzed, it was seen that the comparators in the segmenter and in the block that updates $p(x_i)$ were not properly being put into an idle state, with a high constant power consumption even when they were disabled. Post-layout simulations were run to estimate the array power consumption with the comparator problem solved. The result of such simulations can be seen in the right column of the FoM for this work in Table II, labeled as *. As shown, the estimated FoM is better than that of [14], and of the same order as that of [11]. Moreover, our power consumption could be further improved by more sophisticated power-gating strategies, turning on just the blocks that are being used at each step, and not the entire PE at the processing stage. However, that would require additional control signals that are not implemented on the chip.

## V. Conclusions

This paper addresses the design and hardware implementation of a hardware-oriented version of PBAS on a standard CMOS vision sensor, that aims to fill the gap between literature work that optimizes hardware metrics such as spatial resolution or power consumption at the cost of processing with less elaborate motion detectors, and more challenging state-of-the-art algorithms for background subtraction. For this purpose, the top-ranked background subtraction algorithm PBAS was first modified to reduce its hardware requirements while maintaining its segmentation performance, resulting in HO-PBAS. Validation with the public dataset *changedetection* was run, achieving a slightly better performance for a background model of 8 samples compared with PBAS using 35 samples.

After HO-PBAS was designed, it was implemented on a 98×98 pixel 180 nm CMOS vision sensor, achieving a good algorithm performance when compared with its corresponding implementation in software. A performance study using the dataset *changedetection* was performed, capturing the images with HOPBAS10K from a PC display, and extracting metrics with the computed segmentation result. These tests showed a performance decrease of 6.7% with respect to PBAS and of 27.6% to the software version HO-PBAS. Regarding the power consumption, a sub-optimal strategy was applied in the power-gating of some processing blocks. Post-layout simulations improving this aspect showed that HOPBAS10K would have a reasonable FoM when compared with the state-of-the-art, even accounting for the fact that HO-PBAS is a more elaborated algorithm than those usually run on the focal plane of CMOS vision sensors.

## REFERENCES


[1] R. Gomez-Merchan, J. A. Leñero-Bardallo, and A. Rodríguez-Vázquez, "A self-powered asynchronous image sensor with tfs operation," *IEEE Sensors Journal*, vol. 23, no. 7, pp. 6779–6790, 2023.

[2] M. Adil, V. G. Menon, V. Balasubramanian, S. R. Alotaibi, H. Song, Z. Jin, and A. Farouk, "Survey: Self-empowered wireless sensor networks security taxonomy, challenges, and future research directions," *IEEE Sensors Journal*, vol. 23, no. 18, pp. 20519–20535, 2023.

[3] B. Blanco-Filgueira, D. García-Lesta, M. Fernández-Sanjurjo, V. M. Brea, and P. López, "Deep Learning-Based Multiple Object Visual Tracking on Embedded System for IoT and Mobile Edge Computing Applications," *IEEE Internet of Things Journal*, vol. 6, pp. 5423–5431, June 2019.

[4] M. Maheepala, M. A. Joordens, and A. Z. Kouzani, "Low Power Processors and Image Sensors for Vision-Based IoT Devices: A Review," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1172–1186, 2021.

[5] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11, pp. 31–66, 2014.

[6] L. Maddalena and A. Petrosino, "A self-organizing approach to background subtraction for visual surveillance applications," *IEEE Transactions on image processing*, vol. 17, no. 7, pp. 1168–1177, 2008.

[7] R. R. Fratama, N. D. A. Partiningsih, E. H. Rachmawanto, C. A. Sari, P. N. Andono, *et al.*, "Real-time multiple vehicle counter using background subtraction for traffic monitoring system," in *2019 International Seminar on Application for Technology of Information and Communication (iSemantic)*, pp. 1–5, IEEE, 2019.

[8] K. Goyal and J. Singhai, "Review of background subtraction methods using Gaussian mixture model for video surveillance systems," *Artificial Intelligence Review*, vol. 50, no. 2, pp. 241–259, 2018.

[9] P. Tumas and A. Serackis, "Effective background subtraction algorithm for food inspection using a low-cost near infrared camera," in *2017 Open Conference of Electrical, Electronic and Information Sciences (eStream)*, pp. 1–4, 2017.

[10] 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, 2020.

[11] N. Cottini, M. Gottardi, N. Massari, R. Passerone, and Z. Smilansky, "A 33 uW 64 x 64 Pixel Vision Sensor Embedding Robust Dynamic Background Subtraction for Event Detection and Scene Interpretation," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 3, pp. 850–863, 2013.

[12] X. Zhong, M. Law, C. Tsui, and A. Bermak, "A Fully Dynamic Multi-Mode CMOS Vision Sensor With Mixed-Signal Cooperative Motion Sensing and Object Segmentation for Adaptive Edge Computing," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 6, pp. 1684–1697, 2020.

[13] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference*, pp. 1–8, IEEE, 2012.

[14] Y. Zou, M. Gottardi, M. Lecca, and M. Perenzoni, "A Low-Power VGA Vision Sensor With Embedded Event Detection for Outdoor Edge Applications," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 11, pp. 3112–3121, 2020.

[15] T.-H. Hsu, Y.-K. Chen, J.-S. Wu, W.-C. Ting, C.-T. Wang, C.-F. Yeh, S.-H. Sie, Y.-R. Chen, R.-S. Liu, C.-C. Lo, *et al.*, "A 0.8 V Multimode Vision Sensor for Motion and Saliency Detection with Ping-Pong PWM Pixel," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, pp. 110–112, IEEE, 2020.

[16] W.-C. Chen and S.-S. Jan, "Star tracking algorithm based on local dynamic background reduction for eliminating stray light interference from star spot data," *IEEE Sensors Journal*, vol. 23, no. 18, pp. 21534–21543, 2023.

[17] M. Hofmann, P. Tiefenbacher, and G. Rigoll, "Background segmentation with feedback: The pixel-based adaptive segmenter," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference*, pp. 38–43, IEEE, 2012.

[18] D. García-Lesta, P. López, V. M. Brea, and D. Cabello, "In-pixel analog memories for a pixel-based background subtraction algorithm on CMOS vision sensors," *International Journal of Circuit Theory and Applications*, vol. 46, no. 9, pp. 1631–1647, 2018.

[19] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *European conference on computer vision*, pp. 751–767, Springer, 2000.

[20] M. Suárez, V. M. Brea, J. Fernández-Berni, R. Carmona-Galán, D. Cabello, and Á. Rodríguez-Vázquez, "Low-power CMOS vision sensor for gaussian pyramid extraction," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 2, pp. 483–495, 2017.

[21] L. Stark, J. M. Raynor, F. Lalanne, and R. K. Henderson, "A back-illuminated voltage-domain global shutter pixel with dual in-pixel storage," *IEEE Transactions on Electron Devices*, vol. 65, no. 10, pp. 4394–4400, 2018.

[22] J. A. Leñero-Bardallo, J. Fernández-Berni, and Á. Rodríguez-Vázquez, "Review of ADCs for imaging," in *Image Sensors and Imaging Systems 2014*, vol. 9022, p. 90220I, International Society for Optics and Photonics, 2014.

[23] Y. Zeng, S. Yang, Y. Liu, R. Bao, Z. Zhu, J. Lin, X. Zhou, Y. Chen, J. Yin, P.-I. Mak, and Q. Li, "A digital readout integrated circuit based on pixel-level adc incorporating on-chip image algorithm calibration for irfpa," *IEEE Sensors Journal*, vol. 23, no. 18, pp. 21747–21756, 2023.

[24] Y. Liu, R. C. C. Cheung, and H. Wong, "A Bias-Bounded Digital True Random Number Generator Architecture," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 1, pp. 133–144, 2017.

[25] J. L. Valtierra, E. Tlelo-Cuautle, and Á. Rodríguez-Vázquez, "A switched-capacitor skew-tent map implementation for random number generation," *International Journal of Circuit Theory and Applications*, vol. 45, no. 2, pp. 305–315, 2017.


**Daniel García-Lesta** received his PhD from the University of Santiago de Compostela (USC), Spain, in 2021. His research topic is the design of mixed-signal circuits for efficient computation, especially for video processing. Currently, he is a postdoc researcher at Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Spain.

**Diego Cabello** (M'96) received the BSc and PhD degrees in Physics from the University of Granada, Granada, Spain, and the University of Santiago de Compostela, Santiago de Compostela, Spain, in 1978 and 1984, respectively. Currently, he is a Professor of Electronics at Centro Singular de Investigación en Tecnoloxías da Información (CiTIUS), University of Santiago de Compostela, Spain. He has been the Dean in the Faculty of Physics between 1997 and 2002, and the Head of the Department of Electronics and Computer Science between 2002 and 2006, both in the University of Santiago de Compostela. His main research interests lie in the design of efficient architectures and CMOS solutions for computer vision, especially in early vision.

**Paula López** (M'02) received her PhD degree in Physics in 2003 and is currently an associate professor at the University of Santiago de Compostela. Her research interests include the design of mixed signal integrated circuits for image processing applications and the physical modeling of electronic devices, particularly CMOS image sensors, as well as the translation of these models into hardware description languages. She is author or coauthor of more than 50 research papers.

**Víctor Manuel Brea** received his PhD in Physics in 2003. Currently he is an Associate Professor at Centro Singular de Investigación en Tecnoloxías da Información (CiTIUS), University of Santiago de Compostela, Spain. His main research interests lie in the design of efficient architectures and CMOS solutions for computer vision, especially in early vision, as well as micro-energy harvesting.