A Low-Power VGA Vision Sensor With Embedded Event Detection for Outdoor Edge Applications

Yu Zou, Member, IEEE, Massimo Gottardi, Member, IEEE, Michela Lecca, and Matteo Perenzoni, Senior Member, IEEE

Abstract—We report on a low-power VGA vision sensor embedding event-detection capabilities targeted to battery-powered vision processing at the edge. The sensor relies on an always-ON double-threshold dynamic background subtraction (DT-DBS) algorithm. The resulting motion bitmap is de-noised, projected along xy-axes of the array of pixels and filtered to robustly detect moving targets even in noisy outdoor scenarios. The chip operates in motion detection (MD), applied on a QVGA sub-sampled image, looking for anomalous motion in the scene at 344 μW, and in imaging mode (IM), delivering full-resolution gray-scale images with associated local binary pattern (LBP) coding and motion bitmaps at 8 frames/s and 1.35 mW. The 4-μm pixel vision sensor is manufactured in a 110-nm 1P4M CMOS and occupies 25.4 mm².

Index Terms—Background subtraction (BS), event detection, local binary pattern (LBP), low-power vision sensors, motion detection (MD).

I. INTRODUCTION

LOW power consumption and energy management are of main importance for long-lasting battery-powered sensor nodes. This issue is even more challenging for vision systems at the edge, where visual tasks are executed close to the sensor, given a large amount of information to be managed in real time and with a limited energy budget.

Recently, some image sensors have been reported performing very low power consumption [1]–[9] in the order of tens of μW. Nevertheless, in such sensors, off-chip image processing, and wireless communication still remain the main sources of power consumption, placed about one or two orders of magnitude above that one of the sensors; therefore, they should be used carefully. One approach in this direction is to limit the use of these resources only when really needed, triggered by events occurring in the scene. Here, the most straightforward solution is to provide the image sensor with event detection capabilities so that it generates an alert signal switching ON the external processor, which is normally in idle state, to execute further visual processing. In this regard, different techniques of trigger on-motion have been implemented to activate the external computing resources and to enhance the system energy efficiency [1]–[5]. Nevertheless, the main drawbacks in this approach are the custom pixel design and the reduced performance against the software-based algorithmic counterpart. In fact, while analog processing might be very compact and energy efficient, the algorithm programmability is very limited as well as its performance, reducing the sensor node reliability and its lifetime. This is especially true in outdoor applications, where the event-detection algorithm needs to cope with noisy scenarios, with the risk of generating a large rate of false positives, that activate the external processor uselessly with a dramatic increase of the system average power consumption. The typical approach of published vision sensors with embedded event detection relies on simple motion detection (MD) algorithms, such as frame difference (FD) and background subtraction (BS), that are easy to be implemented on-chip but often unreliable for most real applications. Here, we describe a vision sensor for real-time event detection in outdoor scenarios [10]. The sensor embeds a double-threshold dynamic BS (DT-DBS) algorithm, running continuously and performing MD with energy constraints, and it is connected to an external processor that is activated only when a potential event is revealed. While sensors embedding FD and BS exhibit very low-power performance, this work achieves an accurate detection of the event with a low rate of false positives and limited power resources. Based on this result, we aim at introducing a novel framework for the evaluation of low-power sensors, taking into consideration not only their standalone energy efficiency but also their performance in real-world applications. From this point of view, the proposed sensor outperforms other devices in the state of the art.

This work focuses on the architecture and performance of the chip-embedded algorithm rather than on the image sensor performance.

Although a standard 3T pixel was used, this implementation is fully compliant with a pinned photodiode (PPD) imager.

This article outlines as follows. Section II and III describe, respectively, the chip-embedded algorithm and the overall sensor architecture with its basic building blocks. Section IV presents the experimental results and compares the sensor with similar devices from the electrical and functional point of view. Finally, Section V concludes this article.

II. DOUBLE-THRESHOLD DYNAMIC BACKGROUND SUBTRACTION AND EVENT DETECTION

Detecting an event in a video means to identify one or more objects that change their position or appearance over time,
i.e., over a sequence of frames. This operation is of great
importance for many applications, such as video surveillance,
human behavior understanding and monitoring, traffic control,
and robot navigation, and it is particularly challenging because
of the wide range of circumstances under which an event
can be observed. Variations of the illumination, noisy
acquisitions, changes of perspective, motion speed,
occlusions, complex, and dynamic backgrounds are some of
the many factors that must be taken into account to develop
a robust algorithm for event detection. BS and FD are two
popular methods for which several on-chip implementations
exist [1]–[4], [6]–[8]. BS [11] performs event detection by
subtracting each frame from a reference image, which models
the static environment of the event and which is usually
updated across the video in order to manage possible ambient
changes. Any pixel whose intensity value is significantly
different from the corresponding background intensity value
is labeled as motion pixel. Mathematically, the motion map

\[ H(i) = \begin{cases} 
1, & \text{if } d(V(i), B(i)) > \text{TH} \\
0, & \text{otherwise}
\end{cases} \]  

where \( i \) is a pixel, \( V \) is a frame, \( B \) is the background image,
\( d \) is a function measuring intensity differences, and \( \text{TH} \) is
a pre-defined threshold. In the simplest BS implementation,
\( d \) is the absolute value of the difference between \( V(i) \) and
\( B(i) \). In this case, the on-chip implementation requires a digital
frame buffer to store the background image \( B \) for long terms
in order to be compared with the current frames. A main
drawback of BS is its sensitivity to background variations
and noise that often occur in outdoor scenarios. To overcome
these issues, more sophisticated methods have been developed,
such as adaptive BS techniques, statistic-based approaches,
and multiple thresholds for MD [12]–[16]

FD can be considered as a special case of BS. In fact,
FD defines the motion pixels through the conditional equa-
tion (1) but replaces the background image \( B \) with a frame
\( V' \) consecutive to \( V \). Equation (1) with \( B = V' \) can be imple-
mented on silicon in a straight-forward way with low power
consumption and with low area occupancy. In this case, FD is
much more efficient than BS since it does not require to build,
save, and update a background model. FD is generally more
robust to changes in the environment than BS, which needs to
be updated accordingly; but as a drawback, it is very sensitive
to the frame rate, the speed of the moving objects, and noise.
As a result, in many real-world applications, FD produces a lot
of false positives representing a critical parameter, especially
in energy-efficient systems. Better performance is generally
achieved by temporal difference approaches, where any frame
\( V \) is compared with two or more consecutive frames, e.g., [17],
and by block-wise FD methods, where the FDs are computed
between image patches instead of pixels, e.g., [18].

Statistical analysis, optical flow information, additional fea-
tures, and post-processing algorithms are also employed to
improve the results both on BS and FD [19]–[21], while
some works even propose to combine BS and FD, e.g., [22].
Nevertheless, the computational pipeline of such approaches

\begin{align*}
&\text{if } V_i - V_{\text{Max}} > 0 \text{ then } V_{\text{Max}} \leftarrow V_{\text{Max}} + \Delta_{\text{OPEN}} \\
&\text{if } V_i - V_{\text{Max}} \leq 0 \text{ then } V_{\text{Max}} \leftarrow V_{\text{Max}} - \Delta_{\text{CLOSE}} \\
&\text{if } V_i - V_{\text{Min}} < 0 \text{ then } V_{\text{Min}} \leftarrow V_{\text{Min}} - \Delta_{\text{OPEN}} \\
&\text{if } V_i - V_{\text{Min}} \geq 0 \text{ then } V_{\text{Min}} \leftarrow V_{\text{Min}} + \Delta_{\text{CLOSE}}
\end{align*}

while (4) defines the conditions for \( P_i \) to be hot (HP2 = 1), i.e.,
whose changes are anomalous compared with its past history

\[ (V_i - V_{\text{Min}} < -\Delta_{\text{HOT}}) \land (V_i - V_{\text{Max}} > \Delta_{\text{HOT}}) \]

where \( \Delta_{\text{OPEN}}, \Delta_{\text{CLOSE}}, \text{ and } \Delta_{\text{HOT}} \) are user-defined algorithm
parameters. \( V_{\text{Max}} \) and \( V_{\text{Min}} \) act as two low-pass filters of \( V_i \)

Fig. 1. Operating principle of the sensor embedded DT-DBS. In this case,
the pixel voltage \( V_i \) changes abruptly, while \( V_{\text{Max}} \) and \( V_{\text{Min}} \) approach \( V_i \) at
two different speeds, according to (2) and (3).

is extremely hard to be embedded on a low-power sensor with
memory constraints.

To overcome the mentioned limitations, this work proposes
a novel vision sensor architecture that embeds a DT-DBS
algorithm for the real-time detection of moving objects in
gray-level videos depicting indoor and outdoor scenarios. The
background is here continuously modeled at each pixel by two
thresholds that update over time. A pixel is, thus, detected as
motion pixel, or hot pixel, if its intensity value falls out of
the range bounded by the two thresholds. The resulting motion
label map is de-noised through programmable morphological
filters applied to the image and the horizontal and vertical
projections of the motion pixels. When the sensor detects
sufficiently large regions of motion pixels, it generates an
alarm that is sent to the external processor for further actions.
Despite that the hardware implementation of the algorithm
requires additional resources and hardware overhead compared
with standard FD and BS algorithms, it has been proven to
provide a more reliable result, characterized by a very low
rate of false positives, especially in outdoor scenarios. This
means that the external processor connected to the sensor is
turned on only when the probability of a real alert is high,
granting highly efficient energy management.

A. Double-Threshold Dynamic Background Subtraction

In the proposed algorithm, the background is represented by
two thresholds \( V_{\text{Max}} \) and \( V_{\text{Min}} \), associated with each pixel and
updated at each frame. Let \( V_i \) be the intensity value at the pixel
\( P_i \), and let \( V_{\text{Max}} \) and \( V_{\text{Min}} \) be the corresponding threshold
values in the background model. The operating principle
of the DT-DBS algorithm of Fig. 1 is described by (2) and (3),
which regulates the two thresholds update (\( V_{\text{Max}}, V_{\text{Min}} \))
and have asymmetric behaviors according to their position with respect to \( V_i \). Their asymmetric behavior generates a gray-zone between the two thresholds inside which the pixel is considered normal (CLOSE). If the pixel is outside the safe-zone (OPEN) and sufficiently far from it, as stated by (4), it is labeled as hot pixel (HP\(_i\) = 1).

### B. Post-Processing

The DT-DBS algorithm allows the tuning of the sensitivity of the mechanism that generates the hot pixels so that after a certain time, repetitive intensity variations are ignored. Variations of the pixel signal caused by moving patterns, such as waves or leaves in the wind, are, therefore, suppressed after a certain number of frames. More precisely, the time response of the algorithm can be tuned according with the dynamics of the scene. Although this algorithm requires slightly larger computing resources than FD and BS, it is more efficient in suppressing noise produced by irrelevant events, such as repetitive motion, which is peculiar of outdoor scenarios. Removing noise at the early stage of image processing allows us to discount the computational burden of post-processing algorithms. Nevertheless, our experiments showed that further denoising of the hot-pixel map is, in general, necessary. In our sensor, this operation is performed on-chip by a programmable erosion filter applied on a 3 \( \times \) 3 pixel kernel that removes isolated pixels as well as small connected components. The user-defined erosion threshold must take into account the distance of the sensor from the acquired scene and the characteristics of the scene itself (e.g., forests, urban places) and the objects to be detected in (e.g., cars and humans). By default, the erosion threshold is fixed to 1 pixel. This value performs well in most of our experiments.

After the erosion operation, the resulting hot pixels are projected along the vertical and horizontal axes of the image. These \( x \) - and \( y \)-projections \( P_X \) and \( P_Y \) are designed to remove horizontal or vertical wired regions; thus, they prevent the alarm generation in case of images containing only sparse linear, tiny aggregations of pixels as those produced for repetitive motion, which is peculiar of outdoor scenarios. Removing noise at the early stage of image processing allows us to discount the computational burden of post-processing algorithms. Nevertheless, our experiments showed that further denoising of the hot-pixel map is, in general, necessary. In our sensor, this operation is performed on-chip by a programmable erosion filter applied on a 3 \( \times \) 3 pixel kernel that removes isolated pixels as well as small connected components. The user-defined erosion threshold must take into account the distance of the sensor from the acquired scene and the characteristics of the scene itself (e.g., forests, urban places) and the objects to be detected in (e.g., cars and humans). By default, the erosion threshold is fixed to 1 pixel. This value performs well in most of our experiments.

After the erosion operation, the resulting hot pixels are projected along the vertical and horizontal axes of the image. These \( x \) - and \( y \)-projections \( P_X \) and \( P_Y \) are designed to remove horizontal or vertical wired regions; thus, they prevent the alarm generation in case of images containing only sparse linear, tiny aggregations of pixels as those produced for instance by sea waves. Therefore, \( P_X \) and \( P_Y \) act as an additional de-noising filter. In the proposed sensor, \( P_X \) and \( P_Y \) are stored as 1-D vectors and scanned to identify their maximum numbers of contiguous pixels \( D_X \) and \( D_Y \). These values undergo a test to verify the following condition:

\[
(T_{X_L} < D_X < T_{X_H}) \land (T_{Y_L} < D_Y < T_{Y_H})
\]

where \( T_{X_L}, T_{X_H} \) and \( T_{Y_L}, T_{Y_H} \) are the target constrains, stored in the on-chip REGISTERS and usually related to the aspect ratio of the objects to be detected. If \( D_X \) and \( D_Y \) do not satisfy the condition (5), then no alarm is generated since the moving pixels do not match with the expected size of the objects to be detected.

### C. DT-DBS Versus FD and BS

The DT-DBS algorithm has been tested on a data set of 40 gray-level VGA videos in comparison with FD and BS. The videos used in these experiments depict differently lit outdoor scenarios with moving cars, people, and bikes. The brightness of these videos, computed as the average of the frame brightness over time, vary from 52 to 192 levels of intensity, corresponding to low-light and very lit outdoor scenarios, captured, respectively, in a shadowed environment and an open space at noon. The DT-DBS algorithm, as embedded in the chip, has been simulated through a software implemented in C++. Such a software, which exactly reproduces the sensor response, enables a fair, real-time comparison of our DT-DBS approach with two algorithms exploiting FD and BS, respectively. In this comparative analysis, FD and BS take as input a VGA video and under-sample it to a QQVGA. FD detects the motion map by computing the map \( D \) of the pixel-wise absolute distances between two subsequent frames \( V_1 \) and \( V_2 \), smoothing \( D \) by a Gaussian filter on a 3 \( \times \) 3 kernel and thresholding the result so that only pixels in \( V_2 \) with a smoothed intensity difference above 2 intensity levels are retained. BS implements the method in [23], which detects events by statistical analysis. The background is modeled by a Gaussian mixture model that is iteratively updated over time, and a pixel is labeled as a motion one if its probability density function does not match that of the background. For each video, the resulting binary motion maps output by our sensor, FD, and BS are up-scaled back to VGA. From their qualitative analysis, we observed that: 1) DT-DBS, FD, and BS perform similarly on videos characterized by a good illumination and with well-contrasted objects [see Fig. 2(a)], while their accuracy on event detection decreases in case of low-light or saturation [see Fig. 2(b)]; 2) FD is usually noisier than BS and DT-DBS, especially when removing the Gaussian filter [see Fig. 2(c)]; and 3) the performance of BS strongly depends on the background model and the procedure updating it [see Fig. 2(d)]. Some videos showing the motion maps computed by DT-DBS, FD, and BS are available starting from the web page http://tev.fbk.eu/node/183. As a general conclusion, we observe that DT-DBS is more adequate than FD and BS for detecting moving objects in outdoor noisy scenarios.

### III. Sensor Architecture

The vision sensor architecture, shown in Fig. 3, embeds a VGA imager with column-level readout (amplifiers) and analog-to-digital conversions (ADCs), a processing layer (Processors) executing DT-DBS, to generate the hot-pixel motion bitmap through two-thresholds/pixel \((V_{\text{max}}, V_{\text{min}})\), stored in the 6T SRAM and updated at every frame. Residual noise in the motion bitmap is cleaned up by a programmable erosion filter (Erosion Filter) applied over a 3 \( \times \) 3 pixel kernel to deliver the final bitmap, which is also used to build the two projection vectors (\(x\)-Projection, \(y\)-Projection), generating the alert signal to trigger the external processor. Algorithm parameters and other sensor settings are stored into the 16 \( \times \) 8-b registers (REGISTERS).

#### A. Imager

The imager consists of a VGA rolling shutter array of 3T 4 \( \mu \)m pixels. The column-level single-slope ADCs have a pitch of 8 \( \mu \)m; therefore, the 640 channels have been split...
into two 320 × 2 channels, placed at the top and bottom sides of the array, serving odd and even columns, respectively. Each 320-channel ADC block has its own ramp generator. Under MD, the always-on DT-BSMD is applied on a 120 × 160 pixels image (QQVGA, obtained subsampling the VGA array. This means that while the top-side ADC is off, only half bottom-side ADC bank works (i.e., 160 channels are on) to guarantee the MD algorithm to be continuously active, while minimizing the power consumption of the sensor.

In the imaging mode (IM), the sensor delivers full-resolution VGA image (DATA) by multiplexing the two 8-b outputs (GREY_T/GREY_B).

B. Column-Level Amplifier

Fig. 4 shows the column-level single-slope ADC with the pixel readout and its timing diagram. After row selection (Rsel = EN = H), the pixel voltage $V_p$ is stored onto capacitor C1, with $S = H$ and $S = H$, while the output voltage of amplifier is precharged to $V_{pre}$. Then, C1 is connected to the input of the amplifier ($S = L$), integrating charge onto C2 with a 2× gain. In a second phase, the pixel is reset, and its value $V_{rst} = V_{DDR}$ is stored onto C1 ($S = H$), with inverted polarity $Sch = L$. Then, C1 is connected to the amplifier, subtracting the reset charge from the charge stored on C2. The resulting voltage at the output of the amplifier will be $V_{pre} - 2 \times (V_{rst} - V_p)$.

The pixel readout phase has now concluded, and the amplifier is converted into a voltage comparator by opening the feedback loop ($PRE = H$) and connecting the terminal of C2 to the voltage ramp ($V_{ramp}$). Since $V_{ramp}$ starts with its high value ($V_h$), node A is pulled-up abruptly, unbalancing the comparator, which pulls down $V_{out}$. The decreasing voltage ramp, generated by the DAC_B, can now start, pushing node A down toward the ground. As soon as node A reaches $V_{ref}$, the comparator switches and $V_{out}$ increases toward $V_{dd}$, sampling the content of the digital counter COUNT_B onto the 8-b latches completing the conversion.

C. Column-Level Processor

The processing layer executes the DT-DBS on a sub-sampled image (120 × 160 pixel), as described in [10]. The output is a motion bitmap that is cleaned-up by the programmable erosion filter. The final bitmap is used to build the x- and y-projection vectors that are used to generate the alert signal. The DT-DBS, as depicted in Fig. 1 and described by (2)–(4), is implemented in a mixed mode (analog–digital) by partially exploiting the single-slope ADC operations to compare $V_i$ with the two thresholds and to check if the pixel is inside (CLOSE) or outside (OPEN) the gray-zone and to verify the HP conditions.

To complete the DT-DBS, (3) must be executed on the same pixel. This would require either to duplicate the electronics, exploiting the same voltage ramp, or to run twice the ramp executing (2) and (3) sequentially on the same row. However, both solutions imply a large overhead: the first on silicon area, with column-level processor pitch constraints, and the second on time and power consumption. To address this issue, we made the assumption that neighboring pixels have similar behaviors. Therefore, to simplify the implementation, we applied (2) on a pixel $P_{i,j}$ while (3) on $P_{i+1,j}$ so that we avoid to operate the voltage ramp twice per pixel, exploiting...
Fig. 4. Schematic of the single-slope ADC where, after the pixel readout, the chopper amplifier is converted into a comparator as part of the ADC. On the right, the timing diagram of the pixel readout ($T_{\text{READ}}$), ADC ($T_{\text{ADC}}$), threshold update ($T_{\text{UP}}$), and data dispatching ($T_{\text{OUT}}$) is shown for a sensor frame rate of 8 frames/s.

Fig. 5. Schematic of the circuit for alert generation relying on the HP $XY$-projections.

instead the regular image readout operation of the ADC, thus reducing power consumption and avoiding additional circuitry. Experimental results demonstrated that this assumption is more than acceptable. Moreover, the binary output of the two operations, executed on $P_{i,j}$ and $P_{i+1,j}$, is put in OR, turning into the final pixel status ($HP_{i,j}$). The advantage of exploiting the ADC voltage ramp to partially implement the DT-DBS algorithm is that it simplifies the required electronics: the 8-b comparison is made with an identity comparator using eight XORs in OR; the voltage difference $V_i - V_{\text{Max}}$ [see (4)] is done with a binary counter, which is activated only under OPEN conditions. For each selected pixel, the processor retrieves one of the two thresholds ($V_{\text{Max}}/V_{\text{Min}}$) from the SRAM, while, after signals comparison, the threshold is updated with $\text{OPEN}$ or $\text{CLOSE}$ and restored into the memory.

D. Programmable Erosion Filter

The $120 \times 160$ pixel raw motion bitmap, generated by (2)-(4), is de-noised by a programmable $3 \times 3$ pixel kernel erosion filter.

Generic filter is applied on pixel $P_{i,j}$ whose binary value is stored in $H_{i,j}$. The pixels of the three consecutive rows and the same $j$th column ($H_{i-1,j}$, $H_{i,j}$, and $H_{i+1,j}$) are summed together, and the result is summed to those of the $(j-1)$th and to those of the $(j+1)$th columns, providing a 4-b output $Q(0:3)$. The final result is compared with the user-defined threshold $NH(0:3)$, stored in one of the REGISTERS of Fig. 3

$$\text{if}(H_{i,j} = 1) \land \left( \sum_{j-1}^{j+1} \sum_{k} H_{h,k} \right) \geq NH \rightarrow HP_{i,j} = 1. \quad (6)$$

The 160 erosion filters provide a motion bitmap to generate the alert signal and to be sent off-chip for further processing.

E. Alert Generation

At each row, the detected HPs contribute to the generation of two $XY$ motion projection vectors (see Fig. 3). These vectors are low-pass filtered and binarized with the two user-defined thresholds $DX$ and $DY$. ALARM is generated only when the HPs form a region of a certain size and aspect ratio, which is defined by the constraint of (5). The schematic of the circuits generating the alarm is shown in Fig. 5. The size and the aspect ratio of an object are geometric features that may help to distinguish an object from another and to reject false positives. Of course, the specific values of these features depend on features, such as the distance of the sensor from the acquired scene and the sensor focal length. In most applications, this information is available and can be employed to set up the variability ranges of the size and the aspect ratio for a set of objects of interest, such as cars, humans, and boats.

F. Local Binary Patterns

Local binary patterns (LBPs) [24] are visual features encoding directional local contrasts over a pre-defined image patch. They capture local micro-structures of the image, e.g., edges, corners, flat regions, lines, and codify them in binary vectors. LBPs and their distribution of the image are widely employed in many machine vision applications, such as texture analysis [25], face detection [26], and hand gesture recognition [27]. For this reason, despite that the LBPs are here not involved in the event detection, we decided to embed their computation of the proposed chip and to deliver them to the processor for further visual tasks. Mathematically, the LBPs are defined as follows. Let $x$ be a pixel, and let $y_1, \ldots, y_N$ be $N$ pixels
value of these coordinates do not fall on integer number, the intensity block of the VGA array.

Pixel Pe refers to the neighboring pixels Pe1–Pe8 and is computed with the bottom-side block, while Po refers to Po1–Po8 and is executed by the top-side equi-spaced over a circumference centered at x and with radius R. The LBP code at x is the vector

\[ \text{LBP}_C(x) = [s(V(y_1) - V(x)), \ldots, s(V(y_N) - V(x))] \]  

(7)

where V is a frame and s is the function such that s(t) = 1 if \( t \leq 0 \), while s(t) = 0 otherwise. For any \( j = 1, \ldots, N \), the pixel \( y_j \) has coordinates \( R(\cos 2\pi/N, \sin 2\pi/N) \): when these coordinates do not fall on integer number, the intensity value of \( y_j \) is interpolated by considering its neighbors. By definition, the LBP code is invariant against changes of the illuminant intensity and, thus, against shadows. Invariance against rotations of \((2\pi h/N) \ (h \in \mathbb{Z})^\circ\) can be obtained by a circular bitwise cyclic shift of the entries of \( \text{LBP}_C(x) \). The LBP code is often mapped on a single integer number, which we call LBP value and is obtained as follows:

\[ \text{LBP}(x) = \sum_{j=1}^{N} 2^j s(P(j) - P(x)). \]  

(8)

Some sensors embedding the LBP computation have been recently developed. For instance, the works [28] and [29] propose two low-power sensors that compute the LBPs on a 3 × 3 window, respectively, along the directions \( k\pi/2 \) and \( k\pi/4 \), with \( k \in \mathbb{Z} \). As for the sensor described in [29], the LBP codes computed by the sensor proposed here differ from the standard ones defined in [24] in the geometry of the pixel neighborhood, which is a square instead of a circle. This choice avoids interpolating the intensity value of any pixel with not integer coordinates. For any \( x \), the proposed sensor considers a 5 × 5 window centered at \( x \) and computes the code \( \text{LBP}(x) \) by comparing the intensity of \( x \) with the intensities of the pixels displaced, as shown in Fig. 3. The use of the 5 × 5 windows enables the exploration of a wider region than that considered in [28] and [29], and it is also compliant with the architectural characteristics of the proposed sensor. Precisely, the LBP codes are computed by the sensor pixel-by-pixel during the read-out phase and then output along with the gray level image by simply multiplexing the 8-bit output bus of Fig. 3 (DATA), with SEL0. Since odd and even pixels are read out by the BOTTOM and TOP blocks of Fig. 3, computing LBP codes on a 3 × 3 pixel kernel as in [28] and [29] is complex to be implemented from the layout point of view. Therefore, in the adopted solution, pixels of odd/even rows and columns refer to the 8 pixels of the kernel, as depicted in Fig. 6(c). This allows the LBP processing to be decoupled and executed separately by the TOP and BOTTOM blocks. The correct LBP image readout (SEL0 = H) is performed by multiplexing the output bus (SEL1 = H), in the same way as it is done for the gray-scale image. Fig. 6(a) and (b) shows an example of gray-scale image and related LBP image directly executed by the sensor.

In comparison with the standard LBPs, the LBPs output by the sensor have a lower level of invariance against in-plane rotations: \((k\pi/2)^\circ\) versus the \((k\pi/4)^\circ\) of the standard LBPs, for any \( k \in \mathbb{Z} \). Apart from this difference, we observed that the LBPs of the sensor and the standard ones perform similarly on the description and matching of textured images. In particular, this performance has been measured on a case study, regarding the illuminant invariant texture retrieval. To this purpose, we considered the public data set Outex [30], consisting of 68 textures, each represented by 20 pictures captured under 3K, and those acquired under T3 have been taken, respectively, as queries and references. Therefore, the queries differ from the references only in the illumination. Each image (query and reference) has been split into its three color components, and each component has been described by the distribution of its LBPs values. The three LBP distributions have been then concatenated in a single histogram that has been used as a texture descriptor. For each query \( Q \), the references have been sorted from the most to the least similar to \( Q \). Here, the similarity between \( Q \) and any reference \( R \) was
defined in terms of the $L^1$ difference between their descriptors: the lower this distance, the more similar $Q$ and $R$ are. The accuracy on the matching was measured by the rank $\rho(Q)$, i.e., a parameter related to the position of the reference $R$ corresponding to $Q$ in the sorted list and measured as the ratio $\rho(Q) = (M - PQ / M - 1)$, where $M$ is the number of references and $PQ$ indicates the position of $R$ in the ordered list. The closer $\rho(Q)$ to one, the best is the matching accuracy. For both methods, the mean rank is greater than 0.998. In the 80% of the images, the ranks output by the two methods are the same, while, in the remaining 20%, they differ on average by less than 1%, meaning that the two LBPs computation have similar performance.

IV. EXPERIMENTAL RESULTS AND COMPARISON WITH SIMILAR LOW-POWER SENSORS

Table I shows the main chip characteristics. The electrical characteristics of the sensor have been compared with similar recently published vision sensor chips [1–4] and reported in Table II. It is worth noticing that the sensor can simultaneously deliver a full-resolution VGA gray scale, LBP coding, and QQVGA motion bitmap. On the other side, except [1], the figure of merit (FOM) of this work is not competitive neither in Motion (2.24 nW/pixel*frames/s) nor in IMs (549 pW/pixel*frames/s) compared with the other sensors. In the last two rows of Table II, the values of the power consumption are reported for each chip, both in IM and motion mode. The normalization was done referring each sensor to the same amount of pixels of this work (QQVGA for MD and VGA for IM) and to the same frame rate of 8 frames/s. Table III shows the required energy needed by the sensor to execute the DT-DBS algorithm on-chip with erosion filter and alert detection through $xy$-projections compared with the energy required to executed the same processing off-chip with an external processor [31]. Due to the mixed-mode implementation of the algorithm, it was not possible to estimate the average power consumption of the processing layer. Therefore,
the total digital power of the sensor (168 μW) was used although this value is overestimated since it includes the LBP computation, which cannot be disabled.

Moreover, we tried to compare the performance of event detection of the sensor against those sensors based on FD. For this purpose, we have chosen a scenario in which a boat is approaching the coast on a sunny and windy day. This scenario is very critical since it sets severe requirements for the event detection. Fig. 7 reports a snapshot, extracted from a 8-frames/s, 500-frame video, acquired with the sensor setup of Fig. 8, showing: the VGA gray-scale image [Fig. 7(a)]; the related QQVGA motion bitmap generated by the sensor [Fig. 7(b)]; and a hypothetical motion bitmap [Fig. 7(c)] generated simulating the FD algorithm applied on a sub-sampled format (120 × 160 pixels) of Fig. 7(a) and de-noised with the same 3 × 3 pixel erosion filter as that one used in Fig. 7(b). While the bitmap, delivered by the sensor and shown in Fig. 7(b) with red pixels, is very clean although the noisy scenario, with waves and swaying vegetation, the motion bitmap in Fig. 7(c) with green pixels looks quite noisy so that the boat cannot be even distinguished from the moving background. Plotting the number of HPs generated by the sensor (red) and those of the FD counterpart (green), as in Fig. 9(a), it is visible that the HP activity of FD is about one order of magnitude larger and most of it is due to the noise. According to the video ground truth, the moving boat is inside the scene until frame 90, generating a continuous alert. From frame 90 to frame 500, the boat is outside the scene, and no event is present. The graphs in Fig. 9(b) show the alert signals generated by the two approaches. Starting from frame 90, while the sensor generates 5% of false alerts, the FD counterpart has 46.5% of false positives, thus turning on the external processor more frequently to execute image processing tasks, with a larger waste of power.

Table IV reports an estimation of the required energy per frame vision systems, based on the sensor listed in II, with the best FOM in MD and IM operating modes, interfaced with the processor [31] to execute a people counting task through CNN [32] on alert. As expected, the energy under alert is dominated by the processor. The total estimated energy of the system takes into account the rate of false alerts generated by the two sensors in the case of the video of Fig. 7, from frame 91 to frame 500, where no alert should be generated. It is possible to see how, in this outdoor scenario, the proposed approach has about eight times better energy efficiency against the FD counterpart.

Referring to the estimated energy per frame, reported in Table IV, (9) defines the FOM quantifying the energy efficiency of an event-based vision system. The proposed FOM takes into account the sensor’s energy per frame in the two operating modes (MD and IM) as well as the energy per frame associated with the external processor (sleep and active modes) to execute a desired vision task on a certain image size at a given frame rate.

\[
FOM = \frac{[SE_{MD} + P_{SL} \cdot T]}{1 - R_a}(1) + \frac{[SE_{IM} + (P_{FC} + P_{IM}) \cdot (T_p) + P_{SL} \cdot (T - T_p) \cdot R_a]}{T_p + T_{IP}} (2) \quad (9)
\]

with \(T_p = T_{IT} + T_{IP}\) (Table V).
A low-power VGA vision sensor has been presented embedding a custom DT-DBS technique combined with motion bitmap projections to generate an alert in case of a moving target in the scene. The on-chip algorithm exhibits high reliability in noisy outdoor scenarios, minimizing the rate of false positives, which is one of the most critical parameters in event-based systems. While event-based CMOS vision sensors are typically compared only from the electrical point of view, we demonstrated that performance evaluation should be undertaken holistically, involving sensor, the external processor, and algorithm performance.

In this regard, in Table IV, we compared the estimated energy efficiency of this work with that of [5] by adopting the proposed FOM (9). Although [5] has lower sensor’s FOM, both in MD and IM, the proposed sensor exhibits about seven times better energy performance in a noisy outdoor scenario.

The authors would like to thank Nicola Furlan for his support in the system design and Manuele Rusci for valuable discussions.

REFERENCES
[1] O. Kumagai et al., “A 1/4-inch 3.9Mpixel low-power event-driven back-illuminated stacked CMOS image sensor,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2018, pp. 86–88.

[2] K. D. Choo et al., “5.2 Energy-efficient low-noise CMOS image sensor with capacitor-array-assisted charge-injection SAR ADC for motion-triggered low-power IoT applications,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2019, pp. 96–98.

[3] J. Choi, S. Park, J. Cho, and E. Yoon, “A 3.4 μW CMOS image sensor with embedded feature-extraction algorithm for motion-triggered object-of-interest imaging,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2013, pp. 478–479.

[4] G. Kim et al., “A 467nW CMOS visual motion sensor with temporal averaging and pixel aggregation,” in Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2013, pp. 480–481.

[5] X. Zhong, M. Law, C. Tsui, and A. Bermak, “A fully dynamic multimo-"mode CMOS vision sensor with mixed-signal cooperative motion sensing and object segmentation for adaptive edge computing,” IEEE J. Solid State Circuits, vol. 2, no. 1, pp. 1–14, Jan. 2020.

[6] M.-T. Chung and C.-C. Hsieh, “A 0.5 V 4.95 μW CMOS imager with 82 dB dynamic range and 0.055% fixed-pattern noise,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2012, pp. 114–115.

[7] M. Gottardi, N. Massari, and S. A. Jawed, “A 100 μW 128 × 64 pixels contrast-based asynchronous binary sensor for sensor network applications,” IEEE J. Solid State Circuits, vol. 44, no. 5, pp. 1582–1592, May 2009.

[8] S. Hanson and D. Sylvester, “A 0.45–0.7 V sub-microwatt CMOS image sensor for ultra-low power applications,” in Proc. IEEE Symp. VLSI Circuits, Kyoto, Japan, Jun. 2009, pp. 176–177.

[9] T.-H. Hsu et al., “A 0.8 V multimode vision sensor for motion and saliency detection with ping-pong PWM pixel,” in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2020, pp. 110–111.

[10] Y. Zou, M. Gottardi, M. Lecca, and M. Perenzoni, “A low-power VGA vision sensor with event detection through motion computation based on pixel-wise double-threshold background subtraction and local binary pattern coding,” in Proc. IEEE 45th Eur. Solid State Circuits Conf. (ESSCIRC), Sep. 2019, pp. 97–100.

[11] Y. Benezeth, P. M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, “Review and evaluation of commonly-implemented background subtraction algorithms,” in Proc. 19th Int. Conf. Pattern Recognit., Dec. 2008, pp. 1–4.

[12] A. Mittal and N. Paragios, “Motion-based background subtraction using adaptive kernel density estimation,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), vol. 2, 2004, p. 2.

[13] S. Brutzer, B. Hoferlin, and G. Heidemann, “Evaluation of background subtraction techniques for video surveillance,” in Proc. CVPR, Jun. 2011, pp. 1937–1944.

[14] M. Piccardi, “Background subtraction techniques: A review,” in Proc. IEEE Int. Conf. Syst., Man, Cybern., vol. 4, 2004, pp. 3099–3104.

[15] I. Haritaoglu, D. Harwood, and L. S. Davis, “W4: Who? When? Where? What? A real time system for detecting and tracking people,” in Proc. 3rd IEEE Int. Conf. Autom. Face Gesture Recognit., Apr. 1998, pp. 222–227.

[16] I. Haritaoglu, D. Harwood, and L. S. Davis, “W4: Real-time surveillance of people and their activities,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 809–830, Aug. 2000.

[17] W. Shuigen, C. Zhen, and D. Hua, “Motion detection based on temporal difference method and optical flow field,” in Proc. 2nd Int. Symp. Electron. Commerce Secur., vol. 2, May 2009, pp. 85–88.

[18] H. Wei and Q. Peng, “A block-wise frame difference method for real-time video motion detection,” Int. J. Adv. Robotic Syst., vol. 15, no. 4, Jul. 2018, Art. no. 172988141878363, doi: 10.1177/172988141878363.
based image sensors.

[19] N. M. Oliver, B. Rosario, and A. P. Pentland, “A Bayesian computer vision system for modeling human interactions,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pp. 831–843, Aug. 2000.

[20] M. Heikkila and M. Pietikainen, “A texture-based method for modeling the background and detecting moving objects,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 4, pp. 657–662, Apr. 2006.

[21] S.-G. Wei, L. Yang, Z. Chen, and Z.-F. Liu, “Motion detection based on optical flow and self-adaptive threshold segmentation,” Procedia Eng., vol. 15, pp. 3471–3476, Jan. 2011.

[22] Y. Li, Z.-X. Sun, B. Yuan, and Y. Zhang, “An improved method for motion detection by frame difference and background subtraction,” J. Image Graph., vol. 14, no. 6, pp. 1162–1168, 2009.

[23] Z. Zivkovic and F. van der Heijden, “Efficient adaptive density estimation per image pixel for the task of background subtraction,” Pattern Recognit. Lett., vol. 27, no. 7, pp. 773–780, May 2006.

[24] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 7, pp. 971–987, Jul. 2002.

[25] L. Nanni, A. Lumini, and S. Brahnam, “Survey on LBP based texture descriptors for image classification,” Expert Syst. Appl., vol. 39, no. 3, pp. 3634–3641, Feb. 2012.

[26] T. Ahonen, A. Hadid, and M. Pietikainen, “Face description with local binary patterns: Application to face recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 12, pp. 2037–2041, Dec. 2006.

[27] Y. Ding, H. Pang, X. Wu, and J. Lan, “Recognition of hand-gestures using improved local binary pattern,” in Proc. Int. Conf. Multimedia Technol., Jul. 2011, pp. 3171–3174.

[28] M. Gottardi and M. Lecca, “A 64 × 64 pixel vision sensor for local binary pattern computation,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 66, no. 5, pp. 1831–1839, May 2019.

[29] X. Zhong, Q. Yu, A. Bermak, C.-Y. Tsui, and M.-K. Law, “A 2Pj/pixel/direction MIMO processing based CMOS image sensor for omnidirectional local binary pattern extraction and edge detection,” in Proc. IEEE Symp. VLSI Circuits, Jun. 2018, pp. 247–248.

[30] T. Ojala, T. Maenpaa, M. Pietikainen, J. Viertola, J. Kyllonen, and S. Huovinen, “Outex–new framework for empirical evaluation of texture analysis algorithms,” in Proc. Object Recognit. Interact. Service Robots, vol. 1, 2002, pp. 701–706.

[31] E. Flamand et al., “GAP-8: A RISC-V SoC for AI at the edge of the IoT,” in Proc. IEEE 29th Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP), Jul. 2018, pp. 1–4.

[32] S. Benninger, M. Magno, A. Gomez, and L. Benini, “EdgeEye: A long-range energy-efficient vision node for long-term edge computing,” in Proc. 10th Int. Green Sustain. Comput. Conf. (IGSC), Oct. 2019, pp. 1–8.

Yu Zou (Member, IEEE) received the bachelor’s degree from Tongji University, Shanghai, China, and the Politecnico di Milano, Milan, Italy, in 2010, and the M.S. and Ph.D. degrees from the Politecnico di Milano in 2012 and 2016, respectively. Since 2016, she joined the Fondazione Bruno Kessler, Trento, Italy, as a Research Scientist, where she has been working with the Materials and Microsystems (CMM) Division, Integrated Radiation and Image Sensors (IRIS) Group. Her research interests are in the fields of low-power algorithm on-chip CMOS image sensors and single-photon avalanche diode (SPAD)-based image sensors.

Massimo Gottardi (Member, IEEE) is currently a Senior Researcher with Centre for Materials and Microsystems, Fondazione Bruno Kessler, Trento, Italy. He has served as a reviewer for several international journals and is a coauthor of more than 140 articles in refereed journals and international conferences. His current research interests include energy-efficient CMOS vision sensors, hardware-oriented algorithms for sensor-embedded image processing, memristor-based architectures for adaptive sensory systems, and IoT.

Michela Lecca received the M.S. degree in mathematics from the University of Trento, Trento, Italy, in 2002.

In 2002, she joined the Research Unit Technologies of Vision, Fondazione Bruno Kessler, Trento, where she is currently a Permanent Researcher. She has authored several research articles published in international journals and conference proceedings. Her research interests include all the aspects of low-level image processing, unsupervised object recognition and image retrieval, machine color constancy, and their implementation on smart embedded systems. Ms. Lecca is also a member of the International Association for Pattern Recognition (IAPR), Italian Association for Computer Vision, Pattern Recognition and Machine Learning (CVPL). She has taken part in the Program Committee of many editions of the Applied Computing Machine Symposia and carries on a reviewing activity for several international conferences and journals.

Matteo Perenzoni (Senior Member, IEEE) graduated in electronics engineering from the University of Padua, Padua, Italy, in 2002.

In 2004, he joined the Fondazione Bruno Kessler (FBK), Trento, Italy, as a Researcher, where he has been working with the Integrated Radiation and Image Sensors (IRIS) Research Unit. Meanwhile, he also taught courses on electronics and sensors at the Master and Doctorate School, University of Trento, Trento, from 2006 to 2010. In 2014, he was Visiting Research Scientist with the Microelectronics Department, TU Delft, Delft, The Netherlands. He is currently leading the IRIS Research Unit, FBK, working on the field of radiation and image sensor using custom and CMOS technologies. His research interests include advanced CMOS image sensors with a focus on single-photon detection, THz image sensors, and optimization of analog integrated circuits.

Mr. Perenzoni has been serving as a member of the Technical Program Committee of the European Solid-State Circuit Conference (ESSCIRC) since 2015 and the International Solid-State Circuit Conference (ISSCC) since 2018.